Evaluation and Characterization of Available Bandwidth Probing Techniques

Ningning Hu, *Student Member, IEEE*, and Peter Steenkiste, *Senior Member, IEEE*

***Abstract*—The packet pair mechanism has been shown to be a reliable method to measure the bottleneck link capacity on a network path, but its use for measuring available bandwidth is more challenging. In this paper, we use modeling, measurements, and simulations to better characterize the interaction between probing packets and the competing network traffic. We first construct a simple model to understand how competing traffic changes the probing packet gap for a single-hop network. The gap model shows that the initial probing gap is a critical parameter when using packet pairs to estimate available bandwidth. Based on this insight, we present two available bandwidth measurement techniques, the initial gap increasing (IGI) method and the packet transmission rate (PTR) method. We use extensive Internet measurements to show that these techniques estimate available bandwidth faster than existing techniques such as Pathload, with comparable accuracy. Finally, using both Internet measurements and *ns* simulations, we explore how the measurement accuracy of active probing is affected by factors such as the probing packet size, the length of probing packet train, and the competing traffic on links other than the tight link.**

***Index Terms*—Active probing, available bandwidth, Internet, network measurement.**

I. Introduction

CHARACTERIZING the end-to-end network available bandwidth is a problem that is both intellectually intriguing and of practical importance. However, the scale of the Internet, traffic volume and the diversity of network technologies make it a very challenging task. Furthermore, regular Internet users do not have access to network internals, adding to the complexity of understanding, characterizing, and modeling the performance of the Internet. While the problems of characterizing end-to-end latency and bottleneck link capacity have received a lot of attention [1]–[6], the question that is of most interest to applications is how much bandwidth is available to them along an end-to-end Internet path. What are good techniques for estimating available bandwidth and what factors affect the measurement accuracy are still open questions. Those questions are the focus of this paper.

Network measurement techniques can be classified into two categories: passive measurement [7], [8] and active probing [1]–[6]. Passive measurement tools use the trace history of existing data transmission. While potentially very efficient and accurate, their scope is limited to network paths that have recently carried user traffic. Active probing, on the other hand, can explore the entire network. The packet pair technique is one of the most popular active probing techniques. The basic idea of packet pairs is that the sender sends a pair of packets, which are echoed back by the destination. By measuring the changes in the packet spacing, the sender can estimate the bandwidth properties of the network path. While the packet pair mechanism is a reliable method for measuring the bottleneck link capacity of a network path [1], [2], [5], its use to measure the available bandwidth has had more mixed results.

Let us first define the terms *available bandwidth*, *bottleneck link*, and *tight link* more precisely. Consider an end-to-end path that includes n links $L_1, L_2, \ldots, L_n$. Their capacities are $B_1, B_2, \ldots, B_n$, and the traffic loads on these links are $C_1, C_2, \ldots, C_n$. We define the *bottleneck link* as $L_b (1 \leq b \leq n)$, where

$$B_b = \min(B_1, B_2, \ldots, B_n).$$

The *tight link* is defined as $L_t (1 \leq t \leq n)$, where

$$B_t - C_t = \min(B_1 - C_1, B_2 - C_2, \ldots, B_n - C_n).$$

In the first part of this paper, we assume that the tight link is the bottleneck link; we consider the case where the two are different in Section VIII. The unused bandwidth on the tight link, $B_t - C_t$, is called the *available bandwidth* of the path. The available bandwidth defined here, generally, does not equal the achievable bandwidth for an application. Applications often can not fully utilize the unused bandwidth due to factors such as a small receive socket buffer and packet reordering, which may limit transmission control protocol (TCP) throughput.

This paper makes the following four contributions. First, we develop a *single-hop gap model* that captures the relationship between the competing traffic throughput and the change of the packet pair gap for a single-hop network. We use this gap model to help understand the interaction between the probing packets and the competing traffic, and to identify the conditions under which the packet pair gap can be used to accurately characterize the competing traffic.

Second, based on the insights gained from the gap model, we develop two packet pair techniques—initial gap increasing (IGI) and packet transmission rate (PTR)—to characterize the available bandwidth on a network path. The two techniques experimentally determine an initial packet pair gap that will yield a high correlation between the competing traffic throughput on the bottleneck link and the packet gap at the destination. By

Manuscript received August 19, 2002; revised Februry 26, 2003. This paper was supported in part by the Defense Advanced Research Project Agency under Contract F30602-99-1-0518, monitored by AFRL/IFGA, Rome, NY, and under Contract F30602-96-1-0287, monitored by Rome Laboratory, Air Force Materiel Command, USAF.

N. Hu is with the Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213-3890 USA (e-mail: hnn+@cs.cmu.edu).

P. Steenkiste is with the School of Computer Science and the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213-3890 USA.

Digital Object Identifier 10.1109/JSAC.2003.814505

comparing the available bandwidth and TCP throughput, we show that the relative measurement error, in terms of TCP performance, is generally less than 30%. We also show that the measurement techniques discussed in this paper are much faster than existing methods such as Pathload [9], [10], with comparable measurement accuracy.

Third, we explore how packet train parameters can affect the measurement accuracy. Using Internet measurements, we show that a probing packet size around 700 Byte results in the best accuracy. We also show that the length of the probing packet train should be adjusted based on the burstiness of the competing traffic. Furthermore, we study the potential of IGI and PTR to detect the relative burstiness of Internet background traffic, which is another important metric that may be of interest to applications.

Finally, we use simulations to quantify how various factors impact the accuracy of the algorithms in multihop networks. Specifically, we look at network paths where the tight link is not the bottleneck link, and paths where links other than the tight link carry significant amount of traffic. We show that while these effects can reduce the accuracy of the algorithms, their impact is likely to be minimal in the current Internet.

This paper is organized as follows. We first discuss the related work in Section II. In Section III, we introduce the gap model. The IGI and PTR algorithms are introduced in Section IV. We present our performance evaluation methodology in Section V. The evaluation includes three parts: Section VI studies the performance properties of IGI and PTR, which include a comparison with TCP throughput and Pathload measurement. Section VII studies the impact of two IGI and PTR parameters—the probing packet size and the probing packet train length—on the accuracy of the algorithms. Section VIII uses simulation to quantify possible sources of error for IGI and PTR in multihop network. We conclude in Section IX.

II. Related Work

The problem of estimating the bottleneck link bandwidth using active probing is well studied. The work in [11] classifies the tools into single packet methods and packet pair methods. Single packet methods estimate the link capacity by measuring the time difference between the round-trip time (RTT) to one end of an individual link and that to the other end of the same link. This requires a large numbers of probing packets to filter out the effect of other factors such as queueing delay. Single packet tools include pathchar [4], clink [3], and pchar [6].

Packet pair methods send groups of back-to-back packets, i.e., packet pairs, to a server which echos them back to the sender. As pointed out in an earlier study on TCP dynamics [12], the spacing between packet pairs is determined by the bottleneck link and is preserved by the links with higher bandwidth. Example tools include NetDyn probes [13], packet pairs [14], bprobe [5], [15], and nettimer [1]. Most of these tools use statistical methods to estimate the bandwidth, based on the assumption that the most common value for the packet pair gap captures the bottleneck link transmission delay. In practice, interpreting the packet pair measurements is difficult [16]. Recent work on pathrate [2] addresses these challenges by explicitly analyzing the multimodal nature of the packet gap distribution.

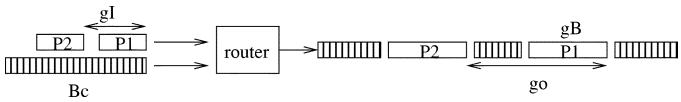


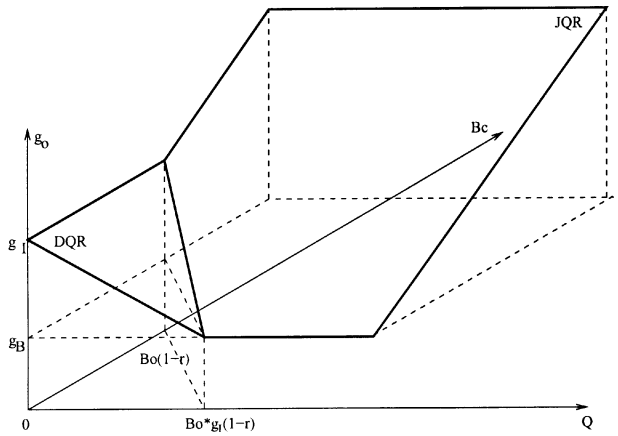
Fig. 1. Interleaving of competing traffic and probing packets. g_I is the initial gap. g_B is the probing packet length on the output link. g_O is the gap after interleaving with the competing traffic. B_C is the competing traffic throughput. Also, refer to Fig. 2 for the symbols' definition.

Characterizing the available bandwidth is more difficult since it is a dynamic property and depends on more factors. Because of the dynamic nature of the available bandwidth, it must be averaged over a time interval. Therefore, active measurement techniques often use packet trains, i.e., longer sequences of packets. A typical example is the packet bunch mode (PBM) method [16]. It extends the packet pair technique by using different-sized groups of back-to-back packets. If routers in the network implement fair queueing, the bandwidth indicated by the back-to-back packet probes is an accurate estimate for the "fair share" of the bottleneck link's bandwidth [14]. Another example, cprobe [5], sends a short sequence of echo packets between two hosts. By assuming that "almost-fair" queueing occurs during the short packet sequence, cprobe provides an estimate for the available bandwidth along the path between the hosts. Treno [17] uses TCP-like flow control and congestion control algorithms to estimate available bandwidth. The work in [2] mentions a technique for estimating the available bandwidth based on the asymptotic dispersion rate (ADR) method. Part of our work is related to ADR, and we share the view that the ADR reflects the effect of all the competing sources along the transmission path. However, we also identify the initial probing packet gap as a critical parameter that must be selected dynamically in order to achieve good accuracy.

Pathload [9], [10] characterizes the relationship between probing rate and available bandwidth by measuring the one way delay of probing packets. By trying different probing rates, a reasonable estimate for the available bandwidth can be found. The work closest to ours is the TOPP method [18]. This method provides a theoretical model for the relationship between available bandwidth and probing packet spacing at both end points. Simulations are used to validate the method. Both of these methods analyze the relationship between probing trains and available bandwidth, but their analysis does not capture the fine-grain interactions between probes and competing traffic. This is useful, for example, to understand the limitations of the techniques.

III. Single-Hop Gap Model

The idea behind using packet pairs to measure available bandwidth is to have the probing host send a pair of packets in quick succession and to measure how the packet pair gap is changed (Fig. 1). As the probing packets travel through the network, packets belonging to the competing traffic may be inserted between them, thus increasing the gap. As a result, the gap value at the destination may be a function of the competing traffic rate, making it possible to estimate the amount of competing traffic.



g_I	the **initial gap**, the time between the first bits of P1 and P2 when they enter the router; it includes P1's transmission delay on the input link
g_B	the **bottleneck gap**, the transmission delay of the probing packet on the output link; it is also the gap value of two back-to-back probing packets on the bottleneck link
g_O	the **output gap**, the time between the first bits of P1 and P2 when they leave the router, i.e., on the bottleneck link
B_O	the bottleneck link capacity
B_C	the competing traffic throughput for the time interval between the arrival of packets P1 and P2
Q	the queue size when packet P1 arrives at the router
L	the probing packet length
r	$r = g_B/g_I$

Fig. 2. Single-hop gap model. The output gap g_O is not affected by B_C in the DQR, while in the JQR, g_O is proportional to B_C. (Transmission delay is defined as the time for a packet to be placed on a link by a sender.)

