

Measuring Capacity Bandwidth of Targeted Path Segments

Khaled Harfoush, *Member, IEEE*, Azer Bestavros, *Senior Member, IEEE*, and John Byers

Abstract—Accurate measurement of network bandwidth is important for network management applications as well as flexible Internet applications and protocols which actively manage and dynamically adapt to changing utilization of network resources. Extensive work has focused on two approaches to measuring bandwidth: measuring it hop-by-hop, and measuring it end-to-end along a path. Unfortunately, best-practice techniques for the former are inefficient and techniques for the latter are only able to observe bottlenecks visible at end-to-end scope. In this paper, we develop end-to-end probing methods which can measure bottleneck capacity bandwidth along *arbitrary, targeted subpaths* of a path in the network, including subpaths *shared* by a set of flows. We evaluate our technique through ns simulations, then provide a comparative Internet performance evaluation against hop-by-hop and end-to-end techniques. We also describe a number of applications which we foresee as standing to benefit from solutions to this problem, ranging from network troubleshooting and capacity provisioning to optimizing the layout of application-level overlay networks, to optimized replica placement.

Index Terms—Bottleneck bandwidth, content distribution, end-to-end measurement, overlay networks, packet-pair.

I. INTRODUCTION

MEASUREMENT of network bandwidth is important for many Internet applications and protocols, especially those involving the transfer of large files and those involving the delivery of content with real-time QoS constraints, such as streaming media. Some specific examples of applications which can leverage accurate bandwidth estimation include end-system multicast and overlay network configuration protocols [8], [24], [2], content location and delivery in peer-to-peer (P2P) networks [43], [5], network-aware cache or replica placement policies [25], [40], and flow scheduling and admission control policies at massively-accessed content servers [9]. In addition, accurate measurements of network bandwidth are useful to network operators concerned with problems such as capacity provisioning, traffic engineering, network troubleshooting and verification of service level agreements (SLAs).

Manuscript received October 21, 2005; revised August 08, 2006 and June 04, 2007; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor M. Roughan. First published December 12, 2008; current version published February 19, 2009. This work was supported in part by the National Science Foundation (NSF) Research Grants, NSF CAREER Award ANIR-0347226, NSF CAREER Award ANI-0093296, ANI Award #9986397, ANI Award #0095988, CISECSR Award #0720604, ENGEFRI Award #0735974, CISECNS Award #0524477, CISECNS Award #0520166, CNSITR Award #0205294, and CISEEIA RI Award #0202067.

K. Harfoush is with the Department of Computer Science, North Carolina State University, Raleigh, NC 27695 USA (e-mail: harfoush@cs.ncsu.edu).

A. Bestavros and J. Byers are with the Department of Computer Science, Boston University, Boston MA 02215 USA (e-mail: best@cs.bu.edu; byers@cs.bu.edu).

Digital Object Identifier 10.1109/TNET.2008.2008702

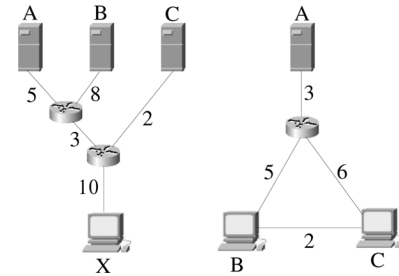


Fig. 1. Leveraging shared bandwidth measurement for optimizing parallel downloads (left) and overlay network organization (right). Numeric labels represent capacity bandwidth of path segments in Mbps.

Bandwidth Measurement: Two different measures used in end-to-end network bandwidth estimation are *capacity bandwidth*, or the maximum transmission rate that could be achieved between two hosts at the endpoints of a given path in the absence of any competing traffic, and *available bandwidth*, the portion of the capacity bandwidth along a path that could be acquired by a given flow at a given instant in time. Both of these measures are important, and each captures different relevant properties of the network. Capacity bandwidth is a static baseline measure that applies over long time-scales (up to the time-scale at which network paths change), and is independent of the particular traffic dynamics at a time instant. Available bandwidth provides a dynamic measure of the load on a path, or more precisely, the residual capacity of a path. Additional application-specific information must then be applied before making meaningful use of either measure. While measures of available bandwidth are certainly more useful for control or optimization of processes operating at short time scales, processes operating at longer time scales (e.g., server selection or admission control) will find estimates of both measures to be helpful. On the other hand, many network management applications (e.g., capacity provisioning) are concerned primarily with capacity bandwidth. We focus on measuring capacity bandwidth in this paper.

Catalyst Applications: To exemplify the type of applications that can be leveraged by the identification of shared capacity bandwidth (or more generally, the capacity bandwidth of an arbitrary, targeted subpath), we consider the two scenarios illustrated in Fig. 1. In the first scenario, a client must select two out of three sources to use to download data in parallel. This scenario may arise when downloading content in parallel from a subset of mirror sites or multicast sources [6], [42], [15], or from a subset of peer nodes in P2P environments [5]. In the second scenario, an overlay network must be set up between a single source and two destinations. This scenario may arise in ad-hoc networks and end-system multicast systems [8], [24].

For the first scenario illustrated in Fig. 1 (left), the greedy approach of selecting the two servers whose paths to the client have the highest end-to-end capacity bandwidth—namely, servers A and B—is not optimal, since the aggregate bandwidth to the client would be limited by the shared 3 Mbps capacity bandwidth from servers A and B to the client. To be able to select the pair of servers yielding the maximum *aggregate* bandwidth of 5 Mbps—namely A and C or B and C—the client needs to measure the shared capacity bandwidth between pairs of servers. Similarly, in the second scenario illustrated in Fig. 1 (right), the identification of the best set of routes for distributing content from source A to destinations B and C hinges on our ability to determine the capacity bandwidth of the shared portion of the AB and AC paths (as well as the end-to-end capacity bandwidth of path BC). Specifically, it is better to use the AB + BC links to provide 3 Mbps to client B and 2 Mbps to client C, rather than the AB + AC links for 1.5 Mbps to each (assuming fair sharing).

Paper Scope, Contributions, and Organization: In this paper we propose an *efficient* end-to-end measurement technique that yields the capacity bandwidth of an *arbitrary subpath* of a route between a set of end-points. By subpath, we mean a sequence of consecutive network links between any two *identifiable* nodes on that path. A node i on a path between a source s and a destination d is identifiable if it is possible to coerce a packet injected at the source s to exit the path at node i . One can achieve this by: 1) targeting the packet to i (if i 's IP address is known), or 2) forcing the packet to stop at i through the use of TTL field (if the hopcount from s to i is known), or 3) by targeting the packet to a destination d' , such that the paths from s to d and from s to d' are known diverge at node i . Our methods are much less *resource-intensive* than existing hop-by-hop methods for estimating bandwidth along a path and much more *general* than end-to-end methods for measuring capacity bandwidth. In particular, our method provides the following advantages over existing techniques: 1) it can estimate bandwidth on links not visible at end-to-end scope, and 2) it can measure the bandwidth of fast links following slow links as long as the ratio between the link speeds does not exceed the ratio between the largest and the smallest possible packet sizes that could be transmitted over these links.

The remainder of this paper is organized as follows. In Section II, we review existing literature. In Section III, we develop a basic probing toolkit, comprising existing methods and our new ideas. We compose several of these tools together in Sections IV and IV.E to measure capacity bandwidth along arbitrary subpaths, and capacity bandwidth shared by a set of flows, respectively. In Sections V, VI and VII, we present results of simulation, controlled laboratory experiments and Internet validation experiments, showing the effectiveness of our constructions.

II. RELATED WORK

One way of classifying bandwidth estimation techniques is based on whether they conduct hop-by-hop [20], [31], [11], [29], [30] or end-to-end [27], [4], [7] measurements.

Hop-by-hop techniques rely on incrementally probing routers along a path and timing their ICMP replies, whereas end-to-end techniques base their bandwidth estimation on end-host replies only. The techniques we present in this paper belong to the latter class, albeit at a granularity finer than that achievable using existing end-to-end techniques. Another classification of bandwidth measurement techniques is based on whether they measure the capacity bandwidth [20], [31], [11], [27], [7], [29], [30], [26] or the available bandwidth [1], [3], [19], [18], [23], [33], [39], [4], [7], [22], [14], [41], [44] of a path. The techniques we present in this paper are aimed at measuring capacity bandwidth. While collecting measurements only at the endpoints, our proposed measurement techniques are able to provide bandwidth estimates along arbitrary segments of a path, which is inherently different from other techniques.

In classifying bandwidth measurement techniques, one can also look at the probing methodology employed—namely, the number and sizes of packets in a probe. Probe structures considered in the literature include: (1) *single packet* probing [4], [20], [31], [11], (2) *packet bunch* probing, employing a group of packets sent back-to-back [7], [30], [45], [35], (3) *uniform packet-pair* probing, employing two back-to-back packets of the same size [27], [7], and (4) *non-uniform packet-pair* probing, employing two back-to-back packets of different sizes [29], [30]. The probing techniques we will propose can be classified as packet-bunch probes with non-uniform packet sizes.

Finally, one can classify bandwidth estimation techniques into *active* and *passive* techniques. Active techniques, comprising most of the work in the literature, send probes for the sole purpose of bandwidth measurement. Passive techniques rely on data packets for probing as exemplified in Lai and Baker's *nettimer* tool [30], which uses a packet-pair technique at the transport level to passively estimate capacity link bandwidth. The techniques we propose in this paper are applied actively.

Several research efforts have conducted thorough analysis of the properties and the limitations of bandwidth measurement techniques. In [10], [22], the authors study the impact of cross-traffic on packet-pair bandwidth estimation techniques. In [36], the authors highlight the problems associated with the use of small packets in packet-pair techniques due to the presence of variable size layer-2 headers in different links along the path. Our proposed techniques do not fall into the packet-pair techniques category and the impact of layer-2 headers on our techniques can be contained by appropriate *sizing* of our probing structures. The probing constructions most closely related to ours are the “packet-pair” [28] and “tailgating” [29] constructions. We discuss relevant technical properties of these constructions, which we employ and build upon in Section III.

III. PROBING TOOLKIT

In this section, we describe basic constructs of our probing sequences and corresponding terminology. With each probing construct, we describe its properties and point to its usefulness as a building block for the end-to-end measurement of subpath capacity bandwidth, which we describe in Section IV.

