Estimating Available Bandwidth Using Multiple Overloading Streams

Minjian Zhang
Software College, Zhejiang University
Email: ben_zhang@zju.edu.cn

Chong Luo
Microsoft Research Asia
Email: cluo@microsoft.com

Jiang Li
Microsoft Research Asia
Email: jiangli@microsoft.com

Abstract— **Available bandwidth measurement is essential to applications running on the best-effort Internet. In this paper, we propose an available bandwidth measurement technique named MoSeab. The idea behind MoSeab is a direct probing technique based on a statistical model for network data transmissions. Differing from the other direct probing techniques, MoSeab does not require any *a priori* knowledge of network path. It has proven valid even when there are multiple bottlenecks. Simulation and real Internet experiments demonstrate the accuracy and robustness of MoSeab, and show its advantages over Spruce, PathChirp, and IGI. Moreover, its probing traffic control mechanism makes MoSeab a non-intrusive solution suitable to be integrated into various network applications.**

I. Introduction

Bandwidth estimation has been an active area of research for several years. It is critical for Internet applications to obtain bandwidth information so that they can adapt quality and bit rate of transmitted content. Examples of such applications include Internet content distribution, media streaming, multiparty conferencing, and Internet gaming.

The bandwidth property of a network path is usually defined by two metrics: capacity and available bandwidth. In literature, there is growing consensus on the definition of these two metrics. Consider a network path P as a sequence of first-come first-serve (FCFS) store-and-forward links $l_0, l_1, ...l_N$ that deliver data from source to destination. Suppose bandwidth, or maximal transmission rate, of link l_i is C_i, then capacity of the path is:

$$C_P = \min_{i=0,...N}\{C_i\} \tag{1}$$

We can see that capacity of a path is determined by the link with the minimal C_i, referred to as *narrow link* [5]. Over the last decade, the measurement of narrow link capacity has been extensively studied and a number of techniques and tools have been developed [10]-[12].

However, with the development of the Internet, network capacity is growing at quite rapid speed. Now, network capacity is usually on the order of hundreds or thousands of Megabytes, far beyond the bandwidth that an application may use. In such a situation, people's interests have slowly shifted to another bandwidth metric: available bandwidth (ABW), which indicates how much bandwidth can actually be used by an application.

Unfortunately, the measurement of available bandwidth is much more difficult than that of capacity. The inherent reason is that the available bandwidth is not a constant value, but a variable that changes over time. Given the starting time T and the duration ΔT, we can define available bandwidth of path P as:

$$A_P(T, \Delta T) = \min_{i=0,...N} \frac{1}{\Delta T} \int_T^{T+\Delta T} C_i(1 - u_i(t))dt \tag{2}$$

where $u_i(t) = 1$ if link l_i is in use at time t, and $u_i(t) = 0$ otherwise. As opposed to the narrow link which defines the capacity of a path, the link with the minimal spare bandwidth, referred to as *tight link*, defines the ABW of the path. In this paper, we will focus discussion on the measurement of available bandwidth.

Recently, research on available bandwidth measurement has been pursued along two directions. Some approaches [1]-[3] directly calculate the ABW based on a statistical model of the network path, while the others [4]-[8] estimate the ABW by iteratively probing a path with different rates. These two types of approaches are usually referred to as *gap model* and *rate model*.

The approaches based on the gap model are evolved from the packet-pair method previously used in capacity measurement. The basic idea is to send two probing packets in a short time interval Δ_{in} and get them queued at the bottleneck (narrow link in capacity measurement, or tight link in ABW measurement). If there is only one bottleneck along the network path, the packet pair will arrive at the receiver with the same spacing Δ_{out} as they left the bottleneck. The only difference between the capacity measurement and ABW measurement is that the former requires the packets to be queued back-to-back in the bottleneck while the latter requires the packet pair to contain some competing traffic between them. Assuming the tight link capacity is *a priori* known, one can calculate the available bandwidth by subtracting the cross traffic rate from the link capacity:

$$A_P = C_t - C_t \times (\frac{\Delta_{out} - \Delta_{in}}{\Delta_{in}}) \tag{3}$$

An important assumption of the gap model is that the tight link capacity is *a priori* known. It is often assumed that the tight link overlaps with the narrow link and the latter's capacity can be estimated by some well-established capacity measurement tools. However, this is not always the case on the real Internet.

1-4244-0355-3/06/$20.00 (c) 2006 IEEE

This assumption is relaxed in the rate model, which determines the ABW using a simple heuristic: the input rate of the probing traffic should be the same as the output rate unless it exceeds the available bandwidth of the path. In order to find the turning point where output rate starts to be smaller than input rate, methods in this category need to iteratively probe the network using various input rates, by either linear or binary search. A major drawback of rate model is that it introduces too much network overhead and may quite easily cause congestion.

Despite of a number of ABW measurement techniques, we have found that most method has certain drawbacks, in terms of robustness, accuracy, or non-intrusiveness. While some drawbacks are inherent to the algorithm, some others can actually be avoided by better design. In this paper, we propose an available bandwidth measurement solution *MoSeab*, which uses *Multiple overloading Streams to estimate available bandwidth*. MoSeab's key algorithm is based on the gap model, but our design has avoided many problems that are common to its kind. Real Internet experiments show that MoSeab constantly achieves better performance than PathChirp [6], Spruce [3] and IGI [2].

