ELSEVIER

Computer Communications 25 (2002) 1699–1710

computer communications

www.elsevier.com/locate/comcom

Accurate bandwidth measurement in xDSL service networks

Liang Cheng*, Ivan Marsic

Department of Electrical and Computer Engineering and the CAIP Center, Rutgers, The State University of New Jersey, Piscataway, NJ 08854-8088, USA

Received 14 May 2001; revised 6 February 2002; accepted 27 February 2002

Abstract

In xDSL service networks, accurate bandwidth measurement is essential for network management and traffic engineering, such as isolating line faults and verifying guaranteed quality of service specifications, where ISP and xDSL service providers may be involved. Existing Internet bandwidth measurement techniques are not suitable because link asymmetry and ATM traffic shaping in xDSL deployments impact the accuracy of their measurements. In this paper, we present a novel stepwise algorithm for bandwidth measurement that performs accurately in xDSL service networks. The stepwise algorithm requires a smooth traffic generator since otherwise the accuracy would be affected by the ATM traffic shaping. A multi-packet technique is used to avoid explicit packet filtering. The scheme is lightweight in terms of implementation complexity and probe traffic overhead. The accuracy of the algorithm is demonstrated by laboratory and field experiments. The results show that the measurement errors in both upstream and downstream cases are within the range of 5% in a commercial ADSL service network tested under the following ATM network configurations: CBR, rt-VBR, and UBR service categories.

Keywords: xDSL networks; Bandwidth measurement; Asymmetric networks; ATM traffic shaping

1. Introduction

The growth in deployment of Internet applications requiring high bandwidth has created a need for faster local loops for users connecting to the Internet through a dial-in Internet Service Provider (ISP). In most cases, a serial modem solution, which offers a maximum bandwidth of up to 56 Kbps, is not sufficient. Therefore, various technologies, such as Digital Subscriber Line (DSL) and cable modems, have emerged to achieve high-speed local-loop connectivity to the Internet. Local phone companies in the United States have been installing commercial xDSL lines since 1998. xDSL refers to different variations of DSL, such as Asymmetric DSL (ADSL), Symmetric DSL (SDSL), High bit-rate DSL (HDSL), Rate Adaptive DSL (RADSL), and Very High-speed DSL (VDSL) [22]. xDSL has several advantages compared to other access technologies [9]: (i) its initial cost can be predicted and spent only when a customer requests the service, (ii) it requires no change to the central office switch software, (iii) it can interface with a number of different premises arrangements, (iv) a DSL line can simultaneously carry both data and voice signals with the data part of the line continuously connected. In addition, xDSL can be used for residential users, Small Office, Home Office (SOHO) users, and large organizations because of its wide range of service rates. For example, ADSL enables receiving data at rates up to 6.1 Mbps (of a theoretical 8.448 Mbps). Typical ADSL connection services provide from 512 Kbps to 1.544 Mbps downstream (from service provider to customer, or from Central Office to Customer Premises) and about 64–128 Kbps upstream. The downstream speeds of VDSL range from 13 to 55 Mbps depending on distance, while its upstream speeds start at 1.5 Mbps and go up to about 26 Mbps [9].

The xDSL deployment architecture in the ADSL case is abstracted in Fig. 1. The customer's premises equipment is an ADSL Termination Unit—Remote (ATU-R) or an ADSL modem. An ADSL Termination Unit—Central Office (ATU-C) is located at a Central Office end of the phone company that provides the xDSL service. The ATU-Cs are grouped into a DSL Access Module (DSLAM) unit that terminates the multiplexed traffic into an ATM switch. In some cases, an IP router can be used instead of the ATM switch depending on budget constraints. However, the performance aspects of ATM networks and services justify the use of ATM switches [9]. In the field experiments

* Corresponding author. Tel.: +1-732-445-0660; fax: +1-732-445-4775.
E-mail addresses: chengl@caip.rutgers.edu (L. Cheng), marsic@caip.rutgers.edu (I. Marsic).

0140-3664/02/$ - see front matter
PII: S0140-3664(02)00104-4

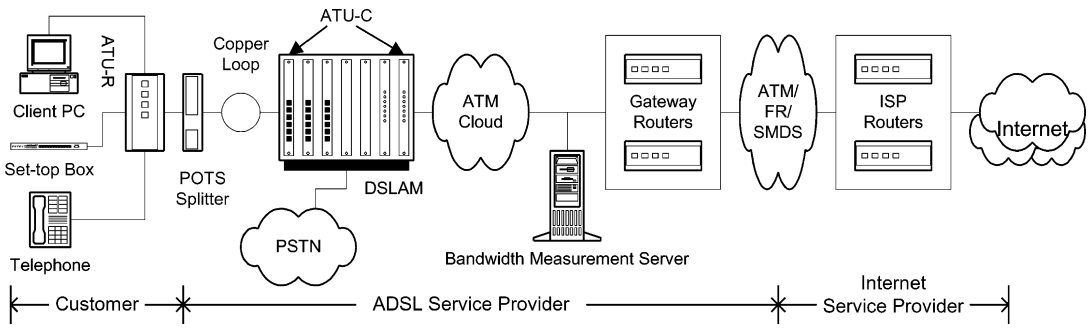


Fig. 1. xDSL deployment architecture (ADSL case).

reported here, an ATM switch is used. The ATU-C terminates multiple subscriber loops and connects the ATU-R to the Internet through a Gateway Router. The deployment and maintenance of the line between the ATU-R and the Gateway Router is the responsibility of the phone company and there is a need to measure the actual physical line speed of the ADSL link for reasons explained below.

1.1. Motivation for accurate bandwidth measurement

In addition to general benefits that can be gained by measuring network link bandwidth, such as application adaptation to network conditions [17], there are specific motivations for accurate bandwidth measurement in xDSL service networks. In this paper, we study the link bandwidth measurement, which is the smallest bandwidth (bottleneck-link bandwidth) along the xDSL data path in the xDSL service network. (The xDSL service network, or 'xDSL network,' refers to the customer and the ADSL service provider parts in Fig. 1 and does not include the ISP part.) It can also be the case that applications themselves might be interested in available bandwidth, given that several applications are simultaneously using the same pipe, however, this is not studied here.

Commercial xDSL networks combine the benefits of the Discrete Multi-Tone (DMT) or Carrierless Amplitude Phase (CAP) modulation and ATM technologies, which result in bandwidth and service flexibility. For example, in ADSL service networks upstream and downstream bit rates can be chosen freely and quasi-continuously up to the maximum physical limits. Based on the maximum possible bit rate (with a predetermined margin), the service management system of an xDSL network can set the bit rate to the level determined by the customer's service profile, thus maximizing noise margin and/or minimizing transmit power. In addition, xDSL networks can support a random mix of services with different bit rates and traffic requirements, which are introduced on a per-customer basis.