A. Basic Definitions and Assumptions

For the purposes of this paper, a *probe* is a sequence of one or more packets transmitted from a common origin. We say that any contiguous subsequence of packets within a probe are transmitted *back-to-back* if there is no time separation between transmission of the individual packets within the subsequence. As detailed in the related work section, back-to-back packets have been widely used in estimating the end-to-end bandwidth of a connection [4], [27], [7], [30], [29]. A *multi-destination* probe is one in which the constituent packets of the probe do not all target the same destination IP address. Multi-destination probes have begun to see wider use as emulations of notional multicast packets—many of the same end-to-end inferences that can be made with multicast packets can be made with multi-destination unicast probes (albeit with added complexity) [12], [17]. A *uniform* probe is one in which all of the constituent packets are of the same size; likewise, a *non-uniform* probe consists of packets of different sizes. Finally, we say that an individual packet is *hop-limited* if its TTL is set to an artificially small value so as not to reach the ostensible destination. Hop-limited packets can be used to trigger an ICMP response from an intermediate router and in other ways that we describe later in the paper.

Throughout the paper we use various probing techniques that rely on sending sequences of probes. The probing techniques differ in the number of packets constituting a probe, the size and the path traversed by each probe packet. They also differ in the host collecting the probing responses and the function used by this host to perform the required estimation.

Each packet p transmitted within a probe is parameterized by its size $s(p)$ in bits and its final destination, $D(p)$. In the event that a packet is hop-limited, it has a third parameter, its maximum hop-count, $h(p)$. To denote a probe, we refer to each probe packet with a distinct lowercase letter, and represent the sequential order in which they are transmitted from the probing host by writing them from left to right.

We denote interpacket spacing with square braces. As an example, $[pq][pq][r]$ would denote transmission of a pair of identical two-packet probes followed by a single packet probe which has different characteristics; packets in each of the two-packet probes are transmitted back-to-back while probes are not transmitted back-to-back. As another example, $[p]^r$ denotes a sequence of r identical probe packets that are sent back-to-back.

We use the term *interarrival time of packets p and q at a link* to denote the time elapsed between the arrival of the last byte of p and the arrival of the last byte of q at that link. Similarly, we use the term *interdeparture time* to denote the time elapsed between the transmission of the last byte of p and the transmission of the last byte of q . By these definitions, the interarrival time of packets p and q at a given link is the same as the interdeparture time of packets p and q at the preceding link on the path.

The constructions and analyses we present later in this paper, are conditioned on a set of basic assumptions about the network. These assumptions, which are common to most probing studies (e.g., [4], [7], [29], [30]), are enumerated below:

- 1) Routers are store-and-forward and use FIFO queueing.
- 2) Probing hosts can inject back-to-back packets into the network.

- 3) Host clock resolution is granular enough to enable accurate timing measurements.
- 4) Analytic derivations assume an environment free from cross-traffic.

Assumption 1 is needed to ensure that probe packet orderings are preserved. Assumptions 2 and 3 are easily enforceable using proper kernel capabilities. Assumption 4, while necessary for analysis, is typically discarded in experimental (simulation or implementation) settings to establish the robustness of the constructions in realistic settings.

B. Existing Probing Methods and Properties

One of the essential techniques that we build upon is the use of “packet-pairs”, originally used by Keshav [28], and subsequently refined by Carter and Crovella [7], Paxson [37], [38], [39] and Lai and Baker [30], to determine capacity bandwidth on a network path. Packet-pair techniques rely on the following property, which holds under an assumed network model discussed later in this section.

Lemma 1: Packet-Pair Property. Consider a path of n physical links L_1, L_2, \dots, L_n with capacity bandwidths b_1, b_2, \dots, b_n respectively. If a probe of the form $[pp]$ is injected at L_1 , with $D(p) = L_n$, then the interarrival time of the two constituent packets of this probe at L_n is $s(p)/(\min_k b_k)$ units of time.

An important corollary to Lemma 1 is that the capacity bandwidth across a set of links ($\min_k b_k$) can be estimated through measurement of packet interarrival times and knowledge of packet sizes.

Another closely related technique also used in our constructions is “packet-tailgating”. This technique was introduced by Lai and Baker in [29] and evaluated within their *nettimer* tool [30] to estimate the capacity bandwidth of all physical links along a path. The packet-tailgating technique hinges on the following property [30], which formulates the condition under which a non-uniform packet-pair remains back-to-back over a sequence of physical links.

Lemma 2: Tailgating Property. Consider a path of n physical links L_1, L_2, \dots, L_n with capacity bandwidths b_1, b_2, \dots, b_n respectively. If a probe of the form $[pq]$ is injected at L_1 , with $D(p) = D(q)$ at the end of link L_n and if $\forall k \leq n$, $\frac{s(p)}{s(q)} \geq \frac{b_{k+1}}{b_k}$, then $[pq]$ will remain back-to-back along the entire path.

C. Ensuring Back-to-Back Queuing at a Given Link

We now describe the first of our constructions—a construction that allows us to establish conditions that guarantee that all constituent packets of a probe will queue up back-to-back at a given intermediate link along a given path. We do so through the use of a (typically large) *pacers* packet, which leads the probe into the network.

Definition 1: A paced probe is a probe X sent back-to-back behind a large *pacers* packet p of the form $[pX]$. The pacers packet has a destination $D(p)$ at an intermediate point in the network. It leads the paced probe (its *followers*) up through this link as part of their trip and all the followers queue behind p in the queue at router $D(p)$. At this point, p is dropped.

The following lemma expresses the condition guaranteeing that probe packets remain back-to-back at the pacer packet's final destination.

Lemma 3: Let L be a sequence of n physical links L_1, L_2, \dots, L_n with capacity bandwidths b_1, b_2, \dots, b_n respectively. Also, let $[pq_1 \dots q_m]$ be a probe consisting of a set of m paced packets which are injected back-to-back behind a pacer packet, where $D(p) = L_k$ and $D(q_i) = L_n$. A sufficient condition for all follower packets q_i to queue behind p at link L_k is

$$\min_{1 \leq w \leq m} \left(\frac{s(p) + \sum_{j=1}^{w-1} s(q_j)}{\sum_{j=1}^w s(q_j)} \right) \geq \frac{b_k}{\min_{i \leq k} b_i}$$

Proof: It is not hard to see that if $[pq_1 \dots q_m]$ probe packets are injected back-to-back at L_1 then the interarrival time Δ_{k-1}^w between p and q_w over link L_{k-1} satisfies the inequality $\Delta_{k-1}^w \leq \frac{\sum_{j=1}^w s(q_j)}{\min_{1 \leq i \leq k-1} b_i}$. In order to guarantee that q_1 will queue behind p over L_k , the transmission time of p over L_k must be at least as large as the maximum Δ_{k-1}^1 value. That is the condition $\frac{s(p)}{b_k} \geq \frac{s(q_1)}{\min_{1 \leq i \leq k-1} b_i}$ (or equivalently $\frac{s(p)}{s(q_1)} \geq \frac{b_k}{\min_{1 \leq i \leq k-1} b_i}$) must be satisfied. Now, assuming that q_1 queues behind p over L_k , in order to guarantee that q_2 will queue behind them over L_k , the transmission time of p and q_1 over L_k must be at least as large as the maximum Δ_{k-1}^2 value. That is the condition $\frac{s(p)+s(q_1)}{b_k} \geq \frac{s(q_1)+s(q_2)}{\min_{1 \leq i \leq k-1} b_i}$ (or equivalently $\frac{s(p)+s(q_1)}{s(q_1)+s(q_2)} \geq \frac{b_k}{\min_{1 \leq i \leq k-1} b_i}$) must be satisfied. In general, assuming that $q_j \forall j = 1, \dots, w-1$ are queued back-to-back over L_k , in order to guarantee that q_w will queue behind them, the condition $\frac{s(p) + \sum_{j=1}^{w-1} s(q_j)}{\sum_{j=1}^w s(q_j)} \geq \frac{b_k}{\min_{1 \leq i \leq k-1} b_i}$ must be satisfied. In order for all the $[pq_1 \dots q_m]$ packets to be queued back-to-back over L_k , the condition $\frac{s(p) + \sum_{j=1}^{w-1} s(q_j)}{\sum_{j=1}^w s(q_j)} \geq \frac{b_k}{\min_{1 \leq i \leq k-1} b_i} \forall 1 \leq w \leq m$ (or equivalently $\min_{1 \leq w \leq m} \left(\frac{s(p) + \sum_{j=1}^{w-1} s(q_j)}{\sum_{j=1}^w s(q_j)} \right) \geq \frac{b_k}{\min_{1 \leq i \leq k-1} b_i}$) must be satisfied. This concludes the proof. ■

D. Preserving Packet Interarrival Times Over a Subpath

Our next construction allows us to tackle another challenge, which is to some extent complementary to pacing—namely how we can ensure that the interarrival time of two packets at a specific link L_i along a path can be *preserved* as these packets traverse additional links *en route* to their common destination L_n . The ability to preserve packet spacing over the subpath $L_i L_{i+1} \dots L_n$ enables us to measure such spacing remotely (at L_n). The following Lemma establishes a necessary condition for the preservation of packet spacing over a sequence of links.

Lemma 4: Preservation of Spacing. Consider a path of n physical links L_1, L_2, \dots, L_n with capacity bandwidths b_1, b_2, \dots, b_n respectively. If a probe of the form $[p][p]$ is injected at L_1 with $D(p) = L_n$ and an interarrival time of Δ , then Δ will be preserved over all links L_i if and only if $\frac{s(p)}{\Delta} \leq \min_{1 \leq k \leq n} b_k$.

Proof: $\frac{s(p)}{\min_{1 \leq k \leq n} b_k}$ is the transmission time of each probe packet, p , over the bottleneck link. If $\frac{s(p)}{\min_{1 \leq k \leq n} b_k} > \Delta$ then

the two probe packets will be transmitted back-to-back over the bottleneck link and Δ will be expanded to $\frac{s(p)}{\min_{1 \leq k \leq n} b_k}$ starting from the bottleneck link till the destination L_n . ■

Lemma 4 shows that in order to avoid skewing the interarrival time (Δ_i) through subpath L_{i+1}, \dots, L_n , the condition $\frac{s(p)}{\Delta_i} \leq \min_{(i+1) \leq k \leq n} b_k$ must hold.

IV. SUBPATH BANDWIDTH MEASUREMENT USING CARTOUCHE PROBING

In this section we present our main probing structures, which enable us to achieve our stated goal of estimating the capacity bandwidth for an arbitrary path segment. In particular, given a path consisting of a sequence of links L_1, \dots, L_n with capacity bandwidths b_1, \dots, b_n , our goal is to estimate the capacity bandwidth of an arbitrary sequence of links along that path, i.e., estimate $\min_{i \leq k \leq j} b_k$, for arbitrary i and j such that $i \leq j \leq n$. We use the shorthand $b_{i,j}$ to denote the capacity bandwidth in the interval between links i and j inclusive.

We proceed by first demonstrating how to estimate the capacity bandwidth over a *prefix* of a path and over a *suffix* of a path. Techniques for handling these two easier cases, which are often useful in their own right, will provide insight as to how to approach the general problem.