The rest of this paper is organized as follows. In Section II, we provide backgrounds on active ABW measurement, and discuss the pros and cons of existing techniques. Section III is devoted to our proposed solution MoSeab. The experimental results are presented in Section IV. Section V concludes the paper.

II. Backgrounds on Active Bandwidth Measurement

Available bandwidth measurement can be carried out by either passive or active means. Passive measurement monitors routers and collects detailed performance statistics. Based on these pieces of information, one can easily calculate ABW during time intervals. The limitation of this approach is the request for special access privileges to all routers along the path, which is usually not possible over the Internet.

Active measurement, on the other hand, does not rely on the access to underlying hardware. The basic idea is to inject a sequence of probing packets at the sender and estimate ABW by analyzing output at the receiver. Therefore, the active measurement technique can be divided into a probing phase and an analysis phase. Next, we will discuss in detail these two phases and revise the treatments for them in existing methods.

A. Probing Phase

The self-injected probing traffic usually takes the form of packet pair, packet train, or packet chirps. We summarized four probing patterns used in existing work, and depicted them in Fig. 1.

Type I is the same as the traditional packet pairs used in capacity measurement. The probing traffic of Spruce [3] takes this form. There are two parameters for this input: the intra-pair interval Δ_{in} and the inter-pair interval τ. While the selection of τ is specific to each method, Δ_{in} is usually set

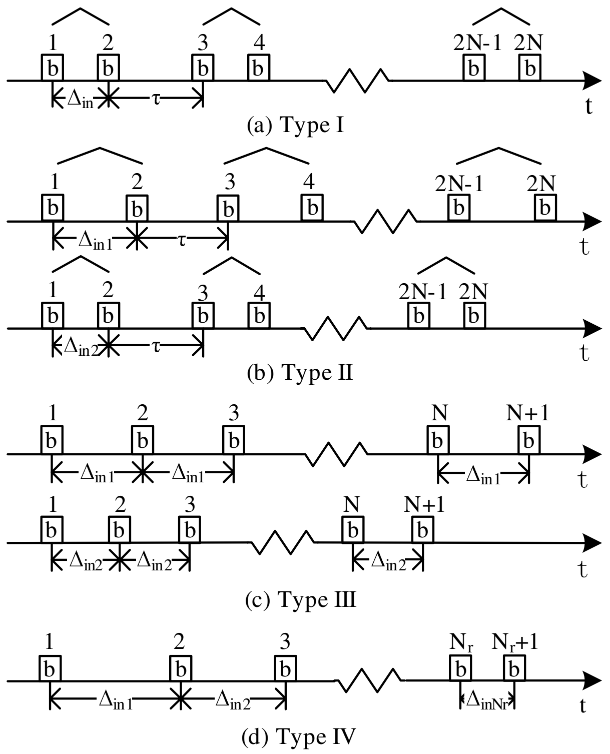

Fig. 1. Four types of probing traffic

to the transmission time of a probe packet on the bottleneck link. In Spruce, one of its main contributions is the use of exponentially distributed τ, which results in a Poisson sampling process. By choosing a large τ, Spruce can keep the probing overhead at a considerably low level, but at the cost of prolonged measurement time.

Type II is different from Type I in that each measurement is composed of multiple probing sequences with different intra-packet intervals. TOPP [4] belongs to this type. By choosing different Δ_{in}, TOPP probes the network with different input rates. The measurement of each rate is the mean of N samples, which are collected by $2N$ packets. The number of probing rates is determined by the predefined measurement region $[o^{min}, o^{max}]$ and step Δo. Hence, there will be $2N \times \lceil \frac{o^{max} - o^{min}}{\Delta o} \rceil$ packets for each measurement. This can be a huge data amount and measurement duration can be extremely length if the inter-pair interval is large enough to avoid congestion.

A more efficient way to collect N samples for a specific probing rate is to send a packet train of $N+1$ packets. This patten, classified as Type III in Fig. 1, is adopted by many methods such as IGI/PTR [2], Pathload [5], and Pathvar [8]. Compared to Type II, the number of packets required by Type III is reduced to $N_r(N+1)$, where N_r is the number of rates to be probed.

Type IV demonstrates the last probing patten, which is a sequence of exponentially spaced packets, known as "chirps." It is adopted by Delphi [1], PathChirp [6], and PathMon [7] for its great efficiency. As Fig. 1(d) shows, a single chirp of

$N_r + 1$ packets is sufficient to probe N_r rates. However, there is always a tradeoff between efficiency and robustness. Since there is only one sample for each probing rate, the stability of the measurement is impaired.

The probing traffic pattern in MoSeab belongs to type III. Packet train is more efficient than packet pairs, and is more robust than chirps. The only problem of packet train is the high probing rate, which tends to cause buffer overflow at the tight link. In MoSeab, we designed a buffer control mechanism that can adaptively adjust the packet train length according to probing rate and estimated link capacity. This mechanism is detailed in Section III.

B. Analysis Phase

The injection of probing traffic is only the first step of measurement. For both direct and iterative probing techniques, the analysis of the measurement data is a more critical task than collecting them. Though all the methods are intrinsically based on the same relationship between the input rate (gap) and the output rate (gap), the choice of the analyzing object does affect the correctness and robustness of the algorithm.

