

# Large-Scale Available Bandwidth Measurements: Interference in Current Techniques

Daniele Croce, Emilio Leonardi, *Senior Member, IEEE*, and Marco Mellia, *Senior Member, IEEE*

**Abstract**—The end-to-end available bandwidth of an Internet path is a desirable information that can be exploited to optimize system performance. Several tools have been proposed in the past to estimate it. However, existing measurement techniques were *not* designed for large-scale deployments. In this paper we show that current tools do not properly work where multiple probing processes share a portion of a path. We provide experimental evidence to quantify the impact of mutual interference between measurements. We further analyze the characteristics of popular tools, quantifying (i) the impact of mutual interference, (ii) the total overhead imposed to the network and (iii) the intrusiveness of the measurement process in a large-scale scenario. Our goal is to effectively quantify the impact of concurrent measurements on current estimation techniques and to offer some simple guidelines for dimensioning a large-scale measurement system.

**Index Terms**—Available bandwidth, measurements, scalability, mutual interference, distributed systems.

## I. INTRODUCTION

The end-to-end available bandwidth (avail-bw) is one of the most important characteristics of an Internet path. This metric is fundamental for the operation of many emerging applications, such as video streaming, on-line gaming, P2P and content delivery systems, dynamic server selection, just to name a few. Thus, in recent years the research community has focused on the problem of the avail-bw estimation, and several techniques and inference methods have been proposed, e.g. [2], [5], [10], [13], [14], [18], [26], [28], [32], [35], [37]. The performance of these techniques has been widely discussed in many articles comparing them in a variety of settings and traffic conditions (see [29] and citations therein).

More recently, the attention of the research community has moved on how the avail-bw knowledge can be exploited to optimize network performance. For example, the avail-bw has been proposed for use in several algorithms for (i) server selection [17], (ii) route selection [20], (iii) admission control [4], (iv) congestion control [21], [33], and (v) for peer-to-peer systems optimization [9], [24], [36]. In these works, a certain measurement technique is assumed to be deployed on Internet hosts to monitor the avail-bw of the paths of interest. This information is then used to optimize the data transfers, or to choose a different path and server if a better one is found.

However, existing avail-bw measurement techniques were *not* designed for this purpose and have never been tested in a

scenario where several nodes measure simultaneously. Indeed, most studies analyze the performance of each technique in a controlled setting, where one single path is probed by only one pair of hosts at-a-time. This scenario is clearly unrealistic on Internet-wide deployments, where millions of nodes may want to estimate the path avail-bw. In such large-scale scenario, different hosts might interfere with each other when performing their measurements. It is then interesting to investigate if and how much the different existing techniques are affected by such events. Intuitively, these measurement tools alter the network conditions introducing probing traffic overhead. When multiple measurements coexist, the tools interfere mutually and the estimate can be expected to be biased somehow. Additionally, the measurement overhead and the impact on the ongoing background traffic might sum up, possibly congesting the network. Finally, different techniques employ different algorithms and probing strategies and, as we show, the measurement bias, the overhead and the intrusiveness grow differently on a tool-by-tool basis. The goal of this paper is to quantify these issues and not to provide new tools or methodologies.

While today the probability of simultaneous measurements (and their impact on the network) might seem negligible, this is expected to change significantly in the near future because the capacity of the Internet access links is rapidly increasing, making avail-bw estimation very appealing to optimize performance. Notice that adoption and implementation of these techniques in applications has already started. For example, as proposed in [21], [27], [33], a certain estimation technique is incorporated into TCP so that the transmitted data packets are shaped to monitor continuously the avail-bw. Or, consider a P2P system in which peers periodically probe each other to optimize the content distribution, as suggested in [9], [24]. In such cases, several measurements simultaneously probe the same Internet segment or even the entire path, e.g. in the case of parallel TCP connections. Even just considering a single host that runs an enhanced TCP version or that runs P2P applications, the interference of coexisting measurements becomes a real issue: in such scenarios, several measurements are likely to operate in parallel, performing simultaneous measurements on paths which share at least the first few hops. As further example, consider a CDN network for file downloading in which clients probe the spare capacity of different servers before starting the content download. Also in this case, the chance that several clients probe the last few hops toward the same server at the same time is large. At last, when the path bottleneck is at the ISP peering links, the large number of probing flows that goes through them makes the interference problem an important issue. Therefore, the risk

D. Croce, E. Leonardi and M. Mellia are with the Department of Electronics, Politecnico di Torino, Italy. E-mail: {firstname.lastname@polito.it}.

This work is supported by the NAPA-WINE Strep Project (FP7-ICT-214412) of the European Union.

Manuscript received January 25, 2011; revised May 30, 2011, and Jun 27, 2011; approved by IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT Editor B. Stiller.

of hosts measuring the avail-bw at the same time is real and hard to eliminate. To the best of our knowledge, we are the first to analyze such problems. A preliminary version of this work appears in [12].

The contributions of this paper are two-fold. First, using a dedicated testbed, we show that mutual interference is indeed an issue for current techniques and it severely impacts both the accuracy of the measurements and the underlying network performance. Second, we provide an in-depth analysis of three popular avail-bw techniques, namely packet pairs, packet chirps and packet trains [3], studying the impact of overlapping measurements on the *accuracy*, on the total *intrusiveness* and on the *overhead* imposed on the network.

Two scenarios are considered: (i) a distributed system which employs avail-bw estimation tools as off-the-shelf applications, running measurements in parallel to the normal functioning of the distributed system itself, and (ii) a distributed system which embeds the measurement technique in its internal functionalities, shaping the data transmission (e.g. modifying TCP or appropriately scheduling UDP packets) so that the measurements are piggy-backed on existing data transfers. In both scenarios, several issues arise when measurements are deployed at large-scale.

In the paper, Sec. II presents an overview of the available bandwidth measurement techniques; Experimental evaluation of the interference is presented in Sec. III, while Sec. IV and V present the analytical models used to assess the performance of the tools. A discussion on the possible implication of interference errors Sec. VI. Finally, Sec. VII presents related works and Sec. VIII summarizes the paper.

## II. BACKGROUND

### A. Available bandwidth techniques

In recent years, the research community has proposed many available bandwidth estimation tools. We can coarsely split them into three classes: *dispersion*-based, *congestion*-based and the emerging class of *delay*-based tools. The first type of tools estimate the path avail-bw by sending two (or more) packets with a specific Inter-Packet Gap (IPG) and observing the IPG increase, or *dispersion*, at the end of the path. This technique is also referred as “direct probing” or Probe Gap Model - PGM [23], [32], because the avail-bw is estimated directly by measuring the gap between probes.

Congestion-based tools, instead, leverage on the *self-induced congestion* principle: if the packet probes are sent at rate  $R$  higher than the avail-bw  $A$ , then they suffer queuing and they reach the receiver with increasing delay and lower rate. Conversely, if  $R < A$ , the probes are received without change. Tools based on this technique, also known as “iterative probing” or Probe Rate Model - PRM, send probes at different rates and detect when the path avail-bw is exceeded.

Finally, delay-based tools estimate the avail-bw by observing the delay experienced by a *single* packet traversing the path. This technique, which we can refer as “isolated probing” or Probe Delay Model - PDM, aims at detecting if the probe has encountered the network empty (no queuing delay) or not.

Tab. I summarizes the characteristics of popular tools and the methodology they adopt. Many other avail-bw techniques

TABLE I  
POPULAR AVAILABLE BANDWIDTH TOOLS.

Tool	Algorithm	Model	Inference Metric
IGI [18]	Closed-loop	PGM	Dispersion
Spruce [32]	Open-loop	PGM	Dispersion
Pathload [13]	Closed-loop	PRM	One-Way Delay
pathChirp [28]	Open-loop	PRM	One-Way Delay
BFind [2]	Closed-loop	PRM	RTT
Forecaster [26]	Open-loop	PDM	One-Way Delay
Divided pairs [6]	Open-loop	PDM	One-Way Delay

exist, e.g. [5], [10], [14], [15], [33], [35], [37], but the probing schemes employed are mostly variations of the tools listed in the table and the analysis undertaken in this paper can be easily extended to these other tools as well. We select Spruce, Pathload and pathChirp as prominent tools that represent classic *Packet Pairs*, *Packet Trains* and *Packet Chirps* methodologies<sup>1</sup>.

### B. Tools overview

1) *Spruce*: Spruce is a well known tool based on the Probe Gap Model. The sender host transmits pairs of packets with input gap  $\Delta_{in}$ , while the receiver measures the output gap  $\Delta_{out}$ . From the gap “dispersion”, the avail-bw  $A$  of the path is computed as

$$A = C \times \left( 1 - \frac{\Delta_{out} - \Delta_{in}}{\Delta_{in}} \right) \quad (1)$$

where  $C$  is the capacity of the bottleneck queue. Let  $S$  be the size of the probes, Spruce sets  $\Delta_{in} = S/C$  so that the packet pair arrives back-to-back on the bottleneck link. Indeed, the PGM paradigm assumes that there is a single bottleneck on the path and that the capacity  $C$  of this bottleneck is known. Furthermore, the bottleneck capacity is supposed to be the minimum of all the links of the path<sup>2</sup>. Spruce repeats the measurement 100 times, with a pausing time between each pair that is exponentially distributed, thus emulating a Poisson sampling process. Thanks to the PASTA property [7], the average avail-bw on the path can then be computed by averaging among all the 100 samples.

2) *Pathload*: Pathload is a classic PRM tool. It sends packet trains of constant rate  $R$  to probe the path, and changes  $R$  to test several rate values. Analyzing the One-Way Delays (OWDs) of the packets, it detects when the sending rate is higher than the avail-bw or not: suppose  $R > A$ ; then, packets suffer some queuing and are received with increasingly high delay. On the contrary, if  $R < A$ , packets reach the receiver without any OWD increase. By iteratively changing  $R$ , Pathload converges at the actual avail-bw value  $A$ . Pathload changes  $R$  in a “binary-search” fashion by updating an upper and lower bound of the avail-bw up to when a predefined measurement resolution is achieved.

<sup>1</sup>PDM tools are still in their early stages and no implementation has been publicly released yet.