Accurate bandwidth measurement in xDSL service networks is essential for service provisioning. For example, customers experiencing slow Internet-connection speeds may want to know what the exact service parameters are. In this case, a bandwidth measurement tool can help the customers and the service providers (both the ISP and the xDSL providers) to verify service quality. Accurate bandwidth measurement thus contributes to better customer service. For example, it can help customers identify which xDSL category is the most appropriate for them. They can determine whether ADSL is sufficient or whether SDSL is needed.

Accurate bandwidth measurement is important for network management in xDSL networks, such as isolating line faults and performing traffic engineering. How xDSL networks should be tested, repaired, and managed is in fact an outstanding issue in the provisioning of xDSL services [9]. From the network administrator's point of view, traffic loads can be balanced knowing the bandwidths in xDSL networks. Lastly, accurate bandwidth measurement is helpful for xDSL-network design. Once a bottleneck link is identified, extra capacity can be added to enhance the network performance.

1.2. Related work

A number of techniques exist for bandwidth measurement. Most of these can be categorized into two groups [16]. One group is a variant of *pathchar* [7,11,12] based on the *one-packet model*. The disadvantage of this group is the heavy overhead in bandwidth consumption. The other is a variant of packet-pair [3,4,17,20,21] based on the *packet-pair model*. This group imposes lower overhead compared to the first group.

In Ref. [16], a multi-packet model is presented to unify the one-packet and the packet-pair model and a packet-tailgating technique *tailgater* is proposed. However, as pointed out in Ref. [15], similar to *pathchar*, the *tailgater* measures individually the bandwidth of each link along a path, which can be time-consuming and unnecessary for applications that only want to know the bottleneck bandwidth such as the xDSL case studied here.

Our focus is on xDSL networks and the common disadvantage of the existing techniques is that their accuracy is not satisfactory for xDSL service provisioning since their error range is greater than 5%, and sometimes 10%. This is mainly due to ATM traffic shaping in xDSL service networks as explained in Section 2.2. For example,

Ref. [15] reports that *nettimer* has a measurement error greater than 10% of the nominal ADSL link bandwidth.

In addition to bandwidth measurement, the related throughput measurement techniques have been used in applications and protocols, such as Network Weather Service (NWS) [23] and TCP. However, these techniques are likewise not accurate and not suitable for bandwidth measurement in xDSL networks. For example, during each measurement in NWS [23], the round-trip time of a single-word packet in a TCP connection is measured and the resulting value is divided by two to yield an approximation of the latency or start-up overhead associated with the communication. Obviously, the approximation is not correct in ADSL's asymmetric case. The approximation is rough without considering the flow and congestion control by TCP. There is also ongoing research [6] addressing problems in bandwidth measurement caused by the ATM network model and link asymmetry. However, no public information is available as of now.

In theory, the one- or two-packet techniques and the tailgating technique can get good estimation of xDSL link bandwidth. However, the existing tools do not regulate the probe packets so that ATM traffic shaper will regulate the probe packets for them by packet delay or packet drop. One possible solution is to explicitly regulate the probe traffic, but then it is difficult to specify the source rate. That is why we propose a stepwise scheme: in the first step we get a rough estimate of the bandwidth and regulate the subsequent probe traffic according to the estimate. We also use multi-packet technique to avoid explicit packet filtering.

In our research, we designed an accurate bandwidth measurement scheme for xDSL service network, implemented it as a Java-based tool, and tested it in both laboratory and commercial setup experiments. The accuracy is achieved by implementing a novel stepwise bandwidth measurement algorithm with an original traffic generator for smooth traffic flow and a multi-packet bandwidth measurement technique. The experiments reported in this paper demonstrate the accuracy of our measurement scheme.

The primary concern of this paper is the measurement accuracy which is essential for xDSL service provisioning. Other concerns could be imposed by protocol or layer constraints (it may be required to use certain protocol, such as TCP, User Datagram Protocol (UDP), ICMP, etc.) or by packet type constraints (it may be required to use payload packets for measurement, as opposed to probe packets which add overhead traffic). We are not considering those concerns here, but we are using transport layer flows. The advantage of using transport layer flows, e.g. a UDP packet flow, is that they can penetrate Network Address Translation (NAT) gateways, which are more and more used in xDSL deployments.

The paper is organized as follows. Section 2 discusses ATM traffic shaping and its effect on the accuracy of bandwidth measurement. Section 3 introduces a multi-packet bandwidth measurement technique and its applications to ADSL. Section 4 presents a stepwise algorithm for accurate bandwidth measurement using a traffic generator with stable and accurate performance. Section 5 presents the evaluation results and Section 6 concludes the paper.

2. ATM traffic shaping

xDSL deployments are typically implemented over ATM to provide different service qualities. For example, the ATU-Cs in ADSL networks are grouped into a DSLAM unit that terminates the multiplexed traffic into an ATM switch (Fig. 1). Based on the synchronization speed of the ADSL modem, the ATM switch sets the service class for the virtual circuit as the class to which a particular customer has been mapped.

There are five service categories/classes in ATM networks specified by the ATM Forum [1]: Unspecified Bit Rate (UBR), Available Bit Rate (ABR), Constant Bit Rate (CBR), Real-Time Variable Bit Rate (rt-VBR), and Non-Real-Time Variable Bit Rate (nrt-VBR) services. UBR service does not have any guarantee with regard to the quality of service (QoS) such as delay, loss or bandwidth. No traffic management is performed in the ATM layer for the UBR service class, so the applications using UBR service should be able to handle the fluctuations in these parameters by using error-correction and flow control techniques. ABR service guarantees only the cell-loss rate (CLR), which is achieved by exchanging resource management cells between the source and sink ATM nodes. CBR service class is suitable for applications having strict QoS requirements in delay and loss because it guarantees that a static amount of bandwidth is always available to an accepted connection. It is characterized by Peak Cell Rate (PCR) and Cell Delay Variation Tolerance (CDVT). In rt-VBR and nrt-VBR service classes, PCR, CDVT, Sustained Cell Rate (SCR), and Maximum Burst Size (MBS) are used to specify the traffic parameters so that they can utilize network resources efficiently by multiplexing several connections with bursty traffic.

2.1. Traffic shaping and buffer management

The goal of ATM traffic shaping is to regulate the traffic flow as per the parameters describing the negotiated QoS to achieve better network efficiency. The basic idea of ATM traffic shaping is buffering the packets/cells and controlling, mostly delaying, their entry into the network, thereby ensuring a more constant flow of traffic in the network. There are two important design factors in ATM traffic shaper: buffer management and scheduling algorithms. As discussed in Section 3.2 below, packet loss is a significant factor affecting the accuracy of bandwidth measurement in xDSL networks. Here we discuss the buffer management since it directly affects the packet loss.

