PAPER

A Simultaneous Inline Measurement Mechanism for Capacity and Available Bandwidth of End-to-End Network Path

Cao LE THANH MAN[† a)], **Go HASEGAWA**[† b)], ***and*** **Masayuki MURATA**[† c)], ***Members***

SUMMARY

We previously proposed a new version of TCP, called Inline measurement TCP (ImTCP), in [2, 3]. The ImTCP sender adjusts the transmission intervals of data packets and then utilizes the arrival intervals of ACK packets for available bandwidth estimation. This type of active measurement is preferred because the obtained results are as accurate as those of other conventional types of active measurement, even though no extra probe traffic is injected onto the network. In the present research, we develop a new capacity measurement function and combine it with ImTCP in order to enable simultaneous measurement of both capacity and available bandwidth in ImTCP. The capacity measurement algorithm is a new packet-pair-based measurement technique that utilizes the estimated available bandwidth values for capacity calculation. This new algorithm promises faster measurement than current packet-pair-based measurement algorithms for various situations and works well for high-load networks, in which current algorithms do not work properly. Moreover, the new algorithm provides a confidence interval for the measurement result.

key words: *TCP, capacity, available bandwidth, packet pair, end-to-end measurement, inline measurement*

1. Introduction

The *capacity* of an end-to-end network path, which is considered to be the smallest capacity of network links along a path, is the maximum possible throughput that the network path can provide. Traffic may reach this maximum throughput when there is no other traffic along the path. The *available bandwidth* indicates the unused bandwidth of a network path, which is the maximum throughput that newly injected traffic may reach without affecting the existing traffic. The two bandwidth-related values are obviously important with respect to adaptive control of the network.

In many cases, both capacity and available bandwidth information are required at the same time. For example, network transport protocols should optimize link utilization according to available bandwidth. However, if a connection tends to fully using the available bandwidth, other connections that join the network later will find it difficult in obtaining bandwidth. Therefore, connections do not share the bandwidth fairly. In this case, if the connections are aware of the capacity, they can quickly change the used bandwidth so that the fairness with newly attended connections is maintained. One method of using capacity and available bandwidth information to optimize both bandwidth utilization and connection fairness for TCP is proposed in [4]. Another example is in large file transfers. Capacity information can be used for the decision of the size of the data to be transferred. The same video data can be recorded in many files with different bit rates, such as Video CD (1.12Mbps), DVD-Video (1.5Mps 9Mbps), etc., and, therefore, different data sizes. Because available bandwidth changes frequently, it is better to use capacity information to decide a suitable file for the transfer. Available bandwidth information is then used to improve the performance of the transmission of the file. Besides, the billing policy of the Internet service provider is based on both the capacity and the available bandwidth of the access link that they are providing to the customer.

Several passive and active measurement approaches exist for capacity or available bandwidth [5-14]. Although active approaches are preferred because of their accuracy and measurement speed, sending extra traffic onto the network is a disadvantage that is common to all active measurement tools. For example, Pathload [5] generates between 2.5 and 10 MB of probe traffic per measurement. The average per-measurement probe traffic generated by Spruce [6] is 300 KB. For routing in overlay networks, or adaptive control in transport protocols, these measurements may be repeated continuously and simultaneously from numerous end hosts. In such cases, the probes will create a large amount of traffic that may degrade the transmission of other data on the network, as well as the measurement accuracy itself.

We therefore propose an active measurement method that does not add probe traffic to the network. The proposed method uses the concept of "plugging" the new measurement mechanism into an active TCP connection (*inline measurement*). We previously introduced ImTCP (Inline measurement TCP) [3], a Reno-based TCP that deploys inline measurement for available bandwidth. The ImTCP sender not only observes the ACK packet arrival intervals in the same manner as TCP Westwood [15], but also actively adjusts the transmission interval of data packets, in the same way that active measurement tools use probe packets. When the corresponding ACK packets return, the sender utilizes the arrival intervals to calculate the measurement values.

The available bandwidth measurement algorithm for ImTCP is described in detail in [3]. For each measurement, the ImTCP sender searches for the available bandwidth only

[†] This paper is based on "An Inline Measurement Method for Capacity of End-to-end Network Path" [1], by the same authors, which appeared in the Proceedings of IM'2005 E2EMON Workshop ©2005 IEEE

[†] The authors are with the Graduate School of Information Science and Technology, Osaka University.

a) E-mail: mlt-cao@ist.osaka-u.ac.jp
b) E-mail: hasegawa@ist.osaka-u.ac.jp
c) E-mail: murata@ist.osaka-u.ac.jp

within a given search range. The search range is a range of bandwidth that is expected to include the current available bandwidth and is calculated statistically from the previous measurement results. Without a search range, measurement tools (such as Pathload) must send packet in many transmission rates, from 0 Mbps to the upper limit of the physical bandwidth, to probe the network. The search range limits the range of the bandwidth that the measurement tool should probe, therefore, probe packets will not be sent in a high rate if not necessary. Thus, the measurements do not cause much effect on other traffic in the network. The search range also allows the number of packets for the measurement to be kept small, so that measurement is still possible when the TCP window size is relatively small. The search range is divided into multiple sub-ranges of identical width of bandwidth. For each of the sub-ranges of the bandwidth, the sender transmits a group of TCP data packets (a packet stream), the transmission rate of which varies to cover the sub-range. The sender then determines whether an increasing trend exists in the transmission delay of packets in each stream when the echoed (ACK) packets arrive at the sender host. Delayed ACKs is supposed to be disabled at the receiver because the ImTCP sender will stop measurement and perform like a normal TCP sender if it finds out that many expected ACKs do not arrive. The increasing trend indicates that the transmission rate of the stream is larger than the current available bandwidth of the network path [5]. This fact allows the sender to infer the location of the available bandwidth in the search range. The simulation results show that the ImTCP sender can perform periodic measurements at short intervals, on the order of several RTTs and the measurements results reflect well the changes in the available bandwidth of the network.

In the present paper, we introduce an inline measurement algorithm for capacity for ImTCP. The proposed algorithm utilizes the arrival intervals of the ACK packets of packet pairs (PPs) that are sent back-to-back. Due to the characteristic of ImTCP that PPs are available after the transmission of each measurement stream, therefore the capacity measurements do not require any further changes in ImTCP. With the proposed method, ImTCP measures the capacity at the early stage of the connection and continues to collect data to improve the measurement accuracy during the transmission. We do not intend to develop a new capacity measurement tool that is better than the existing ones [7-9]. Rather, with the effort of reducing the load over the network caused by probe traffic, our main focus is on how we can extract capacity information from a TCP connection with the smallest change in TCP.

