Bandwidth Estimation: Metrics, Measurement Techniques, and Tools

Ravi Prasad and Constantinos Dovrolis, Georgia Institute of Technology
Margaret Murray and kc claffy, Cooperative Association for Internet Data Analysis (CAIDA)

Abstract

In a packet network, the terms *bandwidth* and *throughput* often characterize the amount of data that the network can transfer per unit of time. Bandwidth estimation is of interest to users wishing to optimize end-to-end transport performance, overlay network routing, and peer-to-peer file distribution. Techniques for accurate bandwidth estimation are also important for traffic engineering and capacity planning support. Existing bandwidth estimation tools measure one or more of three related metrics: capacity, available bandwidth, and bulk transfer capacity. Currently available bandwidth estimation tools employ a variety of strategies to measure these metrics. In this survey we review the recent bandwidth estimation literature focusing on underlying techniques and methodologies as well as open source bandwidth measurement tools.

In physical layer communications, the term *bandwidth* relates to the spectral width of electromagnetic signals or the propagation characteristics of communication systems. In the context of data networks, bandwidth quantifies the *data rate* at which a network link or a network path can transfer. In this article we focus on estimation of bandwidth metrics in this latter data network context.

The concept of bandwidth is central to digital communications, specifically to packet networks, as it relates to the amount of data a link or network path can deliver per unit of time. For many data-intensive applications such as file transfers or multimedia streaming, the bandwidth available to the application directly impacts application performance. Even interactive applications, which are usually more sensitive to latency than throughput, can benefit from the lower end-to-end delays associated with high-bandwidth links and low packet transmission latencies.

Bandwidth is also a key factor in several network technologies. Several applications can benefit from knowing the bandwidth characteristics of their network paths. For example, peer-to-peer applications form their dynamic user-level networks based on available bandwidth between peers. Overlay networks can configure their routing tables based on the bandwidth of overlay links. Network providers lease links to customers and usually charge based on bandwidth purchased. Service level agreements (SLAs) between providers and customers often define service in terms of available bandwidth at key interconnection (network boundary) points. Carriers plan capacity upgrades in their network based on the rate of growth of bandwidth utilization of their users. Bandwidth is also a key concept in content distribution networks, intelligent routing systems, end-to-end admission control, and video/audio streaming.

The term *bandwidth* is often imprecisely applied to a variety of throughput-related concepts. In this article we define specific bandwidth-related metrics, highlighting the scope and relevance of each. Specifically, we first differentiate between the bandwidth of a link and the bandwidth of a sequence of successive links, or *end-to-end path*. Second, we differentiate between the *maximum possible bandwidth* a link or path can deliver (capacity), the *maximum unused bandwidth* at a link or path (available bandwidth), and the achievable throughput of a *bulk transfer TCP connection* (bulk transfer capacity ADR). All these metrics are important since different aspects of bandwidth are relevant for different applications.

An important issue is how to measure these bandwidth-related metrics on a network link or an end-to-end path. A network manager with administrative access to the router or switch connected to a link of interest can measure some bandwidth metrics directly. Specifically, a network administrator can simply read information associated with the router/switch (e.g., configuration parameters, nominal bit rate of the link, average utilization, bytes or packets transmitted over some time period) using the SNMP network management protocol. However, such access is typically available only to administrators and not to end users. End users, on the other hand, can only *estimate* the bandwidth of links or paths from end-to-end measurements, without any information from network routers. Even network administrators sometimes need to determine the bandwidth from hosts under their control to hosts outside their infrastructures, so they also rely on end-to-end measurements. This article focuses on *end-to-end bandwidth measurement techniques* performed by the end hosts of a path without requiring administrative access to intermediate routers along the path.

Differences in terminology often obscure what methodology is suitable for measuring which metric. While all bandwidth estimation tools attempt to identify "bottlenecks," it is not always clear how to map this vague notion of bandwidth to specific performance metrics. In fact, in some cases it is not clear whether a particular methodology actually measures the bandwidth metric it claims to measure. Additionally, tools

---

This work was supported by the SciDAC program of the US Department of Energy (awards # DE-FC02-01ER25466 and # DE-FC02-01ER25467).

0890-8044/03/$17.00 © 2003 IEEE

employing similar methodologies may yield significantly different results. This article clarifies which metric each bandwidth measurement methodology estimates. We then present a taxonomy of major publicly available bandwidth measurement tools, including *pathchar*, *pchar*, *nettimer*, *pathrate*, and *pathload*, commenting on their unique characteristics. Some bandwidth estimation tools are also available commercially, such as AppareNet [1]. However, the measurement methodology of commercial tools is not openly known. Therefore, we refrain from classifying them together with publicly available tools.

The rest of this article is structured as follows. We define key bandwidth-related metrics. The most prevalent measurement methodologies for the estimation of these metrics are described. We present a taxonomy of existing bandwidth measurement tools. We then summarize the article.

Bandwidth-Related Metrics

In this section we introduce three bandwidth metrics: capacity, available bandwidth, and bulk transfer capacity (BTC). The first two are defined for both individual links and end-to-end paths, while BTC is usually defined only for an end-to-end path.

In the following discussion we distinguish between links at the data link layer (layer 2) and links at the IP layer (layer 3). We call the former *segments* and the latter *hops*. A segment normally corresponds to a physical point-to-point link, a virtual circuit, or a shared access local area network (e.g., an Ethernet collision domain or a fiber distributed data interface, FDDI, ring). In contrast, a hop may consist of a sequence of one or more segments, connected through switches, bridges, or other layer 2 devices. We define an *end-to-end path* $\mathcal{P}$ from an IP host $\mathcal{S}$ (source) to another host $\mathcal{V}$ (sink) as the sequence of hops that connect $\mathcal{S}$ to $\mathcal{V}$.

Capacity

A layer 2 link, or segment, can normally transfer data at a constant bit rate, which is the *transmission rate* of the segment. For instance, this rate is 10 Mb/s on a 10BaseT Ethernet segment, and 1.544 Mb/s on a T1 segment. The transmission rate of a segment is limited by both the physical bandwidth of the underlying propagation medium as well as its electronic or optical transmitter/receiver hardware.