<sup>2</sup>Because of these strong assumptions, PGM has been criticized in [23], although its performance is quite similar to other tools in practice [29], [30], [32]. We do not discuss this issue in this paper, since the focus is on the mutual interference and not on the absolute accuracy of the tools.

Each train consists of  $k = 100$  equally spaced packets. The decision if  $A$  is exceeded or not, is taken analyzing the entire OWD *trend*, since this proves to be more accurate than simply measuring the average receiving rate [19]. Two metrics are used to analyze the OWDs: the Pairwise Comparison Test (PCT) tracks the fraction of *consecutive* OWDs that are increasing; and the Pairwise Difference Test (PDT) measures the *overall* OWD variation from the beginning to the end of the train. The exact definition of these metrics can be found in [13]. After a train is received, the values of the PCT and PDT are compared against two thresholds,  $PCT_{th}$  and  $PDT_{th}$  respectively. Only if *both* values are above the corresponding threshold, then the OWDs are said to have an increasing trend, suggesting that  $R > A$ . Pausing times between successive trains guarantee that buffers along the path recover from possible congestion induced by the previous trains. As one can already perceive, this sophisticated algorithm complicates the analysis when several measurements interfere.

3) *pathChirp*: pathChirp is another popular tool based on the self-induced congestion. Differently from classic PRM, it uses particular packet trains, called *chirps*, which consist of  $k$  packets sent with Inter-Packet Gap (IPG) that is exponentially reduced. Considering the packets two-by-two, each consecutive “packet-pair” probes the path with increasing rate, while a single train probes  $k - 1$  different rates at once. The chirp rate varies from a lower rate  $L$  (the first two packets sent) up to an upper rate  $U = \gamma^{(k-2)} \times L$ . For fixed  $k$ , the *spread factor*  $\gamma$  controls both the granularity of the estimates and the range of probed rates. The receiver detects which is the rate  $R_l$  of last packet pair  $l$  received with the expected IPG, before the chirp rate exceeds the avail-bw. Indeed, packet pairs *after* pair  $l$  have rate that is higher than  $A$  and they suffer an increase in the One-Way Delay. Fig. 1 shows an example of the queuing delay seen by the receiver. Together with the final delay increase (or *excursion*) after pair  $l$ , pathChirp also takes into account all “transient” excursions which terminate before  $l$  and considers them as symptoms of self-induced congestion, i.e., a transient period of time during which congestion is detected. Then, the path avail-bw is computed using a weighted average of the “per-packet” avail-bw  $A_i$ , defined as

$$A_i = \begin{cases} R_i & \text{if OWD is increasing AND transient excursion} \\ R_l & \text{otherwise} \end{cases}$$

where  $R_i$  is the rate at which pair  $i$  is received. During a transient excursion,  $R_i$  represents a conservative estimate of the avail-bw. The rate  $R_l$ , instead, marks the beginning of the final excursion and represents the boundary between pairs with rate  $R_i < A$  and  $R_i > A$ . Note that, in absence of transient excursions, pathChirp avail-bw estimate is exactly  $\tilde{A} = R_l$ . Detecting  $R_l$  has thus a large impact on the estimation process and, as we show, is critical for pathChirp accuracy.

### III. EXPERIMENTAL EVALUATION

In this section we experimentally study the performance of the different techniques when multiple hosts are simultaneously estimating the avail-bw, *quantifying* the impact of mutual interference. We use a dedicated testbed composed of 62 identical machines. All hosts used for the experiments run

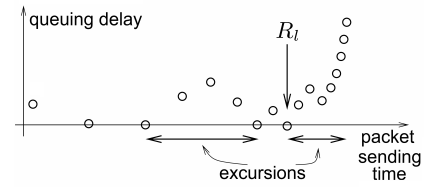


Fig. 1. Queuing delay of a chirp (from [28]).

TABLE II  
QUALITATIVE SUMMARY OF THE RESULTS.

Tool	Sensitiveness to interference	Total overhead	Global intrusiveness
<b>Pathload</b>	Very High	$\sim$ avail-bw	High
<b>Spruce</b>	Low	$O(N)$	Low
<b>pathChirp</b>	High	$O(N)$	Medium

the same version of Linux and are equipped with 100 Mbps Ethernet cards. Additional background traffic can be injected by two separate hosts, using a UDP traffic generator. Finally, a router is routing all traffic between senders and receivers through a 10Mbps Ethernet interface, so to create a *single bottleneck topology* with half of the hosts acting as senders and the other half as receivers. The choice of this simple topology is intentional: it is well known that with a single bottleneck all tools perform best, providing very accurate results. Therefore, this allows us to identify the measurement error caused only by the mutual interference, thus providing a quantitative estimate of the phenomenon. More complex scenarios, such as multiple bottleneck paths, will lead to results in which different impairments will mix together. This leads to superposition of measurement errors hard to be decoupled. Finally, running experiments over wide Internet is problematic due to the lack of ground truth.

The objective of this experimental testbed is to evaluate quantitatively the impact of having  $N$  hosts measuring simultaneously. The results are compared both against the *nominal avail-bw*, i.e., the link capacity minus the eventual background traffic (i.e. ignoring probing traffic generated by other concurrent measurement tools, which should be in principle negligible), and the *actual avail-bw*, i.e., taking into account also the traffic generated by the other  $N - 1$  probing flows. Each avail-bw tool continuously runs on each hosts after an initial exponentially distributed idle time to avoid synchronization between measurements. All tool parameters are left to their default values. The experiments run freely for several minutes (depending on the running time of each tool, up to 5 min/experiment), we collected a large set of measurements, and repeated this process several times. In total, we ran over 900 different experiments. Due to space constraints, we show here a subset of the results obtained and we refer the reader to [11] for additional results. A qualitative summary of the results is reported in Tab. II. A more detailed discussion is presented in Appendix A. Finally, to see if the impact of mutual interference remains the same also on higher-speed links, we recreated the same topology in the *ns-2* simulator considering a 1 Gbps bottleneck link. Tools provided

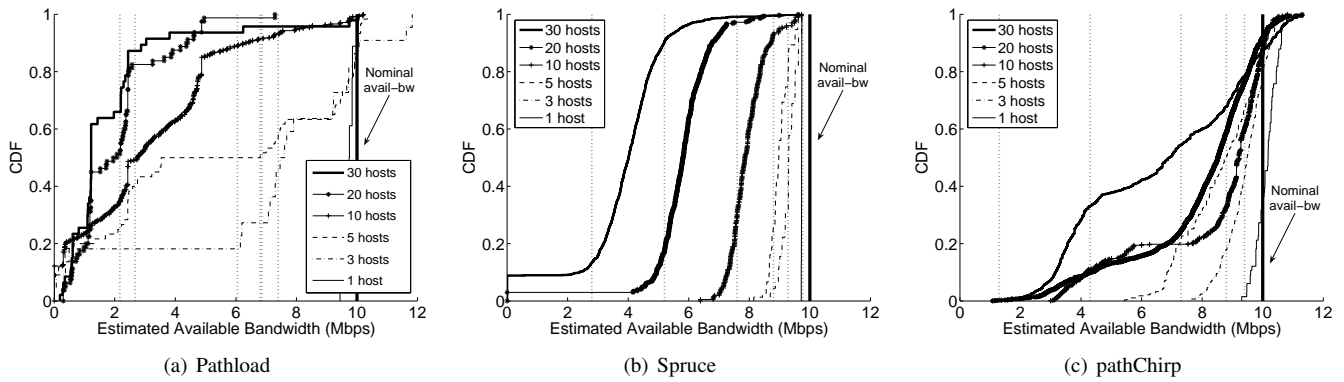


Fig. 2. Cumulative Distribution Function of Avail-bw estimation for different values of  $N$ . No background traffic is present. Vertical lines refer to the actual avail-bw.

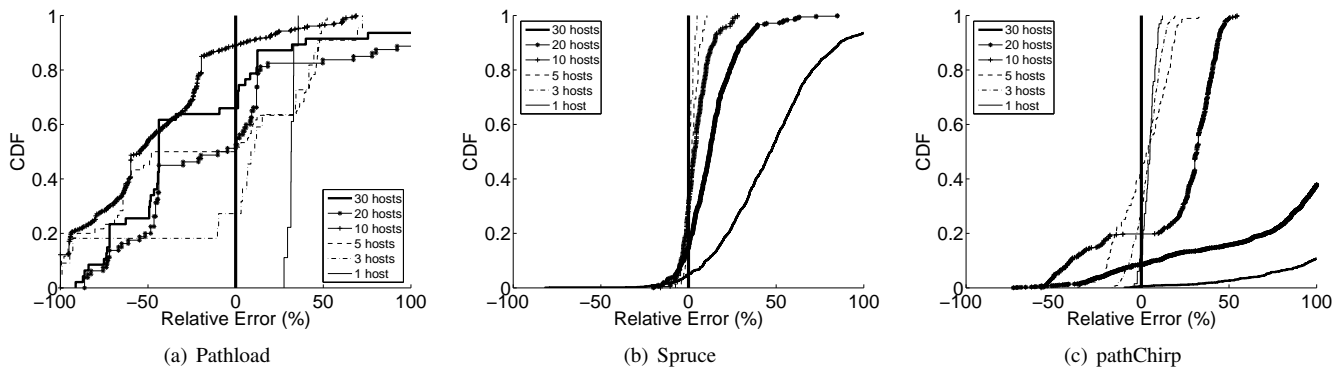


Fig. 3. Cumulative Distribution Function of relative error versus actual avail-bw for different values of  $N$ .