In practice, the way that the competing traffic affects the packet pair gap is much more complex than what is suggested above. In this section, we describe and evaluate a simple model that captures more accurately the relationship between the gap value and the competing traffic load on a single-hop network.

A. Single-Hop Gap Model

The three-dimensional (3-D) graph in Fig. 2 shows the output gap value g_O as a function of the queue size Q and the competing traffic throughput B_C. This model assumes that the routers use first-in first-out (FIFO) queueing and that all probing packets have the same size. It also assumes that the competing traffic is constant in the interval between the arrival of packet P1 and P2; given that this interval is on the order of 1 ms, this is a reasonable assumption.

The model has two regions. As described below, the key difference between these two regions is whether or not the two packets P1 and P2 fall in the same queueing period. A *queueing period* is defined to be the time segment during which the queue is not empty, i.e., two consecutive queueing periods are separated by a time segment in which the queue is empty. For this reason, we call the two regions in the model the disjoint queuing region (DQR) and the joint queuing region (JQR).

If the queue becomes empty after P1 leaves the router and before P2 arrives, then, since we are assuming that B_C is constant in this (short) interval, P2 will find an empty queue. This means that the the output gap will be the initial gap minus the queueing delay for P1, i.e.,

$$g_O = g_I - \frac{Q}{B_O}. \tag{1}$$

Under what conditions will the queue be empty when P2 arrives? Before P2 arrives, the router needs to finish three tasks: processing the queue Q (Q/B_O), processing P1 (g_B), and processing the competing traffic that arrives between the probing packets ($B_C \cdot g_I/B_O$). The router has g_I time to complete these three operations, so the condition is $Q/B_O + B_C \cdot g_I/B_O + g_B < g_I$, which corresponds to the triangular DQR in Fig. 2. In this region, the output gap g_O is independent of the competing traffic throughput B_C. We will refer to the above (1) as the DQR equation.

Under all the other conditions, i.e., in JQR, when P2 arrives at the router, the queue will not be empty. Since we assume B_C is constant, this means that P1 and P2 are in the same queueing period. The output gap consists of two time segments: the time to process P1 (g_B), and the time to process the competing traffic that arrives between the two probing packets ($B_C \cdot g_I/B_O$). Therefore, in this region, the output gap will be

$$g_O = g_B + \frac{B_C \cdot g_I}{B_O}. \tag{2}$$

That is, in this region, the output gap g_O increases linearly with the competing traffic throughput B_C. Equation (2) is referred to as the JQR equation.

This model clearly identifies the challenge in using packet pairs for estimating the competing traffic throughput. If the packet pair happens to operate in the DQR of the bottleneck router, the output gap will bear no relationship with the competing traffic, and using the JQR equation (since the user does not know which region applies) will yield an incorrect result. Furthermore, the estimate obtained using a single packet pair will only provide the average competing traffic over g_I, which is a very short period. Since the competing traffic is likely to fluctuate, one in general will want to average the results of multiple samples, corresponding to independent packet pairs. This of course increases the chance that some of the samples will fall in the DQR.

B. Probing Packet Trains

Equation (2) shows that in the JQR, we can estimate the competing traffic throughput B_C based on the initial gap g_I, the output gap g_O, and the bottleneck gap g_B. However, the single-hop gap model assumes that the competing traffic is a smooth packet stream. In practice, the competing traffic flow will be bursty and a single pair of probing packets will not capture the *average* throughput of the competing traffic. To deal with this problem, people use a packet train [2], [16], i.e., a longer sequence of evenly spaced packets.

The conclusions from the single-hop gap model do not directly apply to a packet train. The main problem is that the "pairs" that make up a packet train are not independent. For example, if one packet pair in the train captures a burst of packets

from the competing flow, it is highly likely that adjacent pairs will not see any competing traffic and will, thus, see a decrease in their packet gap. Intuitively, if we want to estimate the amount of competing traffic, we should focus on the *increased* gaps in a probing packet train since they capture the competing traffic, while decreased gaps saw little or no competing traffic. Note that this observation only applies when the probing packet train operates in the JQR.

More precisely, assume a probing train in which M probing gaps are increased, K are unchanged, and N are decreased. If we now apply (2) to all the increased gaps, we get the following estimate for the competing traffic load:

$$\frac{B_O \sum_{i=1}^{M} (g_i^+ - g_B)}{\sum_{i=1}^{M} g_i^+ + \sum_{i=1}^{K} g_i^= + \sum_{i=1}^{N} g_i^-}. \tag{3}$$

Here, the gap values $G^+ = \{g_i^+ | i = 1, \ldots, M\}$, $G^= = \{g_i^= | i = 1, \ldots, K\}$, and $G^- = \{g_i^- | i = 1, \ldots, N\}$ denote the gaps that are increased, unchanged, and decreased, respectively. In this formula, $B_O \sum_{i=1}^{M} (g_i^+ - g_B)$ is the amount of competing traffic that arrive at router R1 during the probing period. Ideally, $\sum_{i=1}^{M} g_i^+ + \sum_{i=1}^{K} g_i^= + \sum_{i=1}^{N} g_i^-$ is the total probing time. In practice, we exclude gap values that involve lost or reordered packets, so in such cases, the denominator may be smaller than the total probing time. This method of calculating competing traffic load will be used by the IGI algorithm in Section IV, and we call it the IGI formula.

A number of groups have proposed methods to estimate the available bandwidth along a network path [5], [9], [10]. Using the same notation as used above, the equation used in [5] is

$$\frac{(M + K + N)L}{\sum_{i=1}^{M} g_i^+ + \sum_{i=1}^{K} g_i^= + \sum_{i=1}^{N} g_i^-}. \tag{4}$$

Here, L is the probing packet size. This formula represents the average transmission rate of the packet train, measured at the destination. We will also use this formula in the PTR algorithm described in Section IV, and we call it the PTR formula. In contrast, Pathload [9], [10] uses the rate of the packet trains sent by the source.

The gap model shows that the IGI formula only applies in the JQR, and we will show later that the PTR formula is also only valid under similar conditions. Note that the main parameter that is under our control in the single-hop gap model is g_I. It has a large impact on the size of the DQR and, thus, on the region in which the packet train operates. Therefore, the key to an accurate available bandwidth measurement algorithm is to find a g_I value so that the probing packet train operates in the JQR.

Before discussing the details of the algorithms used to achieve that, we first use several simple testbed experiments to illustrate the intuition behind the single-hop gap model and to show how the DQRs and JQRs affect the estimates of the IGI and PTR formulas.

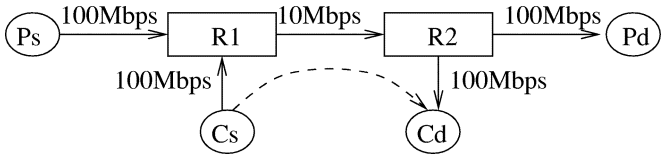


Fig. 3. Testbed configuration.

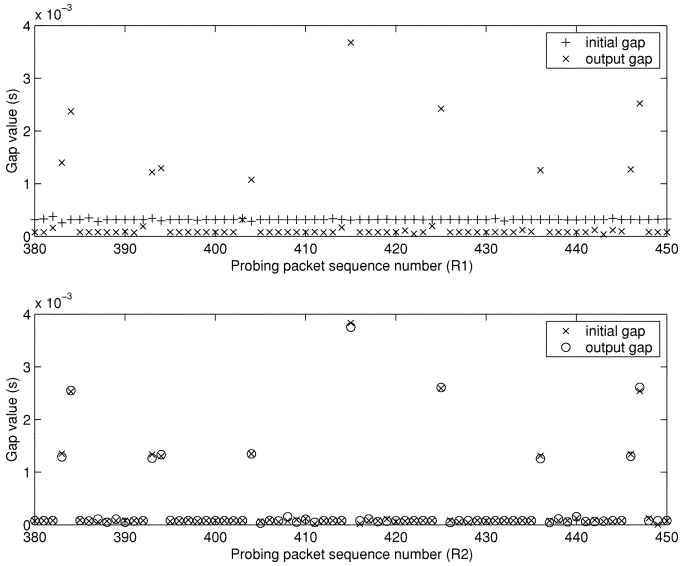


Fig. 4. Effect of JQR. Initial and output gap for routers R1 (top) and R2 (bottom).

C. Testbed Illustration

We run experiments on an isolated testbed. The topology is shown in Fig. 3. In this figure, Ps and Pd are the probing source and destination, and Cs and Cd are used to generate competing traffic. R1 and R2 are FreeBSD-based routers that run *tcpdump* on all relevant interfaces to record packet timestamp information. Ps sends out a series of evenly spaced 100-B packets, each consecutive pair of which can serve as a probing pair. The competing traffic is generated using Iperf [19], which allows us to simulate typical TCP traffic such as FTP traffic. We control the competing traffic throughput by adjusting the TCP window size.

1) Effect of JQR: Capturing Competing Traffic: In this experiment, we use 1024 probing packets[1] of size 100 Byte, so the bottleneck gap is 0.08 ms. The initial gap is set to 0.31 ms, and we use a competing traffic load of 7.2 Mb/s. A typical set of experimental results is shown in Fig. 4: the top graph shows the initial and output gaps measured on R1, and the bottom graph shows the corresponding gaps on R2. The increase in gap values on R1 is caused by competing traffic on the bottleneck link.

The increased gap values in the top graph of Fig. 4 fall into three clusters: 1.2 (the transmission delay of a 1500 Byte competing packet on a 10 Mb/s link), 2.3, and 3.8 ms. The clusters correspond to probing pairs that are separated by exactly one, two, and three competing packets. The fact that at most three packets are inserted in a probing gap should not be a surprise

[1]We use such a large number in order to get a large enough probing period. On our testbed, it does not cause packet drops and it does not significantly affect the competing flow's throughput.

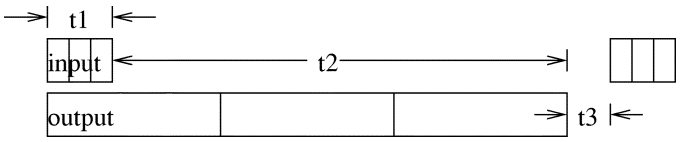


Fig. 5. Bursts of competing packets at the input and output interface of R1. t_1 is the transmission delay of the three competing packets on R1's input link, $t_2 + t_1$ is their transmission delay on the output link, t_3 is the interval between the time when the first three packets finish transmission, and the time when the second three packets arrive.

since the initial gap is 0.31 ms, and the transmission time of the competing packets on their input link is 0.12 ms (note that the input links are ten times faster than the bottleneck link). Besides the increased gap values, most of the other gap values are decreased to 0.08 ms, which is the transmission delay of the probing packets on the 10 Mb/s bottleneck link. The bottom graph shows that the increased gap values are maintained through router R2, because R2 has a higher output rate than input rate.

The changes in the gap values are the direct result of the bursty competing traffic. The *tcpdump* trace on router R1 shows that in some cases, the source Cs sends out three 1500 Byte packets back-to-back (t_1 period in Fig. 5). This builds up the queue in router R1, and the queue will drain during the period t_2. After period t_3, more competing traffic arrives. A packet pair that overlaps with period t_1 will see an increased gap; the gap value depends on whether one, two, or three competing packets are inserted between the packet pair. A packet pair that falls in period t_2 will see its gap reduced to 0.08 ms. In our experiment, because the input link capacity is ten times the output link capacity, t_2 is much longer than t_1, so more packet pair gaps are reduced than increased. Packet pairs can also straddle the t_2 and t_3 periods. In that case, the gap is reduced to a value between $g_B(0.08\ \mathrm{ms})$ and g_I (0.31 ms). This effect corresponds to the DQR, and in this example, it is not very significant.