There are three types of analyzing objects in existing methods: I) the input and output gaps (Δ_{in} and Δ_{out}) between two successive packets; II) the input rate R_i and the output rate R_o of the probing stream; III) the relative one-way delays (OWD) or the queuing delays of a train of packets: $D_i = T_o - T_i$, where T_i and T_o are the input and output time of packet i.

According to our equation (3), most of the direct probing methods [2][3] belong to the first type. In IGI/PTR[2], the probing traffic is a train of evenly spaced packets. Since $\Delta_{out}^k = T_o^{k+1} - T_o^k$, the average of Δ_{out} becomes:

$$\overline{\Delta_{out}} = \frac{T_o^{N+1} - T_o^1}{N} \tag{4}$$

In this equation, only the arrival time of the first and the last packet is used.

As we have mentioned, TOPP [4] is an iterative probing method. It discovers the turning point by solving the linear equation $R_i/R_o = \alpha + \beta R_i$, and therefore belongs to type II. In TOPP, although the outcome of a probing sequence is a series of time stamps for N packet pairs ($2N$ packets), only one R_o value is calculated:

$$R_o = \frac{1}{N}\sum_{k=1}^{n} \frac{b}{\Delta_{out}^k} = \frac{1}{N}\sum_{k=1}^{n} \frac{b}{T_o^{2k} - T_o^{2k-1}} \tag{5}$$

In Type I and II, either Δ_{out} or R_o are averaged among multiple measurements. The differences between successive measurements are either neglected or counteracted.

Type III provides a contrast to the other two types. It captures the relative OWD for every single packet, and uses the whole series of D_i for turning point discovery. Pathload[5], PathChirp[6] and PathMon[7] all belongs to this type. The difficulty of these methods, however, is how to correctly extract useful information from a mass of OWD measurements and eliminate noise.

Pathload is the canonical example of using the OWD information. As it is an iterative probing method, it only needs to judge whether the OWD sequence indicates an upward trend. To this end, Pathload introduces PCT (pairwise comparison test) and PDT (pairwise difference test) measurements, and the upward trend is detected if PCT>0.55 or PDT>0.4. However, PCT is sensitive to network jitters and PDT becomes inaccurate when buffer overflow occurs.

MoSeab also makes use of the OWD information. Differing from iterative probing methods, MoSeab needs to calculate the exact value of OWD increase between successive packets. The method is detailed in Section III.

III. MoSeab - Multiple Overloading Streams to Estimate Available Bandwidth

In this section, we introduce our proposed solution MoSeab. The key concept of MoSeab is a direct probing algorithm which calculates the available bandwidth from the input rate and OWD information. MoSeab does not assume *a priori* knowledge of tight link capacity. It is proved to be valid even under the presence of multiple tight links.

A. Single-Link Model

We start from a single-link model with fluid competing traffic at a constant rate. Let C be the capacity of that link and R_c be the rate of the competing traffic, then the available bandwidth of the link is $A = C - R_c > 0$. This model is actually used by all existing bandwidth measurement techniques. On the Internet, this single-link model can be applied to any paths with a single tight link.

The working conditions of the direct probing technique involve the intra-packet interval Δ_{in} being small enough so that $b/\Delta_{in} > A$, where b is the size of the packet. When the probing train is used, the interval between all the successive packets are the same, therefore, $R_i = b/\Delta_{in} > A$. Now, we consider the situation when a probing train of rate R_i is injected into a single-link path with available bandwidth A, and $R_i > A$.

Given a fixed packet size b, Δ_{in} can be determined by: $\Delta_{in} = b/R_i$. Under the fluid assumption, the competing traffic arrived at the tight link during the time interval of Δ_{in} is $R_c\Delta_{in}$. Hence the total amount of the traffic arrived at the tight link during Δ_{in} is $b + R_c\Delta_{in}$. Recall that the maximal transmission speed of the tight link is C, since $b + R_c\Delta_{in} = (R_i + R_c)\Delta_{in} > C\Delta_{in}$, the extra traffic will be queued at the tight link, and the queue increases by

$$\Delta q = (R_i - A)\Delta_{in} = b \cdot \frac{R_i - A}{R_i} \tag{6}$$

Therefore, the OWD increase between two successive packets can be calculated by the following equation:

$$\Delta D = \frac{\Delta q}{C} = \frac{b}{C}\frac{R_i - A}{R_i} \tag{7}$$

In our system, the packet size b is fixed to 500B. Since the input rate R_i is set by the sender, and ΔD can be measured at