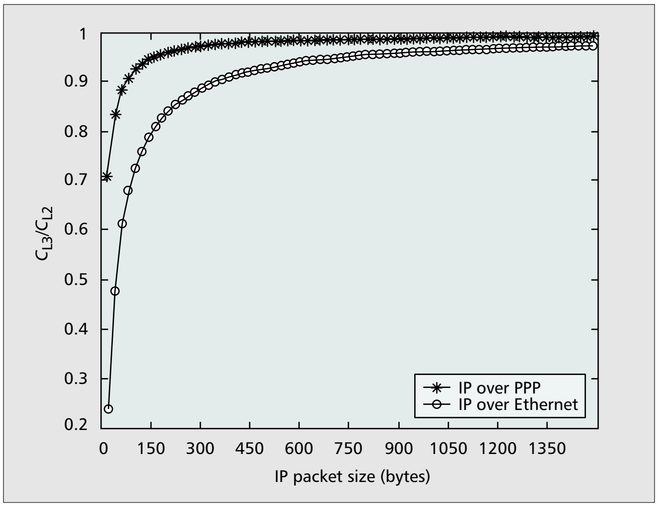
At the IP layer a hop delivers a lower rate than its nominal transmission rate due to the overhead of layer 2 encapsulation and framing. Specifically, suppose the nominal capacity of a segment is C_{L2}. The transmission time for an IP packet of size L_{L3} bytes is

$$\Delta_{L3} = \frac{L_{L3} + H_{L2}}{C_{L2}}, \tag{1}$$

where H_L is the total layer 2 overhead (in bytes) needed to encapsulate the IP packet. So the capacity C_{L3} of that segment at the IP layer is

$$C_{L3} = \frac{L_{L3}}{\Delta_{L3}} = \frac{L_{L3}}{\dfrac{L_{L3} + H_{L2}}{C_{L2}}} = C_{L2}\frac{1}{1 + \dfrac{H_{L2}}{L_{L3}}}. \tag{2}$$

Note that the IP layer capacity depends on the size of the IP packet relative to the layer 2 overhead. For 10BaseT Ethernet, C_{L2} is 10 Mb/s and H_{L2} is 38 bytes (18 bytes for the Ethernet header, 8 bytes for the frame preamble, and the equivalent of 12 bytes for the interframe gap). So the capacity the hop can deliver to the IP layer is 7.24 Mb/s for 100-byte packets, and 9.75 Mb/s for 1500-byte packets. Figure 1 shows the fraction of layer 2 transmission rate delivered to the IP layer as a function of packet size for Ethernet and Point-to-Point Protocol (PPP) layer 2 encapsulations. For PPP transmissions we assume that the maximum transmission unit (MTU) is 1500 bytes while the layer 2 overhead (without any additional data link encapsulation) is 8 bytes.



■ Figure 1. *The fraction of segment capacity delivered to the IP layer, as a function of packet size.*

We define *the capacity* C_i *of a hop* i *to be the maximum possible IP layer transfer rate at that hop*. From Eq. 2 the maximum transfer rate at the IP layer results from MTU-sized packets. So we define *the capacity of a hop as the bit rate, measured at the IP layer, at which the hop can transfer MTU-sized IP packets.*

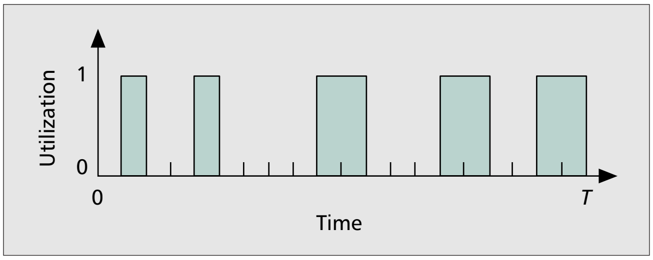
Extending the previous definition to a network path, the capacity C of an end-to-end path is the maximum IP layer rate the path can transfer from source to sink. In other words, the capacity of a path establishes an upper bound on the IP layer throughput a user can expect to get from that path. The minimum link capacity in the path determines the end-to-end capacity C, that is,

$$C = \min_{i=1,\ldots,H} C_i, \tag{3}$$

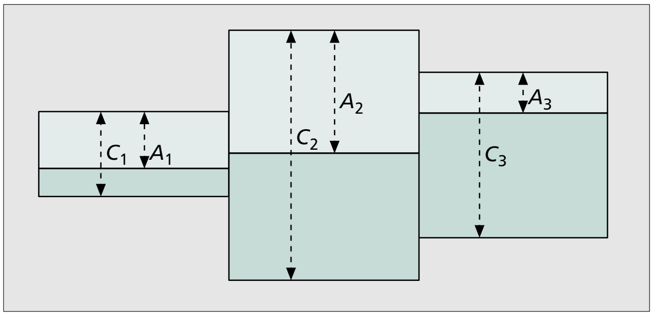
where C_i is the capacity of the ith hop, and H is the number of hops in the path. The hop with the minimum capacity is the *narrow link* on the path.

Some paths include traffic shapers or rate limiters, complicating the definition of capacity. Specifically, a traffic shaper at a link can transfer a peak rate P for a certain burst length B, and a lower sustained rate S for longer bursts. Since we view the capacity as an upper bound on the rate a path can transfer, it is natural to define the capacity of such a link based on peak rate P rather than sustained rate S. On the other hand, a rate limiter may deliver only a fraction of its underlying segment capacity to an IP layer hop. For example, Internet service providers (ISPs) often use rate limiters to share the capacity of an OC-3 link among different customers, charging each customer based on the magnitude of their bandwidth share. In this case we define the capacity of that hop to be the IP layer rate limit of that hop.

Finally, we note that some layer 2 technologies do not operate with a constant transmission rate. For instance, IEEE 802.11b wireless LANs transmit their frames at 11, 5.5, 2, or 1 Mb/s, depending on the bit error rate of the wireless medium. The previous definition of link capacity can be used for such technologies during time intervals in which the capacity remains constant.



■ Figure 2. *Instantaneous utilization for a link during a time period (0,T).*



■ Figure 3. *A pipe model with fluid traffic for a three-hop network path.*

Available Bandwidth

Another important metric is the *available bandwidth* of a link or end-to-end path. The available bandwidth of a link relates to the unused or *spare* capacity of the link during a certain time period. So even though the capacity of a link depends on the underlying transmission technology and propagation medium, the available bandwidth of a link additionally depends on the traffic load at that link, and is typically a time-varying metric.

At any specific instant in time, a link is either transmitting a packet at full link capacity or idle, so the instantaneous utilization of a link can only be either 0 or 1. Thus, any meaningful definition of available bandwidth requires time averaging of the instantaneous utilization over the time interval of interest. The average utilization $\bar{u}(t-\tau, t)$ for a time period $(t-\tau, t)$ is given by

$$\bar{u}(t-\tau,t)=\frac{1}{\tau}\int_{t-\tau}^{t}u(x)dx, \tag{4}$$

where $u(x)$ is the instantaneous available bandwidth of the link at time x. We refer to time length τ as the *averaging timescale* of the available bandwidth. Figure 2 illustrates this averaging effect. In this example the link is used during eight out of 20 time intervals between 0 and T, yielding an average utilization of 40 percent.

Let us now define the available bandwidth of a hop i over a certain time interval. If C_i is the capacity of hop i and u_i is the average utilization of that hop in the given time interval, the average available bandwidth A_i of hop i is given by the unutilized fraction of capacity,

$$A_i = (1-u_i)C_i. \tag{5}$$

Extending the previous definition to an H-hop path, the available bandwidth of the end-to-end path is the minimum available bandwidth of all H hops,

$$A = \min_{i=1,\ldots,H} A_i. \tag{6}$$

The hop with the minimum available bandwidth is called the *tight link*[1] of the end-to-end path.