A. Estimating Bandwidth Over a Prefix of the Path

We begin by estimating the capacity bandwidth along a path prefix, i.e., inferring $b_{1,j}$. Since the packet-pair technique described in Section III provides an estimate for $b_{1,n}$, it follows that if $b_{1,j} \leq b_{j+1,n}$, then $b_{1,j} = b_{1,n}$, giving us a solution. But when $b_{1,j} > b_{j+1,n}$, the packet-pair technique will end up estimating $b_{j+1,n}$. The underlying reason for this is that packet-pair techniques rely on the preservation of packet interarrival times induced at the bottleneck. So while the packet-pair property gives an interarrival time Δ_j at L_j of $\Delta_j = s(p)/(b_{1,j})$, the interarrival time at L_n is $\Delta_n = s(p)/(b_{1,n})$. This suggests a potential solution, namely preserving Δ_j unaltered to the end-host so that the end-host may infer $b_{1,j}$. Indeed, Lemma 4 gives us the condition we must satisfy to ensure such preservation. To do so, we need to generalize the packet-pair construction (spelled out in Lemma 1) to yield an interarrival time that is large enough to satisfy the constraints set by Lemma 4.

Lemma 5: Consider a path of n physical links L_1, L_2, \dots, L_n with capacity bandwidths b_1, b_2, \dots, b_n respectively. If a probe of the form $[\{p\}^{(r+1)}]$ is injected at L_1 and destined towards L_n then the interarrival time (Δ_j) between the first and the last probe packets at the end of every physical link L_j , for $1 \leq j \leq n$, is $\frac{r \cdot s(p)}{\min_{1 \leq k \leq j} b_k}$.

Based on the above lemma, one can generalize the packet-pair technique by using a probe structure consisting of a sequence of packets of the same size, whereby all packets except the first and the last are dropped at the end of L_j . By including enough packets in this sequence, the interarrival time between the first and last packets Δ_j at the end of L_j can be made large enough to be preserved as these two packets traverse links L_{j+1}, \dots, L_n . Indeed, Lemma 5 shows that if we use $r+1$ packets, then Δ_j would be $(r \cdot s(p))/b_{1,j}$. To satisfy the packet interarrival preservation condition, it turns out that we need the condition $r \geq b_{1,j}/b_{j+1,n}$ to be satisfied. That is, we would need as many

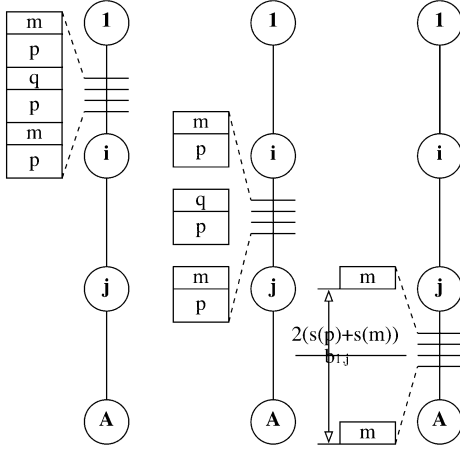


Fig. 2. Illustration of cartouche probing: A cartouche of size $r = 2$ is used to measure $b_{1,j}$. Here, all packets are dropped at link L_j except for the marker packets, and the spacing between the markers is preserved until they reach their target A .

probe packets as the ratio between $b_{1,j}$ and $b_{j+1,n}$, which makes this approach impractical. For example, if the ratio is 100 then a sequence of 100 probes need to be used, which is quite network intrusive. A better approach to preserve the interarrival times of probe packets at an internal link L_j as these packets traverse subsequent links $L_{j+1} \dots L_n$ is to use small packets as “markers” that delimit measurement boundaries. As we elaborate next, the use of small marker packets help preserve the spacing between packets while lowering the heavy requirement of using a sequence of many probe packets.

Definition 2: A cartouche $[pm\{pq\}^{r-1}pm]$ over the set of links $L_1, \dots, L_j, \dots, L_n$ is a sequence of $r + 1$ heterogenous packet-pairs in which $s(p) \geq s(m) = s(q)$, $D(p) = D(q) = L_j$, and $D(m) = L_n$. We refer to the first packet (p) in each pair as the *magnifier* packet, the second packet (m or q) in each pair as the *marker* packet, and r as the *cartouche size*. With the exception of the first and last marker packets m , all packets of the cartouche are targeted to L_j , which is called the *egress link* of the cartouche. L_n , the destination of the first and last marker packets is called the *target* of the cartouche.

Lemma 6: Let L be a sequence of n physical links L_1, L_2, \dots, L_n with capacity bandwidths b_1, b_2, \dots, b_n respectively. Given a cartouche of the form $[pm\{pq\}^{r-1}pm]$ over L with L_j as its egress link, let t_f and t_ℓ be the time that the final byte of the first and last marker packets are received at link L_j , respectively, then $t_\ell - t_f = \frac{r(s(p)+s(m))}{\min_{1 \leq k \leq j} b_k}$.

Lemma 6 provides the most important property of cartouche probing. It defines the interarrival time for the first and last marker packets over every physical link up to the cartouche egress link. Fig. 2 shows the composition and progression of a cartouche of size $r = 2$ injected at link L_1 towards a target end-host $L_n = A$ with link L_j as its egress link.

Corollary 1: Let L be a sequence of n physical links L_1, L_2, \dots, L_n with capacity bandwidths b_1, b_2, \dots, b_n , respectively. Given a cartouche of the form $[pm\{pq\}^{r-1}pm]$ over L with L_j as its egress link, let Δ_j be the interarrival time between the m markers at the end of L_j , then Δ_j will be preserved over L_{j+1}, \dots, L_n if and only if $\frac{b_{1,j}}{b_{j+1,n}} \leq \frac{r(s(p)+s(m))}{s(m)}$.

Corollary 1 follows directly from Lemma 6 and Lemma 4 to derive a sufficient condition for the preservation of markers interarrival times upon exit from the cartouche egress link L_j and throughout the sequence L_{j+1}, \dots, L_n . Note that with $s(p) = 1500$ bytes and $s(m) = 40$ bytes, preservation holds even when $(b_{1,j}/b_{j+1,n}) \leq 38.5r$; that is the interarrival time between the first and last marker packets holds even when $b_{j+1,n}$ is approximately $40r$ times smaller than $b_{1,j}$, where r is the cartouche size. It should be noted that $b_{1,j}$ and $b_{j+1,n}$ values are not known in advance and thus a reasonably large value of r may be needed. In Section V we show that *histograms* leading to bandwidth estimates over a path prefix exhibit consistent patterns as r increases, which helps identify appropriate r values.

Lemma 6 and Corollary 1 are all that are needed to provide a solution to the problem of inferring the capacity bandwidth of a path prefix. Specifically, this is done by: (1) sizing a cartouche to satisfy the conditions of Corollary 1, (2) setting the cartouche egress link to be L_j , (3) injecting the cartouche packets back-to-back at link L_1 , and (4) using the interarrival time of the first and last marker packets at link L_n as an estimate of their interarrival time at link L_j and using the relationship given in Lemma 6 to estimate $b_{1,j}$.

B. Estimating Bandwidth Over a Path Suffix

We now turn our attention to the complementary problem of estimating the capacity bandwidth of a path suffix—namely, $b_{i,n}$ for an arbitrary i such that $1 < i \leq n$. For the case $b_{1,i-1} \geq b_{1,n}$ the task is trivial since $b_{i,n}$ would be equal to $b_{1,n}$, which can be inferred using the packet-pair technique. In the rest of this section, we concentrate on the case when $b_{1,i-1} < b_{1,n}$.

A first-thought approach to obtaining $b_{i,n}$ is to use pacing (as spelled out in Lemma 3) to ensure that cartouche packets queue up (and hence are injected back-to-back) at link L_i , the first link on the suffix of interest. Effectively, this would enable us to “remotely release” a cartouche at link L_i . In [16], we have investigated the effectiveness of such an approach. Our findings show that, due to MTU limitations, the effectiveness of pacing is reduced for large payloads, especially when b_i is much larger than $b_{1,i-1}$.

A better approach to estimating $b_{i,n}$ is to attempt to identify the bottleneck link over the subpath of interest $L_i, \dots, L_{i+1}, \dots, L_n$ and estimate that link’s bandwidth. We do so using *cartouche trains*.

Definition 3: A *cartouche train* over a set of links $L_1, \dots, L_i, \dots, L_j, \dots, L_n$ is a probe consisting of a sequence of $l = j - i + 1$ possibly overlapping cartouches of size r each, whose egress links are L_i, L_{i+1}, \dots, L_j , respectively. Link L_i is called the *initial egress link* of the cartouche train and link L_{i+l-1} is called the *final egress link* of the cartouche train. The number l of possibly overlapping cartouches in a cartouche train is called the *length* of the cartouche train.

A cartouche train is completely defined by its length l , by the size r of its constituent cartouches, by its initial (or final) egress link L_i (or L_j), and by its target L_n . For instance a cartouche train of length 2 and of size 3, whose final egress link is L_5 and whose target is L_8 is given by $[p_3mp_4q_4p_4q_4p_4mp_5q_5p_5q_5p_5m]$, where $D(p_w) = D(q_w) = L_w$, $w = 3, 4, 5$, and $D(m) = L_8$.

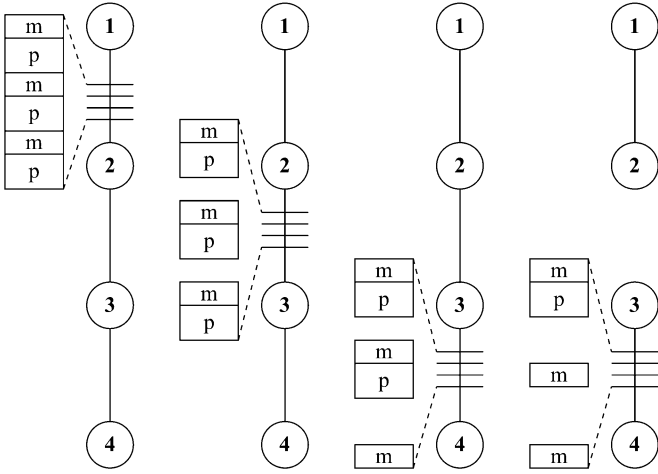


Fig. 3. Illustration of a cartouche train $r = 1$ and $l = 2$ used to estimate $b_{4,5}$.

Consider a cartouche train of the form $[p_{i-1}mp_i mp_{i+1}m \dots p_n m]$, whose target as well as final egress link is L_n . This cartouche train consists of $(n - i + 1)$ overlapping cartouches (each of the form $[p_{w-1}mp_w m]$). Clearly, $D(m) = L_n$ and $D(p_w) = L_w$, which means that all marker packets are targeted to L_n whereas magnifier packets are targeted to successive links starting at L_{i-1} .