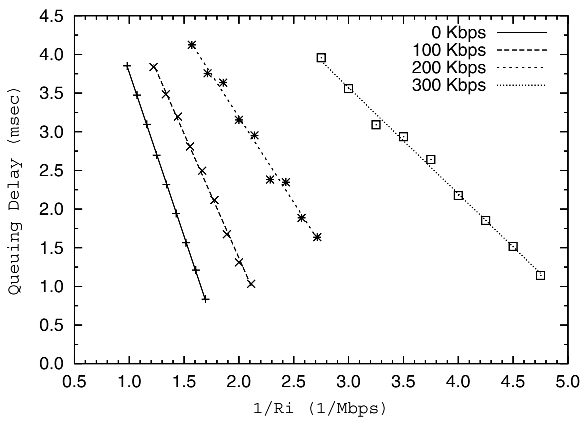

Fig. 2. Relationship between the input rate and the queuing delay

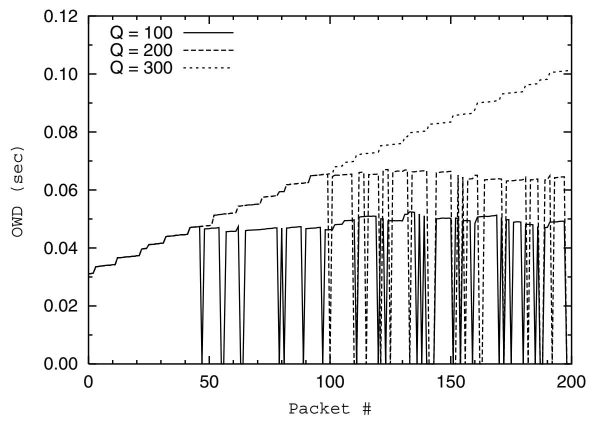

Fig. 3. OWDs under different queue sizes

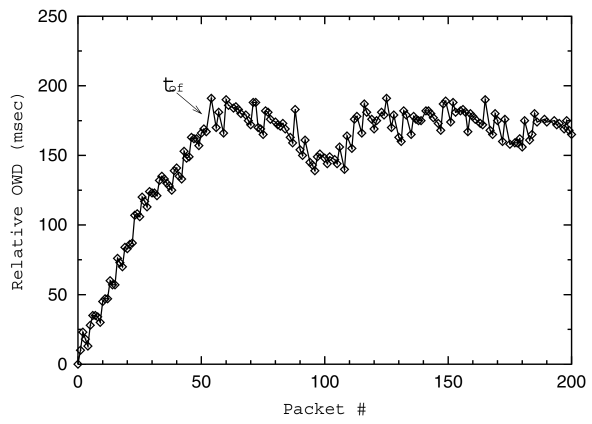

Fig. 4. Real OWD data when buffer overflow occurs at t_{of}

the receiver, there are only two unknown variables A and C in this equation. Theoretically, we can solve them using only two measurements with different R_i. Yet if multiple measurements are available, we can further improve the accuracy of the estimation by rewriting the equation as:

$$\Delta D = \frac{b}{C} - \frac{b \cdot A}{C} \cdot \frac{1}{R_i} = \alpha - \beta \cdot \frac{1}{R_i} \tag{8}$$

We can see that ΔD has a linear relationship with the reciprocal of the sending rate R_i. With multiple measurements of $(\Delta D, R_i)$, we can first estimate α and β with Least Squares Fitting (LSF), and then the available bandwidth and the capacity of the tight link can be estimated by:

$$A = \frac{\alpha}{\beta}, C = \frac{b}{\alpha} \tag{9}$$

Fig. 2 provides an example. It contains data collected from an Internet experiment over ADSL lines. The capacity of the measured link is approximately 560Kbps. The four sets of experiments are performed when there are 0, 100Kbps, 200Kbps, and 300Kbps self-induced cross traffic. We can see that the measurements are well fitted to a line as (8) describes.

B. Probing Phase Control

For packet-pair probing traffic the average input rate is determined by both the intra-packet and the inter-packet intervals:

$$R_i = \frac{2b}{\Delta_{in} + \tau} \tag{10}$$

Although Δ_{in} is small, the probing rate can still be kept low if a large τ is used. However, for packet train probing, R_i is solely determined by Δ_{in}, which is small enough to cause buffer overflow.

Buffer overflow is harmful since it affects OWD measurements, and therefore impairs the accuracy of ABW estimation. Fig. 3 and Fig. 4 demonstrate OWD changes when buffer overflow occurs in both simulation and Internet experiment. Fig. 3 contains three sets of data collected under different queue sizes. All the other settings, including link capacity, available bandwidth, and input rate, remain the same during the three test runs. The turning point where the OWD stops increasing indicates that the buffer is full, and the zero value of OWD implies a loss packet. OWD data collected on the real Internet, as Fig. 4 shows, offer a similar result.

Another negative impact of buffer overflow is that it leads to packet losses of the competing traffic. An applicable ABW measurement technique should be non-intrusive, avoiding unrecoverable impacts to the competing traffic during the measurement. In MoSeab, this is achieved by adjusting the packet train length based on the input rate and the historical information of link capacity and ABW.

Let C and B be the capacity and the buffer size of tight link P. The maximal queuing delay occurs when the buffer is full: $D_{max} = B/C$. When we probe this link with a rate R_i, which is larger than the available bandwidth A, the traffic that gets queued during time interval t is $(R_i - A) \cdot t$. Assume that the buffer is empty before the measurement, and that the packets are dropped using the drop-tail strategy, the maximal time that the probing traffic may last is:

$$T_{max} = \frac{D_{max} \cdot C}{R_i - A} \tag{11}$$

Therefore, with the historical D_{max} information and the previous estimation of C and A, we can estimate the maximal number of packets N_{max} that the sender can inject with rate

R_i:

$$N_{max} = \left\lfloor \frac{D_{max} \cdot C \cdot R_i}{b(R_i - A)} \right\rfloor \quad (12)$$

In MoSeab, the packet train length is the minimum of 100 and N_{max}.