Figure 3 shows a *pipe model with fluid traffic* representation of a network path, where each link is represented by a pipe. The width of each pipe corresponds to the relative capacity of the corresponding link. The shaded area of each pipe shows the utilized part of that link's capacity, while the unshaded area shows the spare capacity. The minimum link capacity C_1 in this example determines the end-to-end capacity, while the minimum available bandwidth A_3 determines the end-to-end available bandwidth. As shown in Fig. 3, the narrow link of a path may not be the same as the tight link.

Several methodologies for measuring available bandwidth make the assumption that the link utilization remains constant when averaged over time (i.e., they assume a *stationary traffic load* on the network path). While this assumption is reasonable over relatively short time intervals, diurnal load variations will impact measurements made over longer time intervals. Also note that constant average utilization (stationarity) does not preclude traffic variability (burstiness) or long-range dependence effects.

Since the average available bandwidth can change over time, it is important to measure it quickly. This is especially true for applications that use available bandwidth measurements to adapt their transmission rates. In contrast, the capacity of a path typically remains constant for long time intervals (e.g., until routing changes or link upgrades occur). Therefore the capacity of a path does not need to be measured as quickly as the available bandwidth.

TCP Throughput and Bulk Transfer Capacity

Another key bandwidth-related metric in TCP/IP networks is the throughput of a TCP connection. TCP is the major transport protocol in the Internet, carrying almost 90 percent of the traffic [2]. A TCP throughput metric would thus be of great interest to end users.

Unfortunately, it is not easy to define the expected throughput of a TCP connection. Several factors may influence TCP throughput, including transfer size, type of cross traffic (UDP or TCP), number of competing TCP connections, TCP socket buffer sizes at both sender and receiver sides, congestion along the reverse path, as well as the size of router buffers and capacity and load of each link in the network path. Variations in the specification and implementation of TCP, such as NewReno [3], Reno, or Tahoe, use of selective ACKs (SACKs) [4] vs. cumulative ACKs, selection of the initial window size [5], and several other parameters also affect TCP throughput.

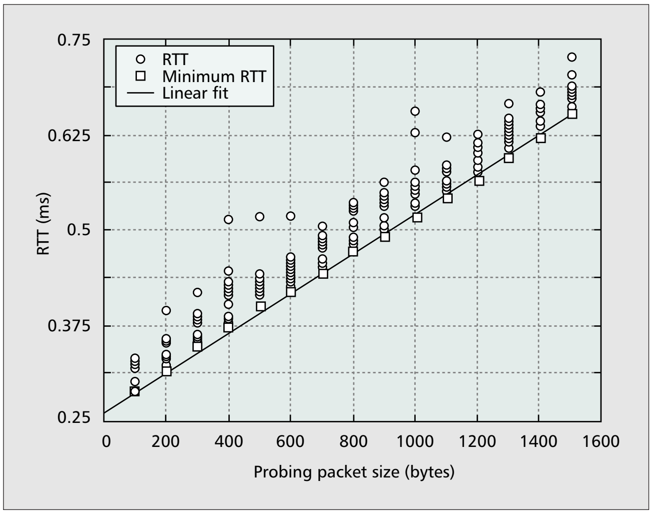
For instance, the throughput of a small transfer such as a typical Web page primarily depends on the initial congestion window, round-trip time (RTT), and slow-start mechanism of TCP, rather than on available bandwidth of the path. Furthermore, the throughput of a large TCP transfer over a certain network path can vary significantly when using different versions of TCP even if the available bandwidth is the same.

The BTC [6] defines a metric that represents the achievable throughput by a TCP connection. *BTC is the maximum throughput obtainable by a single TCP connection*. The connection must implement all TCP congestion control algorithms as specified in RFC 2581 [7]. However, RFC 2581 leaves some implementation details open, so a BTC measurement should also specify in detail several other important parameters about the exact implementation (or emulation) of TCP at the end hosts [6].

Note that the BTC and available bandwidth are fundamentally different metrics. BTC is TCP-specific, whereas the avail-

---

[1] *We choose to avoid the term bottleneck link because it has been used in the past to refer to both the link with the minimum capacity as well as the link with the minimum available bandwidth.*



■ Figure 4. *RTT measurements, minimum RTTs, and the least squares linear fit of the minimum RTTs for the first hop of a path.*

able bandwidth metric does not depend on a specific transport protocol. The BTC depends on how TCP shares bandwidth with other TCP flows, while the available bandwidth metric assumes that the average traffic load remains constant and estimates the additional bandwidth a path can offer before its tight link is saturated. To illustrate this point, suppose a single-link path with capacity C is saturated by a single TCP connection. The available bandwidth in this path would be zero due to path saturation, but the BTC would be about $C/2$ if the BTC connection has the same RTT as the competing TCP connection.

Bandwidth Estimation Techniques

This section describes existing bandwidth measurement techniques for estimating capacity and available bandwidth in individual hops and end-to-end paths. We focus on four major techniques: variable packet size (VPS) probing, packet pair/train dispersion (PPTD), self-loading periodic streams (SLoPS), and trains of packet pairs (TOPP). VPS estimates the capacity of individual hops, PPTD estimates end-to-end capacity, and SLoPS and TOPP estimate end-to-end available bandwidth. There is no currently known technique to measure available bandwidth of individual hops.

In the following we assume that during the measurement of a path $\mathcal{P}$ its route remains the same and its traffic load is stationary. Dynamic changes in routing or load can create errors in any measurement methodology. Unfortunately, most currently available tools do not check for dynamic route or load changes during the measurement process.

Variable Packet Size Probing

VPS probing aims to measure the capacity of each hop along a path. Bellovin [8] and Jacobson [9] were the first to propose and explore the VPS methodology. Subsequent work improved the technique in several ways [10–12]. The key element of the technique is to measure the RTT from the source to each hop of the path as a function of the probing packet size. VPS uses the time-to-live (TTL) field of the IP header to force probing packets to expire at a particular hop. The router at that hop discards the probing packets, returning ICMP time-exceeded error messages back to the source. The source uses the received ICMP packets to measure the RTT to that hop.

The RTT to each hop consists of three delay components in the forward and reverse paths: serialization delays, propagation delays, and queuing delays. The serialization delay of a packet of size L at a link of transmission rate C is the time to transmit the packet on the link, equal to L/C. The propagation delay of a packet at a link is the time it takes for each bit of the packet to traverse the link, and is independent of the packet size. Finally, queuing delays can occur in the buffers of routers or switches when there is contention at the input or output ports of these devices.