The main concept of the proposed capacity measurement algorithm in ImTCP is that the available bandwidth information, which can be yielded periodically due to the deployed available bandwidth measurement mechanism, is exploited. In the existing PP-based capacity measurement algorithm [7-9], the PPs that are cut into by other packets from cross traffic at the bottleneck link causes incorrect capacity estimation and are therefore eliminated from the data used in the calculation. However, in the proposed method, the available bandwidth information is used for estimation of the quantity of the cross traffic that cuts in PPs at the bottleneck link. The interval of the PPs becomes usable for the capacity measurement, which enables ImTCP to collect more information from PPs so that faster and more accurate measurements can be expected. The proposed algorithm also uses statistical analysis to calculate the confidence interval of the delivered results.

Through simulation validations, we show that ImTCP can deliver capacity measurement results quickly, independent of the characteristics of the network. In addition, we find that the capacity measurement algorithm works well in extremely high-load networks, in which current measurement algorithms do not work well.

The remainder of this paper is organized as follows. In Section 2 we discuss PP-based measurement techniques used for inline measurement. In Section 3, we introduce the proposed measurement algorithm for network capacity. In Section 4, we evaluate its performance through simulation experiments. Finally, in Section 5, we present concluding remarks and discuss future projects.

2. Packet-pair-based capacity measurement algorithms

Currently there are various approaches for measuring the capacity of an end-to-end network path [16-20]. Some of these approaches use packets of various size to probe the network and infer the network capacity from the difference in the transmission delays of packets of various sizes [16]. Other approaches use the probe packets in different TTLs (Time To Live) to measure all link bandwidth, rather than just the capacity of the bottleneck link [17-20]. However, the packet size in TCP is always set to path MTU, which is the maximum size a packet can have to avoid fragmentation. A change in packet size can therefore only be done by selecting smaller packets, which requires TCP to send more packets. Setting small TLL values to TCP packets in order to dropt them along the path causes packet retransmissions and reduction in TCP window size. Thus, changes in TCP data packet size or TLLs for the purpose of measurement may cause severe deterioration in the data transmission throughput of TCP so these approaches can not be used for inline measurement.

We found that only PP-based measurement can be used for inline measurement because no changes in packet size or TTL are required, whereas packets that are sent back-to-back can be created with the current ImTCP structure without requiring any changes.

2.1 Packet pair technique

The intuitive rationale of capacity measurement using PPs is that if two packets are sent close enough together in time to cause the packets to queue back-to-back at the bottleneck link, then the packets will arrive at the destination with the same spacing as when they left the bottleneck link [16]. The

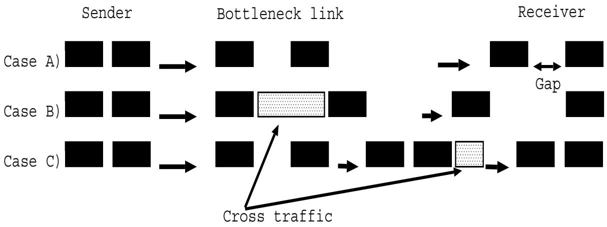


Fig. 1 Three cases showing how the spacing between a pair of packets may change as the pair travels along a path.

spacing is supposed to remain unchanged until the PPs reach the receiver, as shown in Case A of Figure 1, which is a variation of a figure taken from [8].

In this case, the capacity of the bottleneck link (C) can be calculated by the equation:

$$C = \frac{P}{Gap} \tag{1}$$

where P is the size of the PPs, and Gap is the time spacing of the two packets when arriving at the receiver.

However, when a PP travels along the path, two more situations can occur. As shown by Case B in Figure 1, the two packets may be cut into by other packets from cross traffic at the bottleneck link. The result is that, the spacing between the two packets becomes larger than expected. In this case, Equation (1) leads to an under-estimation of the capacity. In another case, indicated by Case C in Figure 1, the PPs may pass back-to-back through the bottleneck link, but in a link downstream of the bottleneck link, the pairs again get in queue, and the spacing between the two packets is shortened. In this case, Equation (1) leads to over-estimation.

Current PP-based measurement techniques use only the PPs described in Case A to calculate capacity. These techniques have various mechanisms for determining the Case-A PPs from all of the received PPs. Some tools assume a high frequency of appearance of Case-A PPs and so search for these PPs from a frequency histogram (Pathrate [7]) or a weighting function (Nettimer [8]). CapProbe [9] repeatedly sends PPs until it discovers a Case-A PP, based on the transmission delay of the packets.

When the network path is almost empty, Case-A PPs may appear with the highest frequency. However, when other traffic appears in the network, there is a high probability that the cross traffic on the tight link (the link having smallest available bandwidth) stretches the PPs so that their intervals become large; the PPs then become Case B. Case-C PPs do also exist, but some probing results from the Internet in [7] show that they are much fewer than Case-B ones. In this case, because Case B PPs occur more often, CapProbe will spend an extremely long time for capacity searching, and Pathrate and Nettimer will deliver incorrect estimations.

Unlike those existing techniques, we propose a new technique by which to calculate capacity that can use both Case-A PPs and Case-B PPs. This is possible because of the available bandwidth information that is available in ImTCP.

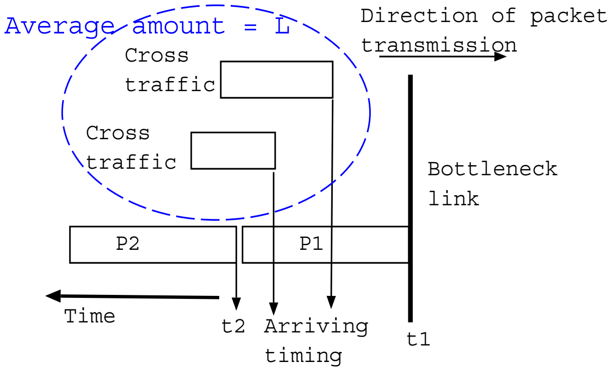


Fig. 2 Arrival time at the bottleneck link of PPs and cross traffic

2.2 Capacity calculation

Let us consider the timing of the arrival at the bottleneck link of a PP (Figure 2). We assume that the first packet arrives at t_1 and the second packet arrives at t_2. During the interval from t_1 to t_2, packets from other traffic may arrive at the bottleneck link. The second packet (P2) must wait in the queue for the processing of the cross packets. Therefore, the time spacing (*Gap*) of the PP after leaving the bottleneck link is the total of the queuing time and the processing time of the second packet. That is:

$$Gap = \frac{P + L}{C} \tag{2}$$

where L is the amount of the cross traffic that arrives at the bottleneck link during the interval (t_1, t_2). Supposing that the bottleneck link of a network path is the link having the smallest available bandwidth, we can then calculate the total transmission rate of the cross traffic at the bottleneck link as: $C - A$, where A is the current available bandwidth. Let δ be the time spacing of the PP upon arrival at the bottleneck link ($\delta = t_2 - t_1$). Then, the average value of L is:

$$L = \delta(C - A) \tag{3}$$

from Equations (2) and (3), we can write:

$$C = \frac{P + \delta(C - A)}{Gap},$$

or

$$C = \frac{P - \delta \cdot A}{Gap - \delta}. \tag{4}$$

Equation (4) enables the calculation of capacity from the PPs for both Case A and Case B. In the next section, we propose the new capacity calculation algorithm based on the equation.