Using the IGI formula, we can obtain an estimated competing traffic throughput of 7.3 Mb/s, and the PTR formula estimates the available bandwidth as 2.4 Mb/s. Both estimates are a good match, given that Iperf reports an average throughput of 7.2 Mb/s.

2) Effect of DQR: Losing Competing Traffic: We now reduce the competing traffic by setting the source TCP socket buffer size to 512 Bytes and the destination TCP socket buffer size to 128 Bytes. This forces the competing traffic source to send roughly one 128 Byte-packet each RTT. The parameters of the probing packet train are kept the same.

Fig. 6 shows that the increased gap values are no longer clustered around a small set of discrete values. When we apply the IGI formula to this experiment, we obtain a competing traffic throughput of 3.8 Mb/s, and PTR estimates the available bandwidth as 2.5 Mb/s (corresponding to 7.5 Mb/s competing traffic throughput). Both are higher than the real competing traffic throughput of 1.4 Mb/s.

To explain this result, Fig. 7 shows a detailed snapshot of the starting and ending time of two competing packets (A and B) and six probing packets (1–6) for both the input and output interfaces of router R1. The lines show the transmission delays of

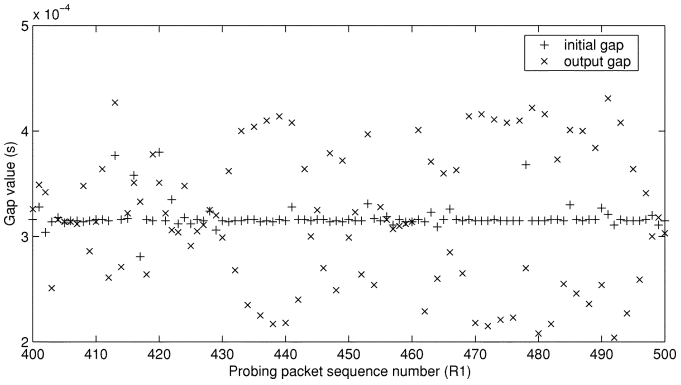


Fig. 6. Effect of DQR. The changes in the gap values are random.

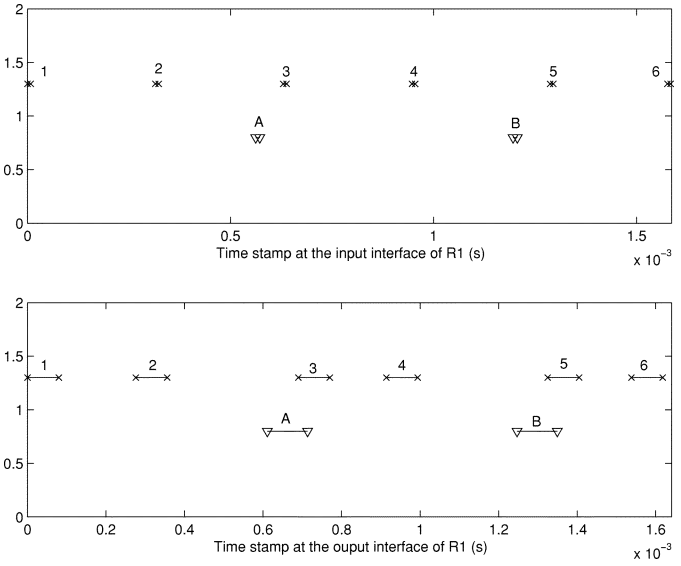


Fig. 7. Snapshot of the interleaving between two competing packets and six probing packets.

the packets.[2] Given the nature of the competing traffic, probing packets will always encounter an empty or very short queue. As a result, it is likely that two consecutive probing packets will fall in different queueing periods, and the changes in gap values are fairly random and not strongly correlated to the competing traffic load. In some cases, we see a gap increase because P2 is delayed, e.g., the pair (2, 3), which has its gap increased to 0.414 ms. In other cases, P1 is delayed, and we end up in region DQR; an example is the pair (3, 4).

D. Discussion

The single-hop gap model and our experiments show the challenges associated with using packet pairs and packet trains to estimate competing traffic on the bottleneck link. To what degree the measured gap at the destination reflects the competing traffic load depends on what region the bottleneck router is operating in. The good news is that when we are operating in the

[2]The time stamp recorded by tcpdump is the time the last bit of a packet passes through the network interface. This means that for the segments in Fig. 7, only the right end points are measured trace data. The left end points are calculated based on the packet length and the corresponding interface's transmission rate. The small overlaps between "3" and "A," and "5" and "B" are not possible and are probably due to the timing error of tcpdump.

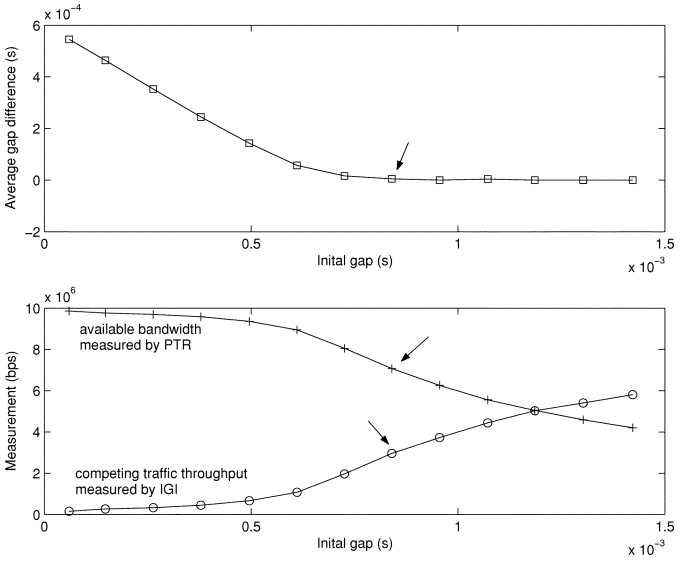


Fig. 8. Impact of the initial gap on available bandwidth measurements. The arrows point out the measurements at the turning point, the smallest initial gap where the average output gap equals the average initial gap.

JQR, there is a proportional relationship between the output gap and the competing traffic. That is the starting point for the algorithms introduced in the next section.

IV. IGI and PTR Algorithms

In this section, we describe how we use the IGI and PTR formulas as the basis for two available bandwidth estimation algorithms. The measurements presented in the previous section clearly show that the initial gap g_I has a large impact on the usefulness of the IGI and PTR formulas, so we first study the role of the initial gap more carefully.

A. Impact of Input Gap g_I

According to the single-hop gap model, if we are in the JQR, the output gap of a packet pair or train can give us an estimate of the competing traffic on the bottleneck link. However, in the DQR, output gap is independent of the competing traffic. We also see that increasing the initial gap will increase the DQR area. This argues for using small initial gaps. In fact, if $g_I \leq g_B$, i.e., if the initial gap is smaller than the probing packet transmission delay on the bottleneck link, the DQR area does not even exist. However, with small initial gaps, such as $g_I \leq g_B$, we are flooding the bottleneck link, which may cause packet losses and disrupt traffic.

In order to better understand the impact of the initial probing gap on the accuracy of the IGI and PTR formulas, we design the following experiment. We send an Iperf TCP competing traffic flow of 3.6 Mb/s over a 10-Mb/s bottleneck link. We then probe the network using a set of packet trains; the packet train length is 256 and the probing packet size is 750 Byte. We start with an initial probing gap of 0.022 ms, which is the smallest gap that we can get on the testbed, and gradually increase the initial gap. Fig. 8 shows the average gap difference (averaged output gap minus the averaged initial gap), the competing traffic throughput estimated using the IGI formula, and the available bandwidth estimated using the PTR formula.

We see that for small initial gaps (smaller than $g_B = 0.6\,\mathrm{ms}$, which is the transmission time on the bottleneck link), we are flooding the network and the measurements underestimate the competing traffic throughput. Note that for minimal initial gaps, the PTR formula is similar to the formula used to estimate the bottleneck link capacity by tools such as bprobe [5], and in fact, the PTR estimate for small initial gaps is close to 10 Mb/s, which is the bottleneck link capacity.

When the initial gap reaches g_B, the DQR effect starts to appear. Note that, unless the network is idle, we are still flooding the bottleneck link. So far, the average output gap at the destination is larger than the initial gap. When we further increase the initial probing gap, at some point (0.84 ms in the figure), the output gap equals the initial gap; we will call this the *turning point*. At this point, the probing packets interleave nicely with the competing traffic, and the average rate of the packet train equals the available bandwidth on the bottleneck link. In this experiment, the IGI estimate for the competing traffic at the turning point is 3.2 Mb/s and the PTR estimate for the available bandwidth is 7.1 Mb/s; both match the actual competing traffic (3.6 Mb/s) quite well. As we continue to increase the initial probing gap, the output gap remains equal to the initial gap since all the packets on average experience the same delay.

We believe that the point where the average output gap equals to the initial gap, i.e., the turning point shown in Fig. 8, is the correct point to measure the available bandwidth. The turning point corresponds to the smallest initial gap value with which we are not flooding the bottleneck link. With respect to the single-hop gap model in Fig. 2 on which the IGI formula is based, this initial gap will result in a packet train that keeps the queue as full as possible without overflowing it; the model shows that this puts us in the JQR. With respect to the PTR formula, the initial gap at the turning point corresponds to the packet transmission rate where the packet trains consume all the available bandwidth without significant interference with the competing traffic. In other words, the packet train behaves like an aggressive, but well behaved (i.e., congestion controlled) application flow, so its rate is a good estimate of the available bandwidth.

The IGI and PTR algorithms discussed below are based on packet trains that operate at the turning point.

B. IGI and PTR Algorithms

The IGI and PTR algorithms send a sequence of packet trains with increasing initial gap from the source to the destination host. They monitor the difference between the average source (initial) and destination (output) gap and they terminate when it becomes zero. At that point, the packet train is operating at the turning point. We then use the IGI and PTR formulas to compute the final measurement.

The pseudocode for the IGI algorithm is shown in Fig. 9. The available bandwidth is obtained by subtracting the estimated competing traffic throughput from an estimate of the bottleneck link capacity. The bottleneck link capacity can be measured using, for example, bprobe [5], nettimer [1], or pathrate [2]. Note that errors in the bottleneck link capacity measurement will affect the accuracy of the available bandwidth estimate, since the bottleneck link capacity B_O is used in the calculation of the bottleneck gap g_B, the competing traffic throughput

```
Algorithm IGI:
{
    /* initialization */
    probe_num = PROBENUM;
    packet_size = PACKETSIZE;
    g_B = GET_GB();
    init_gap = g_B/2;
    gap_step = g_B/8;
    src_gap_sum = probe_num * init_gap;
    dst_gap_sum = 0;

    /* look for probing gap value at the turning point */
    while (!GAP_EQUAL(dst_gap_sum, src_gap_sum)) {
        init_gap+ = gap_step;
        src_gap_sum = probe_num * init_gap;
        SEND_PROBING_PACKETS(probe_num,
          packet_size, init_gap);
        dst_gap_sum = GET_DST_GAPS();
    }

    /* compute the available bandwidth using IGI formula */
    inc_gap_sum = GET_INCREASED_GAPS();
    c_bw = b_bw * inc_gap_sum/dst_gap_sum;
    a_bw = b_bw - c_bw;
}
```

Fig. 9. **Algorithm** IGI. SEND_PROBING_PACKETS() sends out *probe_num packet_size* probing packets with the initial gap set to *init_gap*; GET_DST_GAPS() gets the destination (output) gap values and adds them; GET_INCREASED_GAPS() returns the sum of the initial gaps that are larger than the bottleneck gap; *c_bw, b_bw,* and *a_bw* denote the competing traffic throughput, the bottleneck link capacity, and the available bandwidth, respectively.

c_bw, and the available bandwidth a_bw. However, the analysis of the above mentioned tools and our experience show that the bottleneck link capacity measurement is fairly accurate, so in this paper, we do not consider this factor.