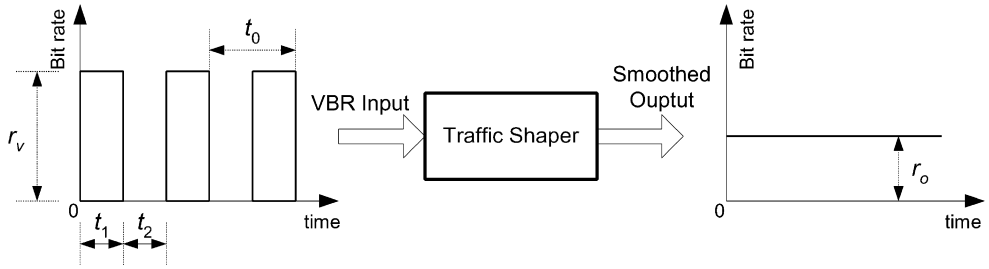


Fig. 2. Traffic shaping illustration diagram.

In general, different service categories are implemented through different queues in an ATM switch. Buffering method defines the amount of buffer space available to a given queue and specifies how the space is shared among the different queues. In the *Complete Partitioning* scheme, each queue gets a fixed amount of the buffer space while in the *Complete Sharing* scheme, all buffer space is fully shared among all the queues. As a compromise between the two schemes, the *Sharing with Minimum Allocation* scheme reserves a minimum buffer space for each queue while the rest of the buffer is completely shared among the queues [19]. Commercial ATM implementations favor buffer sharing schemes because of lower memory requirements, e.g. Cisco LightStream 1010 with total buffer capacity of only 64K cells [18].

Fig. 2 and the following discussion adapted from Ref. [19] illustrates that the amount of the buffer space in the ATM traffic shaper is a major factor in determining if and when cell loss occurs. If we denote the input traffic rate r_v as positive and the output traffic rate r_o as negative values in Fig. 2, the sum of the averages of the input and output traffic rates, r, which is denoted in Eq. (1), will determine the cell/packet loss

$$r = r_o + r_v(t_1/t_0) \tag{1}$$

The variables t_1 and t_2 define the ON and OFF periods of the ON/OFF VBR traffic flow, and $t_0 = t_1 + t_2$ (Fig. 2). If r is positive, it means that the traffic shaper receives data at a greater rate than it can deliver, and thus the cell/packet loss will eventually occur. If it is zero or negative, a zero loss can be achieved by buffering the cells during the bursty period t_1, and the amount of data that must be buffered is $t_1(r_v + r_o)$. In this case some cells will be delayed.

2.2. Effects of ATM traffic shaping on bandwidth measurement

The network model for bandwidth measurement is shown in Fig. 3. The ADSL service rates are independent of the ATM service categories chosen. The ADSL service rates are chosen by using different line cards on the ATU-C unit. We assume that the ADSL link is always the bottleneck link since ATU-R, e.g. ADSL modems, will accommodate ATM transport with variable rates and compensation for ATM overhead [2]. The bearer channel of ADSL interface can be programmed to carry bits in any multiple of 32 Kbps. For example, when ADSL is used for downstream ATM cell transport, service rates of 1.760, 3.488, 5.216, and 6.944 Mbps are the options. The odd-looking speeds are a result of the desire to preserve compatibility with existing AAL1 and circuit definitions already entrenched in ATM specification. ADSL service rates that are not simple multiples of 32 Kbps can be supported by carrying 'extra' bits in the shared overhead area of ADSL frame [9]. Therefore, the measured bandwidth in a general case is the nominal value of the ADSL service rate since the ADSL link is the bottleneck.

However, ADSL link bandwidth can sometimes be larger than the ATM link bandwidth because of the following reasons: (i) the characteristics of a copper telephone line vary over time, even from daylight to night time, due to temperature, other traffic in the same cable bundle, and external RF interferences [10]; and, (ii) the matching difference between the rate of the bearer channel of ADSL interface and the rate of the ATM switch. In these cases, the ATM link will be the bottleneck link. However, because both the ADSL service rate and ATM bit rate are supposedly configured to be the same, the measured bandwidth still represents the nominal value of the ADSL service rate.

In the cases of downstream traffic or upstream traffic with

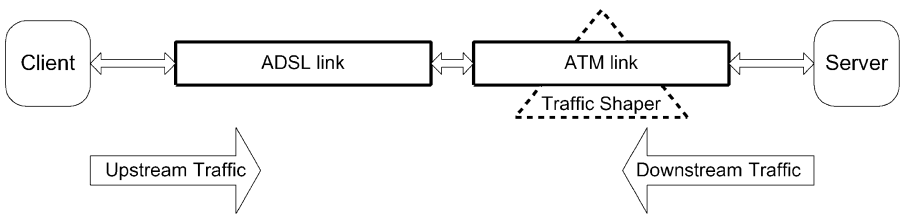


Fig. 3. Network model for bandwidth measurement.

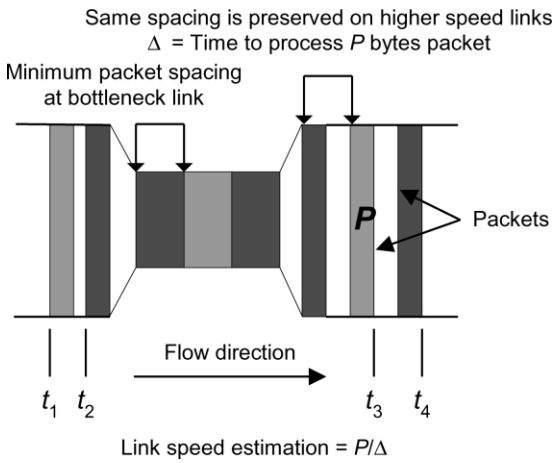


Fig. 4. Packet-pair technique for bandwidth measurement (adapted from Ref. [15]).

ATM bit rate being lower than the ADSL service rate, the ATM traffic shaping mechanism may introduce inaccuracies in bandwidth measurement. Bandwidth measurement techniques, such as *pathchar* in Ref. [11], send a large number of probe packets without explicit traffic regulation, which subjects them the ATM traffic shaping. The shaper either delays the packets or overflows and causes some probe packets to be dropped. The phenomenon is similar to the scenario where for some service classes, such as the CBR service, an increase in the data rate beyond the provisioned bandwidth may result in heavy packet loss caused by the ATM traffic shaping. The effect on the bandwidth measurement is distorting the results since the accuracy of those techniques depends on timely delivery of the probe traffic.

To minimize the effect of ATM traffic shaping, we designed a stepwise bandwidth measurement algorithm. In the first step, a few packets (with the total size less than 10 KB) are used to get the trial bandwidth of the xDSL link. No packet loss is expected because the buffer in ATM switches is generally big enough to tolerate a small burst. The discussion in Ref. [15] about the measurement agility implies that such small amount of data is suitable to get a rough estimate of the nominal bandwidth. Then in the following steps a relatively large number of packets are used for results convergence and consistence. The packets are generated from a smooth traffic generator so that most of them are not dropped by the ATM traffic shaper.