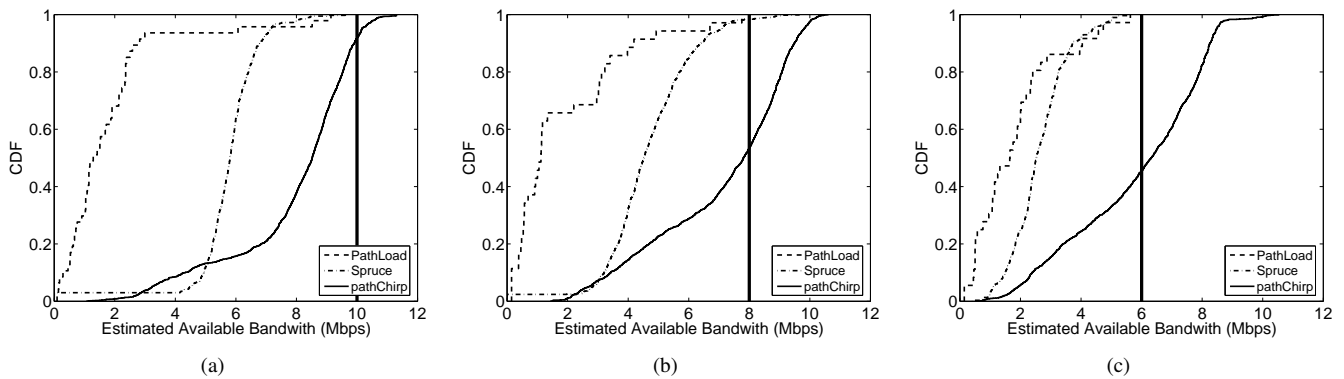


Fig. 4. Cumulative Distribution Function of avail-bw estimation for different values of the background traffic,  $N = 20$ .

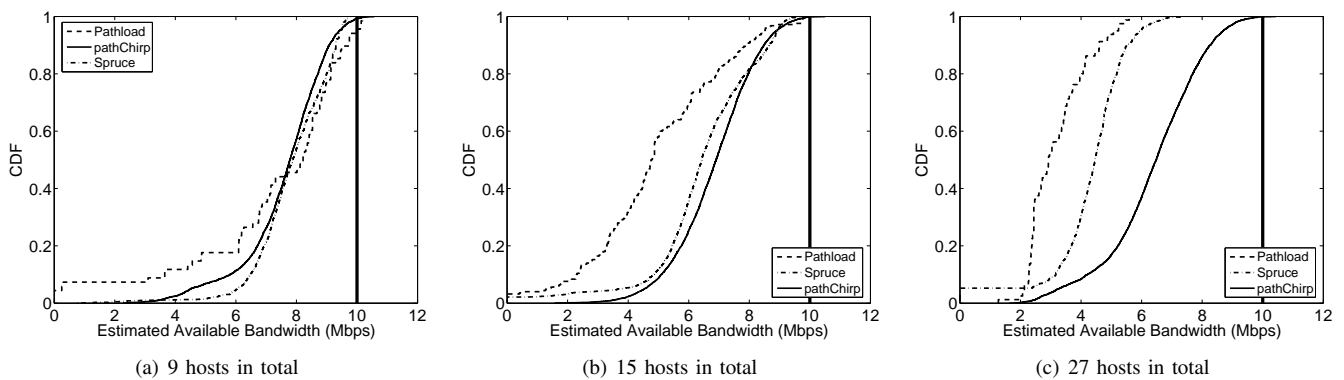


Fig. 5. Cumulative Distribution Function of avail-bw estimation when all techniques run at the same time, no background traffic.

in [29] are used. The simulation results scale similarly to the testbed results and can be found in [11].

#### A. Homogeneous scenario

We evaluate now the impact of mutual interference in an homogeneous scenario, where all hosts run the same application to measure the avail-bw. We compare the performance of Pathload<sup>3</sup>, Spruce and pathChirp when the number of active hosts is  $N = 1, 3, 5, 10, 20, 30$  and we vary the amount of background cross-traffic, thus varying the amount of avail-bw of the path.

1) *No background traffic*: in this scenario, the nominal avail-bw is  $A = 10$  Mbps. For each of the three tools, Fig. 2 reports the Cumulative Distribution Function (CDF) of the estimated avail-bw considering all the results that were generated during each experiment. Results show that (i) when  $N = 1$  (no interference), all estimations are very close to the nominal avail-bw, and (ii) when  $N$  increases, *all* tools suffer from mutual interference and results can significantly differ from the nominal avail-bw. In particular, Pathload (Fig. 2(a)) estimates are highly variable and, interestingly, are always concentrated around certain specified values (10, 5, 7.5, 2.5 Mbps, etc.). The reason behind this lays in the binary-search process used by Pathload, which, depending on the interference, takes wrong decisions at different steps of the avail-bw search, converging then to the same biased values. The results are inaccurate already with very few hosts interfering. Spruce (Fig. 2(b)), reports an avail-bw value which is closer to the *actual* avail-bw (represented by vertical bars at values  $A - (N - 1) \times 240$  kbps, since each Spruce probe injects traffic at an average 240 kbps rate). Spruce estimate clearly far from the *nominal* avail-bw. This means that Spruce underestimate the avail-bw if pure probing traffic is injected in the network, while the estimates are correct if (existing) user traffic is shaped in packet pair probes, i.e., the avail-bw estimation is piggy-backed on actual traffic. We discuss this desirable property later on in the paper. Finally, pathChirp (Fig. 2(c)) seems to deviate less from the nominal avail-bw even when several measurements overlap, but the variance in the results increases rapidly when  $N$  grows. Also, results are very far from the actual avail-bw, as represented by the dotted vertical bars.

Fig. 3 reports the relative error (in percentage), computed against the actual avail-bw, i.e. considering as background traffic the  $N - 1$  other probing flows. Unfortunately, also in this case the results are often far from being precise, with very high variance. Pathload and pathChirp show very inconsistent results, with common errors of  $\pm 100\%$ . Spruce performs better but tends to overestimate the actual avail-bw, up to 50% when  $N = 30$ .

2) *Impact of background traffic*: we repeated these experiments with different nominal avail-bw values by injecting very simple and regular CBR cross-traffic. In general the tools perform poorly and underestimate the nominal avail-bw. For example, Fig. 4 reports the case for  $N = 20$  and values of background traffic equal to 0 Mbps, 2 Mbps and 4 Mbps

from left to right. Results shows that Pathload and pathChirp become almost insensitive to avail-bw changes, and single measurements are very inconsistent, as in Fig. 4(c) where results are almost uniformly distributed for pathChirp. Spruce, instead, coherently provides estimates which are close to the actual avail-bw – about 4.8 Mbps less than the nominal avail-bw.

#### B. Heterogeneous scenario

We analyze now the impact of interference in heterogeneous settings, i.e. when *different techniques* are interacting with each other. We run experiments in which tools coexist two-by-two, and when all the three tools run at the same time. Like before, we repeated the experiments for various values of  $N$  and with different cross-traffic rates. Fig. 5 presents the results in a scenario with no cross-traffic and with all the three tools running. In the experiment we consider  $N = 3, 5, 9$  hosts per different tool, for a total of 9, 15 and 27 senders respectively. In all cases, the nominal avail-bw is underestimated, with more inconsistent results for larger  $N$ . However, the figure shows that the bias is *not the same* for the three tools, which deviate differently from one another for increasing  $N$ . To better understand these results, we studied the case where only two tools coexist simultaneously. Pathload provided similar results both when interfering with Spruce and pathChirp, while Spruce and pathChirp estimates changed significantly in presence of Pathload. Results are not reported here due to lack of space and can be found in [11]. In all cases, however, the variability of the results was very large, ranging in [1,10]Mbps for Pathload, [3.5,9]Mbps for Spruce and [2,10]Mbps for pathChirp. This large variability is reflected in the relative error with respect to the actual avail-bw which reaches  $\pm 100\%$ .

#### C. Impact on the bottleneck buffer

To analyze the *intrusiveness* of the overall measuring process, we computed the queue length at the bottleneck link during the experiments. Fig. 7, shows the queue length distribution using a logarithmic x-axis for  $N = 20$  and in an homogeneous scenario, while Fig. 6 provides a sample time series of the buffer occupation when Pathload (Fig. 6(a)), Spruce (Fig. 6(b)) and pathChirp (Fig. 6(c)) are running. First, notice that there are almost two orders of magnitude difference between the maximum observed queue length: it is never above 15 packets when Spruce is running, 100 packets with pathChirp and almost 1000 packets with Pathload. Additionally, from the traces we can observe that Pathload forces the queue length to large values and for long period of times, which corresponds to hosts injecting bursts of traffic with rates higher than the actual avail-bw. Also, the queue drains slowly because Pathload probing rate tends to consume all the actual avail-bw. Spruce probing process instead is more stationary. This is due to the fact that packet pairs are injected by each host following an exponentially distributed spacing between two successive pairs, so that the resulting process closely resembles a Poisson process. Finally, pathChirp induces a periodic queue increase, due to the deterministic scheme

<sup>3</sup>Pathload provides upper and lower bounds of the estimated avail-bw. For easiness of presentation, we show here the average between the two bounds.

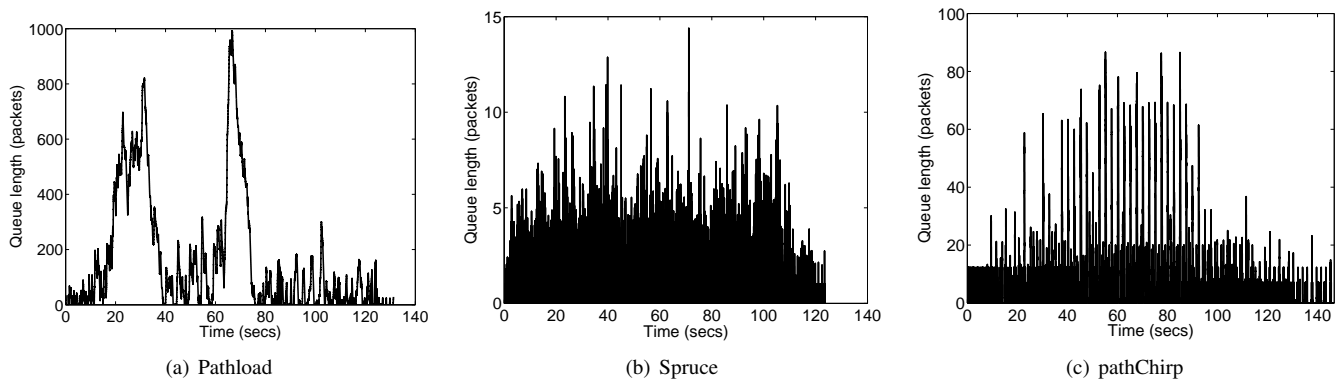


Fig. 6. Queue length under measurement traffic in some sample traces. Note that different  $y$ -scales are used in different plots.

