MultiQ: Automated Detection of Multiple Bottleneck Capacities Along a Path

Sachin Katti MIT CSAIL
Dina Katabi MIT CSAIL
Charles Blake MIT CSAIL
Eddie Kohler UCLA/ICIR
Jacob Strauss MIT CSAIL

ABSTRACT

`multiQ` is a passive capacity measurement tool suitable for large-scale studies of Internet path characteristics. It is the first passive tool that discovers the capacity of *multiple* congested links along a path from a single flow trace, and the first tool that effectively extracts capacity information from ack-only traces. It uses *equally-spaced mode gaps* in TCP flows' packet interarrival time distributions to detect multiple bottleneck capacities in their relative order.

We validate `multiQ` in depth using the RON overlay network, which provides more than 400 heterogeneous, well-understood Internet paths. We compare `multiQ` with two other capacity measurement tools (Nettimer and Pathrate) in the first large-scale wide-area evaluation of capacity measurement techniques, and find that `multiQ` is highly accurate; for instance, though `multiQ` is passive, it achieves the same accuracy as Pathrate, which is active.

Categories and Subject Descriptors: C.2.6 [**Computer Communication Networks**]: Internetworking –*Measurement*
General Terms: Measurement, Management
Keywords: Capacity, Measurement, Modeling

1 INTRODUCTION

Passive estimation of path properties has applications ranging from overlay network path optimization, to building representative descriptions of the current Internet for use in simulation and modeling, to tracking the evolution of the Internet over time using a library of traces collected over multiple years. In this paper, we focus on an important sub-problem—passively estimating multiple bottleneck link capacities along a path from a flow trace.

Current passive tools discover the minimum capacity along a path using logs of data packet interarrival times [10]. They fail, or have greatly reduced accuracy, when run on ack logs, so one cannot learn path capacity from sender-side traces, such as those at a Web server. They also recover only the minimum capacity, obscuring any secondary bottlenecks inside the network; but secondary bottleneck information is vital for network modeling and other applications.

We present `multiQ`, the first passive capacity measurement tool that avoids both these limitations. `multiQ` is based on *equally-spaced mode gaps*, or *EMG*, a new passive technique for inferring *multiple* link capacities from data *or ack* interarrival times. In contrast to prior work, which has inferred link capacity from the *location* of the modes in the packet interarrival distribution [4, 7, 10, 14], EMG uses the *distance between* consecutive modes.

We evaluate `multiQ`'s accuracy using over 10,000 experiments on 400 heterogeneous Internet paths with known likely capacities, and compare it with Nettimer [10], another passive capacity measurement tool, and Pathrate [4], an active tool. Our results include:

- `multiQ` is as accurate as Pathrate, which is active. In particular, 85% of the measurements are within 10% of their correct values.
- `multiQ` is 11% more accurate than Nettimer when both tools are given access only to receiver-side traces (data packet interarrivals). Nettimer needs access to both receiver- and sender-side logs to achieve accuracy comparable to `multiQ`.
- Run on ack traces, 70% of `multiQ`'s measurements are within 20% of their actual values. Though the accuracy is lower than in the case of data traces, it is a substantial improvement over prior tools; for instance, run on the same traces, less than 10% of Nettimer's measurements are within 20% of their actual values.

This material is based in part upon work supported by the National Science Foundation under Grant No. 0230921.

IMC'04, October 25–27, 2004, Taormina, Sicily, Italy.

2 CROSS TRAFFIC: NOISE OR DATA?

The *packet pair* technique has traditionally been used to infer the minimum capacity along a path. A sender emits a pair of probe packets back-to-back; assuming cross traffic does not intervene between the two probes, they arrive spaced by the transmission time on the bottleneck link. The capacity of the bottleneck is computed as $C = S/T$, where S is the size of the second probe and T is the time difference between the probes' arrivals at the receiver.

Cross traffic can cause errors in packet pair-based capacity estimates [4]. Compression errors happen when the first packet of a probe pair gets delayed more than the second, because it gets queued up behind cross traffic downstream of the bottleneck link. This shrinks the arrival spacing, leading to an overestimate. Inflation errors occur when cross traffic intervenes between the probe packets upstream of the bottleneck; this expands the arrival spacing, leading to an underestimate. To eliminate these cross-traffic effects, prior work sends trains of packets (packet bunch mode) [16] or a variety of packet sizes [4]; uses the global mode in the interarrival histograms [10]; and so forth. Yet, as the bottleneck becomes more congested, eliminating the effect of cross traffic becomes more challenging, Given this, is it possible that cross-traffic effects contain any useful information, rather than just being noise? We demonstrate that cross traffic, with proper interpretation, actually helps detect not only the minimum capacity along the path, *but also the capacities of other congested links.*

A *cross-traffic burst* is all traffic that intervenes between two consecutive packets of a flow. We seek to understand the probability distribution of cross-traffic burst sizes: i.e., the chance that a given amount of traffic will intervene between consecutive packets of a flow at a congested link. We examined 375 million packets in 258 NLANR traces, collected at 21 backbone locations, with a total of about 50,000 significant flows. (See Table 1 for a definition of "significant flow" and other important terms.) The diversity and size of this data set makes it a plausible sample of the Internet. For each pair of packets in a significant flow, we compute the intervening cross-traffic burst at the link where the trace is taken. This is repeated for all significant flows. Figure 1a shows the distribution of the sizes of these bursts. Note the surprising regularity: sharp modes separated by equal gaps of 1500 bytes.