The PTR algorithm is almost identical to the IGI algorithm. The only difference is that we need to replace the last three lines in Fig. 9 by

$$ptr = \frac{packet\_size * 8 * (probe\_num - 1)}{dst\_gap\_sum}$$

These formulas assume that there is no packet loss or packet reordering.

In both algorithms, we try to minimize the number of probing phases by carefully selecting the gap_step and $init\_gap$. In step GET_GB(), we first probe using an $init\_gap$ that is as small as possible. This allows us to estimate the bottleneck link capacity and g_B. We then set $gap\_step = g_B/8$, and $init\_gap = g_B/2$. Another key step in both algorithms is the automatic discovery of the turning point. This is done in the procedure GAP_EQUAL(). It tests whether the source and destination gaps are "equal," which is defined as

$$\frac{|src\_gap\_sum - dst\_gap\_sum|}{\max(src\_gap\_sum, dst\_gap\_sum)} < \delta.$$

In our experiments, δ is set to 0.1. These two steps are a key difference between PTR algorithm and other techniques based on (4) since they allow us to quickly find a good initial gap. We evaluate how fast this algorithm converges in Sections VI-B and VII-B.

Besides the initial gap, two other parameters also affect the accuracy of the IGI and PTR algorithms.

1) *Probing packet size*. Measurements using small probing packets are very sensitive to interference. The work in [2] also points out significant post-bottleneck effects for small packets. This argues for sending larger probing packets.
2) *The number of probing packets*. It is well known that the Internet traffic is bursty, so a short snapshot cannot capture the average traffic load. That argues for sending a fairly large number of probing packets. However, sending too many packets can cause queue overflow and packet losses, increase the load on the network, and lengthen the time it takes to get an estimate.

Our experiments show that the quality of the estimates is not very sensitive to the probing packet size and the number of packets, and that there is a fairly large range of good values for these two parameters. For example, a 700-Byte packet size and 60 packets per train work well on the Internet. We discuss the sensitivity to these two parameters in more detail in Section VII.

V. Evaluation Methodology

Our evaluation includes three parts:

1) In Section VI, we compare the performance of IGI, PTR, and Pathload, focusing on the measurement accuracy and the convergence time.
2) In Section VII, we analyze how the probing packet size and the number of probing packets (packet train length) affect the measurement accuracy of IGI and PTR.
3) In Section VIII, we study the performance of IGI and PTR on a network path, where the tight link is not the same as the bottleneck link. We also look into a related issue about the impact of gap timing errors.

The first two parts are based on Internet measurements. The last part is based on ns2 simulations, since we need to carefully control the competing traffic load in the network.

To evaluate the accuracy of the different probing algorithms on the Internet, we interleave probing experiments with large application data transfers that show how much bandwidth is actually available and usable on the network path. However, it is sometimes hard to determine the actual available bandwidth on an Internet path. In principle, we would like the data transfers to use TCP since most applications, especially bulk data transfer applications, use TCP. Unfortunately, for high-bandwidth paths, we find that TCP is often not able to fully utilize the available bandwidth. In most cases, the reason was simply that TCP end-to-end flow control is limiting the throughput, since our guest accounts often do not allow us to increase the socket buffers to large enough sizes. On other paths, we observe a significant amount of packet reordering or unexplained

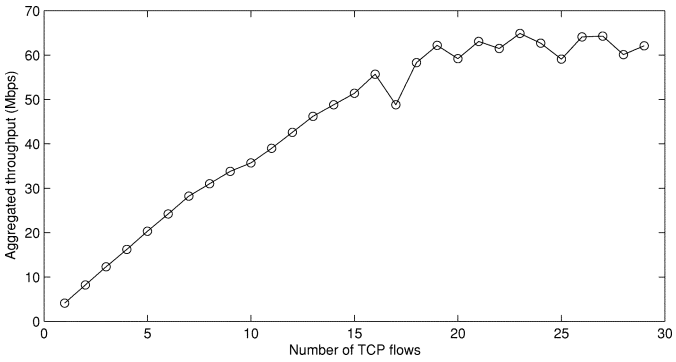


Fig. 10. Throughput of parallel TCP flows on the path ETH → NWU.

packet losses, both of which can have a significant impact on TCP performance.

For the above reasons, we use a mixture of techniques to measure the "true" available bandwidth. When possible, we use a single TCP flow. When small window sizes prevent us from filling the pipe, we use a number of parallel TCP flows. The number of flows is selected on a per path basis. A typical example of how the end-to-end throughput increases with the number of flows is shown in Fig. 10. The throughput increases initially and then flattens out. Typically, 10 or at most 20 flows are sufficient to fill the available bandwidth pipe.

Note that this approach provides only a rough idea of the accuracy of the probing techniques. A first problem is that the probing and the data transfers cannot be run at the same time, so they see different traffic conditions, and we should expect slightly different results. Moreover, because of the bandwidth sharing characteristics of TCP, a single TCP flow and multiple parallel TCP flows are not equivalent. On the other hand, our approach does model the way applications will typically use probing tools, so our approach captures the accuracy that applications will perceive. Our experience with tools such as Remos [20] shows that applications in general only require rough estimates of path properties.

The implementation of the IGI and PTR algorithms needs accurate timestamp measurement. As a result, we would expect the best results with kernel support, such as libpcap [21]. However, for most of the end hosts we use for our experiments, we only have guest accounts, so all the Internet measurements are collected with a user-level implementation. The probing packets are user-defined protocol (UDP) packets, and timestamps are measured when the client or server applications sends or receives the UDP packets.

The Pathload implementation is taken from http://www.cis.udel.edu/~dovrolis/pathload_1.0.2.tar.gz. Pathload returns a measurement interval that should contain the actual available bandwidth. In our analysis, we use the center of the interval returned by Pathload.

VI. Comparative Evaluation

In this section, we analyze the performance of IGI and PTR algorithms using experiments on the Internet. We also compare their performance with that of Pathload.

TABLE I
INTERNET PATHS

ID	*Path* (sender→receiver)	*Capacity* (Mbps)	*RTT (std dev)* (ms)
1	CORNELL→ MA	1.5	27.59 (2.82)
2	SLC1→ CMU2	10	59.65 (0.58)
3	NWU→ CMU1	100	10.29 (0.94)
4	CORNELL→ CMU1	10	13.43 (0.15)
5	ETH→ SWEDEN	10	76.04 (0.35)
6	SLC2→ CMU1	100	83.21 (0.41)
7	SLC2→ NYU	100	53.52 (0.36)
8	ETH→ CMU1	100	125.00 (0.30)
9	ETH→ NL	100	28.21 (0.21)
10	SV→ NYU	2.5	78.29 (0.21)
11	SLC1→ FC	4.5	43.39 (10.10)
12	SLC2→ FC	4.5	80.65 (23.60)
13	NCTU→ CMU3	100	265.54 (0.41)

A. Network Paths

The data presented in this section is collected using a series of experiments where each experiment measures the available bandwidth using the following three methods:

1) IGI and PTR: we use both IGI and PTR algorithms to estimate the available bandwidth. The probing packet size is set to 700 Byte and the probing packet number is 60. We discuss why we choose these two values in Section VII.
2) *Pathload*: The resolution parameter is set to 2 Mb/s.
3) *Bulk data transfer*: We use one or more Iperf TCP flows to probe for the actual available bandwidth. The transmission time is 20 seconds, and the TCP window size at both ends is set to 128 kB, which is supported on all machines we have access to.

We separate the above three measurements by a 5 seconds sleep period to avoid interference between the measurements. We separate experiments by 10 minutes of idle time. The measurements run for anywhere from 6 to 40 hours.

We collect measurements for 13 Internet paths, as listed in Table I.[3] For each path, the first site is the sender, and the second site is the receiver. The capacities in the third column denote the bottleneck link capacities, which we will also refer to as the *path capacity*. The path capacities are measured using bprobe [5], and the RTTs are measured using ping. The path capacities shown in the table are obtained by "rounding" the measured values to the nearest well-known physical link capacity.

B. Measurement Accuracy

Fig. 11 shows the relative measurement error of IGI, PTR, and Pathload.[4] We define the relative measurement error as

$$relative\_error = \frac{|a\_bw_X - throughput_{\mathrm{TCP}}|}{B_O}.$$

Here, a_bw_X can be a_bw_{IGI}, a_bw_{PTR}, and a_bw_{Pathload}, i.e., the available bandwidth estimates generated by the different

[3]CORNELL, CMU [1]–[3], NYU, ETH, and NCTU are machines in Cornell University, Carnegie Mellon University, New York University, ETH Zurich (Switzerland), and National Chiao Tung University (Taiwan), respectively. MA, SLC [1], [2], SV, FC, SWEDEN, and NL are machines on commercial networks, and they are located in Massachusetts, Silicon Valley, Foster City, Sweden, and The Netherlands, respectively.

[4]The Pathload code does not apply to paths with available bandwidth below 1.5 Mb/s (it returns the interval [0, link capacity]), so we have no Pathload measurements for Path 1.

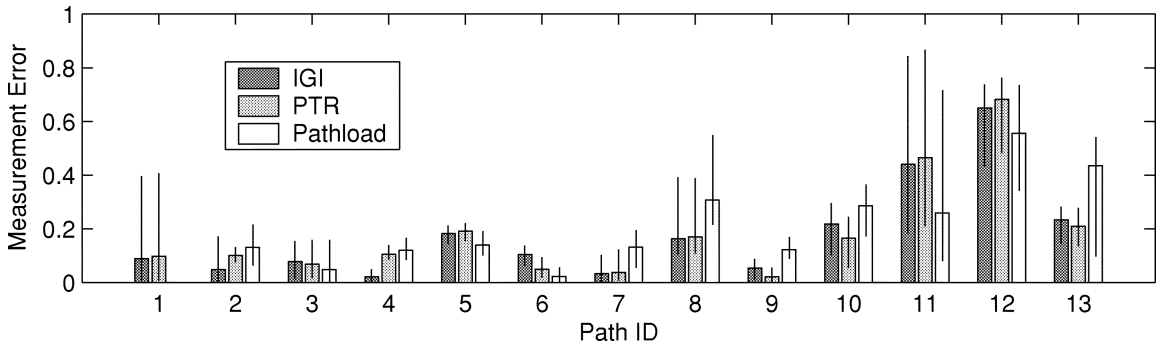


Fig. 11. Available bandwidth measurement error from IGI, PTR, and Pathload. Each bar shows the median value, and the line on each bar shows the 5% and 95% percentile values.

techniques; $throughput_{\mathrm{TCP}}$ is the bulk data transmission rate, and B_O is the bottleneck link capacity. For paths 1–10, we observe that the measurement error is below 30%, and in most cases the error is less than 20%. That is, the estimates produced by the IGI/PTR and the Pathload algorithms match the TCP performance fairly well.