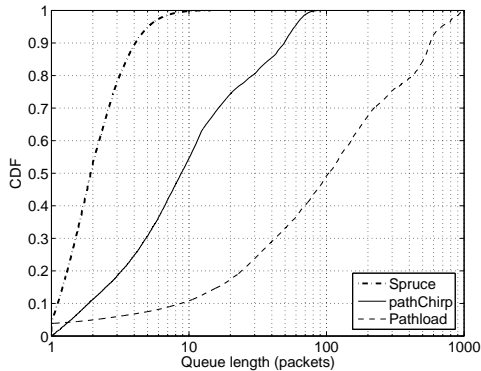


Fig. 7. Distribution of the queue length during the experiments.

of probes which periodically exceed the avail-bw with the impulsive packet “chirps”.

In summary, these experimental results show that mutual interference has a large impact on the accuracy of the estimations and that these measurement techniques introduce significant overhead. Even worse, results are sometimes insensitive to load changes and very unreliable. The intrusiveness is an issue when Probe-Rate-Models are employed, as testified by Pathload which congests the bottleneck buffer for long periods of time.

#### IV. PROBABILITY OF INTERFERENCE

In the previous section we experimentally showed that the measurements may interfere destructively when multiple hosts probe the same path simultaneously. The question we would like to answer now is: *what is the probability of a packet pair, train or chirp being affected by concurrent measurements?* Alternatively, how many coexisting measurements can a path allow for a fixed probability of interference? To answer these questions, we consider a generic shared link and study analytically the probability of having these measurements actually interfering, proving that this probability is high. In the following section we then explain *why* this interference produces large measurement errors, modeling the impact of interference for the three different measurement techniques.

##### A. $N$ hosts continuously monitoring

Consider  $N + 1$  independent entities continuously monitoring the path with probes (either pairs, trains or chirps) of time duration  $\tau$ . The transmission of a probe by a generic host  $i$  is free from interference only if *all* other  $N$  hosts are not transmitting during a period of time  $T = 2\tau$  in which the probe is sent. In a large-scale scenario, the different hosts send their probes independently and we can assume that the arrival process of the different probing streams follows a Poisson process. We can then model the interference probability similarly to an un-slotted Aloha protocol [1]: the so called “vulnerability period” represents the period  $T = 2\tau$  in which the observed probing stream might suffer the interference of other overlapping streams. Since all other  $N$  hosts must be silent for the entire vulnerability period  $T$  for the observed probing stream not to be affected, the probability of “collision” for a stream belonging to the generic host  $i$  is

$$\begin{aligned}
 Pr(\text{interference by host } i) &= \\
 &= 1 - Pr(N \text{ other hosts silent in a period } T) \\
 &= 1 - [Pr(\text{one host silent in a period } T)]^N \\
 &\approx 1 - e^{-N\lambda T} = 1 - e^{-\Lambda} \tag{2}
 \end{aligned}$$

being  $\Lambda = N\lambda 2\tau$ . The approximation comes from the fact that the aggregate probe arrival process tends to a Poisson Process only asymptotically because the probing streams arriving from a same host have finite duration  $T$  and may not overlap – packets arrive no closer than back-to-back. However, the approximation is tight also for moderate  $N$ , as we show later on. The factor  $\lambda\tau$  represents the measuring overhead imposed to the network by a single host. Fig. 8 shows the probability of having a pair affected by mutual interference for different values of the measurement overhead (being 2.4% and 3% of the bottleneck capacity the overhead values during our Spruce and pathChirp experiments respectively). The collision probability is significant even with few tens of interfering hosts.

##### B. $N$ hosts randomly monitoring

Up to now, we have assumed that all the  $N + 1$  hosts in the system are continuously measuring the path avail-bw. We can extend our analysis by considering a more realistic scenario,

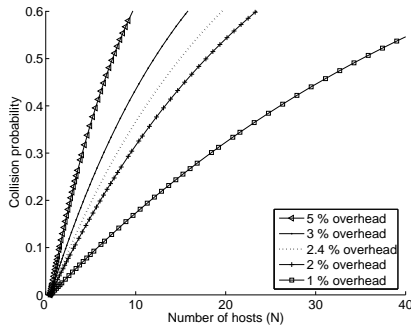


Fig. 8. Probability of having a generic probing stream affected by interference with different overhead values.

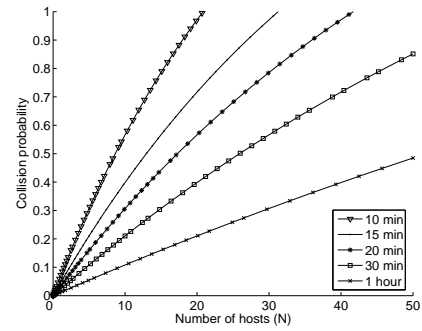


Fig. 9. Probability of a generic probing stream affected by interference for different measurement frequencies.

in which the measuring hosts probe the network just from time to time – perhaps changing their behavior only when a significant variation in the network conditions is detected. Lets suppose, then, that out of the  $N + 1$  total hosts only  $M + 1$  are simultaneously active; also, let the measurement duration last  $D$  seconds on average. For example, in Spruce  $D$  is the time needed to send 100 packet pairs and is about  $D = 10$  seconds. For all tools, time between two successive measurements is chosen according to an exponential distribution. Let  $\lambda_0$  be the corresponding measurement rate. Then the probability of having a collision for a host  $i$  is

$$\begin{aligned} Pr(\text{collision by host } i) &= \\ &= \sum_{M=0}^N Pr(\text{collision} | M \text{ active}) Pr(M \text{ active}) \\ &\approx \sum_{M=0}^N (1 - e^{-M\lambda T}) \frac{(\Lambda_0)^M}{M!} e^{-\Lambda_0} \end{aligned} \quad (3)$$

where  $\Lambda_0 = N\lambda_0 D$ . The terms in the summation decay quickly as  $N$  becomes large and, thus, we can approximate the formula letting the summation run up to infinity. We obtain a closed form of the collision probability:

$$\begin{aligned} Pr(\text{collision by host } i) &\approx \\ &\approx \sum_{M=0}^{\infty} (1 - e^{-M\lambda T}) \frac{(\Lambda_0)^M}{M!} e^{-\Lambda_0} \\ &= 1 - e^{-\Lambda_0(1 - e^{-\lambda T})} \end{aligned} \quad (4)$$

In Eq. (4), it can be shown that the single measurement overhead  $\lambda\tau$  has very little impact on the collision probability – at least within the values of interest, say, less than 10% of the bottleneck capacity. What greatly impacts this probability, instead, is the frequency  $\lambda_0$  of the measurements and their duration  $D$ . For example, suppose that the measurement duration is  $D = 10$ s and that the “per host” measurement overhead is  $\lambda\tau = 1\%$  (as we are going to see, it is generally much higher). Then, Fig. 9 shows the collision probability for different measurement frequencies  $\lambda_0$ , from one measurement every 10 minutes up to one every hour. Already with 10 measuring hosts, the interference between measurements affects up to 50% of the probing streams.

### C. Predicting monitoring interference

The probability of interference we have analyzed up to now is not related to a particular measurement tool, but holds true for *any* technique, as soon as the probing is started according to a Poisson process. In particular, the generality of Eq. (2) and Eq. (4) is fundamental because it provides a working framework useful for the dimensioning of a large-scale avail-bw measurement system, *whatever the chosen technique is*. Indeed, the inversion of Eq. (4) allows to compute the maximum measurement frequency a certain system can afford or, equivalently, the maximum number of users allowed for a given probing rate. For example, suppose that a new P2P system optimizes the distribution of the data using the knowledge of the avail-bw [9], [24]. To make the measurements reliable enough, the P2P application must be designed in such a way that the probability of having measurements interfering with each other is bounded to, say  $p < 10\%$ . Then, when using a certain technique of given measurement duration  $D$  and overhead  $\lambda\tau$  (usually known), Eq. (4) provides an upper bound on the maximum measurement frequency  $\lambda_0$  allowed to each user:

$$\lambda_0 < \frac{\ln(1 - p)}{ND(e^{-\lambda T} - 1)}. \quad (5)$$

This value must be then used by the P2P application to limit the number of per-user measurements and meet the desired collision probability  $p$ . Finally, note that even in the presence of multiple paths (and multiple bottlenecks) it is not necessary to repeat the above analysis on all the links of the network – this would require knowledge of the underlying topology. Instead, it is sufficient to analyze the links with the highest number of end-to-end paths traversing them, i.e., following a worst case approach. Then, the system can be dimensioned on the worst case (maximum link betweenness) or on the average betweenness – more practical but less conservative.

Given this analysis of the interference probability, a natural question worth answering now is what would be the *expected bias* if two or more measurement streams overlap. Clearly, the impact of interference on the measurements depends on the probing strategies and inference mechanisms used by the different tools and, differently from the collision probability, the resulting bias is then an intrinsic characteristic of the technique employed. In the next section we analyze this bias

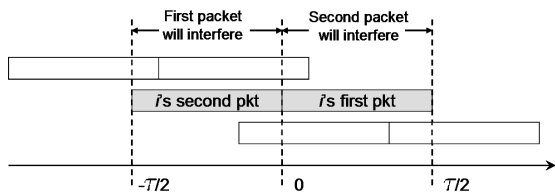


Fig. 10. Interference between packet pairs.

for each tool separately, both through analytical models and simulations.

## V. INTERFERENCE AND OVERHEAD IN CURRENT AVAILABLE BANDWIDTH ESTIMATION TOOLS

We now dig into the properties of the different probing techniques. Keeping in mind the internal functioning of the three tools, as described in Sec. II-B, we analytically model the impact of multiple overlapping measurements on the tools accuracy. We also discuss the overhead imposed to the network as well as the overall intrusiveness of the measurement processes. The validation of our models has been done through *ns-2* simulations. All simulations lasted at least 100s and, on average, over 50,000 avail-bw samples were obtained for each test.