Bottleneck	A link where traffic faces queuing.
Significant flow	A TCP flow that achieves an average packet rate > 10 pps (≈ 1 pkt/RTT), contains at least 50 packets, and has an MTU of 1500 bytes. (The vast majority of medium-to-long data flows have this MTU.)
Cross-traffic burst	Traffic intervening between two consecutive packets of a traced flow.
Capacity	The maximum rate at which packets can be transmitted by a link.
Narrow link	The link with the smallest capacity along a path.
Tight link	The link with minimum available or unused capacity along a path.
Path capacity	Capacity of the narrowest link on that path.

Table 1—Definitions of terms used in this paper.

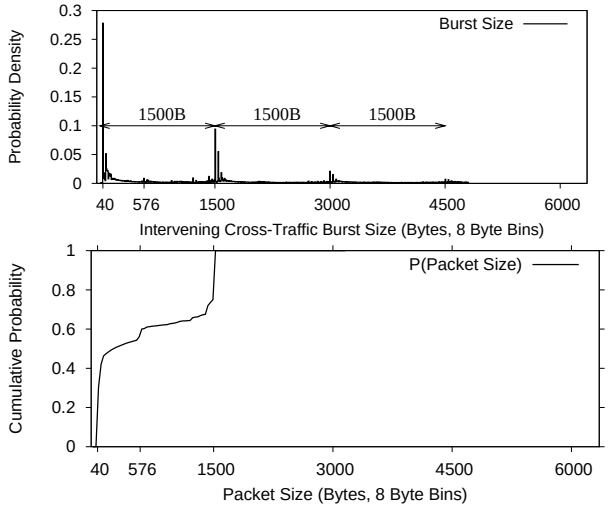
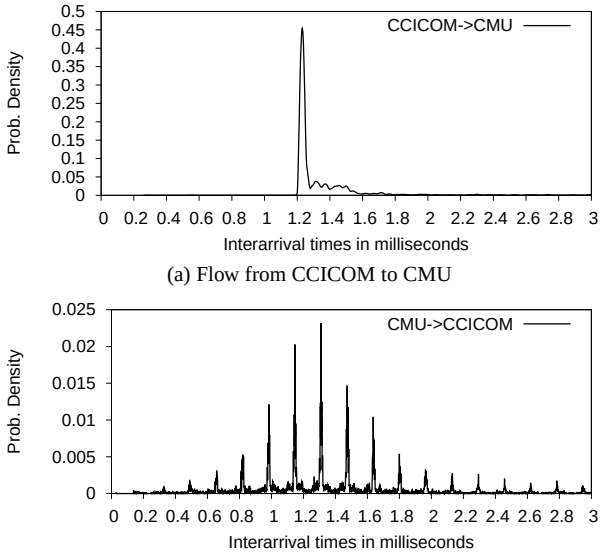


Figure 1—(a) The distribution of cross traffic between consecutive packets in a significant flow has equally-spaced mode gaps of 1500 bytes. (b) The CDF of packet size reveals frequencies of 40- and 1500-byte packets.



(a) Flow from CCICOM to CMU

(b) Flow from CMU to CCICOM

Figure 2—Interarrival PDFs for CCICOM–CMU path in both directions.

To understand this result, see Figure 1b, which shows a cumulative distribution function (CDF) of packet sizes in these traces. The dominant sizes are 40 and 1500 bytes; no other sizes are highly pronounced. (This replicates earlier results [22].) Thus, we would expect that the modes in the burst distribution will stem from 40- and 1500-byte packets; and 1500-byte packets should dominate the modes in Figure 1a, given that they are almost 40 times bigger. The 40-byte packets broaden the 1500-byte modes, and less common sizes create the bed of probability under the modes.

How will these modes be reflected in passive measurements that might not see the physical cross traffic? Once the measured flow reaches a point of congestion—a queue—the idle intervals squeeze out, and the packets (of both our flow and cross traffic) compress nearer in time. Thus, provided subsequent links are uncongested, the interarrival times observed at the receiver are proportional to cross-traffic burst sizes on the congested link. Since the cross-traffic burst size PDF contains modes separated by 1500 bytes, *we expect the PDF of interarrival times in a flow to have modes separated by the transmission time of 1500 bytes at some bottleneck link.*

3 Capacity Estimation with EMG

3.1 Examining an Interarrival PDF

We motivate our work by describing the outcome of a simple experiment. We first download a large file from a machine in CCICOM which has a 100 Mb/s access link, to one at CMU which has a 10 Mb/s access link. Figure 2a shows the interarrival PDF of the data packets. The distribution shows a single spike at 1.2 ms, the transmission time of a 1500-byte packet on a 10 Mb/s link. There is nothing special about this PDF; 10 Mb/s is the minimum capacity link along the path, and the spike in the PDF shows that most packets were queued back-to-back.