Fig. 3 shows the composition and progression of such a cartouche train of length $l = 2$ transmitted from L_1 and targeted to L_4 with $L_3(L_4)$ as its initial (final) egress link. Note that the structure of the cartouche train over the subpath before L_3 (i.e., over L_1 , and L_2) resembles that of a cartouche of size $r = 2$. This means that the interarrival time between any pair of successive marker packets just before the initial egress link L_3 is $(s(p) + s(m))/b_{1,2}$. Also, due to the way the magnifier packets exit the subpath L_3L_4 on successive links, each of these interarrival times may be updated at only one specific link. For instance, the interarrival time at L_2 between marker packets 1 and 2 can only be altered at L_3 .¹ Similarly, the interarrival time between marker packets 2 and 3 can only be altered at L_4 . This key property is formulated in the following lemma, which quantifies the interarrival time (Δ_n^k) between the two marker packets immediately preceding and immediately following a magnifier packet egressing at link L_k .

Lemma 7: Let L be a sequence of n physical links $L_1, \dots, L_i, \dots, L_n$ with capacity bandwidths $b_1, \dots, b_i, \dots, b_n$ respectively, such that $b_{1,i-1} \leq b_{i,n}$. Let $[p_{i-1}m\{p_w m : n \geq w \geq i\}]$ be a cartouche train of size $r = 1$ and length $l = n - i + 1$ over L with L_i and L_n as the initial and final egress links, respectively. If $\frac{s(p)}{s(m)} < \frac{b_w}{b_{w-1}}$, the interarrival time between the two marker packets immediately preceding and immediately following the magnifier packet egressing at L_w is given by $\Delta_n^w = \frac{s(p)+s(m)}{b_{1,i-1}}$, otherwise $\frac{s(p)+s(m)}{b_{1,i-1}} + \frac{s(p)}{b_w} - \frac{s(m)}{b_{1,i-1}} \leq \Delta_n^w \leq \frac{s(p)+s(m)}{b_{1,i-1}} + \frac{s(p)}{b_w} - \frac{s(m)}{b_{w-1}}$. Based on Lemma 7, if $\Delta_n^w \neq \frac{s(p)+s(m)}{b_{1,i-1}}$ then Δ_n^w can fall at any point in the

¹Notice that the only possible alteration is that the interarrival time would grow. This happens if packets p and m satisfy the tailgating property over L_3 ; i.e., if $\frac{s(p)}{s(m)} > \frac{b_3}{b_{1,2}}$.

range $[\frac{s(p)+s(m)}{b_{1,i-1}} + \frac{s(p)}{b_w} - \frac{s(m)}{b_{1,i-1}}, \frac{s(p)+s(m)}{b_{1,i-1}} + \frac{s(p)}{b_w} - \frac{s(m)}{b_{w-1}}]$. This range is of size $\frac{s(m)}{b_{1,i-1}} - \frac{s(m)}{b_{w-1}}$, which is equivalent to the difference between the transmission time of a marker packet at a link with capacity bandwidth $b_{1,i-1}$ and its transmission time at a link with capacity bandwidth of b_{w-1} . The range is thus very small in size and as a result if $\Delta_n^w \neq \frac{s(p)+s(m)}{b_{1,i-1}}$ then it can be approximated as $\Delta_n^w \approx \frac{s(p)+s(m)}{b_{1,i-1}} + \frac{s(p)}{b_w} - \frac{s(m)}{b_{1,i-1}}$. That is $\Delta_n^w \approx \frac{s(p)}{b_{1,i-1}} + \frac{s(p)}{b_w}$, and the error in this approximation is at most $\frac{s(m)}{b_{1,i-1}} - \frac{s(m)}{b_{w-1}}$.

Corollary 2: Let L be a sequence of n physical links $L_1, \dots, L_i, \dots, L_n$ with capacity bandwidths $b_1, \dots, b_i, \dots, b_n$ respectively, such that $b_{1,i-1} \leq b_{i,n}$. Let $[p_{i-1}m\{p_w m : n \geq w \geq i\}]$ be a cartouche train of size $r = 1$ and length $l = n - i + 1$ over L with L_i and L_n as the initial and final egress links, respectively. If at least one link of the subpath L_i, \dots, L_n makes the tailgating property satisfied then $\Delta_n^{k^*} = \max_{i \leq k \leq n} (\Delta_n^k)$ would indicate that L_{k^*} is the subpath bottleneck link and $b_{i,n} \equiv b_{k^*} \approx \frac{s(p)}{(\Delta_n^{k^*} - \frac{s(p)}{b_{1,i-1}})}$.

Corollary 2 spells out how cartouche trains of size $r = 1$ and of length $l = n - i + 1$ could be used to estimate $b_{i,n}$. Note that a necessary condition for the cartouche train construction to work is that $b_{1,i-1}$ be less than $b_{i,n}$.²

Notice that if all Δ_n^k are equal to $(s(p_w) + s(m))/b_{1,i-1}$ then this signifies that links L_i, \dots, L_n are fast enough that none of the marker packets is queued behind its magnifier packet (thereby altering Δ). In this case $b_{i,n}$ cannot be pinpointed and we would then have to rely on an incremental hop-by-hop technique such as pathchar to estimate $b_{i,n}$.

In our presentation so far, we have assumed that cartouche trains are used to estimate the capacity bandwidth over a set of contiguous links (or hops). In general, cartouche trains could be used over a non-contiguous sequence of links as detailed in [16].

C. Estimating Bandwidth Over an Arbitrary Subpath

We are now ready to tackle our main goal of estimating $b_{i,j}$ for arbitrary i, j satisfying $1 \leq i < j \leq n$.

First, we observe that using cartouche probing we can measure $b_{1,i-1}$ and $b_{1,j}$. If $b_{1,i-1} > b_{1,j}$ then $b_{i,j} = b_{1,j}$. Otherwise, we need to find out a way to measure $b_{i,j}$. We do so using cartouche trains.

Consider a cartouche train of length $l = j - i + 1$ targeted at L_n with $L_i (L_j)$ as the initial (final) egress link. Clearly, the interarrival times Δ_j^k $i \leq k \leq j$ between the marker packets at L_j can be used to estimate $b_{i,j}$. Thus, the problem of measuring $b_{i,j}$ reduces to figuring out a way of preserving the spacing³ Δ_j^k between the marker packets of the cartouche train as they travel through links $L_{j+1} \dots L_n$. This can be readily achieved using the results of Lemma 4 which sets the conditions for the preservation of spacing over a subpath.

According to Lemma 4, in order for the marker interarrival times (Δ_j^k $i \leq k \leq j$) at link L_j to be preserved, the condition $(s(q)/\min \Delta_j^k) \leq b_{j+1,n}$ must be satisfied. Using $s(p) = 1500$

²If it is not, then $b_{i,n} = b_{1,n}$ which can be estimated using packet-pair techniques.

³In fact, we only need to ensure that $\Delta_j^{k^*} = \max_{i \leq k \leq j} \Delta_j^k$ is delivered intact to the endpoint.

bytes and $s(m) = 40$ bytes, the Δ_j^k spacing is preserved if $(b_{1,i-1}/b_{j+1,n}) \leq 38.5$. Notice that this bound is similar to the one we obtained for cartouches of size $r = 1$. In order to preserve Δ_j^k over L_{j+1}, \dots, L_n even if $(b_{1,i-1}/b_{j+1,n}) > 38.5$ we need to magnify Δ_j^k (as we did for cartouche probing in Section IV.A) using cartouche trains of size $r > 1$. The following Lemma spells this out.

Lemma 8: Let L be a sequence of n physical links $L_1, \dots, L_i, \dots, L_j, \dots, L_n$ with capacity bandwidths $b_1, \dots, b_i, \dots, b_j, \dots, b_n$ respectively. Let $b_{1,i-1} < b_{i,j}$. Given a cartouche train of length $l = j - i + 1$, size r , and with L_i as its initial egress link and L_j as its final egress link, if $\frac{s(p_w)}{s(m)} < \frac{b_k}{b_{k-1}}$ then $\Delta_j^k = \frac{r(s(p_w)+s(m))}{b_{1,i-1}}$, otherwise $\Delta_j^k \approx \frac{r(s(p_w)+s(m))}{b_{1,i-1}} + \frac{s(p_w)-s(m)}{b_k}$, where $i \leq k \leq j$ and $i - 1 \leq w \leq j$.

Using a maximum packet size (MTU) of 1500 and a minimum packet size of 40, a cartouche train of size r can infer $b_{i,j}$ correctly if $(b_{1,i-1}/b_{j+1,n}) \leq 38.5r$.

D. Summary of Measurement Procedure

We conclude this section with a summary of our procedure for measuring the capacity bandwidth $b_{i,j}$.

Step 1: Using a packet-pair technique, we measure $b_{1,n}$. This will enable us to appropriately size the cartouches used in later steps (using the results of Corollary 1).

Step 2: Using appropriately-sized cartouches, we measure $b_{1,i-1}$ and $b_{1,j}$ using the relationship established in Lemma 6. If $b_{1,i-1} \geq b_{1,j}$ then $b_{i,j} = b_{1,j}$ and we are done, otherwise we proceed to Step 3.

Step 3: Using an appropriately-sized cartouche train of length $l = j - i + 1$, with initial egress link L_i . If estimation of $b_{i,j}$ is possible using Corollary 2 then we are done, otherwise we conclude that our tool cannot accurately measure $b_{i,j}$.

We will argue in our experimental sections that practical situations in which our tool will fail to meet the conditions of Corollary 2 are rare. Alternative measurement techniques such as `pchar`, which have much higher network resource requirements, may be able to provide an estimate for $b_{i,j}$ in such cases by estimating the capacity bandwidth of each individual link and then taking the minimum.

E. Shared Capacity Bandwidth

We now extend our probing technique to enable the inference of the capacity bandwidth along the sequence of links shared by flows emanating from the same server and destined to two different clients as depicted in Fig. 4. Extending the technique to deal with shared links between more than two clients is straightforward and is detailed in [16]. As is clear from the topology in the figure accurate estimation of the capacity bandwidth b_S over the shared links L_S , is tantamount to computing the capacity bandwidth over a path prefix. However, we typically will not have *a priori* knowledge of the length of the shared prefix, nor the IP address of the branching point. One option we have is to use *traceroute* [21] on the path from S to A to determine this missing information, but this method is error-prone and inelegant. A more effective approach is to use a cartouche probe, but instead of using *hop-limited* probe packets as in the previous

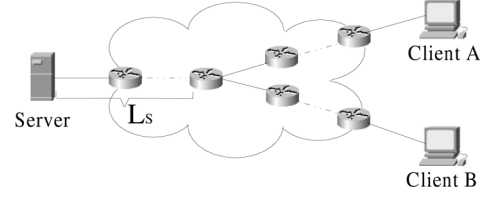


Fig. 4. Topology between a server and two clients.

sections, we use *multi-destination* probes instead. For example, consider a cartouche sent to A . By replacing hop-limited packets by packets destined for B within the cartouche, this method coerces these packets to leave the cartouche precisely when we wish, i.e., after traversing the last physical link in L_S .