VPS sends multiple probing packets of a given size from the sending host to each layer 3 device along the path. The technique assumes that at least one of these packets, together with the ICMP reply it generates, will not encounter any queuing delays. Therefore, the minimum RTT measured for each packet size will consist of two terms: a delay that is independent of packet size and mostly due to propagation delays, and a term proportional to the packet size due to serialization delays at each link along the packet's path. Specifically, the minimum RTT $T_i(L)$ for a given packet size L up to hop i is expected to be

$$T_i(L) = \alpha + \sum_{k=1}^{i} \frac{L}{C_k} = \alpha + \beta_i L, \tag{7}$$

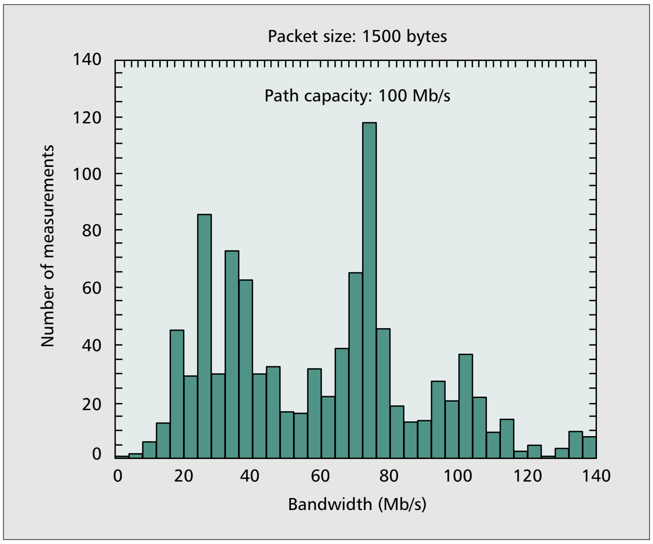
where:

- C_k: capacity kth hop
- α: delays up to hop i that do not depend on the probing packet size L
- β_i: slope of minimum RTT up to hop i against probing packet size L, given by

$$\beta_i = \sum_{k=1}^{i} \frac{I}{C_k}. \tag{8}$$

Note that all ICMP replies have the same size, independent of L; thus, the α term includes their serialization delay along with the sum of all propagation delays in the forward and reverse paths.

The minimum RTT measurements for each packet size up to hop i estimates the term β_i, as in Fig. 4. Repeating the minimum RTT measurement for each hop $i = 1, \ldots, H$, the capacity estimate at each hop i along the forward path is



■ Figure 6. *A histogram of capacity measurements from 1000 packet pair experiments in a 100 Mb/s path.*

$$C_i = \frac{1}{\beta_i - \beta_{i-1}}. \tag{9}$$

Figure 4 illustrates the VPS technique for the first hop of a path. The slope of the linear interpolation of the minimum RTT measurements is the inverse of the capacity estimate at that hop.

Unfortunately, VPS probing may yield significant capacity underestimation errors if the measured path includes store-and-forward layer 2 switches [13]. Such devices introduce serialization delays of the L/C type, but they do not generate ICMP TTL-expired replies because they are not visible at the IP layer. Modifying VPS probing to avoid such errors remains an active research problem [12].

Packet Pair/Train Dispersion Probing

Packet pair probing is used to measure the end-to-end capacity of a path. The source sends multiple *packet pairs* to the receiver. Each packet pair consists of two packets of the same size sent back to back. The dispersion of a packet pair at a specific link of the path is the time distance between the last bit of each packet. Packet pair techniques originate from seminal work by Jacobson [14], Keshav [15], and Bolot [16].

Figure 5 shows the dispersion of a packet pair before and after the packet pair goes through a link of capacity C_i assuming that the link does not carry other traffic. If a link of capacity C_0 connects the source to the path and the probing packets are of size L, the dispersion of the packet pair at that first link is $\Delta_0 = L/C_0$. In general, if the dispersion prior to a link of capacity C_i is Δ_{in}, the dispersion after the link will be

$$\Delta_{out} = \max\left(\Delta_{in}, \frac{L}{C_i}\right), \tag{10}$$

assuming again that there is no other traffic on that link.

After a packet pair goes through each link along an otherwise empty path, the dispersion Δ_R the receiver will measure is

$$\Delta_R = \max_{i=0,\ldots,H}\left(\frac{L}{C_i}\right) = \frac{L}{\min_{i=0,\ldots,H}(C_i)} = \frac{L}{C}, \tag{11}$$

where C is the end-to-end capacity of the path. Thus, the receiver can estimate the path capacity from $C = L/\Delta_R$.

Admittedly, the assumption that the path is empty of any other traffic (referred to here as *cross traffic*) is far from realistic. Even worse, cross traffic can either increase or decrease the dispersion Δ_R, causing underestimation or overestimation, respectively, of the path capacity. Capacity underestimation occurs if cross traffic packets are transmitted between the probing packet pair at a specific link, increasing the dispersion to more than L/C. Capacity overestimation occurs if cross traffic delays the first probe packet of a packet pair more than the second packet at a link that follows the path's narrow link.

Sending many packet pairs and using statistical methods to filter out erroneous bandwidth measurements mitigates the effects of cross traffic. Unfortunately, standard statistical approaches such as estimating the median or the mode of the packet pair measurements do not always lead to correct estimation [17]. Figure 6 illustrates why, showing 1000 packet pair measurements at a path from the University of Wisconsin to CAIDA (at the University of California, San Diego, UCSD), for which the path capacity is 100 Mb/s. Note that most of the measurements underestimate the capacity, while the correct measurements form only a local mode in the histogram. Identifying the correct capacity-related mode is a challenging task.

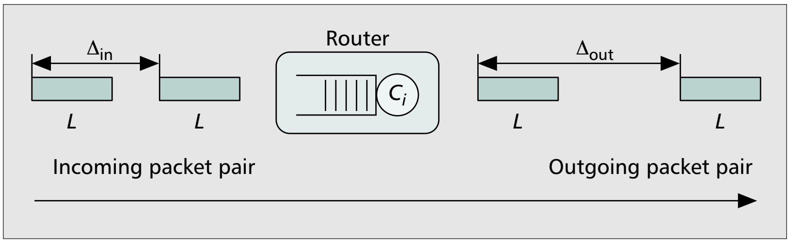


Figure 5. *Packet pair dispersion.*

Several other methodologies proposed in the literature perform capacity estimation using packet pair measurements [17–21]. Reference [18] proposes union and intersection statistical filtering as well as variable-sized packets to reduce the intensity of sub-capacity local modes. Reference [19] proposes an elaborate packet bunch method (PBM) driven by the intensity of local modes in the packet pair bandwidth distribution. Reference [20] uses kernel density estimation instead of histograms to detect the mode of the packet pair distribution, and [17] analyzes the local modes of the packet pair distribution and also uses a lower bound of the path capacity measured with long packet trains. Finally, [21] uses delay variations instead of packet pair dispersion, and peak detection rather than local mode detection. No investigation into the relative merits and drawbacks of these techniques has occurred to date.