C. Queuing Delay Estimation

MoSeab estimates the ABW with two or more $(\Delta D, R_i)$ measurements. As R_i is determined by the sender, ΔD becomes the only parameter that needs to be measured. Since packet train is used in our algorithm, ΔD can be taken on as the increasing rate of the relative OWD. In Moseab, we apply the Least Median of Squares Fitting (LMS) to estimate the ΔD. The benefit of using LMS instead of piece-wise average or Least Squares Fitting (LSF) is the robustness. By using LMS, we are able to eliminate up to 50% outliers, which may be caused by network jitters or context switch in end system.

Note that we are only interested in the relative OWD of the packets so the system clock at the sender and the receiver need not be synchronized.

D. Multiple Tight Links

Under the assumption of FCFS scheduling and random dropping policy of packets at buffer overflow, our methods remains valid in the presence of multiple tight links. In this subsection, we provide proof under the condition that the number of tight links N_t equal to 2. The extension to the cases when $N_t > 2$ is straightforward.

Let link l_u and l_v be the only two tight links in path P, whose capacities are C_u and C_v. Let A be the available bandwidth of these two paths, and there exists a positive number δ which satisfies $A + \delta < minA_i$, $i \neq u, v$. Considering a probing stream of constant rate R_i ($A < R_i < A + \delta$) sent from the sender to the receiver, passing through the tight l_u and then l_v. According to (8), when it leaves the first bottleneck l_u, the increased OWD between two successive packets is:

$$\Delta D_u = \frac{b}{C_u} - \frac{b \cdot A}{C_u} \cdot \frac{1}{R_i} \quad (13)$$

As R_i is smaller than the available bandwidth of any other links along the path, the transmission rate of the probing train remains the same until it reaches the second bottleneck l_v. The input rate of l_v is the output rate of link l_u:

$$R_{i_v} = R_{o_u} = C_u \cdot \frac{R_i}{R_i + (C_u - A)} \quad (14)$$

When the stream exits l_v, the OWD difference between two successive packets is further enlarged by ΔD_v:

$$\Delta D_v = \frac{b}{C_v} - \frac{b \cdot A}{C_v} \cdot \frac{1}{R_{i_v}} \quad (15)$$

Substituting (14) into (15), we have:

$$\Delta D_v = \frac{C_u - A}{C_v} \cdot \left(\frac{b}{C_u} - \frac{b \cdot A}{C_u} \cdot \frac{1}{R_i} \right) \quad (16)$$

Therefore, the OWD difference observed at the receiver becomes:

$$\Delta D = \Delta D_u + \Delta D_v = c \cdot \frac{b}{C_u} - c \cdot \frac{b \cdot A}{C_u} \cdot \frac{1}{R_i} = \alpha - \beta \frac{1}{R_i} \quad (17)$$

where $c = 1 + (C_u - A)/C_v$. It is a constant as long as the capacity and the available bandwidth of both tight links does not change during the measurement. We can see that our calculation of A ($A = \alpha/\beta$) is still correct since α and β are increased by the same constant factor.

E. A Complete Solution

MoSeab is not only an available bandwidth measurement algorithm, but a complete solution that can be integrated into various network applications. It is composed of iterative probing and direct calculation phases.

Since MoSeab does not assume *a priori* knowledge of network path, it starts to probe the network from an initial rate R_{min}, and doubles the probing rate in each consequent run. This process stops when $R > A$, which can be judged from an upward trend of OWD. In our implementation, R_{min} is set to 200Kbps, and the initial packet train length is set to 100. An increasing OWD trend is reported if $\Delta D > \delta$. In our implementation, $\delta = 50\mu s$.

After the iterative probing phase, we can obtain a rough estimation of the ABW as:

$$\widetilde{A} = R/\sqrt{2} \quad (18)$$

As $R > A$ and $R/2 < A$, $\widetilde{A}$ satisfies:

$$\frac{1}{\sqrt{2}} A < \widetilde{A} < \sqrt{2} A \quad (19)$$

Then, in the direct probing phase, we send four trains of probing packets, whose rates are $114\%\widetilde{A}$, $133\%\widetilde{A}$, $160\%\widetilde{A}$, and $200\%\widetilde{A}$ respectively. The purpose of choosing these four rates is to make $1/R_i$ evenly spaced, so that the estimation of A can be equally sensitive to the measurement errors of the four sequences. Besides, these four rates ensures that at least two of them are greater than the actually ABW even in the extreme case that $A = \sqrt{2}\widetilde{A}$.

Given the frequency f, MoSeab makes a measurement for every $\frac{1}{f}$ seconds. Taking the fact that the ABW does not change much during a short time interval, the measurement of the previous round can be looked on as the initial estimation of the next round: $\widetilde{A}_{n+1} = A_n$.

In the rare case that $\widetilde{A}_{n+1} > 160\% A_n$, the four probing sequence fails to provide enough information for solving equation (9). In such a case, if the OWD of the sequence with the maximal probing rate ($R = 200\%\widetilde{A}$) shows an upward trend, we can try to solve the equation with a previous estimation of C. Otherwise, $\widetilde{A}$ has to be re-searched by an iterative process identical to that in the initial phase.