3. Inline measurement algorithm for capacity

3.1 Overview

We introduce an inline PP-based measurement algorithm

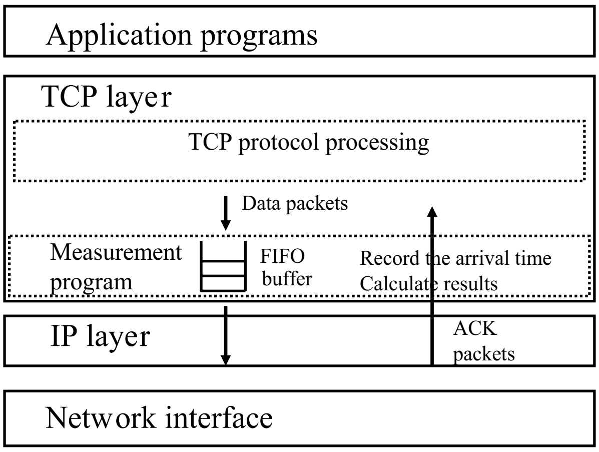


Fig. 3 Placement of measurement program at ImTCP sender

for capacity that utilizes available bandwidth to improve the measurement accuracy. The available bandwidth information can be used because an inline packet stream-based measurement mechanism for it already exists in ImTCP. Some existing available bandwidth measurement tools, such as IGI/PTR [12], take the reverse approach, that is obtaining capacity information first, then using it together with PP probing results to find available bandwidth. Moreover, TOPP [21] measures both available bandwidth and capacity at the same time using PPs. However, as shown in recent experiments in real networks [10], measuring available bandwidth with packet streams is more valid than using packet pairs. Therefore, we think that the approach that we take in ImTCP is better.

The proposed capacity measurement mechanism has the following characteristics:

- The mechanism does not require any change in the current structure of ImTCP. Therefore, it does not affect the data transmission performance of ImTCP.
- The measurement starts and is able to provide results at the early stage of the connection. Unlike other methods such as the work by Hoe [22], the measurement also continues during the transmission. When the connection lasts for a long time, the measurement exploits the accumulated data to improve its accuracy.

3.2 Implementation of packet pairs in ImTCP

As introduced in our previous study [3], a measurement program is inserted into the sender program of TCP Reno to create an ImTCP sender. The measurement program is located at the bottom of the TCP layer, as shown in Figure 3. When a new data packet is generated at the TCP layer and is ready to be transmitted, the packet is stored in an intermediate FIFO buffer. The measurement program waits until the number of packets in the intermediate buffer becomes sufficient and then decides the time at which to send the packets in the buffer in order to create measurement streams. When no measurement stream is needed, the program immediately passes all of the data packets to the IP layer. In the previous version of ImTCP, we decided that the program forms and sends one measurement stream for the available bandwidth in each RTT in order to maintain fairness with respect to traditional TCP Reno.

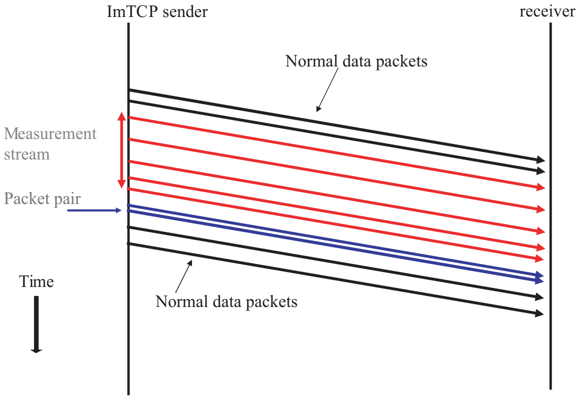


Fig. 4 Creation of PPs in ImTCP

During the transmission of a measurement stream, which includes five packets, there is a high probability that more than two packets arrive at and are stored in the intermediate FIFO buffer. Making use of the fact that after the transmission of a stream, ImTCP sends all stored packets in a bursty fashion, the capacity measurement program considers the first two packets in the burst as a PP to perform the measurement. Thus, there is no effect on the performance of ImTCP by introducing the capacity measurement mechanism.

In ImTCP, 2–4 measurement streams are required in order to determine the available bandwidth. As mentioned above, each PP is formed and transmitted after each measurement stream. Therefore, 2–4 results for PPs can be obtained during the interval of two consecutive measurement results for available bandwidth.

3.3 Proposed measurement algorithm

We next explain the procedure for determining the capacity from the measurement results of PPs using Figure 5. The procedure involves the following steps:

- Grouping of PPs: PPs that sent when the measured available bandwidth remains unchanged are placed in the same `group`. The average value of arrival interval of PPs in a group, denoted by $\overline{Gap}$, is then calculated. To obtain a good average value, the number of PPs in each group should be enough large i.e. larger than or equal to 3, as determined herein. Therefore, after grouping, a group having only one or two PPs will be merged with the group that is collected right after that.
- Calculation: Based on the $\overline{Gap}$ value of a group, a `sample` of capacity is calculated using the following function.

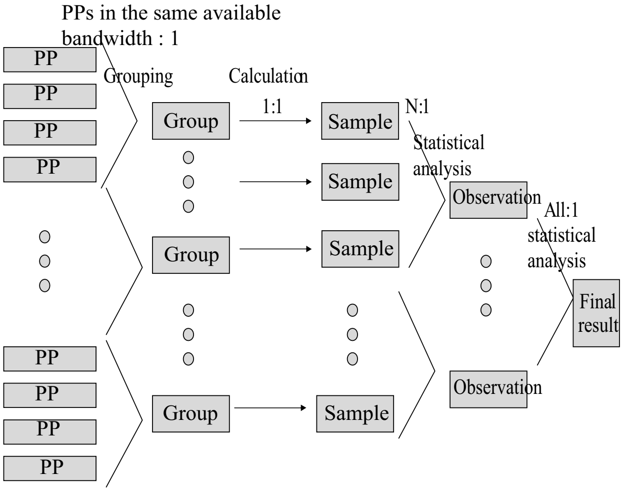


Fig. 5 Proposed algorithm

$$C = \begin{cases} \dfrac{P}{\overline{Gap}} & \text{when } \frac{A}{P/\delta} > \lambda, \quad (5) \\ \dfrac{P - \delta \cdot A}{\overline{Gap} - \delta}, & \text{otherwise}, \quad (6) \end{cases}$$

where λ is the threshold showing the relation between the available bandwidth and the rate of the PPs upon arriving at the bottleneck link, that is defined as P/δ . We assume that the links before the bottleneck link do not have a noticeable effect on the time space, so that δ is approximated by the time interval in which the sender sends the packets. When the available bandwidth is approximately equivalent to the rate of the PPs upon arriving at the bottleneck link, which is considered as $\frac{A}{P/\delta} > \lambda$, the packets may pass through the link without being cut into by other packets (Case A). In this case, Equation (5) (based on Equation (1)) is used. On the other hand, since when the arrival rate of the PPs is much higher than the available bandwidth, which is considered as $\frac{A}{P/\delta} \leq \lambda$, the probability is high that the PP is a Case-B PP, Equation (6) (based on Equation (4)) is used. The changes in δ before the PP arriving at the bottleneck link make the calculation for sample of capacity using Eq.(4) incorrect. However, we believe that the changes are small and do not occur so often. The task of grouping N samples in the next step of the algorithm is is an effort to reduce the effects of the changes.