3. Multi-packet bandwidth measurement

Our bandwidth measurement scheme uses a multiple packet technique instead of two packets in the packet-pair technique in order to achieve accuracy of the measurement, without deploying explicit packet-filtering techniques [16]. Our experience shows that the number of probe packets required to achieve accurate measurement is on the order of hundred for each measurement, while the number of packets used by other currently available bandwidth measurement tools [16] is of the order of a thousand. Thus our scheme is lightweight in terms of implementation complexity and probe traffic overhead. We first describe the packet-pair technique and then present the multi-packet technique.

3.1. Packet-pair technique

The packet-pair technique for bandwidth measurement is based on FIFO queuing network model. Various forms of the packet-pair technique are studied by Bolot [3], Carter and Crovella [4], Paxson [20], and Lai and Baker [17]. Its essential idea is using inter-packet time to estimate the characteristics of the bottleneck link. If two packets, e.g. Internet Control Message Protocol (ICMP) probe packets, travel together so that they are queued as a pair at the bottleneck link with no packet intervening between them, then their inter-packet spacing is proportional to the processing time required for the bottleneck link to transmit the second packet of the pair (Fig. 4). Therefore the link bandwidth of the bottleneck link, b, can be computed as:

$$b = P/\Delta \tag{2}$$

where P is the packet size. Note that the inter-packet spacing at the source must be sufficiently small so that:

$$t_2 - t_1 \leq P/b = t_4 - t_3 \tag{3}$$

and

$$\Delta = t_4 - t_3 \tag{4}$$

We generalized this technique to a multi-packet technique and applied it to achieve accurate bandwidth measurement in xDSL networks, including ADSL networks.

3.2. Multi-packet technique

Since packet-pair methodology depends on the correct inter-packet time, two factors could diminish the accuracy of measurement results [16]. One is called *time compression*, where the first packet of the packet pair gets delayed because of some other cross traffic along the measurement path and the time interval of the packet pair is decreased. Thus the result will be larger than the actual bandwidth. The other is called *time extension*, where the second packet of the packet-pair gets delayed and the time interval is increased. Thus the result will be smaller than the actual value. The packet loss in the packet-pair falls in the latter case. Packet-pair filtering algorithms [16] can be applied to handle these cases so that inaccurate packet-pair time intervals are filtered out and the measurement results become more accurate.

In xDSL networks, customers and administrators use our tool as a network testing or monitoring tool. The data path from the end-user's computer to the bandwidth measurement server is an ATM-circuit-like path. It is safe to assume

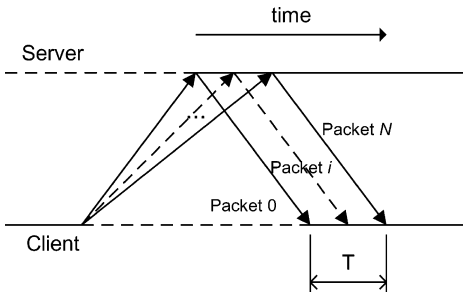


Fig. 5. Upstream multi-packet bandwidth measurement.

that there is no cross traffic, meaning that *time compression* is not a problem in our case. However, *time extension* may occur due to ATM traffic shaping in xDSL service and due to possible packet loss. Hence, we apply a *maximum* operator to handle the inaccuracy caused by time extension.

In order to reduce the effect of *time extension*, we propose to use multiple packets instead of two packets in packet-pair to conduct the bandwidth measurement. The multi-packet technique does not need to deploy explicit packet filtering in order to achieve accurate measurement because of its intrinsic filtering due to the averaging of results over the multiple packets. Theorem 1 provides the foundation for the multi-packet technique.

Theorem 1. *Multi-packet bandwidth measurement. Let* $b_{\min} \leq b_i (\forall i, 1 \leq i \leq m)$, *where m is the number of links in the path and* b_i *is bandwidth of the i-th link between the node* $i-1$ *and the node i. Then if multiple,* $N+1$, *packets of the same size* ($s_0 = s_1 = \cdots = s_N = s$) *are sent at the same time* ($t_0 = t_1 = \cdots = t_N$) *and there is no packet drop and no cross-traffic along the path, they will arrive with a difference in time equal to the sum of all packet sizes divided by the smallest bandwidth along the path, i.e.* $t_{N(m)} - t_{0(m)} = (s \times N)/b_{\min}$, *or* $b_{\min} = (s \times N)/(t_{N(m)} - t_{0(m)})$, *where* $t_{j(\ell)}$ *is the time when packet j fully arrives at node* ℓ.

Using results from Ref. [16], we prove Theorem 1 in Appendix A.

3.3. Discussion

Theorem 1 assumes that there is no packet drop, which means that N should be small enough to ensure that when N packets are simultaneously sent, no packet will be dropped by the ATM traffic shaping. The first step in our stepwise algorithm conforms to Theorem 1 by using a very small number of packets. The algorithm is presented in Section 4 below.

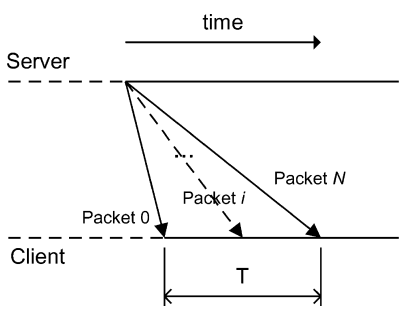


Fig. 6. Downstream multi-packet bandwidth measurement.

The condition of sending the packets at the same time can be relaxed if the time interval between consecutive packets is not greater than the time required to transmit a packet over the bottleneck link. This is illustrated in Fig. 4, where the inter-packet spacing after the bottleneck link does not depend on the inter-packet spacing before the bottleneck link, as long as the latter is smaller. This is also stated in Eq. (3). In the second and subsequent steps of the stepwise algorithm, a smooth traffic generator sends the packets with the inter-packet intervals such that the rate is slightly higher than the bottleneck link bandwidth estimated in the first step. Therefore, Theorem 1 still holds valid.

3.4. Application to asymmetric DSL

The asymmetric nature of the ADSL network makes it necessary to have different measurement methodologies for upstream and downstream cases. The design is based upon the assumption that the upstream bandwidth is lower than the downstream bandwidth.

3.4.1. Upstream methodology

A fixed number, $N+1$, of UDP packets of uniform size, P bytes, are sent from the customer's computer (client) at a rate slightly higher than the nominal bottleneck bandwidth of the ADSL network. Probe traffic with a slightly higher rate (about 10%) is necessary to saturate the pipe. A server process on the Bandwidth Measurement Server (Fig. 1), echoes back the packets as they arrive at the server end. Fig. 5 shows this scheme. The time difference, T, is measured between the arrival of the first packet and the last packet at the client end. The upstream bottleneck link bandwidth, b_1, is computed as:

$$b_1 = (N \times P \times 8)/T \text{ [Kbps]} \tag{5}$$

3.4.2. Downstream methodology

A traffic generator at the Bandwidth Measurement Server generates a downstream traffic. A receiving process at the client measures the arrival time of the packets. Because of the nature of ADSL, such that the upstream bandwidth is smaller than the downstream bandwidth, the client does not echo the probe packets back. Instead, the client computes the downstream bandwidth using the same Eq. (5) as for the upstream case. Fig. 6 shows the downstream scheme. The downstream method can be used for the upstream case, but the results should then be sent from the Bandwidth Measurement Server back to the client for display.