We first study Spruce because the probing consists of only two packets. We exactly predict the amount of measurement bias both in the case of  $N$  hosts which monitor continuously the avail-bw and also in a more realistic ON/OFF scenario, where hosts become active just from time to time to update the measurements. For Pathload and pathChirp, since several packets are sent to probe the path, the interference depends on the arrival instant between the interferers. Thus, the final bias depends jointly on the *time offset* of all the measurements interfering. Since a combinatorial study of all the possible interactions is impossible, we limit the analysis to the case of two hosts interfering, which allows us to explain the root causes of the measurement errors seen in the testbed experiments and, indeed, provides a lower bound of the bias in more complex scenarios. The analysis with more hosts interfering can be extended with numerical simulations.

### A. Spruce – Packet Pairs

1) *Interference analysis*: lets first consider the case of two measuring hosts operating at the same time. Recall that Spruce sends two packets separated in time by  $\Delta_{in} = S/C$ , being  $C$  the capacity of the bottleneck and  $S$  the size of the probes, so that the packet pair arrives back-to-back on the bottleneck link. If two packet pairs overlap in time, one of the probes of the first pair might be transmitted in between the other pair, as shown in Fig. 10. When this happens, since in Spruce  $S$  is constant (1500 bytes by default), the initial probing gap  $\Delta_{in}$  is inflated to  $2 \times \Delta_{in}$ , plus the effect of cross-traffic. Note that already with two overlapping pairs, an output gap of  $\Delta_{out} = 2 \times \Delta_{in}$  leads in Eq. (1) to an avail-bw estimate of zero, i.e., a relative error of 100%. When more packet-pairs interfere (or when the bottleneck link is overloaded),  $\Delta_{out}$

increases further, leading in Eq. (1) to a negative estimate of the avail-bw. Unfortunately, negative values cannot be blindly ignored because they represent transient periods of time in which the bottleneck queue fills up, which happens periodically in normal operating conditions. Therefore, it is not possible to detect and filter affected pairs by discarding negative avail-bw estimates.

Recall that Spruce samples the avail-bw using packet pairs spaced between each other according to an exponentially distributed pausing time, and then it averages the results. Because of the PASTA property, each pair sees the same probability to encounter  $n$  interfering *pairs* and, therefore, the overall (average) bias on the measurement is equal to the average error seen by a *single* packet pair probing the path. Therefore, we can compute the average bias in Spruce measurements simply by analyzing the error of a single pair. From Eq. (1), it is easy to see that the avail-bw  $\tilde{A}$  measured by a packet pair affected by  $n$  interfering pairs has absolute error  $\epsilon = |A - \tilde{A}| = nC$  independently of the network conditions. The relative error, instead, is  $\eta = |A - \tilde{A}|/|A| = n/(1 - \Delta_X/\Delta_{in})$ , being  $\Delta_X$  the dispersion due to cross-traffic, and becomes large when the link is highly loaded. In any case, both  $\epsilon$  and  $\eta$  are directly proportional to the number  $n$  of interfering pairs. To predict the average bias seen by one pair sent by a generic measuring host  $i$ , we must then compute the expected number  $\mathbb{E}[n]$  of interfering pairs. Let  $p_n$  be the probability that exactly  $n$  pairs arrive during the “vulnerability” period. In the particular case of Spruce, the measurement is affected only if the time offset *off* between the interfering pairs and the observed probing pair is less than  $|off| < \tau/2$ , i.e. the vulnerability period is long  $\tau$  instead of  $2\tau$  (see Fig. 10). With the notations previously used and leveraging on the fact that the sampling processes are independent and Poisson, we have

$$\mathbb{E}[n] = \sum_{n=0}^N np_n \approx e^{-\Lambda} \sum_{n=0}^{\infty} n \frac{\Lambda^n}{n!} = NR/C \quad (6)$$

being  $\Lambda = N\lambda\tau = NR/C$ . As in section IV, the approximation comes from the fact that the aggregate probe arrival process tends to a Poisson Process only asymptotically because each the packets have finite length. However, the approximation is tight also for moderate  $N$ , as we show shortly. Eq. (6) says that the bias increases linearly with the probing rate  $R$  and with the number  $N$  of active hosts. The probing rate  $R$  is generally fixed and is set by Spruce to  $R = \min(0.05C, 240\text{kpbs})$ . Note that the testbed results provided in Sec. III confirm this analysis, as the bias shown by Spruce measurements indeed corresponds to  $\epsilon = N \times 240\text{kpbs}$ .

Leveraging on the fact that the aggregate probing process is approximately Poisson, and thus the number of interfering probes is approximately Poisson distributed, the analysis can be easily extended so to characterize the *distribution* of Spruce outcomes. Fig. 11 reports a comparison between the probability density function (pdf) of Spruce avail-bw estimates for testbed results and model predictions; the homogeneous scenario with  $N = 20$  hosts measuring simultaneously is considered. The model follows closely the experimental results. Additionally, Tab. III reports the mean and the standard

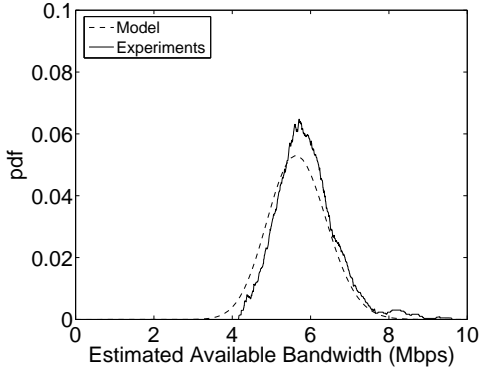
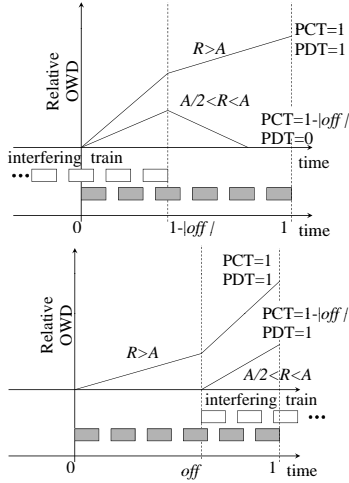

 Fig. 11. Distribution of Spruce results with  $N = 20$  hosts measuring.


Fig. 12. Pathload: OWD when two trains overlap.

deviation of the results in all the experimental scenarios from which it is clear that even for moderate  $N$  the model effectively tackles the experiments, validating it.

		# of hosts $N$	1	3	5	10	20	30
Mean	Experiments		9.73	9.31	9.00	7.94	5.72	3.81
	Model		9.76	9.28	8.80	7.60	5.20	2.80
STD	Experiments		0.01	0.26	0.31	0.58	0.76	0.98
	Model		0.15	0.27	0.35	0.49	0.69	0.85

TABLE III

MEAN AND STANDARD DEVIATION (STD) OF SPRUCE ESTIMATES IN MBPS. THE MODEL FOLLOWS CLOSELY THE EXPERIMENTAL RESULTS.

Finally, the model can be extended to predict the exact amount of bias in an ON/OFF scenario with  $M$  out of  $N$  interfering hosts, similarly to Eq. (6). The generalization is trivial. For what concerns the average error we have:

$$\mathbb{E}[n] = \sum_{M=0}^N \sum_{n=0}^M np_n \frac{(\Lambda_0)^M}{M!} e^{-\Lambda_0}. \quad (7)$$

### B. Pathload – Packet Trains

As discussed previously, in Pathload and pathChirp, the impact of mutual interference depends on the *time offset*

between the probing streams. Since a complete combinatorial study is infeasible, we limit here the analysis to the case of two coexisting measurements which is enough to understand the measurement errors seen in our testbed and, indeed, constitutes a lower bound of the bias in scenarios with more overlapping measurements.

1) *Interference analysis*: Pathload sends several trains of packets at different rates, analyzing the OWD trends and detecting when the avail-bw is exceeded or not looking at the PCT and PDT metrics. Let us first consider the case of two perfectly synchronized hosts, i.e. when the probing trains are sent at same rate  $R$  and overlap *completely* in time. The measurements compete in occupying the avail-bw  $A$ , each one obtaining only half it, and Pathload binary-search stops when  $R \approx A/2$ , resulting in a 50% error<sup>4</sup>.

Things are more complex in the case of two trains which *partially* overlap. Let time be normalized to the train duration, i.e. the trains last  $\tau = 1$ , and let *off* be the normalized offset of the interfering train arrival time from the beginning of the other train. Consider two trains that are sent at the same rate  $R$ . Fig. 12 shows the OWD evolution when  $off < 0$  and  $off > 0$  on the left and right plots respectively. The associated values of the PCT and PDT metrics used to detect the OWD *trend* are also reported. Obviously, the case  $R < A/2$  is not reported because, in this case the rate of the two trains is too low to cause self-induced congestion.

The decision if  $R$  exceeds  $A$  depends on the evolution of the OWD of packets, and how the PCT and PDT metrics are influenced. Let us analyze the PCT first, which simply counts the fraction of time that the OWD increases. A wrong decision is taken if the overlap between the two trains causes the OWDs to increase for at least  $PCTth$  packets. Therefore, the error is a step function of the offset: if  $1 - |off| > PCTth$ , then the PCT test is positive and, just like in the case of two perfectly overlapping trains, the error is 50%. Instead, if  $1 - |off| < PCTth$ , no interference is perceived, and the error is zero. Let us then focus on the PDT, which compares the OWD of the first and the last packets of the train only. If there is a sudden increase in the OWDs of the last packets, the PDT has a high value and  $PDT > PDTth$ . If, on the contrary, the interference involves the beginning of the train, the increase in the OWDs of the first packet is possibly recovered by later packets, as shown in left plot of Fig. 12 when  $A/2 < R < A$ . In particular, the PDT value depends on the train rate  $R$ : it can be easily shown that, for the case of two trains overlapping of the same rate  $R$ , the PDT metric detects an increasing trend only if

$$R \geq A \frac{PDTth(3off - 2) - 1}{2PDTth(2off - 1) - (off + 1)}. \quad (8)$$

Recall that the presence of the interfering train causes a wrong decision only if *both* PCT and PDT metrics exceed the corresponding thresholds. Thus, putting all pieces together, three scenarios are possible, as shown by Pathload error function in Fig. 13. First, in the region where  $off > 0$ , the last packets are always affected, thus  $PDT > PDTth$ . The

<sup>4</sup>Note that, for the sake of simplicity, we assume that the offset remains constant during the whole binary search process.