For paths 11–13, the relative measurement error is much higher. Without information from the service providers, it is hard to tell what causes the higher errors. Because all three methods have low accuracy, we hypothesize that TCP has difficulty using the available bandwidth due to bad path properties. For example, Table I shows that the RTT variances for paths 11 and 12 are large compared with those for the other paths. This may be caused by route flaps, which may negatively influence TCPs performance.

In Fig. 12, we show a more detailed comparison of the bandwidth estimates for six of the paths. We pick three "good" paths with different path properties (paths 1–3, see Table I) and all three of the bad paths (path 11–13).

For paths P1, P2, and P3, the graphs confirm that all three techniques provide good estimates of the available bandwidth, as measured by Iperf. Which technique is more accurate depends on the path. For example, IGI seems more accurate for P2 and Pathload for P3. One notable exception is the period from hour 22 to hour 28 for P1, where both IGI and PTR appear to underestimate the available bandwidth. For this path, the bottleneck link is a digital subscriber line (DSL), which is in general idle, as is shown by the high available bandwidth. During the 22–28 hour interval, the DSL is used. Since only one or a few TCP connections are active, they consume only part of the available bandwidth. The bulk data transfer, however, uses five parallel Iperf flows and appears to be grabbing bandwidth from the other flows. This illustrates that the "available bandwidth" is not necessarily well-defined and depends on how aggressive the sender is. Note that this is a somewhat atypical path: on most Internet paths, individual senders will not be able to affect the bandwidth sharing as easily.

For the three paths where the relative measurement error is high, we see the available bandwidth estimates produced by all three methods are much higher than the bandwidth measured using Iperf. As we already suggested above, this probably means that TCP, as used by Iperf, is not able to function well because of problems such as window size [22], loss rate, and variable RTT [23]. Note that the three bandwidth estimation techniques provide fairly similar results, except for path P13, where the Pathload estimates are extremely high.

For most paths, the IGI and PTR estimates are within 10% of each other. One exception is for path P2 [Fig. 12 (P2)], where the IGI estimates change over a wider range than those provided by the PTR method. We believe this is caused by traffic on links other than the bottleneck link. As we will discuss in Section VIII, the IGI method is more sensitive to competing traffic from nonbottleneck links than the PTR method.

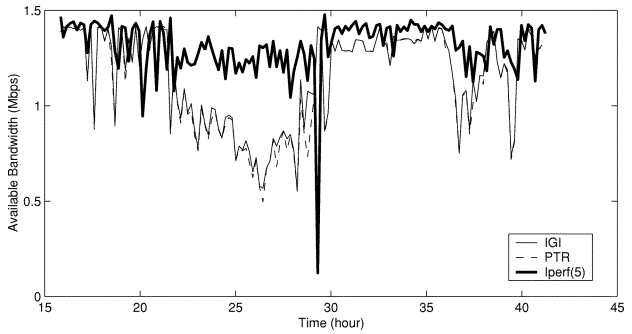
C. Convergence Times

So far our measurements have shown that the three algorithms have similar accuracy in terms of predicting available bandwidth. However, the IGI and PTR methods, which have the same measurement time, are much faster than Pathload, as is shown in Table II. In this table, we show the percentile values of the measurement times at 5%, 50% (median), and 95% for each path for both the IGI/PTR and the Pathload techniques. We see that the IGI and PTR methods typically take about 1–2 s while Pathload takes at least 12 s [9]. We also compute the ratio between Pathload and IGI/PTR for each round of measurements; the median values are listed in the last column of the table. The geometric mean [24] of all ratios shows that the IGI/PTR method is on average more than 20 times faster than Pathload for the 13 paths used in this study.

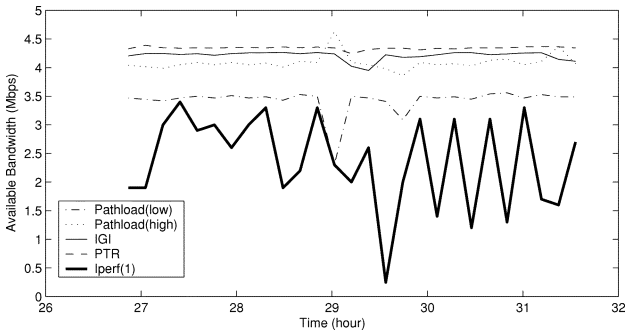
The long measurement time for Pathload is due to its convergence algorithm. Pathload monitors changes in the one-way delay of the probing packets in order to determine the relationship between probing speed and available bandwidth. This can be difficult if probing packets experience different levels of congestion. This can slow down the convergence process and can result in long probing times as shown in Table II. In contrast, the convergence of IGI/PTR is determined directly by the packet train dispersion at the source and destination. Moreover, the IGI and PTR algorithms use the bottleneck link capacity, which is estimated using the same probing procedure, to adjust *init_gap* and *gap_step* so as to optimize convergence.

VII. IGI and PTR Algorithm Properties

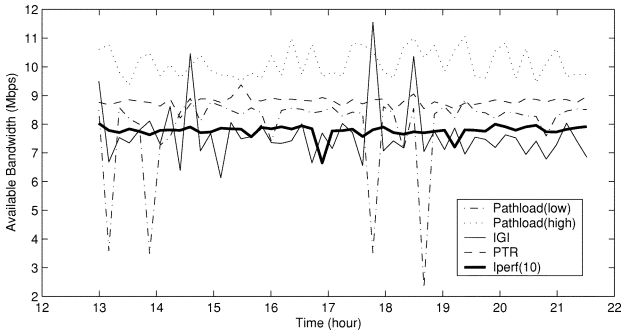
The IGI and PTR algorithms select the appropriate initial gap for the probing trains by searching for the turning point, as described in Section IV. In this section, we use Internet experiments to study the impact of the other two packet train



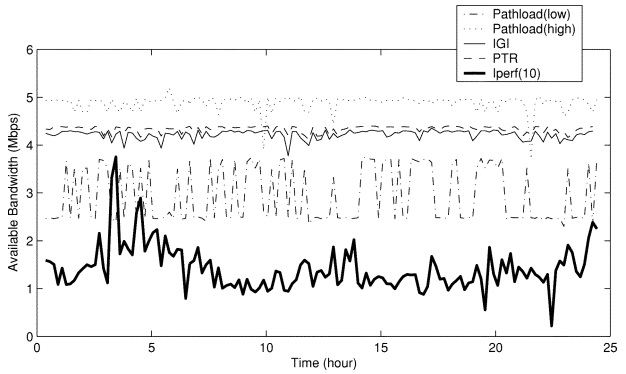
(P1) CORNELL → MA



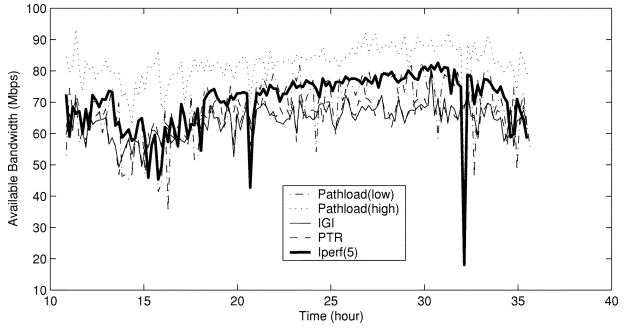
(P11) SLCI → FC



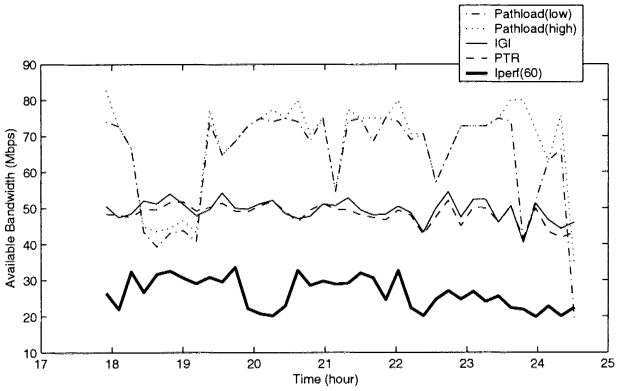
(P2) SLCI → CMU2



(P12) SLC2 → FC



(P3) NWU → CMU1



(P13) NCTU → CMU3

Fig. 12. Available bandwidth measurements and the corresponding TCP performance. The number in the brackets of Iperf is the number of Iperf TCP flows used. X axis is the clock time value, a number larger than 24 is the time next day.

TABLE II
MEASUREMENT TIME

Path ID	*IGI/PTR* (s) (5%, **median**, 95%)	Pathload (s) (5%, **median**, 95%)	Ratio($\frac{Pathload}{IGI/PTR}$) median
1	(1.60, **2.05**, 6.27)	(14.98, **30.56**, 31.03)	13.22
2	(0.58, **0.73**, 1.56)	(13.67, **15.37**, 31.81)	20.86
3	(0.11, **0.11**, 0.18)	(7.55, **13.17**, 14.91)	99.78
4	(0.49, **0.52**, 0.52)	(11.78, **12.26**, 12.76)	23.48
5	(0.78, **0.80**, 0.83)	(15.58, **15.86**, 16.55)	19.75
6	(0.62, **0.80**, 1.20)	(49.07, **56.18**, 62.24)	70.08
7	(0.51, **0.51**, 0.67)	(14.01, **22.40**, 28.51)	45.94
8	(1.01, **1.02**, 1.27)	(27.57, **31.51**, 47.62)	27.80
9	(0.24, **0.30**, 0.30)	(15.35, **16.14**, 27.66)	65.81
10	(1.27, **1.27**, 1.50)	(20.95, **21.04**, 21.77)	16.50
11	(1.03, **1.10**, 2.03)	(19.97, **25.78**, 38.52)	23.45
12	(2.17, **2.32**, 3.60)	(19.24, **21.54**, 42.00)	9.20
13	(1.10, **1.11**, 1.13)	(12.24, **12.76**, 47.22)	11.24
Geometric Mean			26.39

parameters—the probing packet size and the number of probing packets (packet train length).

A. Probing Packet Size

To study the impact of the probing packet size on the measurement accuracy of the IGI and PTR algorithms, we conduct experiments on two Internet paths, using probing packet sizes ranging from 100 to 1400 Byte. We repeat each individual measurement 20 times. The entire experiment takes about 1 h. On the assumption that Internet path properties do not change much on the scale of hours [25], we would expect all measurements to have very similar result.