Table 1
Pseudo-code for the simple traffic generator

```
//Initialize:
//maxNumOfPackets is the maximum number of packets
allowed to be sent
//packetSize is the size of packet in bits, rate is the
expected traffic rate
delay = packetSize/rate;

for (k = 0; k < maxNumOfPackets; k++)
{
  sleep(delay);
  sendPacket();
}
```

4. Stepwise bandwidth measurement algorithm with smooth traffic generator

4.1. Stepwise bandwidth measurement algorithm

Our stepwise algorithm consists of at least two steps. It can be used for both downstream and upstream measurements described in Section 3.4. The generated traffic should be smooth rather than bursty to avoid excessive ATM traffic shaping. We run the stepwise algorithm several times and report the maximum measured value as the link bandwidth.

During the first step, a small number of packets, e.g. 10, are sent sequentially back-to-back from the client to the server. This small number of packets will not likely be affected by ATM traffic shaping, as most ATM networks are capable of handling such small bursts. The computed result is used in the second step as the trial bandwidth of the xDSL link.

The subsequent step(s) assumes that the accurate xDSL bandwidth is close to the trial result obtained from the first step. A larger number of packets than that in the first step are used in the measurement to ensure that the results are convergent and in the end consistent. Had it not been for the first step that approximately determined the bandwidth, the sustained higher-than-provisioned traffic would result in a considerable loss or queuing delay caused by ATM traffic shaping, and this would deteriorate the results, as explained in Section 2.2 earlier.

Table 2
Pseudo-code for the new traffic generator

```
//Initialize:
//maxNumOfPackets is the maximum number of packets
allowed to be sent
//packetSize is the size of packet in bits, rate is the
expected traffic rate
//pps: the number of packets to be sent per second

pps = rate/packetSize;
fraction = 0;
stopSleep = getTimeMillis(); //get the current time in
ms
startSleep = stopSleep;
i = 0; //number of packets have been sent

do
{
  sleep(1); //sleep for 1 ms
  stopSleep = getTimeMillis();

  packetCount = (int) (fraction + pps * (stopSleep -
startSleep)/1000);
  fraction = (fraction + pps * (stopSleep -
startSleep)/1000) - packetCount;

  startSleep = stopSleep;
  upLimit = i + packetCount;
  if(upLimit > maxNumOfPackets)
    upLimit = maxNumOfPackets; //reached the maximum
number of packets allowed to be sent
  for (; i < upLimit; i++)
    sendPacket();
} while(i < maxNumOfPackets);
```

4.2. Smooth traffic generator

As discussed in Sections 2.2 and 3.3 above, the accuracy of the bandwidth measurement depends on whether the traffic generator can generate a smooth traffic at a rate slightly higher than or equal to the nominal bottleneck bandwidth of the xDSL network, thus it is crucial to implement an accurate and stable traffic generator with smooth traffic flow.

A straightforward method of generating packets at a known rate employs the algorithm shown in Table 1.

While the above pseudo-code seems to be correct in theory, it does not yield the expected behavior. Owing to the scheduling mechanisms of operating systems and coarse granularity of timers, the packets are not necessarily sent at the required rate with uniform spacing. Consider the case when the call to `sleep()` returns and the process is scheduled out. Instead of sending a packet immediately, the packet is sent only when the OS reschedules the process. This results in data rate oscillations or bursts in traffic generating, which is shown in Section 4.3 later.

To overcome the randomness caused by the scheduling mechanisms of operating systems and coarse granularity of timers, we devised an alternative way of generating smooth traffic at the desired rate. The pseudo-code for the improved algorithm is shown in Table 2. The algorithm achieves the stable and smooth traffic flow by adjusting the traffic nearly every millisecond. It is similar to the leaky bucket algorithm; see, e.g. Section 13.3.4 in Ref. [13]. Intuitively, for every millisecond the traffic generator accumulates tokens into `packetCount`, which is a real number, e.g. 1.3. The algorithm checks the `packetCount` every millisecond and the integer part represents the number of packets to be sent (if greater than zero). The remainder, called `fraction`, e.g. 0.3, is then used in the next millisecond's accumulation. The performance of the new traffic generator is shown in Section 4.3.

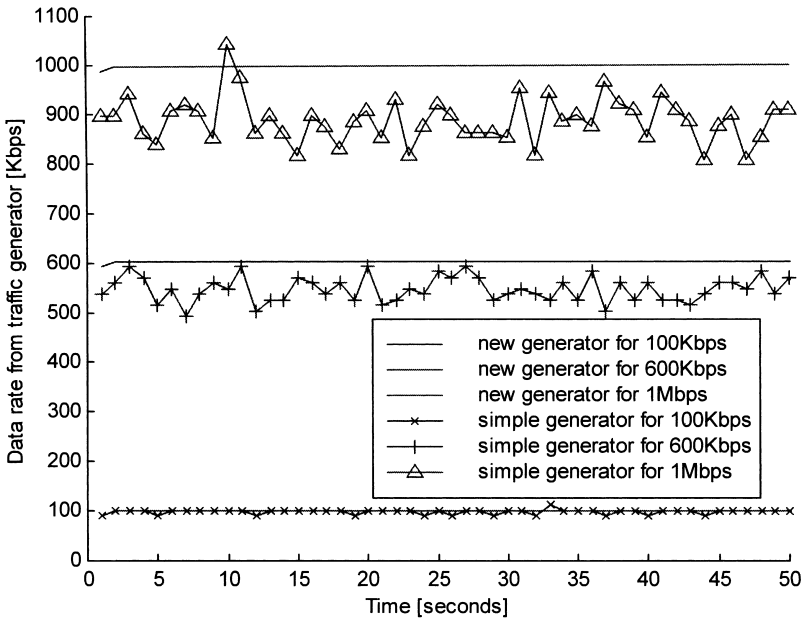


Fig. 7. Performance comparison for traffic generators. The parameters used in simulation are: $a = 1$, $b = 10$.

The packet size is computed as:

$$P = \text{MAC header} + \text{IP header} + \text{UDP header} + \text{payload} \tag{6}$$

The default payload size is 1400 bytes for packets to fit the Ethernet frame size. The payload is a ZIP-compressed file to avoid the effects of the compression on the V90 analog modem.