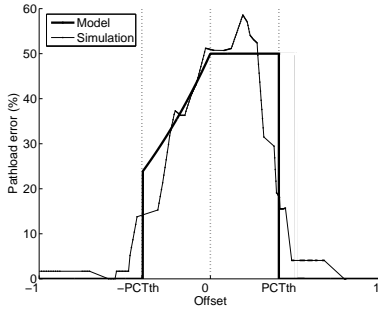


Fig. 13. Pathload error as a function of the offset for two trains of same rate.

resulting error is then determined by the PCT metric and the bias is the step function described above for the PCT. Second, if  $-PCTth \leq off \leq 0$ , then  $PCT > PCTth$ , and the wrong decision triggers according to the PDT metric, following inequality (8). Finally, in the region  $-1 < off < -PCTth$ , the PCT is not affected and thus the error is zero.

To validate our analysis, we ran *ns-2* simulations using the implementation of Pathload provided in [29]. We consider two hosts probing the same path with variable time offset between the beginning of the measurements. All simulation lasted at least 100s. Fig. 13 compares the results obtained against our analytical model, showing that the analysis is consistent.

Complicating the analysis further, it is possible to study the bias for hosts interfering at various stages of the binary search, and for multiple overlapping trains. This is best done via numerical simulations because analytically predicting the final bias suffered after all iterations of the binary search with multiple trains interfering becomes quickly intractable. Indeed, depending on the interaction between the measurements, an erroneous decisions can occur at various stages of the binary-search process: an error in the final stages has smaller impact on the result because the upper and lower bounds are already close to each other; vice-versa, a wrong decision at the beginning of the process leads to larger measurement errors. This can produce severe errors in the measurements (such as the one observed in our experiments), because the binary search algorithm used to seek for the correct rate  $R$  is not able to recover from wrong decisions taken at any previous step. In summary, our analysis confirms that already two hosts can destructively interfere, heavily biasing the results, as reported by the error function of Fig. 13. Finally, since the effect of interference is always to underestimate the avail-bw, Fig. 13 constitutes a lower bound on the estimation error when more hosts are interfering simultaneously.

### C. PathChirp – Packet Chirps

1) *Interference analysis*: pathChirp uses a sophisticated algorithm to compute the avail-bw. It takes into account both the last rate  $R_l$  before the last OWD excursion starts, and also the evolution of the queuing delay across the rest of the chirp before this point. Fig. 14 shows the OWD time series of a chirp in presence of a second chirp interfering. Three scenarios are possible: (i) the second chirp creates a

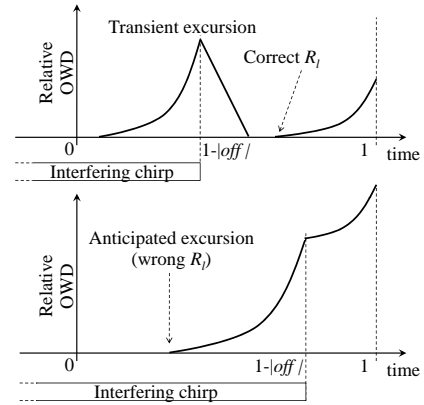


Fig. 14. Evolution of the queuing delay when two chirps overlap with different offsets.

transient excursion in the beginning of the first chirp, which is re-absorbed before  $R_l$  is reached (top of Fig. 14); (ii) the excursion caused by the interfering chirp adds up with the last excursion, potentially inflating it and *anticipating* the point where  $R_l$  is detected (bottom of Fig. 14); or (iii) the second chirp arrives *after*  $R_l$  is already reached, sharpening the last excursion but without inflating it. In the first case, when the interference generates a transient excursion, the avail-bw estimate is affected proportionally to the duration and intensity of the excursion, but the detection of the  $R_l$  rate, the most critical part, remains unchanged. Instead, if the two excursions combine together anticipating the point where  $R > A$  (second case), the detection of the limit rate  $R_l$  is affected. As Fig. 14 shows, the limit rate  $R_l$  can be significantly modified and this leads to an underestimation of  $A$  which can be large, as we explain shortly. Finally, if the interfering chirp arrives after  $R_l$  is reached, there is no impact on the estimation at all.

More quantitatively, Fig. 15 shows the resulting error as a function of the offset according to the analytical model we now present. Again, we validated the model against *ns-2* simulations, with two hosts measuring the same path, and we varied the time offset between the measurements. Fig. 15 shows that the analytical results are consistent with the simulations. Let  $\tau = 1$  be the normalized duration of the chirps and  $off$  the (normalized) offset between the beginning of the two chirps. Recall that a chirp probes a variety of rate values, from a lower rate  $L$  (the first two packets of the chirp) up to an upper rate  $U = \gamma^{(k-2)} \times L$ . The instantaneous chirp rate can be approximated by a continuous function  $R(t) = L\gamma^{(k-2)t} = L\Gamma^t$ , being  $\Gamma = \gamma^{(k-2)}$ . It is easy to see that when the interfering chirp overlaps with the final excursion, the avail-bw is reached in advance and the  $R_l$  rate is detected at  $\tilde{R}_l = A/(1 + \Gamma^{-|off|})$ . Since in absence of transient excursions pathChirp avail-bw estimate is exactly  $\tilde{A} = R_l$ , the measurement error is

$$\eta = 1 - \frac{\tilde{R}_l}{A} = \frac{\Gamma^{-|off|}}{1 + \Gamma^{-|off|}}. \quad (9)$$

This is represented in the central part of Fig. 15 labeled “Z3”. Note that already with two interfering chirps, the error reaches above 80%.

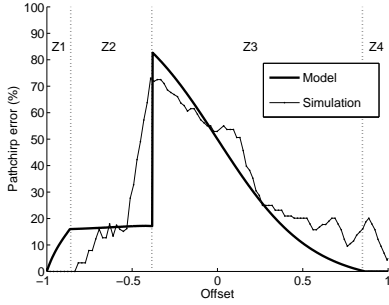


Fig. 15. PathChirp measurement error as a function of the offset between the two chirps.

Consider now the case of an interfering chirp generating a transient excursion but without affecting the estimation of  $R_l$ , i.e.  $\tilde{R}_l = A$ . As explained in Sec. II-B3, in this scenario the estimated avail-bw  $\hat{A}$  is the weighted average between the received rate  $R_i$  during the transient excursion, and  $R_l = A$  for the rest of the train duration.  $R_i$  is the fair share between the two competing trains and is dominated by the interfering chirp which rate has already exceeded  $A$ . The error becomes

$$\eta \approx 1 - \frac{\ln(A/L)}{\ln \Gamma} - \frac{U - A}{A} \frac{\Gamma^{|off|}}{(1 - \Gamma^{-|off|}) \ln \Gamma}. \quad (10)$$

The corresponding range is labeled in Fig. 15 as “Z2”. When  $off < \log_{\Gamma}(A/L) - 1$  (label “Z1”), the formulation is similar but the effect is reduced by the fact that the interfering chirp is progressively disappearing. The derivation of the error in this particular case and the exact (unapproximated) computation of Eq. (10) can be found in the Appendix B. Finally, at the extreme right of Fig. 15, when  $off > \log_{\Gamma}(A/L)$  (label “Z4”), the interfering chirp arrives after  $R_l = A$  is reached and the error is zero.

An additional complication in predicting the accuracy of pathChirp comes from the fact that the results are averaged among several tests. The de-synchronization among interfering hosts causes variable overlapping periods, thus the error must be computed considering the (possibly different) time offsets of all the interfering chirps on which the average is being taken. As reported in Fig. 15, pathChirp errors are largely dependent on the offset between interfering chirps, becoming very large in some cases. As a result, the estimates may exhibit an high variance as emerged from our testbed results. Again, since the interference only causes underestimations, the analysis reported here constitutes a lower bound of the measurement bias in more complex scenarios and may be easily extended through numerical simulations.

## VI. DISCUSSION AND CONCLUSIONS

Both our analysis and testbed results have shown that current measurement paradigms can not be used off-the-shelf in large-scale deployments. All methods underestimate the avail-bw significantly: additional overhead, mutual interference and intrusiveness impair the both estimations themselves and overall performance of the network.

### A. Overhead

The overhead problem can be eliminated if the measurements are piggy-backed on ongoing data flows. That is, instead of injecting additional probes, the existing “data” traffic is shaped and sent in packet pairs, trains or chirps. This removes the overhead problem. The drawback is that (i) there must enough data to be sent on the desired path, and (ii) the data transmission can be delayed to form packet pairs, trains or chirps. While Spruce and PathChirp require relatively few probing phases, pathLoad requires to shape the traffic in relatively long trains of various rates. This might add too much delay, especially for real time applications such as video streaming.

### B. Mutual interference

However, even when measurements are piggy-backed, the problem of mutual interference affecting the accuracy of the measurements is unchanged. In particular, the analysis undertaken in the previous section, confirmed by our testbed results, shows that the sophisticated technique used in pathChirp provides variable results and might not be robust enough to tolerate multiple overlapping chirps. Pathload can not be used because the binary-search algorithm leads to unpredictable errors in presence of interference. Spruce seems to maintain a certain coherence in the results: if the probes are considered as actual traffic, then the estimated avail-bw is fairly accurate, i.e., it reflects the actual avail-bw left free by the total probing process. This suggests that if the probing is performed as “in-band” process, it can be successfully exploited. In large-scale scenarios, shaping actual data packets and sending them in pairs should be an easy task, whereas injecting additional pure probing packets causes biased results.

### C. Intrusiveness

Regarding the measurement intrusiveness, Pathload has a deep impact on the queuing delay and keeps the buffers full for large amounts of time. The periodic probing of pathChirp also causes some negative effects, especially if several measurements are in-phase with each other. The impact on the network would probably be lower if the measurements are spread and randomized in time. However, our analysis also shows that the exponential rate increase is probably too bursty and a more gradual shape could be needed in large-scale deployments. Finally, although Spruce sends packets at full capacity rate, the overall intrusiveness is small as only two packets are sent at-a-time, with almost no impact on queues.