The first Internet path we use is from NWU to CMU. It has a path capacity of 100 Mb/s. The measurement results are shown in Fig. 13(a) and (c) shows how the available bandwidth measurements change with the probing packet size. The available

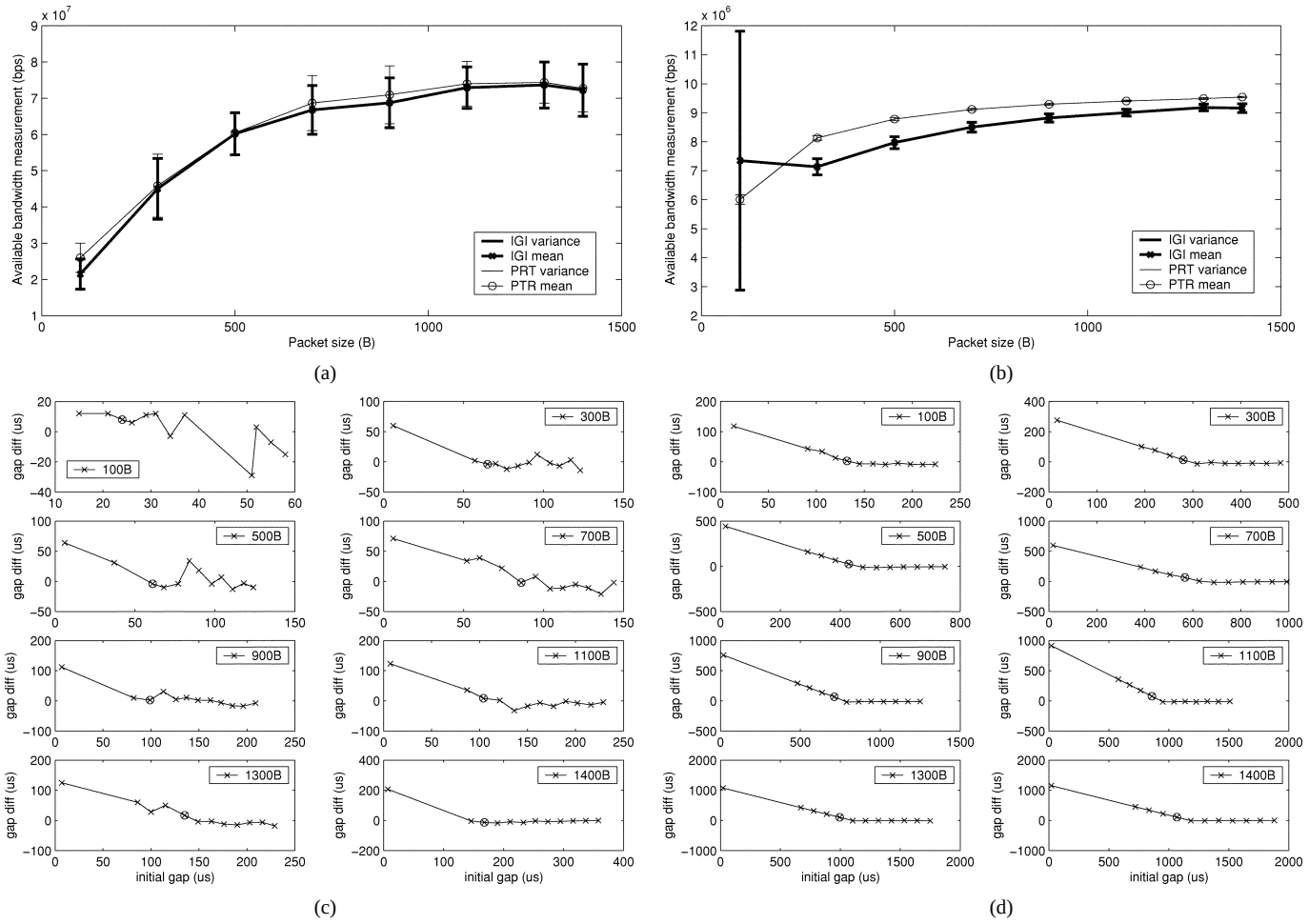


Fig. 13. IGI and PTR measurements with different probing packet sizes on two Internet paths. Graphs (a) and (b) show the final available bandwidth estimates. Graphs (c) and (d) show the gap convergence for individual measurements. The x axis is the initial source gap, and the y axis is the gap difference, i.e., the destination (output) gap value minus the source (input) gap value. The points marked with circles are the turning points where the final estimates are computed.

bandwidth measured using a TCP bulk data transfer (based on the method discussed in Section VI) is 64 Mb/s. The packet sizes that result in the closest estimates are 500 and 700 Byte. For smaller packet sizes, both methods underestimate the available bandwidth by a significant margin. For larger probing packet sizes, the two methods overestimate the available bandwidth by a much smaller amount.

There are at least two reasons why small probing packet sizes can result in high errors in the available bandwidth estimation. First, as illustrated in Fig. 13(c), at the turning point the gap value is proportional to the packet size. This means that with small packet sizes, we will have small gaps, especially if the available bandwidth is high, as is the case for the NWU to CMU path. The resulting probing train is more sensitive to the burstiness of the competing traffic. The graph for 100 Byte probing packets in Fig. 13(c) confirms this: the gap difference does not converge as nicely as it does with larger probing packets. The second reason is that the small gap values that occur with small probing packets are harder to measure accurately, so measurement errors can affect the result significantly. Gap values on the order of 10 μs are hard to generate and measure accurately, especially for user-level applications.

It is less clear why with larger probing packets, the available bandwidth estimates further increase and in fact exceed the measured bulk throughput. We conjecture that this is a result of the aggressiveness of the probing packet train flow. Probing flows with larger packets are more aggressive than probing flows with smaller packets, so they "observe" a higher available bandwidth. The packet size distribution on Internet has clusters around 40, 500, and 1500 Byte [26], so a flow with only 1200 or 1500 Byte packets, for example, is more aggressive than average. A TCP bulk data transfer is likely to use mostly maximum-sized packets (1500 B in this case), but its dynamic congestion control behavior reduces how much bandwidth it can use.

The second experiment is on the path from CORNELL to CMU. The results are summarized in Fig. 13(b) and (d). The link capacity of the bottleneck link is only 10 Mb/s, as opposed to 100 Mb/s for the NWU to CMU path. As a result, the available bandwidth is significantly lower. The results confirm the main results of the measurements for the NWU to CMU path. First, the available bandwidth estimates increase with the packet size. Second, since the available bandwidth is much lower, we are seeing fairly smooth convergence of the gap difference, even for small probing packet sizes [Fig. 13(d)]. Finally, even though we observe nice convergence, the burstiness of the competing traffic does affect the probes with small packets more than the probes with larger packets. For the IGI algorithm, the results with 100 Byte probing packet are

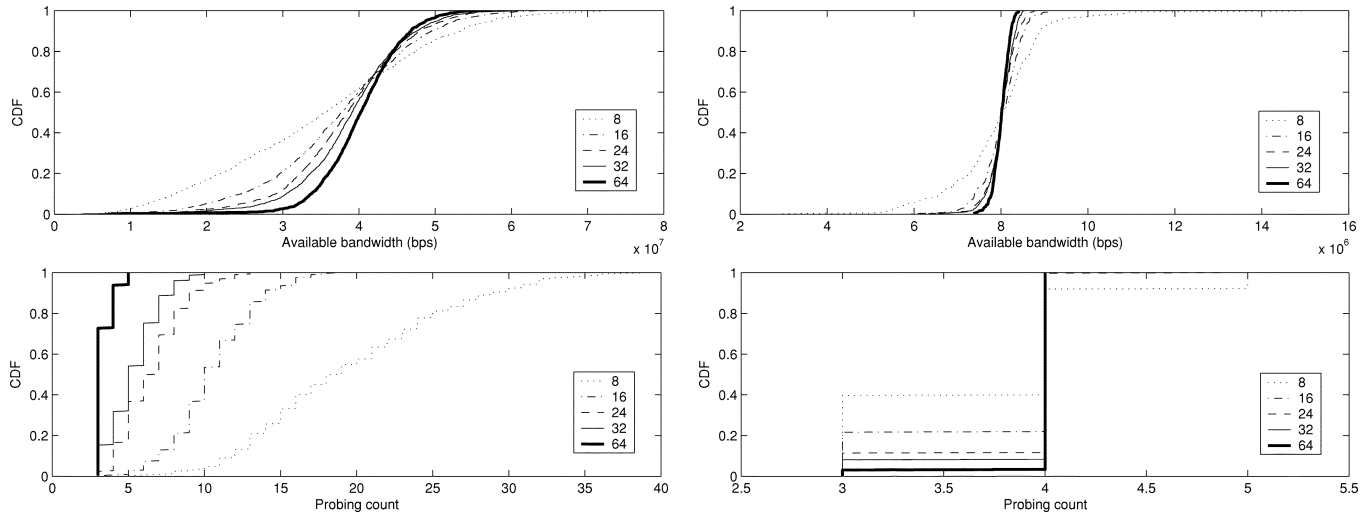


Fig. 14. Performance with packet trains of different lengths.

suspicious and have a large variance. Because the IGI algorithm uses the changes in individual gap values instead of the average packet train rate (as used by PTR), it is more sensitive to small changes in gap values, for example as a result of bursty traffic or traffic on nonbottleneck links. We discuss this point in more detail in Section VIII.

Our conclusion is that in general, average-sized probing packets of about 500 to 700 Byte are likely to yield the most representative available bandwidth estimate. Smaller packet sizes may underestimate the available rate and may be more sensitive to measurement errors, while larger probing packet sizes can overpredict the available bandwidth.

B. Packet Train Length and Number of Probing Phases

The packet train length has a large impact on the cost of the PTR and IGI algorithms, since it affects both the number of packets that are sent (i.e., the load placed on the network) and the probing time (i.e., the latency associated with the probing operation). Another important parameter, the number of phases needed to converge on the best initial gap value (the turning point), is tied very closely to the packet train length. Intuitively, shorter packet trains provide less accurate information, so more phases may be needed to converge on the turning point. For this reason, we will study the packet train length and the number of phases in the IGI/PTR algorithm together.

In Section IV, we mentioned that trains of 60 packets work well. In this section, we experimentally evaluate how much we can reduce this number without a significant loss in accuracy. We conduct experiments over the same two Internet paths as in the previous section, i.e., NWU to CMU and CORNELL to CMU. For each path, we use packet trains of different lengths to estimate the available bandwidth. The measurements take about two hours. Since the available bandwidth over the Internet is fairly stable [25], we do not expect the available bandwidth to change significantly during the 2-h period. The measurements with different train lengths are also interleaved to further reduce any possible bias toward a specific train length.

Fig. 14 shows the cumulative distribution function (CDF) of the estimated available bandwidth using IGI (top), and the number of probing phases needed to converge on the turning point (bottom). The distributions for the PTR measurement are similar and are not included here. Each graph has five curves, corresponding to five different packet train lengths: 8, 16, 24, 32, and 64. First, we observe that shorter packet trains need more phases to converge, which we had already conjectured earlier. The measurements also show, again not surprisingly, that shorter packet trains result in a wider range of available bandwidth estimates, as shown by a CDF that is more spread out. The reason is that the competing traffic (and, thus, the available bandwidth) is bursty, and since a shorter packet train corresponds to a shorter sampling interval, we are seeing a wider range of estimates. Note, however, that as the packet train length increases, the impact of the packet train length on the distribution of the bandwidth estimates becomes smaller, i.e., the estimates converge on a specific value.

It is interesting to compare the results for the two paths. For the NWU to CMU path, changing the packet train length has a fairly significant impact on the distributions for both the available bandwidth and the phase count. In other words, increasing the packet train length helps in providing a more predictable available bandwidth estimate. Using longer trains is also "rewarded" with a reduction in the the number of probing phases. In contrast, for the CORNELL to CMU path the CDF functions for both the available bandwidth and phase count are fairly similar for train lengths of 16 packets or more. The reason is that the competing traffic on this path is not as bursty as that on the NWU to CMU path.