- Statistical analysis:
  - We form `obsevations`, each of which is the average value of N samples. N should be large enough so that each observation has high accuracy. But when N is too large, the time required to finish an observation is long. This means that the proposed algorithm can not deliver the measurement results quickly. In the present paper, based on empirical experiments, we recommend $N = 10$.
  - The average value of the observations are calculated as the "final result". The 90% confidence interval is also calculated to show the degree of fluctuation of the capacity.

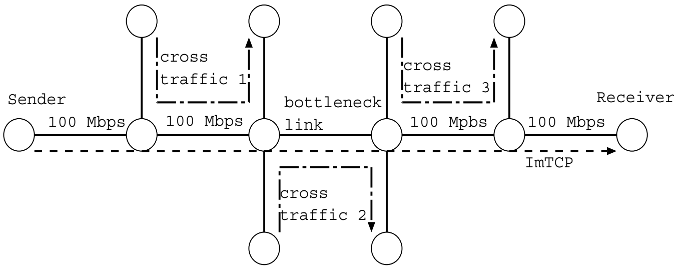


Fig. 6 Simulation topology

Table 1 Distribution of packet size of the cross traffic.

Packet size (Bytes)	Proportion of bandwidth (%)
28	0.08
40	0.51
44	0.22
48	0.24
52	0.45
552	1.10
576	16.40
628	1.50
1420	10.50
1500	37.10
40–80 (range)	4.60
80–576 (range)	9.60
576–1500 (range)	17.70

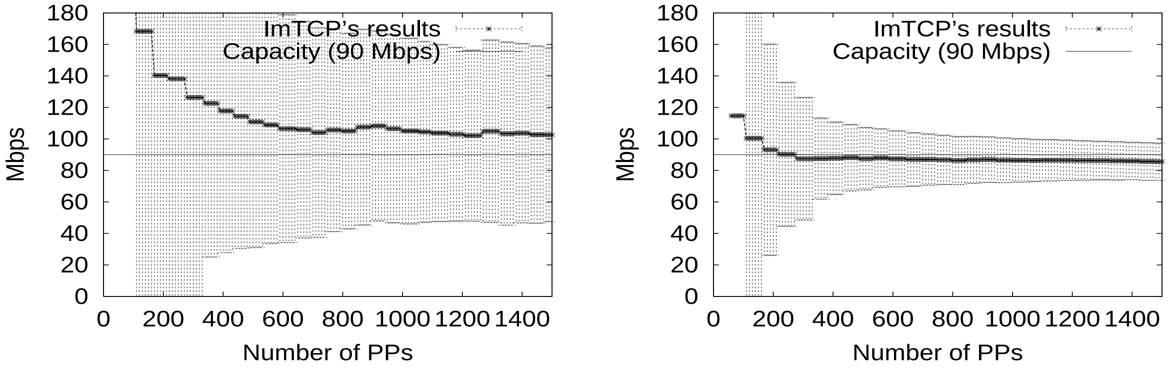
4. Simulation experiments

In this Section, we examine the measurement results of the proposed capacity measurement algorithm through ns-2 [23] simulations. We also compare the proposed algorithm with two existing algorithms, CapProbe [9] and Pathrate[7]. We compare the algorithms in the scope of inline measurement, because we only focus on how to extract capacity information from a TCP connection without introducing any extra probe traffic in to the network.

We use the simulation topology shown in Figure 6. The transmission rate of Cross traffic 1 is fixed to 5 Mbps and that of Cross traffic 3 is fixed to 15 Mbps. The packet size distribution of cross traffic is set to the statistical results for the Internet traffic reported in [24], as shown in Table 1. This mixture has an average packet size of 404.5 bytes and has a correlation value of 0.999 when compared to realistic Internet traffic.

4.1 Effect of parameters

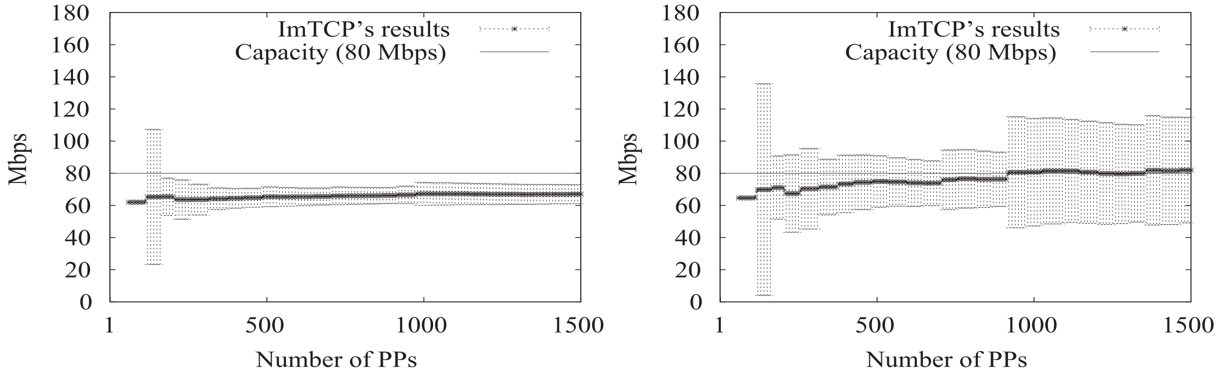
- Value of λ
  We set the bottleneck link capacity to 90 Mbps and the transmission rate of Cross traffic 2 to 5 Mbps and examine the measurement results when $\lambda = 0.9$ (Figure 7(a)) and λ = 0.8 (Figure 7(b)). These figures show



(a) $\lambda = 0.9$

(b) $\lambda = 0.8$

Fig. 7 Measurement results for the proposed algorithm when Cross traffic 2 is 5 Mbps. The errors bars show the 90% confidence interval of the correspondent results. $\lambda = 0.8$ gives more accurate results than $\lambda = 0.9$.