Packet train probing extends packet pair probing by using multiple back-to-back packets. The dispersion of a packet train at a link is the amount of time between the last bit of the first and last packets. After the receiver measures the end-to-end dispersion $\Delta_R(N)$ for a packet train of length N, it calculates a *dispersion rate* D as

$$D = \frac{(N-1)L}{\Delta_R(N)}. \tag{12}$$

What is the physical meaning of this dispersion rate? If the path has no cross traffic, the dispersion rate will be equal to the path capacity, the same as with packet pair probing. However, cross traffic can render the dispersion rate significantly lower than the capacity.

To illustrate this effect, consider the case of a two-hop path. The source sends packet trains of length N through an otherwise empty link of capacity C_0. The probing packets have a size of L bytes. The second link has a capacity $C_1 < C_0$, and carries cross traffic at an average rate of $R_c < C_1$. We assume that the links use first come first served (FCFS) buffers. The dispersion of the packet train after the first link is $\Delta_1 = L(N - 1)/C_0$, while the train dispersion after the second link is

$$\Delta_2 = \frac{(N-1)L + X_c}{C_1}, \tag{13}$$

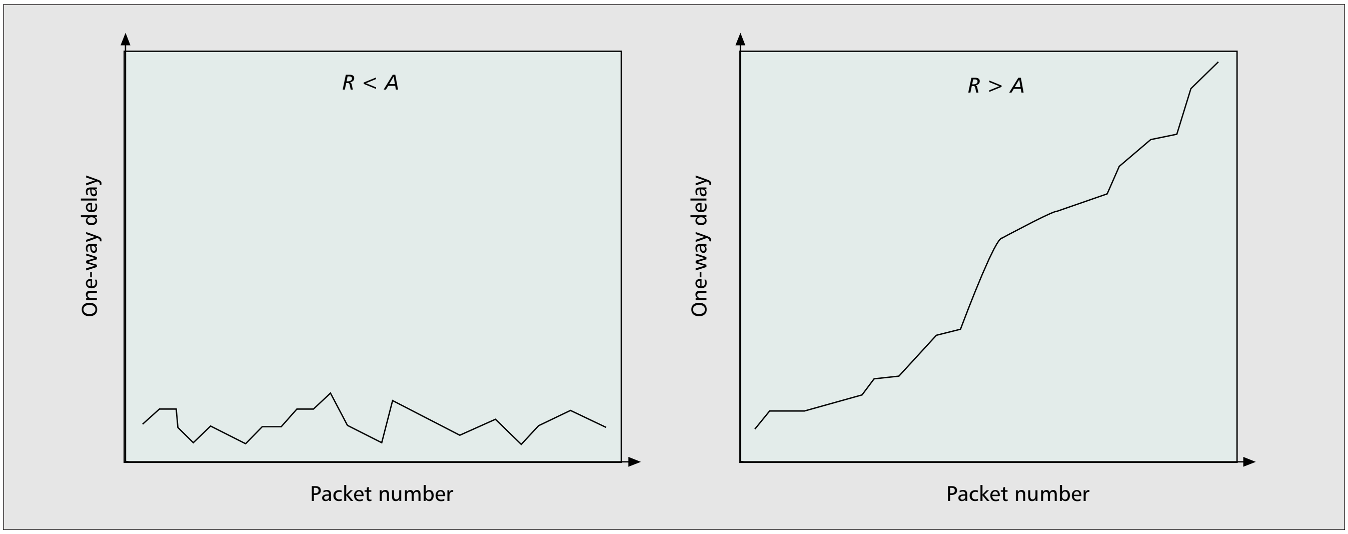
where X_c is the amount of cross traffic (in bytes) that will arrive at the second link during the arrival of the packet train at that link. The expected value of X_c is

$$E[X_c] = R_c\Delta_1 = R_c\frac{(N-1)L}{C_0}, \tag{14}$$

so the average dispersion rate ADR the receiver measures is

$$ADR \overset{\Delta}{=} \frac{(N-1)L}{E[\Delta_2]} = \frac{C_1}{1+\frac{R_c}{C_0}} \le C_1. \tag{15}$$

As the train length N increases, the variance in the amount of cross traffic X_c that interferes with the probing packet train decreases, reducing also the variance of the dispersion rate D.



■ Figure 7. *One-way delays increase when the stream rate* R *is larger than the available bandwidth* A*, but do not increase when* R *is lower than* A.

Equation 15 shows the following important properties for the average dispersion rate ADR. First, if $R_c > 0$, ADR is less than the path capacity. Second, ADR *is not related to the available bandwidth in the path* (as was previously assumed in [18]), which is $A = C_1 - R_c$ in this example. In fact, it is easy to show that ADR is larger than the available bandwidth (ADR > A) if $R_c > 0$. Finally, ADR *is independent of the packet train length* N. However, N affects the variance of the measured dispersion rate D around its mean ADR, with longer packet trains (larger N) reducing the variance in D.

PPTD probing techniques typically require double-ended measurements, with measurement software running at both the source and the sink of the path. It is also possible to perform PPTD measurements without access at the sink, by forcing the receiver to send some form of error message (e.g., ICMP port-unreachable or TCP RST packets) in response to each probe packet. In this case the reverse path capacities and cross traffic may affect the results.

Self-Loading Periodic Streams (SLoPS)

SLoPS is a recent measurement methodology for measuring end-to-end available bandwidth [22]. The source sends a number $K \approx 100$ of equal-sized packets (a *periodic packet stream*) to the receiver at a certain rate R. The methodology involves monitoring variations in the one-way delays of the probing packets. If the stream rate R is greater than the path's available bandwidth A, the stream will cause a short-term overload in the queue of the tight link. One-way delays of the probing packets will keep increasing as each packet of the stream queues up at the tight link. On the other hand, if the stream rate R is lower than the available bandwidth A, the probing packets will go through the path without causing increasing backlog at the tight link, and their one-way delays will not increase. Figure 7 illustrates the two cases.

In SLoPS the sender attempts to bring the stream rate R close to the available bandwidth A, following an iterative algorithm similar to binary search. The sender probes the path with successive packet trains of different rates, while the receiver notifies the sender about the one-way delay trend of each stream. The sender also makes sure that the network carries no more than one stream at any time. Also, the sender creates a silent period between successive streams in order to keep the average probing traffic rate to less than 10 percent of the available bandwidth on the path.

The available bandwidth estimate A may vary during the measurements. SLoPS detects such variations when it notices that the one-way delays of a stream do not show a clear increasing or nonincreasing trend. In that case the methodology reports a *grey region*, which is related to the variation range of A during the measurements.