The difference between the two paths raises the question of what packet train length we should use for available bandwidth estimation. Clearly, the most appropriate train length depends on the path. For the NWU to CMU path, we probably would want to use a fairly large value (32 or 64 packets), while for the CORNELL to CMU path, a train length of 16 packets is sufficient. Since the difference between the paths appears to be caused by the burstiness of the traffic, we decide to use the changes in the packet gaps to characterize the burstiness of the

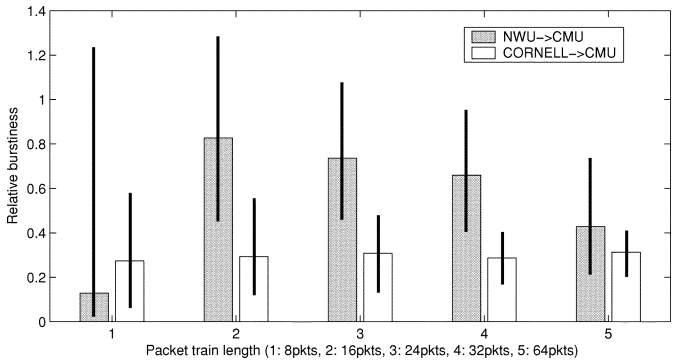


Fig. 15. Relative burstiness measurements based on the gap values of a probing train. Each bar shows the median value and the lines on each bar show the 5 and 95% values.

competing traffic. Specifically, we define the *relative burstiness* as

$$relative\_burstiness = \frac{\frac{1}{N-1}\sum_{i=2}^{N}|g_i - g_{i-1}|}{\frac{1}{N}\sum_{i=1}^{N} g_i}$$

where $g_i (1 \le i \le N)$ are the N gap measurements of a probing train.

Fig. 15 shows the relative burstiness of the IGI measurements at the turning point for the two paths and for the different packet train lengths. We record the detailed gap values at the turning point for 65 measurements (around 20% of the measurements collected). The relative burstiness for the path from NWU to CMU is significantly higher than that for the path from CORNELL to CMU. Interesting enough, the results for eight-packet probing trains do not follow this trend. We suspect that eight packets is simply not long enough to get a reliable measurement (note the wide spread).

These results suggest that we can reduce the cost of probing by dynamically adjusting the length of the packet train. For example, we could use a packet train of 32 packets for the first few phases and use the burstiness results of those phases to adjust the length of later packet trains. We decide not to do this because, as the results in Table II show, the IGI/PTR algorithm is already quite fast. The distribution of the probing phase counts shows that 80% of the measurements only need 4–6 phases to converge to the turning point, so the corresponding probing time is around 4–6 RTTs. Dynamically adjusting the packet train length is, thus, not likely to have a large impact on the probing time. Of course, we could make the burstiness information available to users so they can know how variable the available bandwidth is likely to be for short data transfers.

VIII. Multihop Effects

The IGI and PTR algorithms are based on the gap model presented in Section III. It is derived for a simple single-hop network, or more generally, for a network in which the bottleneck link is the tight link and the effect of all other links can be ignored. In this section, we use simulations to study more general multihop networks. Specifically, we address two questions. First, how should we interpret the model if the tight link is not the bottleneck link, and what are the implications for the IGI and PTR method? Second, how does the competing traffic on links other than the tight link affect the accuracy of the algorithms?

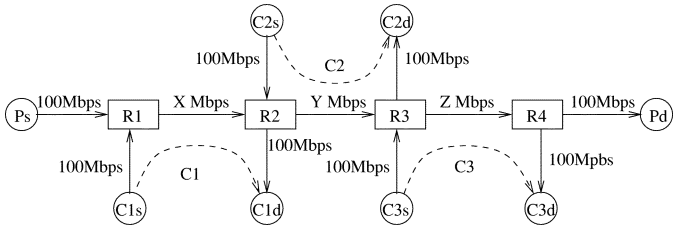


Fig. 16. Simulation configuration. Ps and Pd are used for probing. C1s, C1d, C2s, C2d, C3s, and C3d are used for the competing traffic generation.

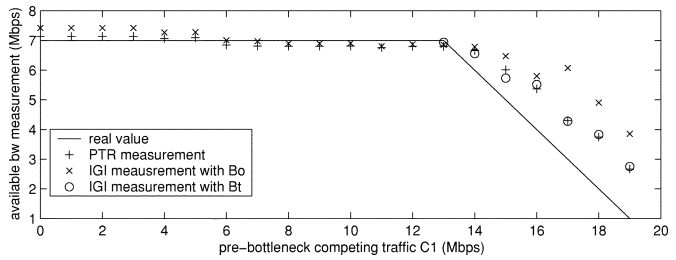


Fig. 17. Pretight link effect. Here, B_O is the bottleneck link capacity, and B_t is the tight link capacity.

A. Tight Link Is Not the Bottleneck Link

When the tight link and the bottleneck link are different, the gap model shows that the IGI algorithm should use the B_O and g_B values for the tight link when estimating the available bandwidth. Unfortunately, tools such as bprobe only estimate the capacity of the bottleneck link. This will have an impact on the accuracy of the method. Note that PTR does not use the B_O and g_B values explicitly, so it will not be affected by this tight link issue.

In the remainder of this section, we will use ns2 [27] simulation to evaluate the accuracy of both algorithms in this scenario. While simulation has the drawback that it leaves out many real-world effects, it has the advantage that we can study topologies that are difficult or impossible to build.

We use the simulation topology shown in Fig. 16, using 20, 10, and 20 Mb/s for the link capacities X, Y, and Z, respectively. By changing the competing loads C1, C2, and C3 we can change the tight link of the path and also change the level of traffic on links other than the tight link. The probing packet size used in the simulation is 700 Byte and the probing packet train length is 60. The competing traffic consists of CBR UDP traffic. Note that by picking link capacities that are fairly close, the available bandwidths on different links are likely to be close as well, which is a challenging case.

In the first set of simulations, we set C2 to 3 Mb/s and change C1 from 0 to 19 Mb/s. When C1 is in the range 0–13 Mb/s, the bottleneck link $\langle$R2, R3$\rangle$ is also the tight link, but when C1 falls in 13–19 Mb/s, the tight link is $\langle$R1, R2$\rangle$. Fig. 17 presents the simulation results. We see that when the bottleneck link is equal to the tight link ($0 \le C1 \le 13$ Mb/s), the IGI method accurately predicts the available bandwidth, as expected. When $\langle$R1, R2$\rangle$ is the tight link, we show the IGI estimates based on the

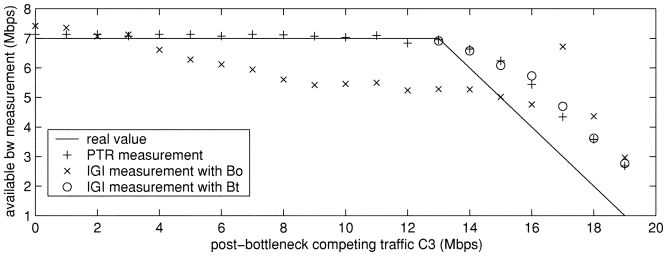


Fig. 18. Posttight link effect. Here, B_O is the bottleneck link capacity, and B_t is the tight link capacity.

B_O and g_B values for both the tight ("o" points) and bottleneck links ("x" points). We see that the results using the tight link values are much closer. The error is the result of interference from competing traffic on the "nontight" link, as we discuss in more detail in the next subsection.

Next, we run a similar set of simulations, but we now keep C2 fixed to 3 Mb/s and change the competing traffic C3 from 0 to 19 Mb/s. The tight link switches from $\langle$R2, R3$\rangle$ to $\langle$R3, R4$\rangle$ when C3 goes above 13 Mb/s. Fig. 18 shows that the results are similar to those in Fig. 17: when the tight link is not the bottleneck link ($13 \leq C3 \leq 19$ Mb/s), using the B_O and g_B values for the tight link gives a more accurate prediction for the available bandwidth on the path. However, the results when $0 \leq C3 \leq 13$ Mb/s are less clear than for the pretight link case in Fig. 17, we will explain it in the next section.

In Figs. 17 and 18, we also plot the corresponding PTR values. The PTR estimates are almost identical to the IGI estimates that use the B_O and g_B values for the tight link. The reason is that the PTR formula does not explicitly use any information about the tight link capacity.

The fact that the IGI algorithm uses the capacity of the tight link explicitly is a problem because we only have techniques for identifying the link capacity of the bottleneck link, not the tight link. In practice, this is not likely to be a problem: we expect that on many paths, the access link from the client network to the ISP will be both the bottleneck and the tight link. Our Internet measurements in Section VI confirm this.

B. Interference From Traffic on "Nontight" Links

In a multihop network, each link will potentially affect the gap value of a packet pair or packet train, so we have to effectively concatenate multiple instances of the single-hop gap model. Such a multihop gap model is hard to interpret. However, it is fairly easy to see that it is the link with the lowest unused bandwidth (i.e., the tight link) that will have the largest impact on the gap at the destination. The intuition is as follows. On links that have a lot of unused bandwidth, the packets of the probing flow are likely to encounter an empty queue, i.e., these links will have a limited impact on the gap value. Of course, these links may still have some effect on the gap values, as we analyze in this section using the simulation results from the previous section.

The results in Fig. 17 for $0 \leq C1 \leq 13$ Mb/s show that both IGI and PTR are very accurate, even when there is significant competing traffic on a link preceding the tight link. Interesting enough, the second set of simulations show a different result.

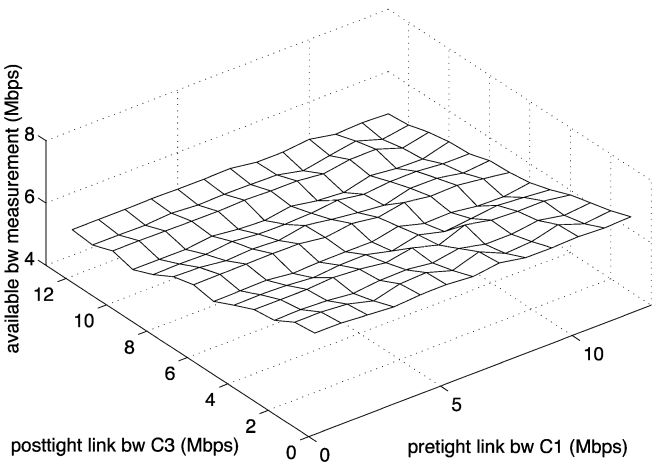


Fig. 19. Combined pretight and posttight link effects with 20 Mb/s pretight and posttight link capacities.

The results in Fig. 18 for $0 \leq C3 \leq 13$ Mb/s correspond to the case that there is significant competing traffic on a link following the tight link. We observe that while PTR is still accurate, the IGI accuracy suffers.

The different impact on IGI of competing traffic in links upstream and downstream of tight link can be explained as follows. Changes in gap values before the tight link will be *reshaped* by the router which the tight link connects with, and the competing traffic on the tight link ends up having the dominating impact. In contrast, any changes in gap values that are caused by traffic on links following the tight link will directly affect the available bandwidth estimates, so they have a larger impact. Since IGI is based on more fine-grain information than PTR, it is more sensitive to this effect.

In Fig. 19, we show the available bandwidth, as estimated by IGI, when there is significant competing traffic on both the links before and after the tight link. The actual available bandwidth is 7 Mb/s for all data points. It is determined by link $\langle$R2, R3$\rangle$, which has 10 Mb/s capacity and 3 Mb/s competing traffic (C2). The results confirm the above observation. Even significant competing traffic before the tight link has almost no impact on the accuracy: the curve is basically flat along the C1 axis. Competing traffic after the tight link does, however, have an effect and, not surprisingly, its impact increases with the level of competing traffic.