IV. Experimental Results

We evaluated MoSeab's efficiency in simulation and Internet experiments.

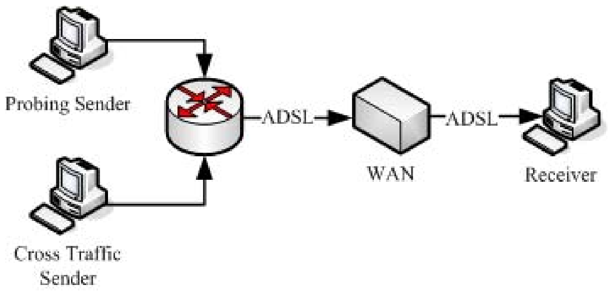

Fig. 5. Configuration of the real Internet experiments

A. Internet Experiments

As the MRTG data are not available in our real Internet experiments, we can only evaluate MoSeab through differential test (D-test) [3] and through comparisons with the other methods, including: Spruce[3], IGI[2], and PathChirp[6].

The experiment environment is configurated as shown in Fig. 5. There are three end systems in this configuration: the sender and the receiver of the measured path, denoted as S and R, and a cross traffic generator G connected to the same router as S. All these end systems are connected to the Internet through ADSL lines with an uplink capacity of approximately 560Kbps, as estimated by pathrate [12].

Four sets of experiments were conducted: one without cross traffic, and the other three with the cross traffic of 100Kbps, 200Kbps, and 300Kbps. The cross traffic is a stream of constant bit rate (CBR) UDP packets generated by Iperf [13]. The cross traffic rate is adjusted by changing the inter-packet interval and the packet size is maintained at 300B. For each configuration, we did 10 experiments for every evaluated tool. Hence, 40 measurements of ABW estimation, measurement time, and network overhead are collected for each method. Fig. 6 shows the comparison among the four methods.

The x-axis of this figure is $R_c + A_e$, where R_c is the injected cross traffic rate and A_e is the ABW estimated by each method. The y-axis is the Cumulative Distribution Function (CDF) of the measurement. We can see that the average estimation of MoSeab is very close to that of Spruce, which has been proven to be more accurate than Pathload and IGI in [3]. And the standard deviation of MoSeab, which is only 6.57Kbps, is less than half of that of Spruce (13.13Kbps).

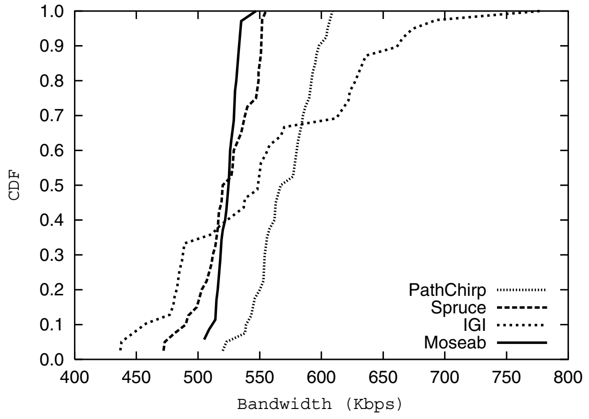

Fig. 6. Comparison among Spruce, IGI, PathChirp, and MoSeab

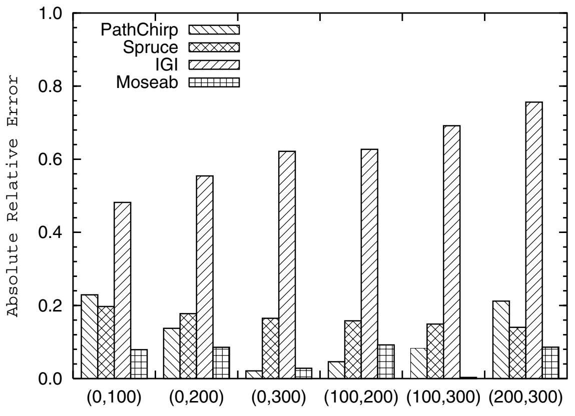

Fig. 7. D-test results of Spruce, IGI, PathChirp, and MoSeab

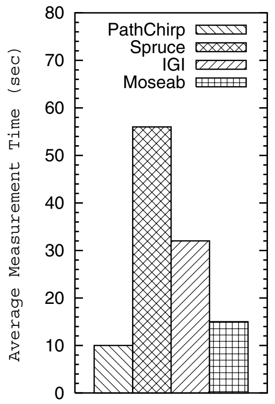

Fig. 8. Average measurement time

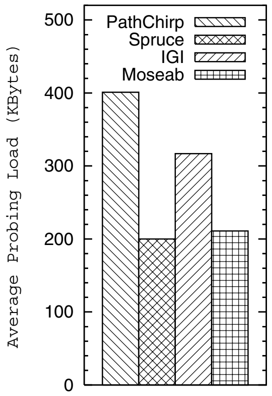

Fig. 9. Average probing load

We also did D-tests for each method and compared the results in Fig. 7. let R_c^i and R_c^j be the cross traffic rate in measurement M_i and M_j, and A_i and A_j be the estimated ABW. Then the relative error E_r between these two measurements is defined as:

$$E_r = \left| \frac{(A_j - A_i) - (R_c^j - R_c^i)}{(R_c^j - R_c^i)} \right| \tag{20}$$

We can see from the figure that MoSeab has the smallest relative errors in comparison with the other three methods. The average relative error of MoSeab is only 3.36%, while those of Spruce, IGI, and PathChirp are 16.44%, 62.21%, and 12.14%.