As a detailed illustration, we send from source S a $[pm\{pq\}^{r-1}pm]$ cartouche probe sequence where $D(m) = A$ and $D(p) = D(q) = B$. The two markers m destined to A travel together with the other probe packets destined to B only over the links in L_S , which is exactly the desired outcome. Client A then measures the interarrival time Δ between marker packets m and computes $(r(s(p) + s(m)))/\Delta$ to estimate b_S .

V. IMPACT OF CROSS TRAFFIC

In Section IV, we presented an analysis of our end-to-end capacity bandwidth estimation procedures. As stated in Section III, the analysis assumes an environment free from cross-traffic, and it is under this idealistic assumption that we prove the various properties of cartouche probing. Clearly, in any practical setting, cross-traffic cannot be ignored. In this section, we present results from simulations intended to characterize the impact of cross traffic on cartouche probing. Our goal in this section is to identify *traffic conditions* under which cartouche probing is and is not effective. We also find that cross-traffic is not our only worry; we demonstrate scenarios in which structural characteristics of the *network path* itself impact our results. However, we find that cartouche probing is highly resilient to both the impact of cross-traffic (much more so than packet-pair and tailgating techniques) and structural issues along the network path. This is further reinforced in Section VI and in Section VII, in which results from controlled laboratory experiments and Internet validation experiments are presented.

Prelude: Recall that cartouche probing relies on the preservation of spacing between marker packets to estimate the capacity bandwidth of a path segment at endpoints. In general, cross traffic may impact marker spacing in two possible ways: it may cause *marker compression*, i.e., inter-packet spacing between a pair of markers is reduced in transit, or *marker expansion*, i.e., inter-packet spacing between a pair of markers is increased in transit. Both compression and expansion can result from the arrival of cross-traffic at a link [10]. For example, a burst arriving before the first marker causes the first packet to queue, and results in compression; a burst arriving *between* the markers can cause the second marker to be delayed, resulting in expansion.

Marker compression is also possible even in the absence of cross traffic. Recall our constructions in Section IV.A. There, we preserved spacing between the two markers used to measure

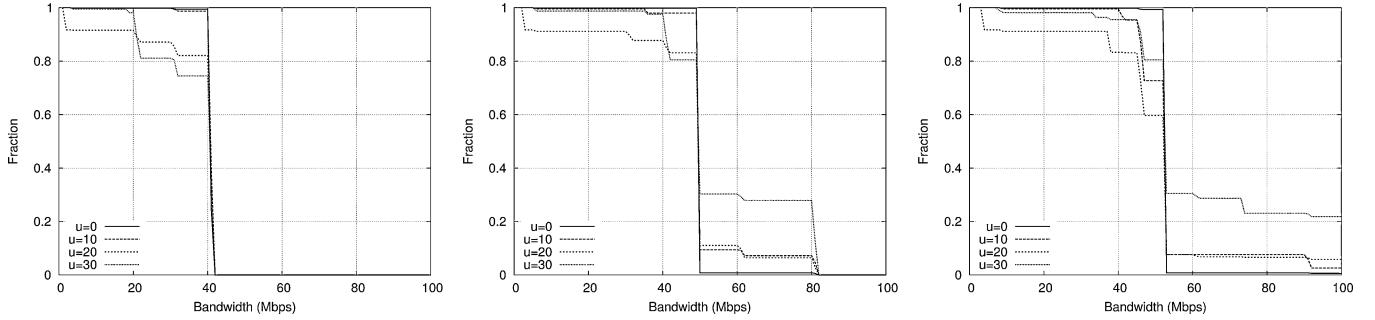


Fig. 5. CDF distribution of estimated $b_{1,10}$ values after using a sequence of 1000 $[pm\{pq\}^{r-1}pm]$ cartouche probes of dimensionality $r = 1$ (left), $r = 2$ (middle), and $r = 3$ (right), for different link utilization, u . Actual $b_{1,10} = 50$ Mbps and $b_{11,20} = 1$ Mbps.

the capacity bandwidth $b_{i,j}$ as the markers travel over subsequent links $L_{j+1} \dots L_n$. But if $b_{j+1,n}$ is small enough to violate the condition stated in Corollary 1, interpacket spacing is not preserved. To avoid such compression, the size r of the cartouches employed must be increased so as to satisfy the conditions of Corollary 1. In effect, Corollary 1 sets a lower bound on r , which must be satisfied for our probing constructions to work (as spelled out in the procedure in Section IV.D), and is due entirely to the static properties of the path.

To reduce the effects of cross traffic, a capacity bandwidth measurement experiment must be conducted repeatedly and estimates that may have been affected by marker compression or expansion must be identified and excluded using heuristics [10]. All of our methods require the end-host A conducting the experiment to compile a histogram of the frequency of each estimate it obtains (or a Cumulative Distribution Function CDF). One simple heuristic is to pick the bin with the largest frequency, i.e., the mode (the largest *dip* in the CDF curve). But with a more refined understanding of how marker compression or marker expansion affects our capacity bandwidth estimation in specific experiments, we develop *better alternative heuristics* to simply picking the mode. In all our experiments, we use a fixed bin width of 1 Mbps for the histograms.

From equations in Section IV, one can see that marker compression results in *overestimation* of capacity bandwidth, whereas marker expansion results in *underestimation* of capacity bandwidth. Moreover, as we will subsequently demonstrate, marker compression due to cross traffic is more prevalent in experiments involving path prefixes, whereas marker expansion is more prevalent in experiments involving path suffixes and targeted path segments. This suggests that picking the *mode* of a histogram in prefix experiments and picking the *last mode* of a histogram for suffix/subpath experiments are better heuristics to use for filtering the effects of cross traffic.

Experimental Setup: We used the Network Simulator (ns) [34] to simulate a path L connecting two hosts A and B . L consists of 20 physical links L_1, L_2, \dots, L_{20} . Link bandwidth values b_1, b_2, \dots, b_{20} were hand-picked to illustrate various scenarios but the default link bandwidth value is set to 100 Mbps. Link latencies d_1, d_2, \dots, d_{20} are all set to 10 msec since they have no impact on the results. Cross-traffic is modeled using a combination of TCP and UDP flows generating equal bit rates (using 32 Kb/sec as the mean flow rate). Cross-traffic flows are *hop-persistent*, that is any flow traverses only one link. By varying the

number of cross-traffic flows over each link we control the utilization of each of the links, u . Packet sizes of cross-traffic flows are equally distributed between 40, 576 or 1500 bytes as suggested in measurement studies on network traffic traces [13]. Probe transmission, time measurements, logging and estimation functions were all performed at host A .

In our experiments, we vary a number of parameters to study the effects of cross traffic. These include: link utilization u , cartouche size r , length l of the subpath (whether a prefix, suffix, or arbitrary segment) under consideration, and the ratio of the actual bandwidth of the segment under consideration to that of the entire path. Any one of these parameters may alter the spacing of marker packets and thus the accuracy of our bandwidth estimates.

In all experiments presented in this section, we use the following settings in the construction of cartouches: $s(p) = 1500$ bytes, $s(m) = s(q) = 40$ bytes.

Path Prefix Experiments: As described in Section IV.A, our technique for measuring the capacity bandwidth of path prefix relies on sending a sequence of $[pm\{pq\}^{r-1}pm]$ cartouches from source A , with L_i as the egress link. Host A monitors the interarrival time Δ of the responses to the marker packets m , and uses the formula $(r(s(p) + s(m)))/\Delta$ to estimate $b_{1,i}$.

Fig. 5 shows the CDF curves of the bandwidth estimates that we obtain (at host A) trying to infer $b_{1,10}$ using a sequence of 1000 $[pm\{pq\}^{r-1}pm]$ cartouches of sizes $r = 1$ (left) $r = 2$, (middle), and $r = 3$ (right), with an actual $b_{1,10} = 50$ Mbps and $b_{11,20} = 1$ Mbps.

Examining the results in Fig. 5 we observe that the $r = 2$ and $r = 3$ cases lead to a correct $b_{1,10} = 50$ estimate while the $r = 1$ case does not. Lemma 6 explains why this happens. Using $s(p) = 1500$ bytes and $s(m) = 40$ bytes the condition $b_{1,i}/b_{i+1,n} \leq 38.5r$ must hold to deliver unperturbed marker interarrival times to the endhost. Since $b_{1,10}/b_{11,20} = 50$, the condition holds only for $r = 2$ and $r = 3$, but not for $r = 1$. Thus, when $r = 1$, our cartouches were undersized, resulting in marker expansion and an underestimate of the value of $b_{1,i}$.

The histograms corresponding to $r = 1, 2$ and 3 show few instances of underestimation of $b_{1,10}$ especially as u grows (indicated by short *dips* in the CDF curves). These are examples of marker expansion due to bursty cross traffic, leading us to underestimate the value of $b_{1,i}$. Notice that the incidence of underestimation is more pronounced for $r = 3$. This is due to the fact that larger cartouches imply longer marker interarrivals, which

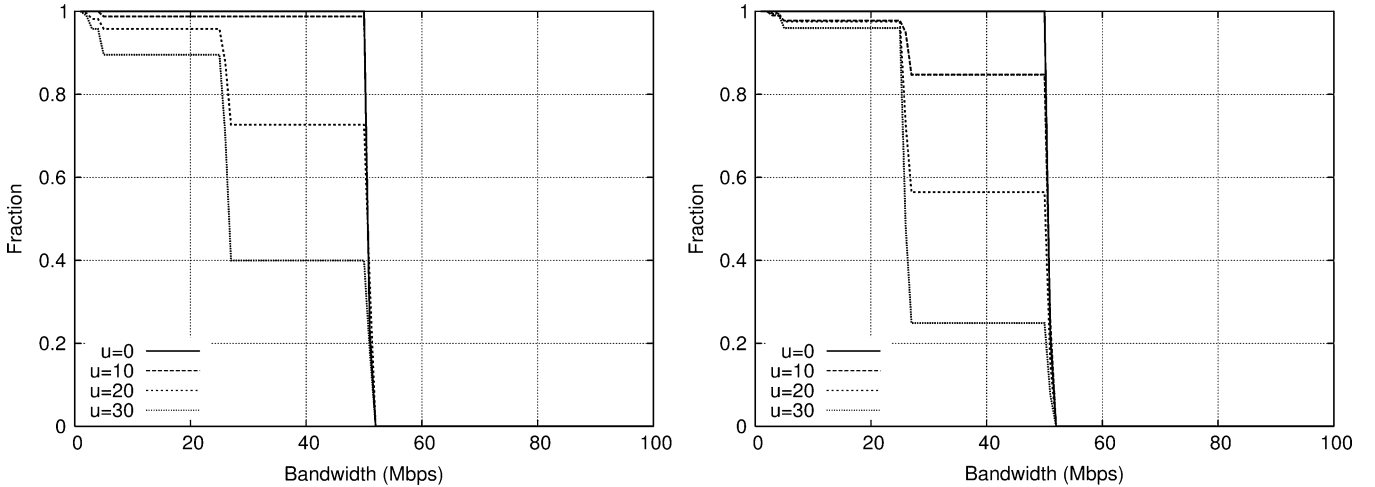


Fig. 6. CDF distribution of estimated $b_{i,20}$ values using $i = 15$ (left) and $i = 10$ (right) for different values of u , using a sequence of 1000 cartouche train probes. Actual $b_{i,20} = 50$ Mbps.

in turn leads to a higher probability of cross traffic bursts further separating the markers.