4.3. Performance study of the traffic generator

We simulated in Matlab both the simple traffic generator (Table 1) and our original traffic generator (Table 2) and compared their performance in terms of stability and accuracy. The simulation models can be expressed as follows:

1. Physical network interface of the data source is 10/100 Mbps Ethernet, which is popular in xDSL implementation with PPPoE (Point-to-Point Protocol over Ethernet).
2. The randomness caused by the scheduling mechanisms of operating systems is additive and limited to a certain range. For example, since the minimum time used to send a packet of 1400 bytes is $t_{\min}$ milliseconds ($t_{\min} = 1400 \times 8 \times 1000/10^7$ in case of the 10 Mbps Ethernet), the actual time used to send a packet t_{send} can be expressed as:

$$t_{\text{send}} = \text{Round}(t_{\min} + R_{\text{os}}) \tag{7}$$

where $R_{\text{os}} = aR_{\text{n}}$, R_{n} is a uniformly distributed real random number on [0,1), Round(·) is an operator that rounds a number towards the nearest integer, and a is the upper limit (in ms) of the additive random component caused by the scheduling mechanisms of operating systems.

3. The randomness caused by the coarse granularity of timers is multiplicative because it is accumulated by each tick of the timer, and it is limited to a certain range. For example, in case a thread is set to sleep for t_{sleep} milliseconds, the actual time the thread will sleep, t_{delay}, can be expressed as:

$$t_{\text{delay}} = t_{\text{sleep}} + R_{\text{timer}} \tag{8}$$

where

$$R_{\text{timer}} = \text{Sgn}(R_{\text{n}})\text{Min}(\text{Abs}(\text{Round}(t_{\text{sleep}}R_{\text{n}})), b) \tag{9}$$

where Sgn(·) is the signum function, Abs(·) is the absolute value function, Min(·) is the minimum value function, and $b (b > 0)$ is the upper limit on the random value caused by the coarse granularity of timers (in ms).

The simulation results are shown in Fig. 7. The data rate from the new traffic generator is stable and smooth. Compared to the oscillations and deviations in traffic rate generated by the simple traffic generator, the new generator has much better performance than the simple one. The new traffic generator is an essential part of our accurate bandwidth measurement scheme.

5. Evaluation

We have implemented the above bandwidth measurement scheme as a Java-applet based tool and deployed it in a commercial setup [5]. We first evaluated the tool using a serial link in a laboratory testbed network in which the

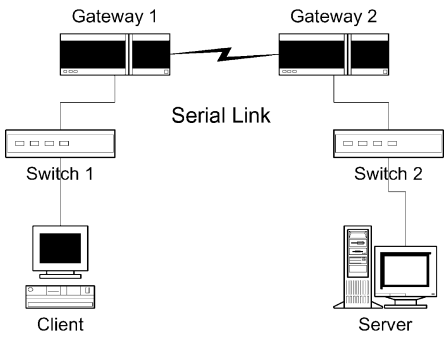


Fig. 8. Testbed topology.

bandwidth could be altered manually. Next, we evaluated the tool over commercial ADSL service networks.

5.1. Laboratory experiments

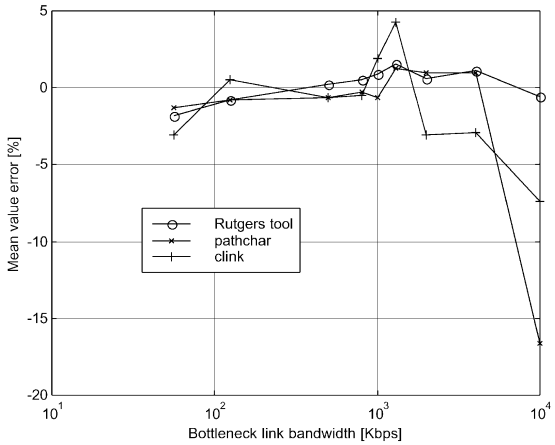
Fig. 8 shows the topology of the testbed. The client and the server are in different subnets that are connected by a serial link by introducing a High-Speed Serial Interface (HSSI) between the two gateways. The serial link is the bottleneck link. The endpoints for the tests were located across the two ends of the HSSI. There is no cross traffic.

We compared our measurement tool with other popular tools, such as *pathchar* [7,11], *clink* [8], and *nettimer* [14]. The experiments were performed on an Intel Pentium II 266 MHz machine with Red Hat Linux 7.2 installed. For a certain bandwidth setting of the HSSI, the test was repeated 100 times for the Rutgers tool which generates measurement results in seconds, and 10 times for both *pathchar* and *clink* which generate results in minutes, as illustrated in Table 3. The test was also repeated 10 times for the *nettimer* tool for several parameter and HSSI bandwidth settings. The results were generated within seconds or minutes, depending on the *nettimer* parameter settings.

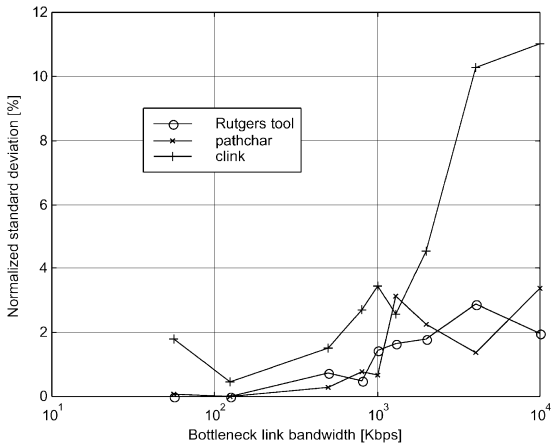
5.1.1. Rutgers tool vs. pathchar and clink

Table 3 shows the experimental results for the case of 10 Mbps link. Both *pathchar* and *clink* use ICMP packets in estimating the link characteristics. The Rutgers tool is implemented as a Java applet which uses UDP packets and it is thus platform-independent and very easy to use. Unlike *pathchar* and *clink*, no root or super-user privileges are required to use our tool.

Fig. 9 compares the performance of our bandwidth measurement tool to that of *pathchar* and *clink* on a range of bottleneck link bandwidths. The figure shows that our tool compares favorably with the other tools in terms of measurement accuracy. In addition, it is much faster to run, consumes much less bandwidth, and it is platform independent. Both *pathchar* and *clink* show significant deterioration of measurement accuracy starting at about 10 Mbps. We did not perform measurements for higher bandwidths, but it is reported in Ref. [16] that the accuracy becomes worse with increasing bandwidth.



(a)



(b)

Fig. 9. Comparison of tools for bandwidth measurement. The HSSI clock is only able to accurately set the link speeds up to 4000 Kbps. Consequently there is a gap from 4000 to 10 000 Kbps, which is the link speed for a 10 Mbps Ethernet LAN.

5.1.2. Rutgers tool vs. nettimer

Nettimer version 2.3.8 was used in the tests. It was run in the mode that actively probes the bottleneck link bandwidth by sending a given number of TCP FIN packets to ports on the destination host. These packets cause TCP RESET packets to be sent back to the measurement host so that *nettimer* captures these packets and makes calculations.