### D. Possible Solutions

In general, both our analysis and testbed results suggest that avail-bw measurements in large-scale systems are *possible* as long as the measurement traffic is (i) piggy-backed on existing data flows, (ii) the traffic shape is “smooth”, with minimum impact on the network queues, and (iii) the inference algorithm is direct or needs less iterations as possible. To this extent, PGM, as the one implemented in Spruce, seem to be the preferred candidate. However, the possible inaccuracies

pointed out in [23] remain unchanged. Using trains longer than two packets, as suggested in [25], could increase the accuracy of PGM tools but might have a higher impact on the network, as several packets would be sent at full line rate. In this sense, there is probably some research space for finding an optimal tradeoff. Note that in presence of different applications exploiting different measurement techniques, the picture gets more complicated and the resulting interference is similar or worse than considering a homogeneous set. A necessary step is to have a common methodology that is elected to standard.

Finally, up to now the research in the field of avail-bw estimation has only been directed towards the development of new techniques, with little interest in integrating existing probing strategies. However, it would be interesting to develop new hybrid tools able to empower the strengths of the different PRM, PGM and PDM paradigms. For example, OWD trends and packet dispersions could be exploited together, and the first of the sent probes can also provide an information on the queuing delay probability. Then, the measurement can be done by using Pathload OWD analysis, the dispersion technique of [37] (a generalized form of Spruce) and the idea implemented in Forecaster [26], thus exploiting the potentiality of the new PDM model. Another interesting idea worth to be explored would be to apply network tomography concepts to the avail-bw estimation and try to measure several Internet paths with the synergy of many cooperating hosts. This would reduce the overall amount of measurements on the core paths of the network (the most critical ones, with higher betweenness), making the measurement system much more scalable. Finally, the cooperation of the ISPs, as advocated in [36], could pave the way to new, highly optimized, applications truly aware of the network avail-bw conditions.

## VII. RELATED WORK

Given the large amount of papers related to available bandwidth measurement, here we restrict our overview to those papers that present comparison of tools and techniques. To the best of our knowledge, we are the first to focus on the mutual interference suffered in large scale deployment of avail-bw estimation techniques [12]. Instead, a variety of related works focus on the performance comparison of existing avail-bw tools, tested in different network topologies and traffic conditions. In [30] several tools have been tested in high-speed links, while in [8] the focus was on lower speed links and on the impact of the router queuing disciplines. The authors in [22] compare the performance of avail-bw tools in different access technology, such as Wi-Fi or Cable. An interesting calibrating framework was proposed in [31] to accurately tune the different tool parameters. In [16] and [3], the different tools are tested separately in various network scenarios, analyzing their performance and their sensitiveness to packet loss, cross-traffic rate, packet size, etc. Finally, the work in [29] made a step towards a reliable tool comparison, by decoupling the implementation technology (and its issues) from the probing technique performance. All these studies analyze the performance of each tool separately, with only one technique probing one single path at-a-time, which is clearly unrealistic in large-scale scenarios.

## VIII. SUMMARY

In this paper we analyzed quantitatively the impact of interference when current avail-bw tools are deployed in large-scale settings. We showed that existing measurement tools are affected by this issue and we quantified (i) the impact of mutual interference, (ii) the total overhead imposed to the network and (iii) the intrusiveness of the measurement process in a large-scale scenario. Our findings offer some simple guidelines for the dimensioning of large-scale measurement systems and effectively predict the impact of concurrent measurements on existing estimation techniques.

In the future, the development of new hybrid tools, able to empower the strengths of the different measurement paradigms, and the application of network tomography concepts to the avail-bw estimation (perhaps with the cooperation of the ISPs) will open the way to new network-aware applications which make a intelligent use of the avail-bw. The framework and models introduced in this article are a fundamental building block for the development of such smart applications and will be useful in preventing measurement errors and congestion problems, potentially arising in Internet-wide deployments of avail-bw measurements.

## REFERENCES

- [1] N. Abramson. The aloha system: another alternative for computer communications. *AFIPS (Fall): Proceedings of the November 1970, fall joint computer conference*, 1970.
- [2] A. Akella, S. Seshan, and A. Shaikh. An empirical evaluation of wide-area internet bottlenecks. *SIGMETRICS*, 2003.
- [3] L. Angrisani, S. D'Antonio, M. Esposito, and M. Vadursi. Techniques for available bandwidth measurement in IP networks: A performance comparison. *Computer Networks*, Vol.50, n.3, pp.332-349, 2006.
- [4] L. Breslau, E. W. Knightly, S. Shenker, I. Stoica, and H. Zhang. Endpoint admission control: architectural issues and performance. *SIGCOMM*, 2000.
- [5] A. Cabellos-Aparicio, F. Garcia, and J. Domingo-Pascual. A novel available bandwidth estimation and tracking algorithm. *NOMS*, Salvador da Bahia, BR, 2008.
- [6] Y. Cheng, V. Ravindran, A. Leon-Garcia, and H.-H. Chen. New exploration of packet-pair probing for available bandwidth estimation and traffic characterization. *ICC*, Glasgow, UK, 2007.
- [7] L. Kleinrock. *Queueing Systems*. J. Wiley, NY, 1975.
- [8] F. Coccetti and R. Percacci. Bandwidth measurements and router queues. Tech. report, Istituto Nazionale Di Fisica Nucleare, Trieste, IT, 2002.
- [9] A. P. Couto da Silva, E. Leonardi, M. Mellia, M. Meo. A Bandwidth-Aware Scheduling Strategy for P2P-TV Systems. *P2P*, Aachen, DE, 2008.
- [10] D. Croce, T. En Najjary, G. Urvoy Keller, and E. W. Biersack. Fast Available Bandwidth sampling for ADSL links: rethinking the estimation for larger-scale measurements. *PAM*, Seoul, SK, 2009.
- [11] D. Croce, E. Leonardi, M. Mellia. Large-Scale Available Bandwidth Measurements: Interference in Current Techniques. <http://www.tlc-networks.polito.it/croce/techrep190609.pdf>.
- [12] D. Croce, E. Leonardi, M. Mellia. The Quest for Bandwidth Estimation Techniques for Large-Scale Distributed Systems. *HOTMETRICS*, Seattle, WA, 2009.
- [13] C. Dovrolis and M. Jain. End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput. *SIGCOMM*, Pittsburgh, PA, 2002.
- [14] S. Ekelin, M. Nilsson, E. Hartikainen, A. Johnsson, J.-E. Mangs, B. Melander, and M. Bjorkman. Real-time measurement of end-to-end available bandwidth using kalman filtering. *NOMS*, Vancouver, CA, 2006.
- [15] E. Goldoni, G. Rossi, and A. Torelli. ASSOLO, a new method for available bandwidth estimation. *ICIMP 2009*, Venice, IT, 2009.
- [16] C. Guerrero, and M. A. Labrador. On the applicability of available bandwidth estimation techniques and tools. *Computer Communications*, Vol.33, n.1, pp.11-22, 2010.

- [17] K. Hanna, N. Natarajan, and B. Levine. Evaluation of a novel two-step server selection metric. *ICNP*, Riverside, CA, 2001.
- [18] N. Hu and P. Steenkiste. Evaluation and characterization of available bandwidth probing techniques. *IEEE JSAC*, Vol. 21, N.6, pp.879–894, 2003.
- [19] M. Jain and C. Dovrolis. Ten fallacies and pitfalls on end-to-end available bandwidth estimation. *IMC*, 2004.
- [20] M. Jain and C. Dovrolis. Path selection using available bandwidth estimation in overlay-based video streaming. *Computer Networks*, Vol. 52, n.12, pp.2411–2418, 2008.
- [21] V. Konda and J. Kaur. RAPID: Shrinking the Congestion Control Time Scale. *INFOCOM*, Rio de Janeiro, BR, 2009.
- [22] K. Lakshminarayanan, V. N. Padmanabhan, and J. Padhye. Bandwidth estimation in broadband access networks. *IMC*, 2004.
- [23] L. Lao, C. Dovrolis, and M. Y. Sanadidi. The probe gap model can underestimate the available bandwidth of multihop paths. *Computer Communication Review*, Vol.36, n.5, pp.29–34, 2006.
- [24] N. Laoutaris, D. Carra, and P. Michiardi. Uplink allocation beyond choke/unchoke. *CoNEXT*, Madrid, ES, 2008.
- [25] X. Liu, K. Ravindran, and D. Loguinov. A stochastic foundation of available bandwidth estimation: Multi-hop analysis. *IEEE/ACM Transactions on Networking*, Vol.16, n.1, pp.130–143, 2008.
- [26] M. Neginhal, K. Harfoush, and H. Perros. Measuring bandwidth signatures of network paths. *NETWORKING*, Atlanta, GA, 2007.
- [27] P. Papageorge, J. McCann, and M. Hicks. Passive Aggressive Measurement with MGRP. *SIGCOMM*, Barcelona, ES, 2009.
- [28] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell. pathchirp: Efficient available bandwidth estimation for network paths. *PAM*, La Jolla, CA, 2003.
- [29] A. Shriram and J. Kaur. Empirical evaluation of techniques for measuring available bandwidth. *INFOCOM*, Anchorage, AK, 2007.
- [30] A. Shriram, M. Murray, Y. Hyun, N. Brownlee, A. Broido, M. Fomenkov, and K. C. Claffy. Comparison of public end-to-end bandwidth estimation tools on high-speed links. *PAM*, Boston, MA, 2005.
- [31] J. Sommers, P. Barford, and W. Willinger. Laboratory-based calibration of available bandwidth estimation tools. *Microprocessors and Microsystems*, Vol.31, n.4, pp.222 – 235, 2007.
- [32] J. Strauss, D. Katabi, and F. Kaashoek. A measurement study of available bandwidth estimation tools. *IMC*, 2003.
- [33] T. Tsugawa, C. L. T. Man, G. Hasegawa, and M. Murata. Inline network measurements: Implementation difficulties and their solutions. *IEEE/IFIP E2EMON*, Munich, DE, 2007.
- [34] D. Van Aken. Encapsulation overhead(s) in ADSL access networks, Dec. 2000. [www.thomsontelecompartner.com/getfile.php?id=525](http://www.thomsontelecompartner.com/getfile.php?id=525)
- [35] Q. Wang and L. Cheng. FEAT: Improving accuracy in end-to-end available bandwidth measurement. *Globecom*, San Francisco, CA, 2006.
- [36] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. Liu, and A. Silberschatz. P4P: Provider portal for applications. *SIGCOMM*, Seattle, WA, 2008.
- [37] D. Xu and D. Qian. A bandwidth adaptive method for estimating end-to-end available bandwidth. *ICCS*, Guangzhou, CN, 2008.