Trains of Packet Pairs

Melander *et al.* proposed a measurement methodology to estimate the available bandwidth of a network path [23, 24]. TOPP sends many packet pairs at gradually increasing rates from the source to the sink. Suppose a packet pair is sent from the source with initial dispersion Δ_S. The probing packets have a size of L bytes; thus, the offered rate of the packet pair is $R_o = L/\Delta_S$. If R_o is more than the end-to-end available bandwidth A, the second probing packet will be queued behind the first probing packet, and the *measured rate* at the receiver will be $R_m < R_o$. On the other hand, if $R_o < A$, TOPP assumes that the packet pair will arrive at the receiver with the same rate it had at the sender (i.e., $R_m = R_o$). Note that this basic idea is analogous to SLoPS. In fact, most of the differences between the two methods are related to the statistical processing of the measurements. Also, TOPP increases the offered rate linearly, while SLoPS uses a binary search to adjust the offered rate. An important difference between TOPP and SLoPS is that TOPP can also estimate the capacity of the tight link of the path. Note that this capacity may be higher than the capacity of the path if the narrow and tight links are different.

To illustrate TOPP (Fig. 8), consider a single-link path with capacity C, available bandwidth A, and average cross traffic rate $R_c = C - A$. TOPP sends packet pairs with an increasing offered rate R_o. When R_o becomes larger than A, the measured rate of the packet pair at the receiver will be

$$R_m = \frac{R_o}{R_o + R_c} C \tag{16}$$

or

$$\frac{R_o}{R_m} = \frac{R_o + R_c}{C}. \tag{17}$$

TOPP estimates the available bandwidth A to be the maximum offered rate such that $R_o \approx R_m$. Equation 17 is used to estimate the capacity C from the slope of R_o/R_m vs. R_o.

In paths with multiple links, the R_o/R_m curve may show

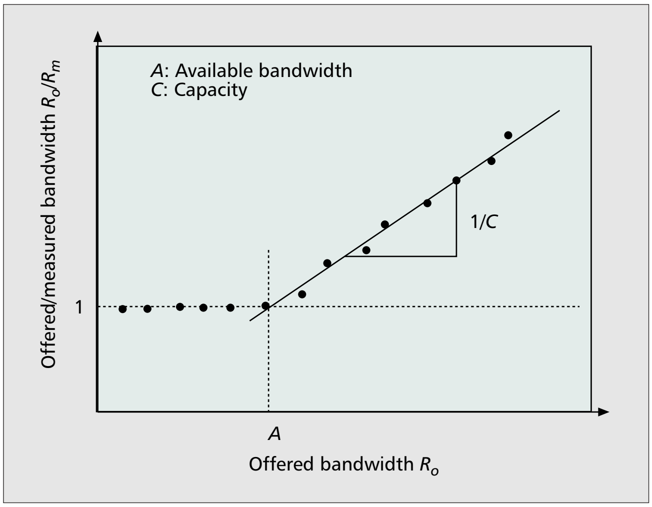


Figure 8. *Offered bandwidth over measured bandwidth in TOPP for a single-hop path.*

multiple slope changes due to queuing at links having higher available bandwidth than A. Unfortunately, the estimation of bandwidth characteristics at those links depends on their sequencing in the path [24].

Other Bandwidth Estimation Methodologies

Several other bandwidth estimation methodologies have been proposed in the last few years. We cannot present these methodologies in detail due to space constraints. In summary, [25] defines the *available capacity* as the amount of data that can be inserted in the network in order to meet some permissible delay. The estimation methodology of [26] estimates the available bandwidth of a path if queuing delays occur only at the tight link. Reference [27] estimates the utilization of a single bottleneck, assuming Poisson arrivals and either exponentially distributed or constant packet sizes. References [28, 29] propose available bandwidth estimation techniques similar to SLoPS and TOPP, but using different packet stream patterns and focusing on reducing measurement latency. Finally, [30] uses packet dispersion techniques to measure the capacity of targeted subpaths in a path.

A Taxonomy of Bandwidth Estimation Tools

This section provides a taxonomy of all publicly available bandwidth estimation tools known to the authors. Table 1 gives the names of these tools together with the target bandwidth metric they try to estimate and the basic methodology used. Due to space constraints we do not provide URLs for these tools, but they can be found with any Web search engine. An up-to-date taxonomy of network measurement tools is maintained online at [31].

Per-Hop Capacity Estimation Tools

These tools use the VPS probing technique to estimate the capacity of each hop in the path. The minimum of all hop estimates is the end-to-end capacity. These tools require superuser privileges because they need access to raw IP sockets to read ICMP messages.

Pathchar was the first tool to implement VPS probing, opening the area of bandwidth estimation research. This tool was written by Van Jacobson and released in 1997 [9]. Its source code is not publicly available.

Clink provides an open source tool to perform VPS probing. The original tool runs only on Linux. *Clink* differs from *pathchar* by using an "even-odd" technique [10] to generate interval capacity estimates. Also, when encountering routing instability, clink collects data for all the paths it encounters until one of the paths generates enough data to yield a statistically significant estimate.

Pchar is another open source implementation of VPS probing. *Libpcap* is used to obtain kernel-level timestamps. *Pchar* provides three different linear regression algorithms to obtain the slope of the minimum RTT measurements against the probing packet size. Different types of probing packets are supported, and the tool is portable to most UNIX platforms.

End-to-End Capacity Estimation Tools

These tools attempt to estimate the capacity of the narrow link along an end-to-end path. Most of them use the packet pair dispersion technique.

Bprobe uses packet pair dispersion to estimate the capacity of a path. The original tool uses SGI-specific utilities to obtain high-resolution timestamps and to set a high priority for the tool process. *Bprobe* processes packet pair measurements with an interesting union and intersection filtering technique in an attempt to discard packet pair measurements affected by cross traffic. In addition, *bprobe* uses variable-sized probing packets to improve the accuracy of the tool when cross traffic packets are of a few fixed sizes (e.g., 40, 576, or 1500 bytes). *Bprobe* requires access only at the sender side of a path, because the target host (receiver) responds to the sender's ICMP echo packets with ICMP echo replies. Unfortunately, ICMP replies are sometimes rate-limited to avoid denial-of-service attacks, negatively impacting measurement accuracy.

Nettimer can run as either a VPS probing tool or a packet pair tool. However, the documentation on how to use it as a VPS tool is not available, so it is primarily known as a capacity estimation packet pair tool. *Nettimer* uses a sophisticated statistical technique called *kernel density estimation* to process packet pair measurements. A kernel density estimator identifies the dominant mode in the distribution of packet pair measurements without assuming a certain origin for the bandwidth distribution, overcoming the corresponding limitation of histogram-based techniques.

Pathrate collects many packet pair measurements using various probing packet sizes. Analyzing the distribution of the resulting measurements reveals all local modes, one of which typically relates to the capacity of the path. Then *pathrate* uses long packet trains to estimate the average dispersion rate ADR of the path. ADR is never larger than the capacity, so it provides a reliable lower bound on path capacity. Eventually, *pathrate* estimates C as the strongest local mode in the packet pair bandwidth distribution that is larger than ADR. *Pathrate* does not require superuser privileges but does require software installation at both end hosts of the path.