Table 4 illustrates the measurement results. For each HSSI bandwidth setting, the *nettimer* was tested 10 times with each of these three parameter settings: (1) all parameters of *nettimer* were set as default, (2) the probe packet size `active_probing_packet_size` was set as 1500 bytes, and (3) the probe packet size `active_probing_packet_size` was set as 1500 bytes, the minimum number of samples, `active_probing_min_samples`, and the maximum number of probe packets sent, `active_probing_max_packets_sent`, were set as 5000. Setting the value of probe packet

Table 3
Laboratory experiment results in case of a 10 Mbps link

Measurement tool	Version	Mean value (Mbps)	Standard deviation	Number of probe packets	Mean time per measurement
Rutgers tool	1.0	9.95	197 Kbps	450[a]	0.09 min
Pathchar [7]	Alpha; April 21, 1997	8.34	336 Kbps	*h* 2880[b]	7.50 min
Clink [8]	1.0; August 14, 1998	9.26	1.10 Mbps	*h* 1488	1.33 min[c]

[a] The number of probe packets can be configured. In all experiments reported here, 450 packets have been transmitted for each measurement. This number is statistically significant enough to offset the effect of the system timer inaccuracy.
[b] *h* is the number of hops between the two endpoints.
[c] This time is variable and reaches 8.50 min for smaller HSSI bottleneck link bandwidths.

Table 4
Measurement results using nettimer in active probing mode

Measurement tool	Nominal bandwidth	56 Kbps	125 Kbps	500 Kbps	2 Mbps	10 Mbps
Rutgers tool		55.0 Kbps	124.0 Kbps	501.2 Kbps	2.01 Mbps	9.95 Mbps
Nettimer [14]	Parameter setting #1	57.1 Kbps	133.3 Kbps	400 Kbps	1.48 Mbps[a]	9.65 Mbps[a]
	Parameter setting #2	54.5 Kbps	122.5 Kbps	600 Kbps		
	Parameter setting #3	54.9 Kbps	133.3 Kbps	600 Kbps		

[a] The data is reproduced from Ref. [15] since we were unable to obtain reasonable results at high-speed rates in our laboratory. In our correspondence with the author, he indicated that problem probably arises due to the limitation of the timer resolution in our configuration. If that is the case, such sensitivity to timer resolution is not desirable for general xDSL user community.

size to 1500 bytes is suggested in *nettimer* manual [14] for better performance when measuring high bandwidth links on slow hosts. The minimum number of samples and the maximum number of probe packets send were set to 5000 since it was observed that the large values ensure the convergence of the measurement results. The results demonstrate that our tool has generated more accurate measurement results in all cases that we were able to measure. For the high-speed rates we use the results reported in Ref. [15], where error varies from 3.5 to 7.0% in the 10 Mbps case, but the results for our tool are still more accurate (error of 0.5%) as shown in Table 4.

5.2. Field experiments

The field experiments were performed in the deployment abstracted as in Fig. 1. The accuracy of our bandwidth measurement tool is evaluated in the following ATM service categories in an ADSL service network:

- Constant Bit Rate (CBR) service category
- Real-Time Variable Bit Rate (rt-VBR) service category
- Unspecified Bit Rate (UBR) service category

For each category, the upstream and the downstream link bandwidths are measured for several nominal ADSL service rates: 90 and 680 Kbps in the upstream case, and 640 Kbps, 1.7 and 7.1 Mbps in the downstream case. Each measurement was repeated 100 times and the mean values and standard deviations were computed. Figs. 10 and 11 show that our tool generates accurate results. The standard deviations are omitted in the graphs but they follow similar trend as in Fig. 9, never exceeding 5%.

The possible reasons for not achieving 100% accuracy could be some of those discussed in Ref. [16]. Of those, the most feasible ones seem to be unreachable nominal values and the timing resolution of the operating system.[1]

To verify the necessity of stepwise algorithm, we also performed field measurements with the single-step algorithm. The experiment explicitly sets the traffic generator's sending rate in the client, lets it send sustained traffic at that rate, and then records the result at the measurement server. The results show that the measurement inaccuracy of the single-step algorithm ranges up to 25%. Since the stepwise algorithm prevents packet loss caused by ATM traffic shaping in xDSL service networks, it increases the accuracy of the bandwidth measurement.

6. Conclusion

Telecom operators use xDSL service networks to provide high-speed digital services by taking advantage of their copper infrastructure via overlay and without interfering with the traditional analog telephone service, such as Plain Old Telephone Service (POTS). Because xDSL is introduced on a per-user basis and has full bandwidth and service flexibility, it is essential for both customers and xDSL-service providers to have tools to accurately measure the xDSL link bandwidth.

We presented a novel stepwise bandwidth measurement

[1] In addition to the timer used in traffic generation, there are also times recorded at the beginning and end of each measurement cycle.

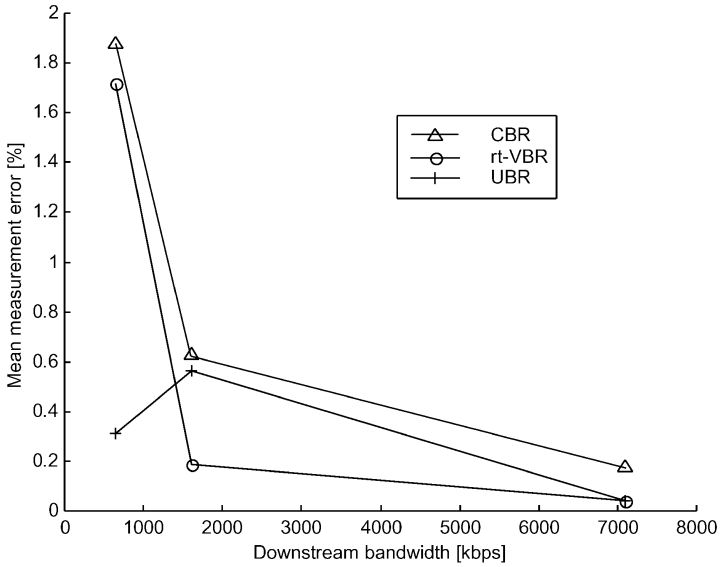


Fig. 10. Measurement accuracy in field experiments of ADSL downstream link.

scheme for accurately measuring link bandwidth in xDSL service networks, considering the impact of ATM traffic shaping and link asymmetry. It uses a multi-packet technique and a smooth traffic generator. The scheme solves the inaccuracy problem caused by ATM traffic shaping encountered by bandwidth measurement techniques that do not regulate their traffic. Given that the ATM shaper rate is greater than or equal to the ADSL service rate, our tool measures the actual ADSL link bandwidth. The deployment of the multi-packet technique does not require packet filtering to achieve accurate measurements. It is lightweight in terms of implementation complexity and probe traffic overhead. Our new traffic generator helps to achieve the measurement accuracy of the multi-packet technique.

Evaluation experiments demonstrate that our bandwidth measurement tool offers accurate bandwidth measurement in xDSL service networks. The measurement errors in the field experiments in both upstream and downstream cases are within the range of 5%, tested in an ADSL network under the following ATM network configurations: CBR, rt-VBR, and UBR service categories. We even plan to further improve the accuracy of our tool by using the arrival time of other packets instead of just the first and the last packet in the multi-packet technique. Our future work also includes more extensive evaluations and embedding this scheme in network-aware adaptive applications.