## APPENDIX A

### INTRUSIVENESS AND OVERHEAD

#### A. Spruce

It is a common belief that PGM-based tools are relatively non intrusive. The reason is that they are capable of estimating the avail-bw without having to significantly overload the bottleneck buffer. While it is true that the measurements consists of only a packet pair, however, these are sent at a rate equal to the capacity of the bottleneck link which is potentially much higher than the avail-bw. In other words, if PGM appears not intrusive it is just because the measurement is very limited in time. Consider, for example, the idea suggested in [25] of using longer *packet trains* instead of packet pairs. The authors prove that long trains reduce the “elastic bias” and improve Spruce estimation performance. In this case, however, a burst of several back-to-back packets would be sent at the bottleneck link capacity, and the intrusiveness of the measurement would be much higher. The issue is amplified when several hosts

simultaneously measure the same path, or when buffers are small, as several overlapping pairs could easily fill-up the bottleneck buffer, eventually causing losses to on-going traffic.

Finally, let us consider the overall measurement overhead. Spruce sets the rate of the probes to  $R = \min(0.05C, 240\text{kbps})$ . While this overhead seems acceptable for a single running instance, it becomes intolerable already with a handful of measuring hosts (4 hosts alone would consume 20% of the bottleneck capacity), with the double risk of (i) quickly congesting the bottleneck link and (ii) obtaining useless avail-bw estimates, because of the high probability of pairs interfering with each other. Note that reducing the probing rate  $R$  reduces the total overhead, the intrusiveness, and the probability of interference. However, this comes at the cost of waiting proportionally longer times to get the final estimate. Therefore, there is a tradeoff to be considered between measurement duration and overhead.

#### B. Pathload

In [29], it has been shown that Pathload can increase the response times of on-going TCP connections. This is a direct consequence of the measurement principle of PRM tools, the self-induced congestion: when the probing rate  $R$  is higher than the avail-bw, the buffer fills up and all following packets are delayed. When multiple Pathload instances probe the path simultaneously, the situation worsens because more trains with  $R > A$  might be sent at the same time, eventually causing buffer overflows and severely impacting regular traffic. In our experiments, this appeared clearly in Fig. 6(a), where the buffer occupation showed high values. However, trains sent at rate  $R < A$  have no impact on the network. Thus, an idea for dramatically reducing the intrusiveness of Pathload measurements would be to change the binary search algorithm in a more conservative “slow start” TCP-like approach, in which the probing rate  $R$  is progressively increased and exceeds the avail-bw only at the last few iterations. The drawback is the eventual longer time to estimate  $A$ .

Regarding the measurement overhead, PRM tools probe the path with different rates until they converge to  $R \approx A$ . This means that Pathload measurements tend to consume *all* the avail-bw, which is one of the main critics moved to PRM. We shall not forget, however, that the probing is not continuous. In particular, when the bandwidth is low, the pausing time between two consecutive trains is larger and Pathload sets the overhead to be approximately 10% of the avail-bw. Nevertheless, when more measurements coexist, the residual avail-bw may be consumed by other concurrent runs, eventually consuming all the residual bandwidth.

However, compared to PGM, the overhead introduced by PRM is proportional to  $A$  instead of  $C$ . This has two important advantages: first, the additional traffic is guaranteed to consume “only” the unused bandwidth; second and most importantly, *the overhead automatically scales with the avail-bw  $A$  and is accordingly reduced when  $A$  is low.*

#### C. pathChirp

The key in pathChirp algorithm is to detect the rate  $R_l$  when the train exceeds the avail-bw  $A$ . The chirp upper rate

$U = L\gamma^{k-2}$  must be set higher than the path avail-bw. Therefore, even when only one host is measuring, part of the chirp consumes more than what the path can afford and this creates some short-term congestion. While buffers could easily absorb the impact of one chirp (by default the chirp length is  $k = 30$  packets), several overlapping chirps can create sudden increase of load which can quickly fill the bottleneck buffer, as shown in our testbed results (see Fig. 6(c)). Indeed, bursty traffic arriving in the queues might have severe consequences on the network performance, reflected in high delay, jitter and losses.

PathChirp has many tunable parameters which affect both the accuracy and the impact on the network. The spread factor  $\gamma$ , for example, determines the shape of the exponential and thus the intensity of the last excursion. Indeed, a sharp rate increase (large  $\gamma$ ) translates in a wider range of rates probed for, but increases the chirp burstiness as well. On the other extreme, if  $\gamma$  is small, the probing tends to be linear and it becomes “slow”. In this case, a binary-search scheme like the one in Pathload would be more effective. Another trade-off to consider is on how to set the upper and lower rates  $U$  and  $L$  so that  $L < A < U$ . Clearly, if  $U$  is close to  $A$ , the impact of the chirp on the path is limited. However, because of the exponential increase in the probing rate, the granularity of the estimates is low on the last part of the chirp while it is high at the beginning. Thus, for a better probing resolution, it is somehow convenient to set  $L$  close to  $A$ , which implies high measurement intrusiveness. All this is a direct consequence of using an exponential shape and has motivated the works in [15], [35], where a different shape is used.

Another tunable parameter in pathChirp is the average probing rate  $R = kS/W$ , where  $W$  is the chirp inter-departure time and  $S$  the packet size. By default, in pathChirp the pausing time  $W$  is chosen so that  $R = 300$  kbps, independently of the capacity and avail-bw of the path. Therefore when  $N$  measurements coexist, the overhead scales linearly as  $NR$ . Note that reducing the average probing rate  $R$  increases the pausing time between chirps but does not reduce the total amount of probing packets nor the impact of the chirps on the queues. This also leads to slower estimations.

#### APPENDIX B PATHCHIRP THEORETICAL RESULTS

We now compute the *exact* bias in the ranges labeled as “Z2” and “Z1” of Fig. 15. Let us first consider the full transient excursion (“Z2”). To determine the estimated avail-bw, we must compute the duration of the excursion. We thus consider two points in time: the beginning of the excursion  $t'$ , i.e. the instant at which the sum of the two chirps reaches the avail-bw  $A$ , and the instant  $t'' = 1 - |off|$  at which the interfering chirp ends.  $t'$  must thus satisfy  $L\Gamma^{t'} + L\Gamma^{t'-|off|} = A$  and is thus  $t' = \log_{\Gamma} \left( \frac{A/L}{1 + \Gamma^{-|off|}} \right)$ .

In the algorithm used by pathChirp, the estimated avail-bw  $\tilde{A}$  is computed as the weighted average between the received rate  $R_i$  during the increasing part of the excursion, and  $\tilde{R}_i = A$  for the rest of the train duration. Since  $R_i$  is the fair share between the two competing trains, i.e.  $R_i = L\Gamma^{t'}/(1 + \Gamma^{-|off|})$ , the estimated avail-bw is

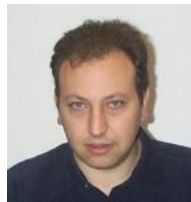
$$\begin{aligned} \tilde{A} &= R_i(t' + |off|) + \int_{t'}^{t''} \frac{L\Gamma^t}{(1 + \Gamma^{-|off|})} dt \\ &= A(t' + |off|) + \frac{L}{(1 + \Gamma^{-|off|}) \ln \Gamma} \left[ \Gamma^{1-|off|} - \frac{A/L}{1 + \Gamma^{-|off|}} \right] \end{aligned} \quad (11)$$

Note however, that in the range “Z2” the error varies little with the offset. This is because the interfering chirp dominates compared to the other one. We can thus approximate this equation by assuming that  $t'$  is such that the interfering chirp alone has reached the avail-bw  $A$ , i.e.  $L\Gamma^{t'-|off|} = A$ . From simple manipulations, this leads to the approximated form of Eq. (10).

In the range  $off < \log_{\Gamma}(A/L) - 1$  (label “Z1”), the excursion is truncated on the left side because the interfering chirp has already exceeded the avail-bw  $A$  but the other chirp has not yet started. The excursion starts immediately at the beginning of the chirp and thus  $t' = 0$ . This value can be substituted above in Eq. (11) from which the error in the range “Z1” can be easily derived.



**Daniele Croce** received his Ph.D. in 2010 jointly from Politecnico di Torino and Université de Nice-Sophia Antipolis. In 2006 he obtained a double MSc. degree from Politecnico di Torino and EURECOM in Networking Engineering. In the same year, he also obtained a Research Master diploma in Networking and Distributed Systems from UNSA, with a six month internship in Cisco Systems’s Technology Center in Sophia Antipolis (France).



**Emilio Leonardi** (SM09) received a Ph.D. in Telecommunications Engineering in 1995 Politecnico di Torino where he is currently an Associate Professor. In 1995, he visited the Computer Science Department of UCLA; in summer 1999 he joined the Bell Laboratories/Lucent Technologies, Holmdel (NJ); in 2001, the Electrical Engineering Department of the Stanford University and finally in 2003, the Sprint Advanced Technologies Laboratories, Burlingame (CA). He is the scientific coordinator of the European 7-th FP STREP project NAPA-WINE. His research interests are in the field of performance evaluation of wireless networks, P2P systems.



**Marco Mellia** (SM08) received his Ph.D. degree in telecommunications engineering in 2001 from Politecnico di Torino. During 1999, he was with the CS Department at Carnegie Mellon University. During 2002 he visited the Sprint Advanced Technology Laboratories Burlingame, CA. Since April 2001, he is with Electronics Department of Politecnico di Torino as Assistant Professor. He has co-authored over 140 papers, and he participated in the program committees of several conferences. His research interests are in the fields of traffic measurement and modeling, P2P application analysis and energy aware network design.