The histograms corresponding to $r = 2$ and $r = 3$ show instances of overestimation of $b_{1,10}$. These are examples of marker compression due to bursty cross traffic, leading us to overestimate the value of $b_{1,i}$. Our overestimates of $b_{1,10}$ are capped at 77 Mbps and 115.5 Mbps, for $r = 1$ and $r = 3$, respectively. Again, this is a direct consequence of the inequality in Lemma 6, which in effect specifies an upper bound on the maximum observable value for $b_{1,i}$ using cartouches of size r . This bound (confirmed in Fig. 5) is $r * b_{i+1,n} * 38.5$, which is 38.5 Mbps and 77 Mbps and 115.5 Mbps for $r = 1, 2$ and 3 respectively. Clearly, when $r = 2$ and $r = 3$, our filtering approach was successful in hiding both underestimates and overestimates of $b_{1,10}$ when the utilization is reasonably low (up-to $u = 0.2$ in this case), but filtering becomes more challenging as the utilization u increases. As u increases, incorrect estimates due to marker compression become much more significant than those resulting from marker expansion. Marker compression is either caused by a violation of the preservation of spacing lemma, and/or by cases in which the first marker is delayed more than the second marker in the network due to cross traffic. Note that the first marker is more likely to be delayed since a delay of the first marker is due to queued cross traffic at any router, while a delay in the second marker is only due to cross traffic induced during the gap between the markers. For larger utilization values filtering out the last *pronounced* mode, corresponding to marker *expansion* helps to locate the correct bandwidth value.

Path Suffix Experiments: As described in Section IV.B, our technique to measure path suffix capacity bandwidth relies on sending a cartouche train of size $r = 1$ and of length l equal to the suffix length. As before, host A monitors the interarrival time Δ of the responses to markers m , then uses the largest interarrival time between any 2 successive marker packets at the endhost, Δ_n^{k*} , and $b_{1,i-1}$ to estimate $b_{i,n}$.

Fig. 6 shows the CDF of the bandwidth estimates we obtain trying to infer $b_{15,20}$ (left) and $b_{10,20}$ (right) after using a sequence of 1000 cartouche trains of length $l = 5$ and $l = 10$, respectively for different link utilization values, u , with actual

$b_{15,20} = 50$ Mbps and $b_{10,20} = 50$ Mbps, with the overall path capacity $b_{1,20}$ being set to 1 Mbps. Fig. 6 shows that incorrect estimates due to *marker expansion* are non-negligible, whereas those due to *marker compression* are virtually non-existent. This is in sharp contrast to our results for prefix capacity bandwidth estimation, in which we have shown that incorrect estimates due to *marker compression* are more prevalent. In path suffix measurements, marker expansion exists if cross-traffic interferes between any pair of successive markers, making the separation larger than the separation induced by the bottleneck link. Marker compression is unlikely as it can only happen if cross traffic is injected before the first marker corresponding to the bottleneck, reducing the gap between the marker packets corresponding to the bottleneck link without increasing the gap between other marker packets. Picking the mode (largest CDF dip) is fine as long as u is reasonably small (upto $u = 0.2$ in our simulations). For larger utilization values filtering out the first pronounced mode, corresponding to marker *expansion* helps to locate the correct bandwidth value.

Fig. 6 also shows that incorrect estimates due to marker expansion are more prevalent in the histogram on the right (corresponding to the longer path suffix). This is explained by noting that the length l of the cartouche train grows in tandem with the length of the path suffix—the longer the cartouche train, the higher the probability of cross-traffic interference. Again, this is in sharp contrast to capacity bandwidth estimation for path prefix, in which the prefix length did not play a significant factor.

Arbitrary Path Segments Experiments: As described in Section IV.C, our technique to measure arbitrary subpath bandwidth relies on sending a sequence of appropriately-sized cartouche trains from source A . Host A monitors the interarrival time Δ of the responses to markers m , then uses the largest interarrival time between any 2 successive marker packets, Δ_n^{k*} , and $b_{1,i-1}$ to estimate $b_{i,j}$.

Fig. 7 shows the histograms we obtain trying to infer $b_{5,15}$ using 1000 generalized cartouche probes of dimension $r = 2$ (left) and $r = 3$ (right), for different link utilization values, u . The setup is such that $b_{5,15} = 50$ Mb/s, $b_{1,4} = 1$ Mb/s, $b_{16,20} = 5$ Mb/s. Fig. 7 shows $b_{5,15}$ is correctly estimated. It

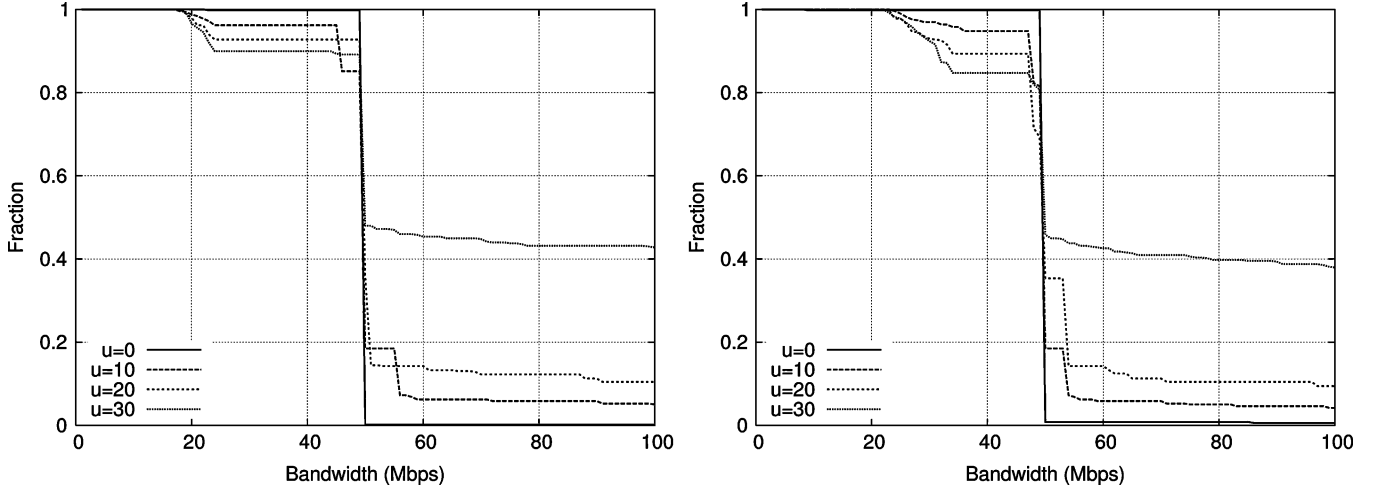


Fig. 7. Distribution of estimated $b_{5,15}$ values after using a sequence of 1000 generalized probes of dimension $r = 2$ (left) and $r = 3$ (right). Actual $b_{5,15} = 50$ Mbps.

also shows that underestimates (due to marker expansion) and overestimates (due to marker compression) exist. In fact, when $u = 0.3$ the majority of our trials resulted in underestimates and overestimates. When u is smaller then the highest mode (largest CDF dip) is obvious and corresponds to the correct bandwidth estimate. When u is larger, then filtering out the modes corresponding to marker compression and expansion is needed. The same conclusion holds when the targeted path is long as the gaps between probe packets are more prone to dispersion. These results suggest that for long subpaths, it is not advisable to simply use (correspondingly) long cartouche trains. A better divide-and-conquer alternative may be to partition a long sub-path into segments, to which shorter cartouche trains could be applied.

Postlude: We conclude this section with a summary of our findings regarding the susceptibility of our constructions to cross traffic. Specifically, we observe that, in highly congested setups: 1) Marker compression and hence overestimation of $b_{1,i}$ presents the most significant hurdle for capacity bandwidth estimation of path prefix using cartouches. 2) Marker expansion and hence underestimation of $b_{j,n}$ presents the most significant hurdle for capacity bandwidth estimation of path suffix. 3) Both marker compression and expansion are prevalent in arbitrary path segments bandwidth measurement. This difficulty can be alleviated through the use of the smallest cartouches that would satisfy the structural constraints imposed by Lemma 6, through appropriate filtering techniques, and through a divide-and-conquer to limit the size of the cartouche probes approach.

VI. CONTROLLED LABORATORY EXPERIMENTS

To evaluate our mechanisms, we incorporated our cartouche probing functionality into a measurement toolkit developed in our laboratory. This toolkit is partially implemented in user space and partially implemented in the kernel (Linux). We use BBSCOPE to refer to this embodiment of cartouche probing in our toolkit. The functionalities of BBSCOPE implemented in the kernel include orchestration of cartouche and cartouche

train transmissions. This includes checking whether packet reordering occurs on the path, ensuring the back-to-back transmission of the constituent packets of cartouche probes (and cartouche trains), and the collection of probing results. This is done by pairing up responses (ECHO REPLY) to markers and using timestamps to estimate marker interarrivals and hence bandwidth estimates. The functionalities of BBSCOPE implemented in user space include various user interfaces for the collection of experiment parameters and for communicating these parameters with the probing engine in the kernel. Also, it includes “filtering” processes (similar to those discussed in Section V) as well as other processes for calculating various statistics (e.g., confidence intervals). BBSCOPE requires *no* support from clients or intermediate routers beyond the ability to respond to ICMP ECHO REQUESTs.