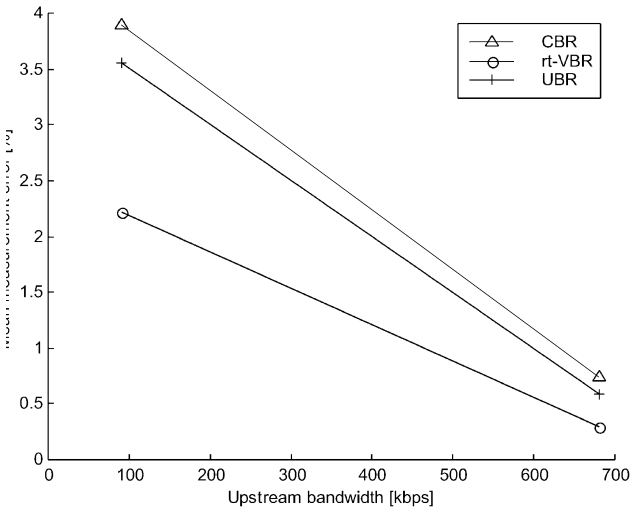


Fig. 11. Measurement accuracy in field experiments of ADSL upstream link.

Acknowledgements

The authors are grateful to Senthilkumar Rajasekharan for helping perform the measurements and collecting data in the field experiments. The authors would like to thank anonymous reviewers and to Dr Edward J. Devinney for their valuable comments. This research is supported by grants from Verizon, Inc. (formerly Bell Atlantic), Cisco Systems, Inc., and NSF KDI IIS-98-72995, and by the Rutgers Center for Advanced Information Processing (CAIP).

Appendix A. Multi-packet bandwidth measurement

Proof. We perform induction on N, the number of packet sent. For $N = 1$,

$$t_{N(m)} - t_{0(m)} = t_{1(m)} - t_{0(m)}$$

According to Ref. [16],

$$t_{N(m)} - t_{0(m)} = s/b_{\min}$$

that is:

$$t_{N(m)} - t_{0(m)} = (s \times N)/b_{\min}$$

for $N = 1$. This proves the $N = 1$ case.
For $N > 1$, assume the equation holds for $N = n$ case, i.e.

$$t_{n(m)} - t_{0(m)} = (s \times n)/b_{\min}$$

then for $N = n + 1$ case,

$$t_{n+1(m)} - t_{0(m)} = (t_{n+1(m)} - t_{n(m)}) + (t_{n(m)} - t_{0(m)})$$

$$= (t_{n+1(m)} - t_{n(m)}) + (s \times n)/b_{\min}$$

According to Ref. [16], since there is no cross traffic,

$$t_{n+1(m)} - t_{n(m)} = t_{1(m)} - t_{0(m)} = s/b_{\min}$$

therefore,

$$t_{n+1(m)} - t_{0(m)} = (s \times (n + 1))/b_{\min}$$

This proves the $N = n + 1$ case.

References

[1] ATM Forum, Online at: http://www.atmforum.com/.
[2] DSL Forum, Online at: http://www.adsl.com/.
[3] J.C. Bolot, End-to-end packet delay and loss behavior in the Internet, Proceedings of the ACM SIGCOMM '93 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (1993). pp. 289–298 San Francisco, CA, USA.
[4] R.L. Carter, M.E. Crovella, Measuring bottleneck link speed in packet-switched networks, Technical Report TR-96-006, Department of Computer Science, Boston University, March 1996, Online at: http://www.cs.bu.edu/faculty/crovella/papers.html.
[5] L. Cheng, I. Marsic, Java-based bandwidth measurement tool for digital subscriber line networks, Proceedings of the Ninth Mediterranean Conference on Control and Automation, Dubrovnik, Croatia, June (2001).
[6] A.B. Downey, From pathchar to netchar: keeping an eye on your ISP, Presentation at Cisco Systems Inc., Online at: http://rocky.wellesley.edu/downey/clink/.
[7] A.B. Downey, Using pathchar to estimate Internet link characteristics, Proceedings of the ACM SIGCOMM '99 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Cambridge, MA (1999) 241–250. August/September.
[8] A.B. Downey, Clink: a tool for estimating Internet link characteristics, Wellesley College, Online at: http://rocky.wellesley.edu/downey/clink/.
[9] W. Goralski, ADSL and DSL Technologies, McGraw-Hill, New York, 1998.
[10] D. Greggains, ADSL and high bandwidth over copper lines, International Journal of Network Management 7 (5) (1997) 277–287.
[11] V. Jacobson, Pathchar, 1997, Online at: ftp://ftp.ee.lbl.gov/pathchar.
[12] V. Jacobson, Pathchar—A tool to infer characteristics of Internet paths, Presented at the Mathematical Science Research Institute, April 21, 1997, Online at: ftp://ftp.ee.lbl.gov/pathchar.
[13] S. Keshav, An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network, Addison-Wesley, Reading, MA, 1997.
[14] K. Lai, Nettimer manual, Online at: http://gunpowder.stanford.edu/~laik/projects/nettimer/#Documentation, last visited and downloaded on January 15, 2002.
[15] K. Lai, M. Baker, Nettimer: a tool for measuring bottleneck link bandwidth, Proceedings of the third USENIX Symposium on Internet Technologies and Systems, San Francisco, CA, March (2001) pp. 122–133.
[16] K. Lai, M. Baker, Measuring link bandwidths using a deterministic model of packet delay, Proceedings of the ACM SIGCOMM 2000 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, Stockholm, Sweden, August (2000) pp. 283–294.
[17] K. Lai, M. Baker, Measuring bandwidth, Proceedings of the Conference on Computer Communications (IEEE INFOCOM '99), New York, NY, USA, March (1999) 235–245.
[18] A. Lin, LightStream 1010 Switch Architecture and Traffic Management, White Paper, Cisco Systems Inc., July 2000, Online at: http://www.cisco.com/warp/public/cc/so/neso/vvda/atm/lsatm_wp.htm.
[19] T. Lizambri, F. Duran, S. Wakid, Priority scheduling and buffer management for ATM traffic shaping, Proceedings of Seventh IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS '99), Cape Town, South Africa December (1999) 36–43. Online at: http://w3.antd.nist.gov/Publications/Hsnt/lizambri_1299.html.
[20] V. Paxson, Measurements and Analysis of End-to-End Internet Dynamics, PhD Thesis, University of California, Berkeley, April, 1997.
[21] V. Paxson, End-to-end Internet packet dynamics, Proceedings of the ACM SIGCOMM '97 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (1997) pp. 139–152 Cannes, France.
[22] The DSL Forum, Online at: http://www.adsl.com.
[23] R. Wolski, Dynamically forecasting network performance to support dynamic scheduling using the network weather service, Proceedings of the Sixth IEEE Symposium on High-Performance Distributed Computing, August (1997). pp. 316–325 Portland, OR, USA.