Note that the above simulations results are designed to highlight a particularly challenging case. In practice, it is not common to have links with capacities and/or available bandwidths that are this similar. In such cases, the effect of competing traffic on other links is very minimal. For example, we run a set of simulations similar to those described above, but with the $\langle$R1, R2$\rangle$ and $\langle$R3, R4$\rangle$ set to 100 Mb/s instead of 20 Mb/s. The capacity of $\langle$R2, R3$\rangle$ and it competing traffic throughput (C2) keep to be 10 and 3 Mb/s, respectively, i.e., the available bandwidth is still 7 Mb/s. The results are shown in Fig. 20. We see that the IGI method gives accurate results—the mean value for the data points in this figure is 7.24 Mb/s, and the standard deviation is 0.10 Mb/s. The fact that IGI and PTR typically produce very similar estimates in our Internet

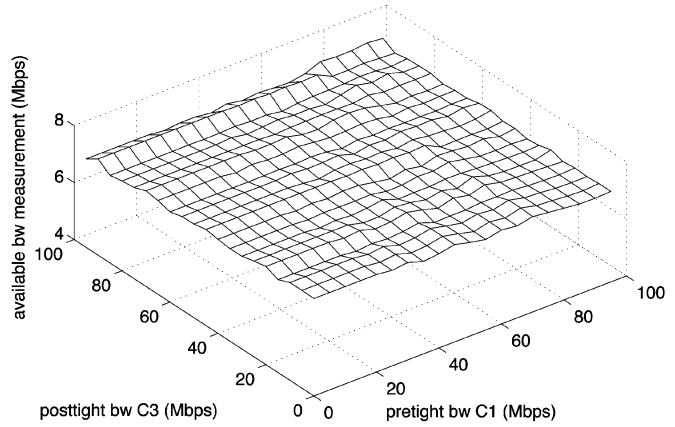


Fig. 20. Combined pretight and posttight link effects with 100 Mb/s pretight and posttight link capacities.

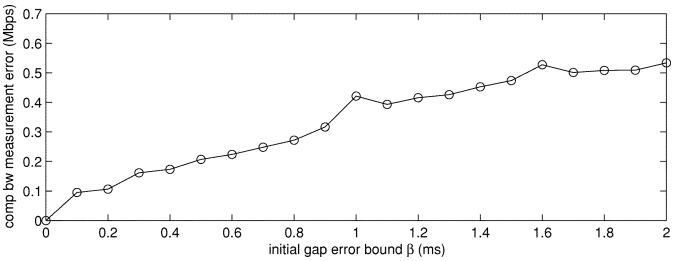


Fig. 21. Impact of initial gap error.

experiments shows that the results in Fig. 20 are much more typical than the worst case results in Figs. 17 and 18.

C. Timing Errors

Another factor that can reduce the accuracy of the IGI and PTR algorithms is the measurement errors in the gap values. There are two types of gap measurement errors: the errors in the initial gap value generated by the source host and the measurement errors in the final gap value measured on the destination host.

To illustrate the effect of source gap generation error, we use the topology shown in Fig. 16, with X, Y, and Z set to 20, 10, and 20 Mb/s, respectively. The flow C2 is the only competing flow and we change its throughput in the range of 0–9 Mb/s. For each experiment, the initial gap (g_I) is incremented by a random value ϵ that is uniformly distributed in $(-\beta, \beta)$, i.e.,

$$g = \max(0, g_I + \epsilon), -\beta < \epsilon < \beta.$$

We run simulations for β ranging from 0–2 ms, and for each β value, we collect results when C2 changes between 0 and 9 Mb/s. Fig. 21 shows the average absolute error in the IGI estimate as a function of β. We see that the error is small. Note the turning gap value for this simulation is 0.3–1.7 ms, so the errors inflicted on the initial gap are quite large. We believe that the reason for the high-error tolerance is the same as the reason for the low sensitivity of IGI to the pretight link traffic. Specifically, the tight link ends up reshaping the gaps according to the competing traffic, thus, in effect hiding the initial gap errors.

Clearly, measurement errors on the destination side will have a more significant impact since they will directly change the gap values that are used in the IGI and PTR formulas.

IX. Conclusion

In this paper, we present a simple gap model that captures how competing traffic on a network link affects the gap value of packet pairs and packet trains. The gap model helps us identify under what conditions packet pairs probing yields useful information about the available bandwidth along a path. It also shows that the key to get useful measurements is to control the initial gap of the packet train. The most accurate results are obtained when the average output gap at the destination equals the average initial gap at the source.

We design two techniques for estimating available bandwidth based on the gap model. The IGI algorithm uses the information about changes in gap values of a packet train to estimate the competing bandwidth on the tight link of the path. The PTR method uses the average rate of the packet train as an estimate of the available bandwidth.

We compare the estimates of the IGI and PTR algorithms with Pathload estimates and measured TCP throughput on the Internet. The results show that all three methods (IGI, PTR, and Pathload) have a similar measurement error: in most cases the error is less than 30%. IGI and PTR typically finish in under 2 s while Pathload takes a lot longer. An analysis of the algorithm properties provides suggestions on how to choose the probing packet size and the probing packet train length in order to achieve the best measurement accuracy with the least overhead.

In the last part of this paper, we use simulations to study the dynamics of the methods in networks with significant traffic on multiple links along the path. We show that the IGI method loses accuracy if the tight link is not the bottleneck link, or if there is significant competing traffic on links following the tight link. Competing traffic before the tight link has, however, little impact on the accuracy. Since the PTR method does not make use of the detailed changes in the gap values in the probing packet train, it is much less sensitive to the presence of traffic on links other than the tight link. These results suggest that the PTR method is the preferred method for estimating available bandwidth.

Acknowledgment

The authors would like to thank D. Andersen, Y.-H. Chu, P. Dinda, and R. Karrer for allowing us to use their resources in our Internet experiments. They would also like to thank M. Agrawal, R. Balan, Y.-H. Chu, J. Gao, A.-C. Huang, J. Lopez, and S. Rao for their useful comments.

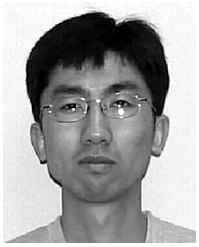
References

[1] K. Lai and M. Baker, "Nettimer: A tool for measuring bottleneck link bandwidth," in *Proc. USENIX Symp. Internet Technologies and Systems*, Mar. 2001, pp. 123–134.

[2] C. Dovrolis, P. Ramanathan, and D. Moore, "What do packet dispersion techniques measure?," in *Proc. Conf. Computer Communication*, Apr. 2001, pp. 905–914.

[3] A. B. Downey. Clink: A tool for estimating internet llink characteristics. [Online]. Available: http://rocky.wellesley.edu/downey/clink/

[4] V. Jacobson, "Pathchar – A tool to infer characteristics of internet paths," presented at the Mathematical Sciences Research Institute, Apr. 1997, [Online]. Available: ftp://ftp.ee.lbl.gov/pathchar/msri-talk.pdf.
[5] R. L. Carter and M. E. Crovella, "Measuring bottleneck link speed in packet-switched networks," Boston Univ., Boston, MA, Comput. Sci. Dept., Tech. Rep., Mar. 1996.
[6] B. A. Mah. (2001) Pchar: A tool for measuring internet path characteristics. [Online]. Available: http://www.employees.org/bmah/Software/pchar/
[7] S. Seshan, M. Stemm, and R. H. Katz, "SPAND: Shared passive network performance discovery," in *Proc. 1st Usenix Symp. Internet Technologies Systems*, Monterey, CA, Dec. 1997, pp. 135–146.
[8] M. Stemm, S. Seshan, and R. H. Katz, "A network measurement architecture for adaptive applications," in *Proc. Conf. Computer Communication*, Mar. 2000, pp. 285–294.
[9] M. Jain and C. Dovrolis, "End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput," in *Proc. ACM SIGCOMM Symp. Communications Architectures Protocols*, Pittsburgh, PA, Aug. 2002, pp. 295–308.
[10] ——, "Pathload: A measurement tool for end-to-end available bandwidth," in *Proc. Passive Active Measurements*, Fort Collins, CO, Mar. 2002.
[11] J. Curtis and T. McGregor, "Review of bandwidth estimation techniques," in *Proc. New Zealand Computer Science Research Students' Conf.*, vol. 8, New Zealand, Apr. 2001.
[12] V. Jacobson, "Congestion avoidance and control," in *Proc. ACM SIGCOMM Symp. Communications Architectures Protocols*, Aug. 1988, pp. 314–329.
[13] J.-C. Bolot, "End-to-end packet delay and loss behavior in the Internet," in *Proc. ACM SIGCOMM Symp. Communications Architectures Protocols*, San Francisco, CA, Sept. 1993, pp. 289–298.
[14] S. Keshav. Packet Pair Flow Control. [Online]. Available: http://www.cs.cornell.edu/skeshav/doc/94/2–17.ps
[15] R. L. Carter and M. E. Crovella, "Dynamic server selection using bandwidth probing in wide-area networks," Boston Univ., Boston, MA, Comput. Sci. Dept., Tech. Rep., Mar. 1996.
[16] V. Paxson, "Measurements and Analysis of End-to-End Internet Dynamics," Ph.D. dissertation, Comput. Sci. Div., U.C. Berkeley, Berkeley, CA, May 1996.
[17] M. Mathis and J. Mahdavi, "Diagnosing Internet congestion with a transport layer performance tool," presented at the INET Conf., Montreal, QC, Canada, June 1996.
[18] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Proc. IEEE GLOBECOM–Global Internet Symp.*, San Francisco, CA, Nov. 2000, pp. 415–420.
[19] A. Tirumala and J. Ferguson. Iperf. [Online]. Available: http://dast.nlanr.net/Projects/Iperf/
[20] P. A. Dinda, T. Gross, R. Karrer, B. Lowekamp, N. Miller, P. Steenkiste, and D. Sutherland, "The architecture of the Remos system," in *Proc. 10th IEEE Int. Symp. High Performance Distributed Computing*, Aug. 2001, pp. 252–265.
[21] Libpcap [Online]. Available: ftp://ftp.ee.lbl.gov/libpcap.tar.Z
[22] J. Semke, J. Mahdavi, and M. Mathis, "Automatic TCP buffer tuning," in *Proc. ACM SIGCOMM Symp. Communications Architectures Protocols*, Vancouver, BC, Canada, Sept. 1998, pp. 315–323.
[23] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *Proc. ACM SIGCOMM Symp. Communications Architectures Protocols*, Vancouver, BC, Canada, Sept. 1998, pp. 303–314.
[24] R. Jain, *The Art of Computer Systems Performance Analysis*. New York: Wiley, 1991.
[25] Y. Zhang, N. Duffield, V. Paxson, and S. Shenker, "On the constancy of Internet path properties," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, San Francisco, CA, Nov. 2001, pp. 197–211.
[26] K. Claffy, G. Miller, and K. Thompson, "The nature of the beast: Recent traffic measurements from an internet backbone," presented at the ISOC INET Conf., July 1998.
[27] Ns2 [Online]. Available: http://www.isi.edu/nsnam/ns



Ningning Hu (S'02) received the B.S. degree in computer science from Nanjing University, Nanjing, China, in 1998 and is working toward the Ph.D. degree in the Computer Science Department, Carnegie Mellon University, Pittsburg, PA.

His research interests are in network measurement, networking protocols, and active network services.

Peter Steenkiste (M'00–SM'01) received the B.S. degree in electrical engineering from the University of Ghent, Belgium, in 1982, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1983 and 1987, respectively.

He is a Professor in the Department of Computer Science and the Department Electrical and Computer Engineering, Carnegie Mellon University, Pittsburg, PA. His research interests are in the areas of networking and distributed system.