In our lab, we installed a path L connecting two hosts, A and B , running RedHat Linux 2.2.14 on Pentium III processors. The hosts A and B are running BBSCOPE to validate the effectiveness of the cartouche probing constructions. The path L has three Ethernet links, L_1 , L_2 and L_3 , inter-connected through Cisco 7200 series routers. Link bandwidth values, b_1 , b_2 and b_3 , are 10 Mbps, 100 Mbps, and 10 Mbps, respectively. In all experiments, we use magnifier probe packets of size 1500 bytes and marker packets of size 60 bytes. Cartouche probes were sent once per second until 1000 bandwidth estimates are collected. As used in the simulations described in Section V, cross-traffic is *hop-persistent*, consists of a combination of TCP and UDP flows, and its packet sizes are distributed between 40, 576 and 1500 bytes. We vary the number of flows at each link to vary the utilization of the links. In Fig. 8 we plot the CDF of our estimates of b_1 , b_2 , and b_3 , for different utilization values. It is clear from the CDF curves, that the bandwidth estimates mostly reflect the accurate values.

VII. INTERNET MEASUREMENT EXPERIMENTS

In this section, we present results of Internet experiments we have conducted to validate our capacity bandwidth measure-

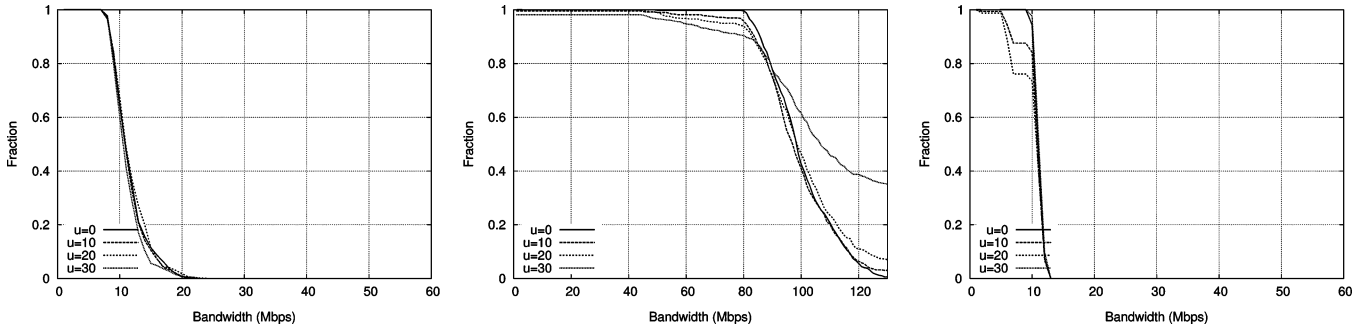


Fig. 8. CDF Distribution of estimated b_1 (left), b_2 (middle), and b_3 (right) values after using a sequence of 1000 cartouche probes of dimensionality $r = 3$ (left) 1000 cartouche trains of dimensionality $r = 3$ (middle) and 1000 cartouche trains of dimensionality $r = 1$ (right). Different curves correspond to different link utilization values, u . Actual $b_1 = 10$ Mbps, $b_2 = 100$ Mbps, and $b_3 = 10$ Mbps.

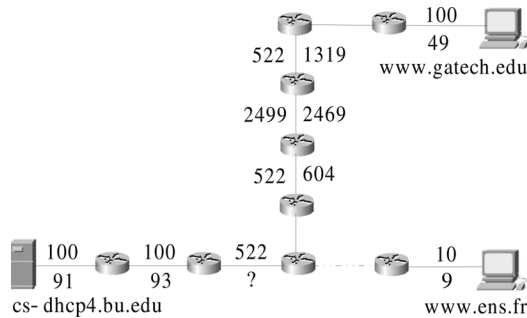


Fig. 9. The Internet paths used in our validation experiments. Labels above links are *a priori*-known link bandwidths. Labels below links are estimates obtained by `pchar`. All units are in Mbps.

ment techniques and compare its performance and efficiency an existing hop-by-hop techniques (namely `pchar`).⁴

Experimental Setup: Two Internet paths connecting our laboratory to two different universities have been handpicked to demonstrate the different scenarios that we discussed in Section V. These universities are Georgia Tech in the US and Ecole Normale Superieure in France. Fig. 9 shows that these two paths share the first three hops and then diverge. The figure also shows two sets of labels for the link bandwidth of a subset of the hops along these paths. The first set of labels (shown above a link) reflects the *a priori*-known capacity bandwidth of links in our own laboratory, links of Internet2 hops published by Abilene [32] on January 30, 2002 and links on the far end of the paths obtained through personal contacts. The second set of labels (shown below a link) reflects link bandwidths measured from our laboratory on June 30, 2002 using `pchar`. Notice that, in some cases, `pchar` often fails to obtain reliable link bandwidth estimates (hence the “?” label on some of the links in Fig. 9).

We installed BBSCOPE in our laboratory on a Pentium III processor running RedHat Linux 2.2.14) over a 100 Mbps LAN. In all experiments, we use magnifier probe packets of size 1500 bytes and marker packets of size 60 bytes. Experiments were conducted once per second until we obtain 100 valid results for a given path—recall, that packet reordering or packet losses invalidate an experiment.

Results: As indicated in Fig. 9, the link bandwidth of many of

the hops along the set of paths we considered are fairly well established, thus giving us a reliable reference against which to test the performance of BBSCOPE.

Fig. 10 shows the histograms we obtained when using BBSCOPE to estimate $b_1, b_{1,3}, b_3, b_5, b_6, b_{5,7}, b_9$ for the path to Georgia Tech and the bandwidth of the last link for the Ecole Normale Superieure path. Clearly, the estimated values are close to the *a priori*-known bandwidth values.

Comparison to `pchar` and `nettimer`: Both `pchar` [31] and `nettimer` [30] are hop-by-hop techniques which means that in order to estimate the capacity bandwidth along a path segment they need to run tests over every hop in the segment to estimate its bandwidth and provide the lowest estimate as the final result. On the other hand, Cartouche probing directly targets the capacity bandwidth of the segment.

In comparing against `pchar` and `nettimer`, we need to consider two measures: 1) *time efficiency*: which reflects the time it takes to get a reliable estimate, and 2) *byte efficiency*: which is a measure of the number of bytes injected into the network to get a reliable estimate. In terms of time efficiency, Cartouche probing is more efficient since it does not have to orchestrate a round of probing for every hop in a segment before returning the final estimate. Also, in terms of byte efficiency, cartouche probing is more efficient than `pchar` that uses linear regression in its statistical analysis for every hop which requires injecting the network with lots of extra traffic. In fact, `pchar` default settings, using a packet size increments of 32 bytes and 32 repetitions per hop, leads to more than 1 MB injected in the network per hop. Cartouche probing needs less than half this number to estimate to capacity bandwidth of a path segment. Cartouche probing is also more byte efficient than `nettimer` when estimating the capacity along a path prefix in case the path prefix length is larger than the cartouche size r and is almost as byte efficient as `nettimer` when estimating the capacity bandwidth along arbitrary segments. A cartouche of size r has as many bytes as $r + 1$ `nettimer` tailgated pairs and a cartouche train of length l and size r has as many bytes as $(r + 1)l - 1$ tailgated pairs.

VIII. CONCLUSION

We have described an end-to-end probing technique which is capable of inferring the capacity bandwidth along an arbitrary set of path segments in the network, or across the portion of a

⁴We were not able to find a working version of `nettimer`.

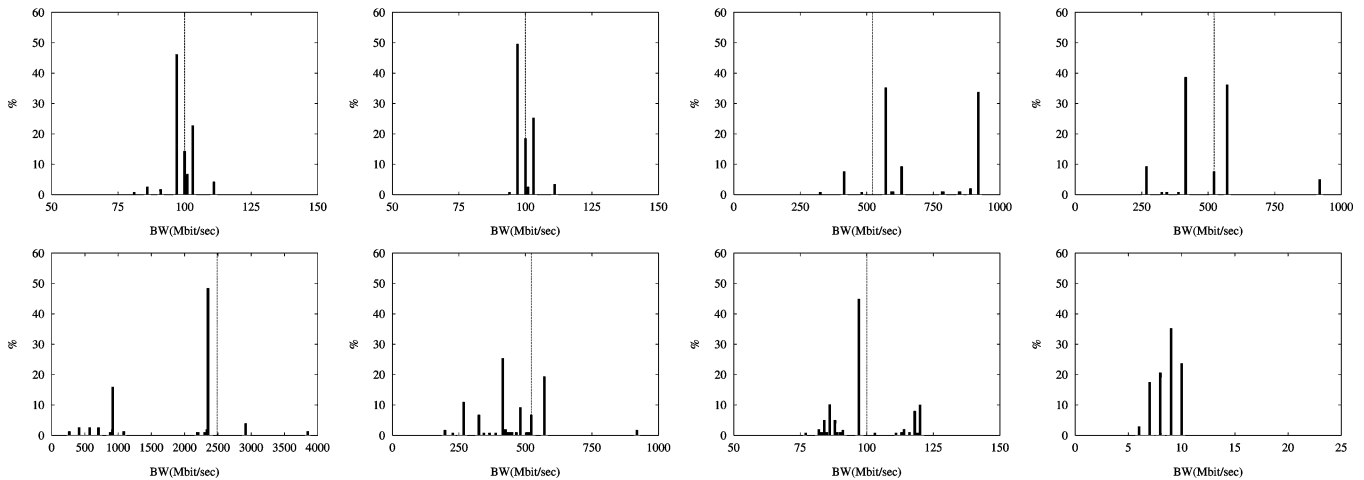


Fig. 10. Histograms of estimated capacity bandwidth along different segments of the paths to Georgia Tech and Ecole Normale Superieure. The histograms in the top row (from left to right) are for Georgia Tech b_1 , $b_{1,3}$, b_3 , b_5 and in the bottom row are for Georgia Tech b_6 , $b_{5,7}$, b_9 and for the last link of the path to Ecole Normale Superieure. Dotted lines represent a *priori*-known bandwidth values.

path shared by a set of connections, and have presented results of simulations and preliminary Internet measurements of our techniques. The constructions we advocate are built in part upon packet-pair techniques, and the inferences we draw are accurate under a variety of simulated network conditions and are robust to network effects such as the presence of bursty cross-traffic.

While the end-to-end probing constructions we proposed in this paper are geared towards a specific problem, we believe that there will be increasing interest in techniques which conduct remote probes of network-internal characteristics, including those across arbitrary subpaths or regions of the network. We anticipate that lightweight mechanisms to facilitate measurement of metrics of interest, such as capacity bandwidth, will see increasing use as emerging network-aware applications optimize their performance via intelligent utilization of network resources.