Fig. 8 and Fig. 9 compare the average measurement time and the probing load of the four evaluated tools. Spruce takes the longest measurement time since it needs to send 100 pairs of packets at a considerably low rate. The binary search manner of IGI costs heavy network load and long convergence time. PathChirp is the fastest method for the chirp style probing traffic. However, the probing load is extremely high. Comparing to other methods, MoSeab takes considerably short time to converge while keeping the network overhead

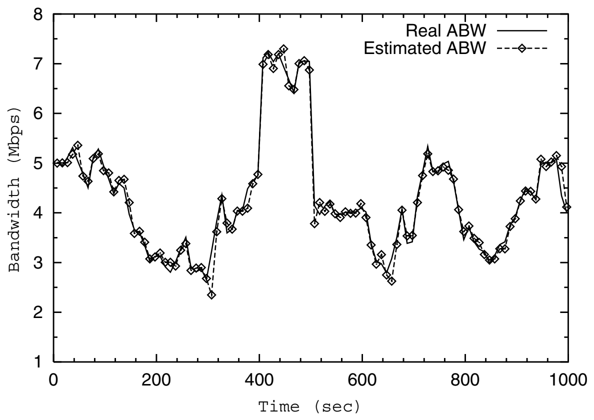

Fig. 10. ABW estimation results of MoSeab

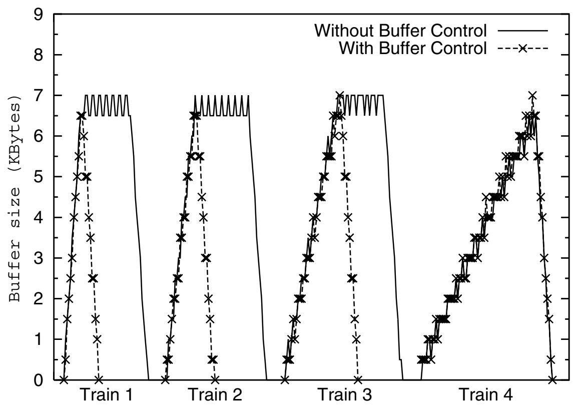

Fig. 11. Buffer size control

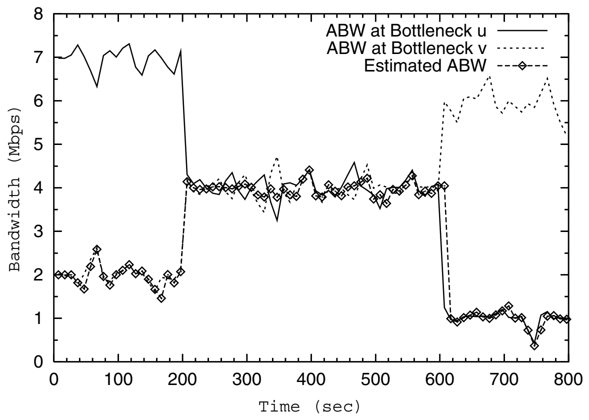

Fig. 12. ABW estimation under multiple bottlenecks

at a quite low level. In these two figures, the measurement time and the network load occurred in the iterative probing phase are not included; they are one-off costs for a series of measurements.

B. Simulations

The simulation experiments allow us to access the router data, so that we can evaluate the accuracy of our ABW estimation, and the effect of the buffer control mechanism. In this section, the experimental results of the NS-2 simulations are discussed.

In the first experiment shown in Fig. 10, we examine the performance of MoSeab in a single-link path. The available bandwidth at time t is generated by the following function:

$$ABW(t) = b(t) + d_1(t) + d_2(t) \tag{21}$$

where $b(t)$ is the base function, and $d_1(t)$, $d_2(t)$ are two disturbances at different scales. In order to examine MoSeab under different ABW changing conditions, $b(t)$ is designed as a segmented function: 1) constant ABW during $[0, 100]$, $[200, 300]$, $[400, 500]$, and $[500, 600]$; 2) linear increasing and decreasing during $[100, 200]$ and $[300, 400]$; 3) abrupt change at 400 and 500; 4) sine wave during $[600, 1000]$. $d_1(t)$ is a random disturbance at relative large time scale. For every 23.4 seconds, a random value $r_1(t)$ between -0.5MB and 0.5MB is generated and is added to the base line. Then, for every 0.497 second, $d_2(t)$ contributes a smaller disturbance in range [-50K,50K].

MoSeab measures the ABW every 10 seconds. The four probing sequences are sent with 1 second interval. Hence, with the smaller scale disturbance, the ABW changes during a single measurement. From Fig. 10, we can see that MoSeab achieves high accuracy and is alert to network changes.