(a) $\lambda = 0.5$

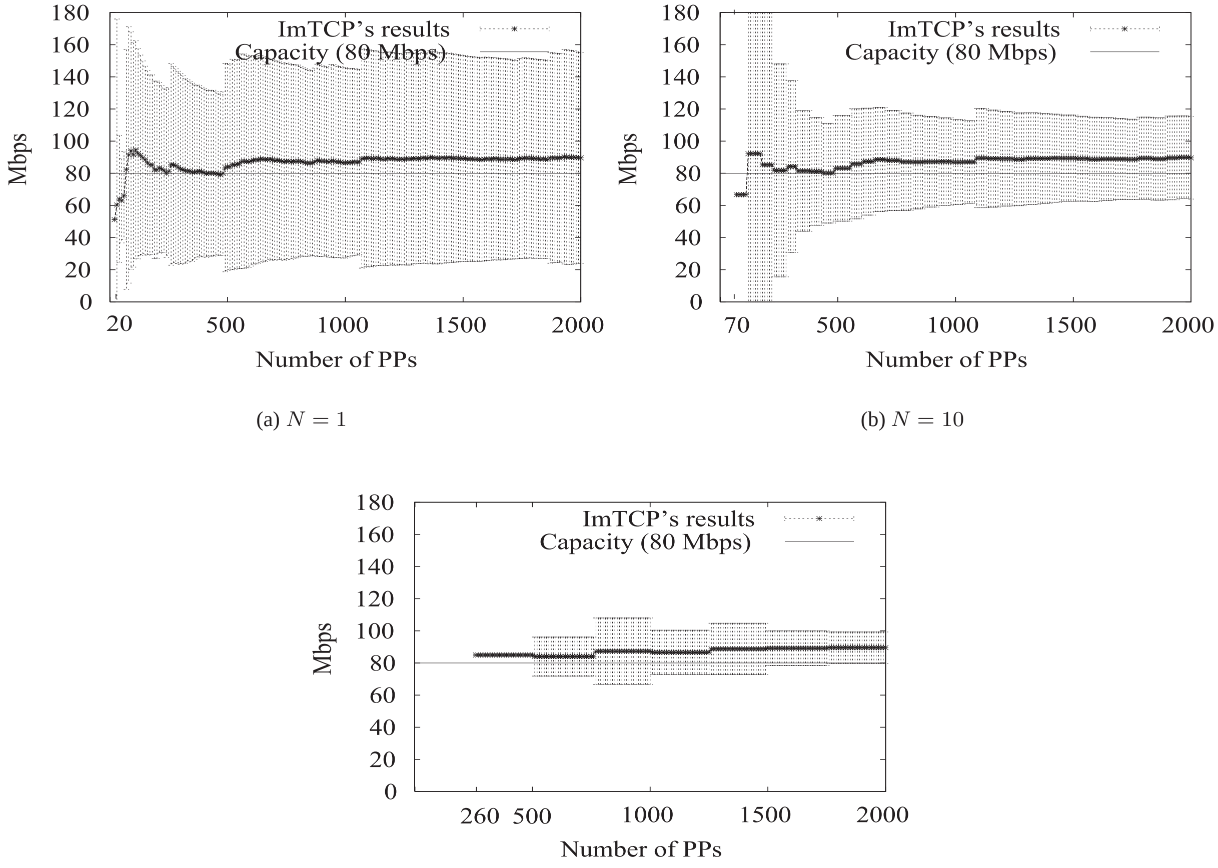
(b) $\lambda = 0.8$

Fig. 8 Measurement results for the proposed algorithm when Cross traffic 2 is 20 Mbps. $\lambda = 0.8$ gives more accurate results than $\lambda = 0.5$.

the changes of the capacity measurement results as the number of PPs sent for the measurement increases. The errors bars show the 90% confidence interval of the correspondent results. For the first some PPs, there is at most one "observation" is delivered therefore ImTCP can not calculate the confidence interval. In this case, the load on the bottleneck link is low, so the Equation (5) should normally be used. The setting $\lambda = 0.9$ does not allow the Equation (5) to be used so frequently and therefore leads to a bad result, that can be seen in large confidence intervals. We see that in this case $\lambda = 0.8$ (or lower than 0.8) is a better setting.

We next show the case when the capacity is 80 Mbps and the rate of Cross traffic 2 is set to 20 Mbps in Figure 8(a) ($\lambda = 0.5$) and 8(b) ($\lambda = 0.8$). In this case, the rate of the cross traffic is high, so Equation (6) should normally be used. Therefore, a small value of λ, such as 0.5, gives incorrect results for the capacity, and, again, $\lambda = 0.8$ is a good setting in this case. Thus, $\lambda = 0.8$ is a suitable setting for the two cases above, and we found that it is a good setting in many other cases. Therefore, in the following simulations, we used $\lambda = 0.8$. However, this is cannot be proved to be a suitable setting for all of the cases. Finding an optimal value for λ is one of our future goals.

In general, for longer connections, because the larger number of PPs is sent, ImTCP's results approach nearer to the right value. However, we can see that the measurement results of ImTCP are sometimes not exactly the right value (for example results in Figure 7(b)) even when the connection lasts for along time. The reason for this is that, we suppose that the amount of the traffic that cut in every PPs is the average value of that ($L = \delta(C - A)$), but the amount of traffic that cut in a certain PP is sometimes too large or too small in comparison with L. In these cases, the Sample cal-



(a) $N = 1$

(b) $N = 10$

(c) $N = 50$

Fig. 9 Measurement results for the proposed algorithm when N changes. N=10 is the best setting.

culated from these outstanding values (using Equation (6)) is far from the right value of Capacity, this leads to a slight inaccuracy in the final result of ImTCP.

- Value of N
  N is the number of samples to form an observation. We set the bottleneck link capacity to 80 Mbps and the rate of Cross traffic 2 to 40 Mbps. Figures 9(a), 9(b) and 9(c) show the measurement results when N is set to 1, 10 and 50, respectively. We can see that in Fig. 9(a), the results are yielded after only 20 PPs are sent, while in Fig 9(b), the number is 70, and in Figure 9(c), it is 260. The large confidence interval in Figure 9(a) indicates that a small value of N ($N = 1$) is not suitable. On the other hand, Figure 9(c) indicates that a value of N ($N = 50$), that is too large, is unsuitable as well, because in this case the time required for the results to be yielded is long. Figure 9(b) shows the results with the proposed setting ($N = 10$), which can provide faster and better results. Unlike the results in Fig 7 and 8, the measurement results in Fig. 9(a), 9(b) and 9(c) are all accurate from the first values. We can not expect the results to be better, as we have explained in Section 4.1 Therefore we could not see an improvement in the measurement accuracy in the figures, as the number of PPs increases.
  We use $N = 10$ for the following simulations. In fact, N must be set by applications or programs for that the measurement results are collected, depending on its uses. For example, if the application needs the measurement results to be updated in short intervals, it may choose a small value for N. In future works, we will examine in detail the requirement of the application or any program that uses the measurement results and create an outline for setting N based on it.