REFERENCES

- [1] B. Ahlgren, M. Bjorkman, and B. Melander, "Network probing using packet trains," Swedish Inst., Technical Report, Mar. 1999.
- [2] D. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay networks," in *SOSP 2001*, Banff, Canada, Oct. 2001.
- [3] S. Banerjee and A. Agrawala, "Estimating available capacity of a network connection," in *IEEE Int. Conf. Networks (ICON 01)*, Bangkok, Thailand, Oct. 2001.
- [4] J. C. Bolot, "End-to-end packet delay and loss behavior in the Internet," in *Proc. ACM SIGCOMM'93*, Sep. 1993, pp. 289–298.
- [5] J. Byers, J. Considine, M. Mitzenmacher, and S. Rost, "Informed content delivery across adaptive overlay networks," in *ACM SIGCOMM'02*, Pittsburgh, PA, Aug. 2002.
- [6] J. Byers, M. Luby, and M. Mitzenmacher, "Accessing multiple mirror sites in parallel: Using Tornado codes to speed up downloads," in *Proc. IEEE INFOCOM'99*, Mar. 1999, pp. 275–83.
- [7] R. L. Carter and M. E. Crovella, "Measuring bottleneck link speed in packet switched networks," *Performance Evaluation*, vol. 27&28, pp. 297–318, 1996.
- [8] Y.-H. Chu, S. Rao, and H. Zhang, "A case for end-system multicast," in *ACM SIGMETRICS'00*, Santa Clara, CA, Jun. 2000.
- [9] M. E. Crovella, R. Frangioso, and M. Harchol-Balter, "Connection scheduling in web servers," in *Proc. 1999 USENIX Symp. Internet Technologies and Systems (USITS'99)*, Oct. 1999.
- [10] C. Dovrolis, P. Ramanathan, and D. Moore, "What do packet dispersion techniques measure?," in *INFOCOM'01*, Anchorage, AK, Apr. 2001.
- [11] A. Downey, "Using pathchar to estimate internet link characteristics," in *ACM SIGCOMM'99*, Boston, MA, Aug. 1999.
- [12] N. Duffield, F. L. Presti, V. Paxson, and D. Towsley, "Inferring link loss using striped unicast probes," in *IEEE INFOCOM 2001*, Apr. 2001.
- [13] NLNLR Network Traffic Traces. NLNLR: National Lab. for Applied Network Research, 2003 [Online]. Available: <http://www.caida.org/publications/papers/2003/nlanr/nlanr-overview.pdf>
- [14] M. Goyal, R. Guerin, and R. Rajan, "Predicting TCP throughput from non-invasive network sampling," in *IEEE INFOCOM'02*, Jun. 2002.
- [15] K. Hanna, N. Natarajan, and B. Levine, "Evaluation of a novel two-step server selection metric," in *9th Int. Conf. Network Protocols (ICNP)*, Riverside, CA, Nov. 2001.
- [16] K. Harfoush, "A framework and toolkit for the effective measurement and representation of internet internal characteristics," Ph.D. dissertation, Boston Univ., Boston, MA, Jun. 2002.
- [17] K. Harfoush, A. Bestavros, and J. Byers, "Robust identification of shared losses using end-to-end unicast probes," in *8th Int. Conf. Network Protocols (ICNP)*, Nov. 2000.
- [18] N. Hu, L. Li, and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE J. Sel. Areas Commun.*, vol. 21, no. 8, pp. 879–974, Aug. 2004.
- [19] N. Hu, L. Li, and P. Steenkiste, "Locating internet bottlenecks: Algorithms, measurements, and implications," in *ACM SIGCOMM 2004*, Portland, OR, Sep. 2004.
- [20] V. Jacobson, *pathchar*: A tool to infer characteristics of Internet paths. [Online]. Available: <ftp://ftp.ee.lbl.gov/pathchar>
- [21] V. Jacobson, *Traceroute*. 1989 [Online]. Available: <ftp://ftp.ee.lbl.gov/traceroute.tar.Z>
- [22] M. Jain and C. Dovrolis, "End-to-end available bandwidth: Measurement methodology, dynamics, and relation with TCP throughput," in *ACM SIGCOMM'02*, Aug. 2002.
- [23] M. Jain and C. Dovrolis, "Pathload: A measurement tool for end-to-end available bandwidth," in *Passive and Active Measurement (PAM) Workshop 2002*, Fort Collins, CO, Mar. 2002.
- [24] J. Jannotti, D. Gifford, K. Johnson, M. F. Kaashoek, and J. O'Toole, Jr., "Overcast: Reliable multicasting with an overlay network," in *Proc. OSDI 2000*, San Diego, CA, Oct. 2000.
- [25] J. Kangasharju, J. Roberts, and K. W. Ross, "Object replication strategies in content distribution networks," in *Proc. WCW'01: Web Caching and Content Distribution Workshop*, Boston, MA, Jun. 2001.
- [26] R. Kapoor, L. Chen, L. Lao, M. Gerla, and M. Sanadidi, "CapProbe: A simple and accurate capacity estimation technique," in *ACM SIGCOMM 2004*, Portland, OR, Sep. 2004.
- [27] S. Keshav, "A control-theoretic approach to flow control," in *ACM SIGCOMM'91*, Sep. 1991.
- [28] S. Keshav, "Congestion control in computer networks," Ph.D. dissertation, University of California, Berkeley, CA, Sep. 1991.
- [29] K. Lai and M. Baker, "Measuring link bandwidths using a deterministic model of packet delay," in *ACM SIGCOMM'00*, Stockholm, Aug. 2000.
- [30] K. Lai and M. Baker, "Nettimer: A tool for measuring bottleneck link bandwidth," in *Proc. USITS'01*, Mar. 2001.

- [31] B. Mah, pchar. 2000 [Online]. Available: <http://www.ca.sandia.gov/bmah/Software/pchar>
- [32] The Abilene Network Logical Map. Jan. 30, 2002 [Online]. Available: <http://www.abilene.iu.edu/images/logical.pdf>
- [33] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *IEEE GLOBECOM 2000*, San Francisco, CA, Nov. 2000.
- [34] ns: Network Simulator [Online]. Available: <http://www-mash.cs.berkeley.edu/ns/ns.html>
- [35] A. Pasztor and D. Veitch, "Attila pasztor and darryl veitch, active probing using packet quartets," in *ACM SIGCOMM Internet Measurement Workshop 2002*.
- [36] A. Pasztor and D. Veitch, "The packet size dependence of packet pair like methods," in *Tenth International Workshop on Quality of Service (IWQoS)*, 2002.
- [37] V. Paxson, "End-to-end routing behavior in the Internet," in *ACM SIGCOMM'96*, Stanford, CA, Aug. 1996.
- [38] V. Paxson, "End-to-end Internet packet dynamics," in *ACM SIGCOMM'97*.
- [39] V. Paxson, "Measurements and analysis of end-to-end internet dynamics," Ph.D. thesis, Comput. Sci. Dept., University of California, Berkeley, CA, 1997.
- [40] P. Radoslavov, R. Govindan, and D. Estrin, "Topology-informed internet replica placement," in *WCW'01: Web Caching and Content Distribution Workshop*, Boston, MA, Jun. 2001.
- [41] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell, "PathChirp: Efficient available bandwidth estimation for network paths," in *Passive and Active Measurement (PAM) Workshop 2003*, La Jolla, CA, Apr. 2003.
- [42] P. Rodriguez, A. Kirpal, and E. Biersack, "Parallel-access for mirror sites in the internet," in *IEEE INFOCOM'00*, Mar. 2000.
- [43] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *ACM SIGCOMM'01*, San Diego, CA, Aug. 2001.
- [44] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," in *ACM Internet Measurement Conf. (IMC) Tools 2003*, Miami, FL, Oct. 2003.
- [45] Z. ZiXuan, B. Lee, C. Fu, and J. Song, "Packet triplet: A novel approach to estimate path capacity," *IEEE Commun. Lett.*, vol. 9, no. 12, 2005.



Khaled A. Harfoush (M'02) received the Ph.D. degree in computer science from Boston University, Boston, MA, in 2002.

He is currently an Associate Professor in the Department of Computer Science at North Carolina State University, Raleigh, which he joined in 2002. His research interests are in the general areas of network modeling, Internet measurement, peer-to-peer systems and network security.

Prof. Harfoush is a recipient of the prestigious NSF CAREER award. He serves on the program committees for numerous conferences including IEEE INFOCOM and IEEE ICNP. He has been a member of the ACM and IEEE since 2002.



Azer Bestavros (SM'08) received the Ph.D. degree in computer science from Harvard University, Cambridge, MA, in 1992.

He is a Professor in and former Chairman of the Computer Science Department at Boston University, Boston, MA. His research interests are in networking and in real-time systems. His research contributions include his pioneering of the push content distribution model adopted years later by CDNs, his seminal work on traffic characterization and reference locality modeling, his work on various network transport, caching, and streaming media delivery protocols, his work on e2e inference of network caricatures, his work on identifying and countering adversarial exploits of system dynamics, his work on game-theoretic approaches to overlay and P2P networking applications, his generalization of classical rate-monotonic analysis to accommodate uncertainties in resource availability/usage, his use of redundancy-injecting codes for timely access to periodic broadcasts, his work on verification of network protocol compositions, including the identification of deadlock-prone arrangements of HTTP agents, and his work on virtualization services and programming environments for embedded sensor networks. His work has culminated so far in 10 Ph.D. theses, more than 80 masters and undergraduate student projects, five US patents, two startup companies, and over 3000 citations. His research has been funded by more than 15 million of government and industry grants. He has served as the general chair, program committee chair, officer, or PC member of most major conferences in networking and real-time systems.

Prof. Bestavros has received distinguished service awards from both the ACM and the IEEE, and is a senior member of both the IEEE and ACM. He is the Chair of the IEEE Computer Society Technical Committee on the Internet and a distinguished speaker of the IEEE.



John W. Byers is Associate Professor and Associate Chair of Computer Science at Boston University (B.U.), Boston, MA. Prior to joining B.U., he completed the Ph.D. degree in computer science at the University of California at Berkeley in 1997 and was a postdoctoral researcher at the International Computer Science Institute in Berkeley in 1998. His research interests include algorithmic aspects of networking, Internet content delivery, and pay-per-click advertising.

Dr. Byers received a National Science Foundation CAREER Award in 2001, and the IEEE ICDE Best Paper Award in 2004. He serves on the executive committee of ACM SIGCOMM as Awards Chair and served on the editorial board of IEEE/ACM Transactions on Networking from 2003 to 2006. He has won awards for his teaching, is a coinventor on several patents, and is an author of 50 widely-cited journal and conference publications. He has served as Chief Technology Officer at Adverplex, Inc., a search engine marketing firm based in Cambridge, MA, since the company's founding in 2005. Previously, he served for many years as Affiliated Scientist at Digital Fountain, Inc., and as an Educational Consultant to Hewlett-Packard.