Sprobe is a lightweight capacity estimation tool that provides a quick capacity estimate. The tool runs only at the source of the path. To measure the capacity of the forward path from the source to a remote host, *sprobe* sends a few packet pairs (normally TCP SYN packets) to the remote host. The remote host replies with TCP RST packets, allowing the sender to estimate the packet pair dispersion in the forward path. If the remote host runs a Web or gnutella server, the tool can estimate the capacity in the reverse path — from the remote host to the source — by initiating a short file transfer from the remote host and analyzing the dispersion of the packet pairs TCP sends during slow start.

Available Bandwidth Estimation Tools

Cprobe was the first tool to attempt to measure end-to-end available bandwidth. *Cprobe* measures the dispersion of a train of eight maximum-sized packets. However, it has been

Tool	Author	Measurement metric	Methodology
pathchar	Jacobson	Per-hop capacity	Variable packet size
clink	Downey	Per-hop capacity	Variable packet size
pchar	Mah	Per-hop capacity	Variable packet size
bprobe	Carter	End-to-end capacity	Packet pairs
nettimer	Lai	End-to-end capacity	Packet pairs
pathrate	Dovrolis-Prasad	End-to-end capacity	Packet pairs and trains
sprobe	Saroiu	End-to-end capacity	Packet pairs
cprobe	Carter	End-to-end available bandwidth	Packet trains
pathload	Jain-Dovrolis	End-to-end available bandwidth	Self-loading periodic streams
IGI	Hu	End-to-end available bandwidth	Self-loading periodic streams
pathChirp	Ribeiro	End-to-end available bandwidth	Self-loading packet chirps
treno	Mathis	Bulk transfer capacity	Emulated TCP throughput
cap	Allman	Bulk transfer capacity	Standardized TCP throughput
ttcp	Muuss	Achievable TCP throughput	TCP connection
Iperf	NLANR	Achievable TCP throughput	Parallel TCP connections
Netperf	NLANR	Achievable TCP throughput	Parallel TCP connections

Table 1. *Taxonomy of publicly available bandwidth estimation tools.*

previously shown [17, 23] that the dispersion of long packet trains measures the *dispersion rate*, which is not the same as the end-to-end available bandwidth. In general, the dispersion rate depends on all links in the path as well as on the train's initial rate. In contrast, the available bandwidth only depends on the tight link of the path.

Pathload implements the SLoPS methodology. *Pathload* requires access to both ends of the path, but does not require superuser privileges because it only sends UDP packets. *Pathload* reports a range rather than a single estimate. The center of this range is the average available bandwidth during the measurements, while the range itself estimates the variation of available bandwidth during the measurements.

More recently, two new tools have been proposed for available bandwidth estimation: *IGI* [29] and *pathChirp* [28]. These tools modify the self-loading methodology of TOPP or SLoPS, using different probing packet stream patterns. The main objective in *IGI* and *pathChirp* is to achieve similar accuracy to *pathload* but with shorter measurement latency.

TCP Throughput and BTC Measurement Tools

Treno was the first tool to measure the BTC of a path. *Treno* does not perform an actual TCP transfer, but instead emulates TCP by sending UDP packets to the receiver, forcing the receiver to reply with ICMP port-unreachable messages. In this way *Treno* does not require access at the remote end of the path. As with *bprobe*, the fact that ICMP replies are sometimes rate-limited can negatively affect the accuracy of *Treno*.

Cap is the first canonical implementation of the BTC measurement methodology. The National Internet Measurement Infrastructure (NIMI) [32] uses *cap* to estimate the BTC of a path. It has recently been shown that *cap* is more accurate than *Treno* in measuring BTC [33]. *Cap* uses UDP packets to emulate both the TCP data and ACK segments, and it requires access at both ends of the measured path.

TTCP, **NetPerf**, and **Iperf** are all benchmarking tools that use large TCP transfers to measure the achievable throughput in an end-to-end path. The user can control the socket buffer sizes and thus the maximum window size for the transfer. *TTCP* (Test TCP) was written in 1984, while the more recent *NetPerf* and *Iperf* have improved the measurement process and can handle multiple parallel transfers. All three tools require access at both ends of the path, but do not require superuser privileges.

Intrusiveness of Bandwidth Estimation Tools

We close this section with a note on the *intrusiveness* of bandwidth estimation tools. All active measurement tools inject probing traffic in the network and thus are intrusive to some degree. Here we make a first attempt to quantify this concept. Specifically, we say that *an active measurement tool is intrusive when its average probing traffic rate during the measurement process is significant compared to the available bandwidth in the path.*

VPS tools that send one probing packet and wait for an ICMP reply before sending the next are particularly nonintrusive since their traffic rate is a single packet per RTT. PPTD tools, or available bandwidth measurement tools, create short traffic bursts of high rate, sometimes higher than the available bandwidth in the path. These bursts, however, last for only a few milliseconds, with large silent periods between successive probing streams. Thus, the average probing traffic rate of these tools is typically a small fraction of the available bandwidth. For instance, the average probing rate in *pathload* is typically less than 10 percent of the available bandwidth. BTC tools can be classified as intrusive because they capture all of the available bandwidth for the duration of the measurements. On the other hand, BTC tools use or emulate TCP, and thus react to congestion in a TCP-friendly manner, while most of the VPS or PPTD tools do not implement congestion control and thus may have a greater impact on the available bandwidth. The benefits of bandwidth estimation must always be weighed against the cost and overhead of the measurements.

Summary

IP networks do not provide explicit feedback to end hosts regarding the load or capacity of the network. Instead, hosts use active end-to-end measurements in an attempt to estimate the bandwidth characteristics of paths they use. This article surveys the state of the art in bandwidth estimation techniques, reviewing metrics and methodologies employed and the tools that implement them. Several challenges remain. First, the accuracy of bandwidth estimation techniques must be improved, especially in high-bandwidth paths (e.g., greater than 500 Mb/s). Second, bandwidth estimation tools and techniques in this article assume that routers serve packets on a first come first served basis. It is not clear how these techniques will perform in routers with multiple queues (e.g., for different classes of service) or with virtual output-input queues. Finally, much work remains on how to best use bandwidth estimates to support applications, middleware, routing, and traffic engineering techniques, in order to improve end-to-end performance and enable new services.

References

[1] Jaalam Technologies, "The Apparent Network : Concepts and Terminology," http://www.jaalam.com/, Jan. 2003.

[2] S. McCreary and K. C. Claffy, "Trends in Wide Area IP Traffic Patterns," Tech. rep., CAIDA, Feb. 2000.

[3] S. Floyd and T. Henderson, "The NewReno Modification to TCP's Fast Recovery Algorithm," RFC 2582, Apr. 1999.

[4] M. Mathis *et al.*, "TCP Selective Acknowledgment Options," RFC 2018, Oct. 1996.