4.2 Comparision with CapProbe

We implement the CapProbe algorithm in TCP in order to compare the performance with the greatest possible impartiality. The difference from the original CapProbe algorithm proposed in [9] is that the packet size remains unchanged over the "runs" in the algorithm, because in TCP connections, changing the data packet size may have a bad effect on the TCP performance. The restriction on the packet size may be the reason for the poor performance of CapProbe in

Table 2 Number of PPs required for the first measurement result of the proposed algorithm and CapProbe. CapProbe requires more PPs to deliver the first result.

Capacity	Cross traffic 2	Proposed Algorithm	CapProbe
10 (Mbps)	1 (Mbps)	60 (PPs)	87 (PPs)
10	2	60	85
10	4	60	92
10	5	60	159

the following simulations. This means that CapProbe is not suitable for inline measurement.

4.2.1 Small capacity, low network load scenario

The capacity is set to 10 Mbps, and the rate of Cross traffic 2 is set to 4 Mbps. Figures 10(a) and 10(b) show the measurement results for the proposed algorithm and CapProbe, respectively. Both of the measurement results are good. Moreover, we can see that the results obtained by CapProbe have high accuracy, because when CapProbe successfully finds the PP in Case A, the capacity can be calculated exactly. Another advantage of CapProbe is that, compared with the proposed algorithm, CapProbe is simple because it requires no complicated calculations. Howerver, CapProbe only delivers a measurement result after sending a large number of PPs. Table 2 shows the number of PPs sent until the proposed algorithm and CapProbe deliver the first measurement result. Here, the capacity of the bottleneck is kept unchanged while the Cross traffic 2 is varied from 1 Mbps to 5 Mbps. The table shows that, CapProbe only delivers a measurement result after 85 PPs or more are sent. The required number of PP is larger as the network load increases. In contrast, the proposed algorithm delivers good measurement results faster, after sending 60 PPs.

4.2.2 Small capacity, high network load scenario

We next change the Cross traffic 2 to 9 Mbps to form a high network load environment. In this case, ImTCP still delivers good measurement results, as shown in Figure 11(a). On the other hand, CapProbe, as can be seen in Figure 11(b) introduces fewer results. It also delivers one incorrect measurement result. The reason for this is that, when the bottleneck link is crowded, many PPs are cut into by cross traffic so most of PPs are in Case-B. It is easy for CapProbe to mistake a Case-A PP for a Case-B PP.

4.2.3 Large capacity, high network load scenario

The capacity is set to 80 Mbps, and the rate of Cross traffic 2 is set to 60 Mbps. In a network with such a heavy load, the proposed algorithm can perform well (Figure 12(a)), whereas CapProbe can not deliver accurate results (Figure 12(b)), because, in this case, most of the PPs are cut into by other traffic so there are few Case-A PPs. In Figure 12(b) we also show the measurement results of CapProbe when the Cross traffic 2 is decreased to 50 Mbps. These measurement results are still far from the correct value. We believe that CapPobe will perform better if the size of PPs is adapted appropriately, instead of being unchanged in the simulations, but changing packet size is not suitable with inline measurement.

4.3 Comparision with Pathrate

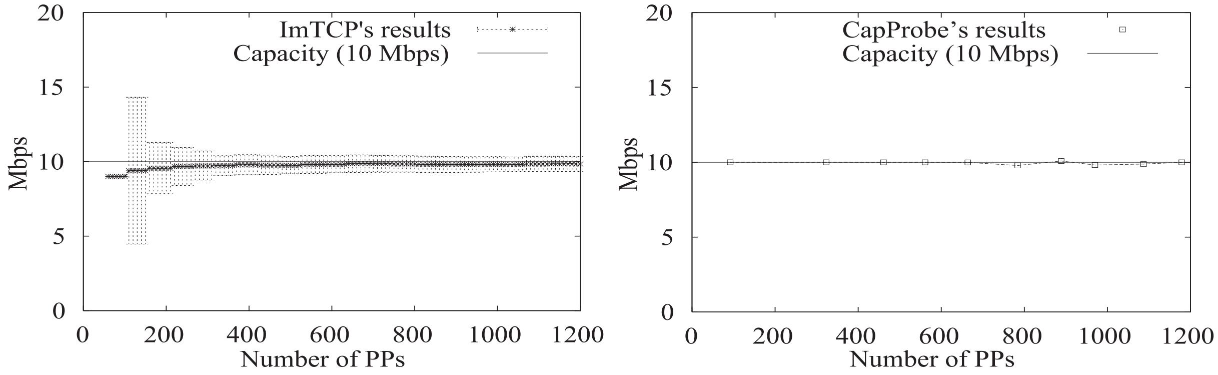
In order to accommodate the Pathrate algorithm into TCP, we use the interval of PPs delivered in ImTCP to form the histogram to be used in Pathrate. Pathrate also requires the measurement results of packet trains, referred to as the Average Dispersion Rate (ADR) in the Pathrate algorithm [7]. However, integrating the packet train into TCP is difficult because this has an adverse effect on the performance of TCP. Therefore, we perform the packet train measurement separately from TCP connection, in the same environment as that in the simulation with the ImTCP connection. The result of ADR is then used to find the measurement result for Pathrate.

We use the same network topology as that for the above-described simulations. The capacity is set to 80 Mbps and the transmission rate of Cross traffic 2 is variable. We show the case when the cross traffic contains mainly packets of small size, by randomly varying the packet sizes of the cross traffic within the range of 400 to 600 B, because in this environment the difference between the proposed algorithm and Pathrate appears clearly. The performance of the proposed program in this environment is also examined, and the measurement results are listed in Table 3. In this case, since most PPs are cut into by cross traffic packets, Pathrate should not work very well. On the other hand, the proposed can yields good measurement independent on the value of cross traffic. However, when the cross traffic is small, (Cross traffic 2 is 10Mb/s), many PPs do not stretched at the bottleneck link so they do not become Case-B PPs. Instead, they become Case C PPs due to the effect of Cross traffic 3. Because the number Case-B decreases, the measurement algorithm introduces larger confidence intervals.

We explain in detail the respective behaviors of these two algorithms in Figures 13(a) and 13(b). In Figure 13(a), the “Raw data” histogram indicates the measurement results calculated using Equation (1) that are used in Pathrate, and in Figure 13(b), the “Proposed method” histogram shows the “observation” results obtained using proposed algorithm, when the Cross traffic 2’s rate is 75 Mbps. In this case, Pathrate fails to deliver good measurement results because in this case number of Case-A PPs are fewer than Case-B PPs. This can be seen in some high peaks near 50 Mbps (while the correct value of capacity is 80 Mbps) in Figure 13(a). In contrast, the proposed algorithm can deliver good results, because the “observation” values always concentrate at the correct value of capacity, regardless of the network load.