Next, we conducted experiments to validate our buffer control mechanism. During the experiment, ABW was kept static and the buffer size was fixed to 7Kbps. We performed two test runs: in the first run, a fixed number of 100 packets were sent in each probing train; in the second run, our buffer control mechanism was applied. In each measurement, we sent four probing trains with the rates of $200\%\widetilde{A}$, $160\%\widetilde{A}$, $133\%\widetilde{A}$, and $114\%\widetilde{A}$.

In Fig. 11, we compare the typical measurements of these two approaches. The four strikes correspond to the four probing trains. In the first test run, which is described by the solid line, the probing packets continue to arrive after the buffer is full. We can see that buffer overflow lasts for quite a long time, and during this period, the queue size fluctuates between 6.5Kbps and 7.0Kbps. This is because that the packet size is 500B. Hence when a packet is sent out, the queue becomes 6.5Kbps before the next packet arrives.

However, if we apply our buffer control algorithm, we can estimate that the maximal number of probing packets to be 29, 35, 69, and 114 for the input rates of $200\%\widetilde{A}$, $160\%\widetilde{A}$, $133\%\widetilde{A}$, and $114\%\widetilde{A}$. As a result, no buffer overflow is observed in the second test run, which is described by the dotted lines with crosses.

We have proved in Section III that our method remains valid even when there are multiple bottlenecks. Here, we present the experimental results that support our judgment. In this experiment, the measured path consists of two links l_u and l_v. At time 200, 600, and 800 seconds, the ABW of l_u changes from $7M$, $4M$, to $1M$ and the ABW of l_v changes from $2M$, $4M$, to $6M$. Therefore, during time interval $[200, 600]$, there are two bottlenecks in the path. Disturbances are added in the same way as the single-link experiment. From Fig. 12, we can see that MoSeab can accurately measure the ABW in the presence of two bottlenecks.

V. Conclusion

In this paper we proposed an active available bandwidth measurement technique named MoSeab. It is a direct probing technique that calculates available bandwidth from the relationship between input rate and increased output delay. MoSeab does not assume *a priori* knowledge of a network path, and remains valid when multiple bottlenecks exist.

We also studied the four different patterns of the probing traffic, and discussed their pros and cons. According to these discussions, we adopted packet train as our probing traffic, and designed an adaptive buffer control mechanism which can effectively avoid buffer overflow.

The real Internet experiments demonstrated the efficiency of MoSeab, through the comparison with three other techniques known as Spruce, IGI, and PathChirp. NS-2 simulations showed that MoSeab was accurate, robust, and non-intrusive. We believe that MoSeab is suitable to be integrated into various network applications.

Acknowledgment

The authors would like to thank Dwright Daniels for proofreading the paper.

* Minjian Zhang was a visiting student of Microsoft Research Asia. The work presented in this paper was carried out in Microsoft Research Asia.

References

[1] V.J. Ribeiro, M. Coates, R.H. Riedi, S. Sarvotham, and R.G. Baraniuk, *Multifractal cross traffic estimation*. Procedings of ITC Specialist Seminar on IP Traffic Measurement, September 2000.

[2] N. Hu and P. Steenkiste, *Evaluation and characterization of available bandwidth probing techniques*. Selected Areas in Communications, IEEE Journal on Volume 21, Issue 6, August 2003.

[3] J. Strauss, D. Katabi, and F. Kaashoek, *A measurement study of available bandwidth estimation tools*. Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement, October 2003.

[4] B. Melander, M. Bjorkman and P. Gunningberg, *A new end-to-end probing and analysis method for estimating bandwidth bottlenecks*. Global Telecommunications Conference, 2000. GLOBECOM '00. IEEE Volume 1, December 2000.

[5] M. Jain and C. Dovrolis, *End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput*. IEEE/ACM Transactions on Networking (TON), Volume 11 Issue 4, August 2003.

[6] V.J. Ribeiro, R.H. Riedi, R.G. Baraniuk, J. Navratil, and L. Cottrell, *pathChirp: Efficient available bandwidth estimation for network paths*. In Passive and Active Measurements Workshop, April 2003.

[7] D. Kiwior, J. Kingston and A. Spratt, *PathMon, a methodology for determining available bandwidth over an unknown network*. Advances in Wired and Wireless Communication, 2004 IEEE/Sarnoff Symposium on 26-27, April 2004.

[8] M. Jain and C. Dovrolis, *Network performance measurements: End-to-end estimation of the available bandwidth variation range*. Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, June 2005.

[9] M. Jain, C. Dovrolis, *Lessons learned: Ten fallacies and pitfalls on end-to-end available bandwidth estimation*. Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, October 2004.

[10] V. Jacobson, *pathchar: A Tool to Infer Characteristics of Internet Paths*. ftp://ftp.ee.lbl.gov/pathchar/, Apr.1997

[11] R.L. Carter and M.E. Crovella, *Measuring Bottleneck Link Speed in Packet-Switched Networks*. Performance Evaluation, vol.27,28, pp.297-318, 1996.

[12] C. Dovrolis, P. Ramanathan, and D.Moore, *What do Packet Dispersion Techniques Measure?*. in Proceedings of IEEE INFOCOM, April 2001. http://www.cc.gatech.edu/fac/Constantinos.Dovrolis/pathrate.html

[13] http://dast.nlanr.net/projects/Iperf/