[5] M. Allman, S. Floyd, and C. Partridge, *Increasing TCP's Initial Window*, RFC 3390, Oct. 2002.

[6] M. Mathis and M. Allman, *A Framework for Defining Empirical Bulk Transfer Capacity Metrics*, RFC 3148, July 2001.

[7] M. Allman, V. Paxson, and W. Stevens, "TCP Congestion Control," IETF RFC 2581, Apr. 1999.

[8] S. Bellovin, "A Best-Case Network Performance Model," Tech. rep., ATT Research, Feb. 1992.

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

[10] A.B. Downey, "Using Pathchar to Estimate Internet Link Characteristics," *Proc. ACM SIGCOMM*, Sept. 1999, pp. 222–23.

[11] K. Lai and M. Baker, "Measuring Link Bandwidths Using a Deterministic Model of Packet Delay," *Proc. ACM SIGCOMM*, Sept. 2000, pp. 283–94.

[12] A. Pasztor and D. Veitch, "Active Probing using Packet Quartets," *Proc. Internet Measurement Wksp.*, 2002.

[13] R. S. Prasad, C. Dovrolis, and B. A. Mah, "The Effect of Layer-2 Store-and-Forward Devices on Per-Hop Capacity Estimation," *Proc. IEEE INFOCOM*, 2003.

[14] V. Jacobson, "Congestion Avoidance and Control," *Proc. ACM SIGCOMM*, Sept. 1988, pp. 314–29.

[15] S. Keshav, "A Control-Theoretic Approach to Flow Control," *Proc. ACM SIGCOMM*, Sept. 1991, pp. 3–15.

[16] J. C. Bolot, "Characterizing End-to-End Packet Delay and Loss in the Internet," *Proc. ACM SIGCOMM*, 1993, pp. 289–98.

[17] C. Dovrolis, P. Ramanathan, and D. Moore, "What do Packet Dispersion Techniques Measure?," *Proc. IEEE INFOCOM*, Apr. 2001, pp. 905–14.

[18] R. L. Carter and M. E. Crovella, "Measuring Bottleneck Link Speed in Packet-Switched Networks," *Perf. Eval.*, vol. 27, 28, 1996, pp. 297–318.

[19] V. Paxson, "End-to-End Internet Packet Dynamics," *IEEE/ACM Trans. Net.*, vol. 7, no. 3, June 1999, pp. 277-92.

[20] K. Lai and M. Baker, "Measuring Bandwidth," *Proc. IEEE INFOCOM*, Apr. 1999, pp. 235–45.

[21] A. Pasztor and D. Veitch, "The Packet Size Dependence of Packet Pair Like Methods," *IEEE/IFIP Int'l. Wksp. QoS*, 2002.

[22] M. Jain and C. Dovrolis, "End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput," *Proc. ACM SIGCOMM*, Aug. 2002, pp. 295–308.

[23] B. Melander, M. Bjorkman, and P. Gunningberg, "A New End-to-End Probing and Analysis Method for Estimating Bandwidth Bottlenecks," *IEEE Global Internet Symp.*, 2000.

[24] B. Melander, M. Bjorkman, and P. Gunningberg, "Regression-Based Available Bandwidth Measurements," *Int'l. Symp. Perf. Eval. Comp. and Telecommun. Sys.*, 2002.

[25] S. Banerjee and A. K. Agrawala, "Estimating Available Capacity of a Network Connection," *Proc. IEEE Int'l. Conf. Networks*, Sept. 2001.

[26] V. Ribeiro *et al.*, "Multifractal Cross-Traffic Estimation," *Proc. ITC Specialist Seminar on IP Traffic Measurement, Modeling, and Management*, Sept. 2000.

[27] S. Alouf, P. Nain, and D. Towsley, "Inferring Network Characteristics via Moment-Based Estimators," *Proc. IEEE INFOCOM*, Apr. 2001.

[28] V. Ribeiro *et al.*, "pathChirp: Efficient Available Bandwidth Estimation for Network Paths," *Proc. Passive and Active Measurements Wksp.*, Apr. 2003.

[29] N. Hu and P. Steenkiste, "Evaluation and Characterization of Available Bandwidth Probing Techniques," *IEEE JSAC*, 2003.

[30] K. Harfoush, A. Bestavros, and J. Byers, "Measuring Bottleneck Bandwidth of Targeted Path Segments," *Proc. IEEE INFOCOM*, 2003.

[31] CAIDA, http://www.caida.org/tools/taxonomy, Oct. 2002.

[32] V. Paxson, J. Adams, and M. Mathis, "An Architecture for Large-Scale Internet Measurement," *IEEE Commun. Mag.*, vol. 36, no. 8, 1998, pp. 48–54.

[33] M. Allman, "Measuring End-to-End Bulk Transfer Capacity," *Proc. ACM SIGCOMM Internet Measurement Wksp.*, Nov. 2001, pp. 139–43.

Biographies

RAVI PRASAD (ravi@cc.gatech.edu) is a graduate student working toward his Ph.D. at the College of Computing of the Georgia Institute of Technology (Georgia Tech). His current area of interest is networking, focusing on network measurements, bandwidth estimation methodologies, and their applications. He obtained his B.Tech. degree in ocean engineering and naval architecture from the Indian Institute of Technology, Kharagpur, and his M.C.E. in coastal engineering from the University of Delaware, Newark.

MARGARET MURRAY (marg@caida.org) is director of the Cooperative Association for Internet Data Analysis (CAIDA), based at the San Diego Supercomputer Center (SDSC). In addition to managing CAIDA people and funding, she runs CAIDA's bandwidth estimation test laboratory within SDSC's Network Performance Reference Laboratory. Her research interests include bandwidth estimation tool comparisons and DNS performance analysis. She received her Ph.D. degree in music (critical studies and experimental practices) from the University of California, San Diego. She also has many years of computer-related experience in system configuration and software development, including extensive work managing and coordinating heterogeneous systems in a variety of local and wide area networks.

CONSTANTINOS DOVROLIS [M'93/ACM'96] (dovrolis@cc.gatech.edu) is an assistant professor at the College of Computing of Georgia Tech. He received a computer engineering degree from the Technical University of Crete (Greece) in 1995, an M.S. degree from the University of Rochester in 1996, and a Ph.D. degree from the University of Wisconsin-Madison in 2000. His research interests include methodologies and applications of network measurements, bandwidth estimation, service differentiation, and router architectures.

KC CLAFFY (kc@caida.org) is founder and principal investigator of CAIDA, based at the SDSC. Her research focuses on collection, analysis, and visualization of wide-area Internet data on topology, workload, performance, and routing. CAIDA seeks, through the development of neutral open tools and methodologies, to promote cooperation among Internet service providers in face of commercialization and competition, and to meet engineering and traffic analysis requirements of the commercial Internet community. She received her M.S. and Ph.D. degrees in computer science from the University of California, San Diego.