4.4 Measurement in Web traffic environment

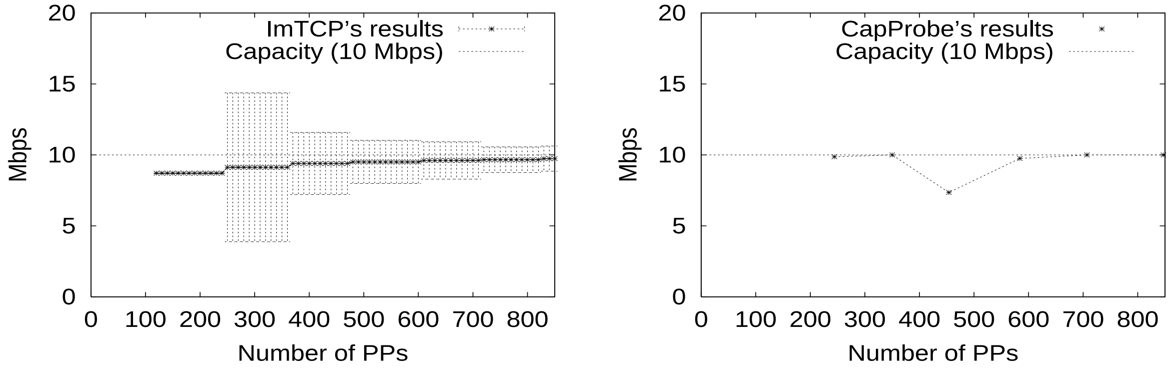
We finally investigate the measurement results for ImTCP in



(a) ImTCP

(b) CapProbe

Fig. 10 Measurement results for the proposed algorithm (ImTCP) and CapProbe in small capacity, low network load scenario. Both deliver accurate results.



(a) ImTCP

(b) CapProbe

Fig. 11 Measurement results for the proposed algorithm (ImTCP) and CapProbe in small capacity, high network load scenario. CapProbe can not deliver accurate results.

Table 3 Measurement results of ImTCP and Pathrate when Cross traffic 2 changes. The proposed algorithm can deliver accurate results in high-load network, in which Pathrate does not work well.

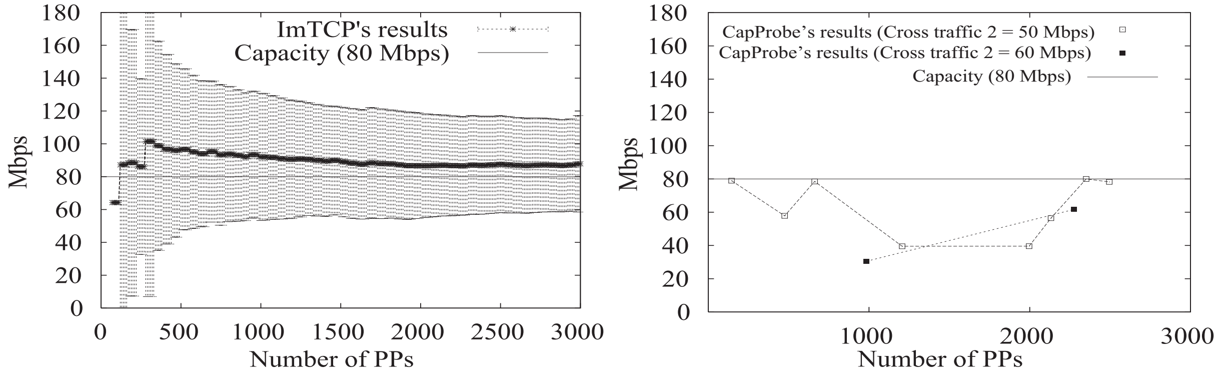
Cross traffic 2	ImTCP's results (90%confidence interval)	Pathrate
75 (Mbps)	79.35 (18.26)	49.00
60	80.24 (23.03)	48.00
40	78.32 (26.04)	80.00
10	81.57 (46.98)	80.00

the network model depicted in Figure 6 with the Cross traffic 1 and 3 turn off. Cross traffic 2 is now changed to Web traffic involving a large number of active Web document accesses. We use a Pareto distribution for the Web object size distribution. We use 1.2 as the Pareto shape parameter with 12 KBytes as the average object size. The number of objects in a Web page is eight. The capacity of the bottleneck link is set to 50 Mbps.

Figure 14(a) shows the measurement results for available bandwidth given by ImTCP. Also shown are the correct values of available bandwidth. We can see that the available bandwidth fluctuate frequently, because of the changes in the number of TCP connections in the crossing Web traffic. Figure 14(b) shows the measurement results for capacity also introduced by ImTCP in this case. The results are always approximately 50 Mbps, the correct value. The figure confirms that, even when the available bandwidth fluctuates frequently, ImTCP can deliver good measurement results for capacity.

5. Conclusion and future works

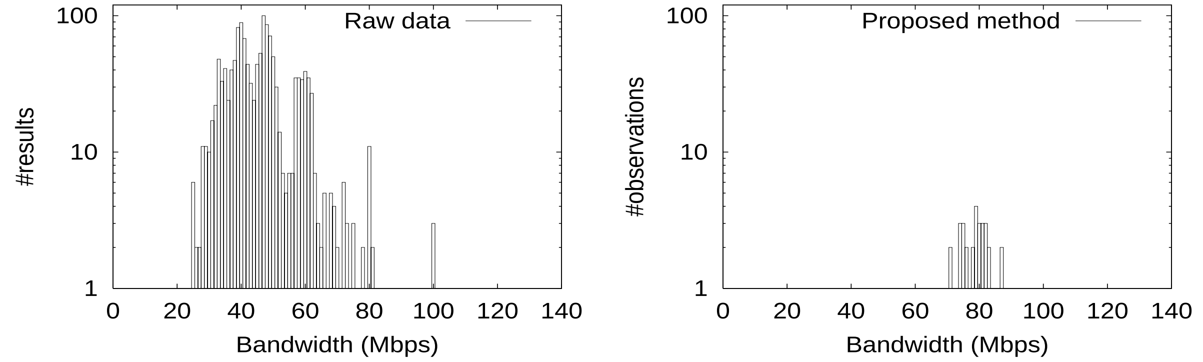
In this paper, we have proposed a new capacity measurement technique that is suitable for use in TCP connections. In contrast to existing techniques, the proposed mechanism uses available bandwidth information that is available in ImTCP, which enables the utilization of packet pairs that can not be used in existing techniques to calculate the capacity. The simulation results show that, the proposed tech-



(a) ImTCP

(b) CapProbe

Fig. 12 Measurement results for the proposed algorithm (ImTCP) and CapProbe in high network load scenario. CapProbe can not deliver accurate results.



(a) Data collected for Pathrate algorithm.

(b) Observation value calculated by the proposed algorithm

Fig. 13 Comparison of the histograms collected by Pathrate and the proposed algorithm in heavy load network (Cross traffic 2's rate is 75 Mbps).

nique can deliver measurement results quickly, even for a heavily loaded network, in which other techniques do not work well.

As our future works, we should find a proper setting for parameters such as λ and N. We are currently implementing ImTCP using the proposed technique on a FreeBSD system. We will also consider a bandwidth measurement algorithm that can be deployed at the TCP receiver.

References

[1] C. Man, G. Hasegawa and M. Murata, "An inline measurement method for capacity of end-to-end network path," in *3rd IEEE/IFIP Workshop on End-to-End Monitoring Techniques and Services (E2EMON 2005)*, May 2005.

[2] C. Man, G. Hasegawa and M. Murata, "Available bandwidth measurement via TCP connection," in *Proceedings of the 2nd Workshop on End-to-End Monitoring Techniques and Services E2EMON*, Oct. 2004.

[3] C. Man, G. Hasegawa and M. Murata, "ImTCP: TCP with an inline measurement mechanism for available bandwidth," *to appear in Computer Communications Special Issue: Monitoring and Measurements of IP Networks*.

[4] T. Iguchi, G. Hasegawa and M. Murata, "A new congestion control mechanism of TCP with inline network measurement," in *Proceedings of ICOIN 2005*, Jan. 2005.

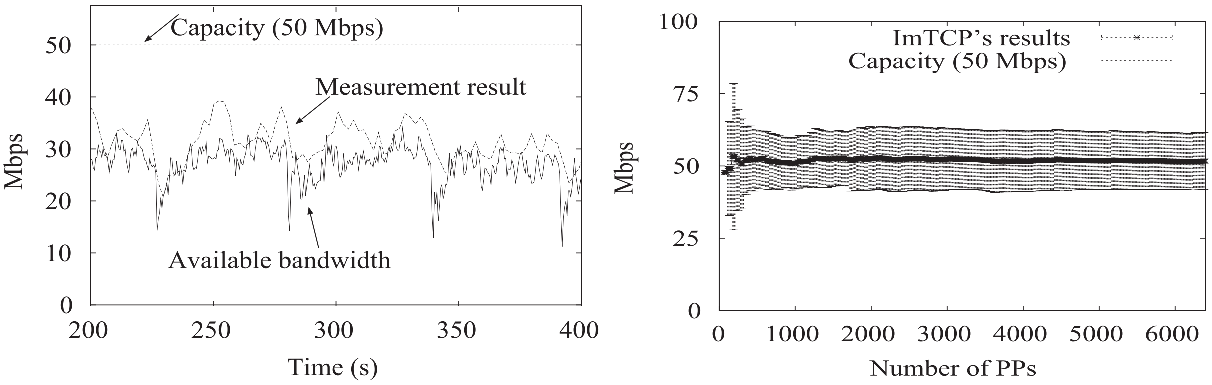
[5] M. Jain and C. Dovrolis, "End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput," in *Proceedings of ACM SIGCOMM 2002*, Aug. 2002.

[6] J.Strauss, D.Katabi and F.Kaashoek, "A measurement study of available bandwidth estimation tools," in *Proceedings of Internet Measurement Conference 2003*, Oct. 2003.

[7] C. Dovrolis, P. Ramanathan and D. Moore, "Packet dispersion techniques and capacity estimation," *IEEE/ACM Transactions on Networking*, vol. 12, pp. 963–977, Dec. 2004.

[8] K. Lai and M. Baker, "Nettimer: A tool for measuring bottleneck link bandwidth," in *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, Mar. 2001.

[9] R. Kapoor, L. Chen, L. Lao, M. Gerla and M. Sanadidi, "CapProbe: a simple and accurate capacity estimation technique," in *Proceedings of the 2004 Conference on Applications, Technologies, Archi-*



(a) Available bandwidth measurement results for ImTCP

(b) Capacity measurement results for ImTCP

Fig. 14 Measurement in Web traffic environment

tectures, and Protocols for Computer Communications, 2004.

[10] A. Shriram, M. Murray, Y. Hyun, N. Brownlee, A. Broido, M. Fomenkov and k claffy, "Comparison of public end-to-end bandwidth estimation tools on high-speed links," in *Proceedings of the 6th Passive and Active Measurement Workshop PAM 2005*, Mar. 2005.

[11] J. Navratil and R. Cottrell, "ABwE: A practical approach to available bandwidth estimation," in *Proceedings of the 4th Passive and Active Measurement Workshop PAM 2003*, Apr. 2003.

[12] N.Hu and P.Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 21, Aug. 2003.

[13] R.Anjali, C.Scoglio, L.Chen, I.Akyildiz and G.Uhl, "ABEst: An available bandwidth estimator within an autonomous system," in *Proceedings of IEEE GLOBECOM 2002*, Nov. 2002.

[14] S. Seshan, M. Stemm, and R. H. Katabi, "SPAND: Shared passive network performance discovery," in *Proceedings of the 1st Usenix Symposium on Internet Technologies and Systems (USITS '97)*, pp. 135–146, Dec. 1997.

[15] M.Gerla, B.Ng, M.Sanadidi, M.Valla, R.Wang, "TCP Westwood with adaptive bandwidth estimation to improve efficiency/friendliness tradeoffs," *Computer Communication*, vol. 27, pp. 41–58, Jan. 2004.

[16] K. Lai and M. Baker, "Measurering link bandwidths using a deterministic model of packet delay," in *Proceedings of ACM Sigcomm*, Aug. 2000.

[17] Bruce A. Mah, "Pchar," available at `http://www.ca.sandia.gv/~bmah/Software/pchar`.

[18] V. Jacobson, "Pathchar-A tool to infer characteristics of Internet paths," 1997. available at `http://www.caida.org/tools/utilities/others/pathchar/`.

[19] A. B. Downey, "Using pathchar to estimate internet link characteristics," in *Proceedings of ACM SIGCOMM*, 1999.

[20] M. Goutelle and P. Vicat-Blanc, "Study of a non-intrusive method for measuring the end-to-end capacity and useful bandwidth of a path," in *Proceedings of the 2004 IEEE International Conference on Communications*, 2004.

[21] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Proceedings of IEEE GLOBECOM 2000*, Nov. 2000.

[22] J. C. Hoe, "Improving the start-up behavior of a congestion control sheme for TCP," in *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, vol. 26,4, pp. 270–280, ACM Press, 1996.

[23] NS Home Page, available at `http://www.isi.edu/nsnam/ns/`.

[24] NLANR web site, available at `http://moat.nlanr.net/Datacube/`.