Next, we repeat the experiment along the reverse path and plot the interarrival PDF seen at CCICOM in Figure 2. The envelope of the distribution is again centered near 1.2 ms, because of the upstream 10 Mb/s link; but it is modulated with sharp spikes separated by *equally-spaced mode gaps* (EMGs) of 0.12 ms, which is the transmission time of a 1500-byte packet on a 100 Mb/s link.

To understand this PDF, consider what happens as packets go from CMU to CCICOM. As packets traverse the 10 Mb/s CMU access link, they become spaced by 1.2 ms, the transmission time of one packet on that link. The interarrivals remain relatively unperturbed as the packets cross the Internet backbone. Then the packets reach the 100 Mb/s CCICOM access link, where the flow faces congestion again. There, the spacing of two consecutive packets changes in one of three ways:

(a) *Neither packet is queued* (Figure 3a). The interarrival, or the time between the trailing edges of the two packets, remains 1.2 ms.

(b) *Either packet is queued and the queue empties between the departure time of the two packets.* Figure 3b shows an example where the first packet arrives while a cross-traffic packet is in the process of being transmitted. The packet has to wait for that transmission to finish, plus any remaining cross-traffic packets in the queue. This waiting time takes values spread over a wide range, depending on the total number of bytes that must be transmitted before our packet. If the second packet is not queued, then the interarrival time becomes 1.2 ms minus the delay of the first packet. Interarrival samples of this type are spread over a wide range with no pronounced values or modes, and contribute to the bed of probability under the spikes in Figure 2. A similar argument applies if the second packet is queued and the first is not, or even if the two packets are both queued, as along as the packets belong to different queuing epochs (the queue empties between their departures).

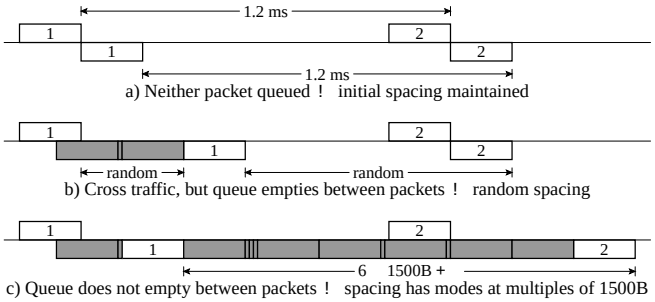


Figure 3—Various cases of packet spacing on CCICOM's access link. Arrivals are shown above the line, departures below the line. Light packets are from the traced flow and dark packets are cross traffic. Cross traffic arrivals are not shown.

(c) *Either packet is queued and the queue does* not *empty between the departure times of the two packets* (Figure 3c). The resulting interarrival is the transmission time of the intervening cross-traffic burst plus the second packet in the pair. Since cross-traffic bursts have modes at multiples of 1500 bytes, interarrival samples of this type will show modes spaced by 0.12 ms (the transmission time of 1500 bytes on 100 Mb/s). The input interarrival of 1.2 ms is a factor of 10 higher than this mode spacing, so these modes will be centered around 1.2 ms unless the queuing is extremely bursty.

Figure 2b also shows some symmetry around 1.2 ms. Our traced packets arrive at the CCICOM queue equally spaced by 1.2 ms. If cross-traffic effects stretch a pair of packets in the traced flow, the resulting interarrival sample will lie to the right of the 1.2 ms mode; if they squeeze the pair, the interarrival sample lies to the left of the 1.2 ms mode. On this link, it seems that the probability of stretching and squeezing were close.

This simple experiment teaches us two lessons: (1) Equally-spaced mode gaps (EMGs) in a flow's interarrival PDF correspond to the transmission times of 1500-byte packets on some bottleneck along the path. (2) The envelope of the PDF describes the minimum-capacity congested link along the path, whose output gets modulated by downstream congested links.

3.2 Interarrival PDF Variations

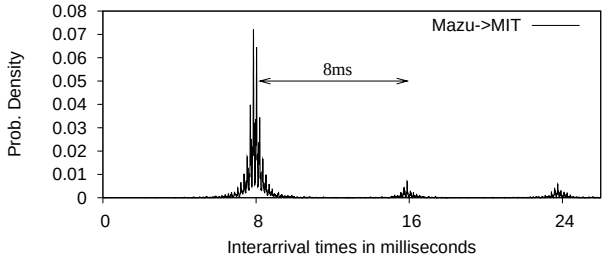
Inspection of interarrival PDFs for over 400 different Internet paths from the RON testbed (described in §5) shows that most exhibit equally-spaced mode gaps separated by the transmission time of a 1500-byte packet on a well-known link capacity. For lack of space we show only a few PDFs, chosen to exemplify the possible shapes.

Figure 4a shows a flow going from a lower-capacity bottleneck to a higher-capacity one. This time the upstream bottleneck (a T1) is highly congested, so the 8 ms primary EMGs are modulated by smaller EMGs of 0.12 ms corresponding to the 100 Mb/s link.

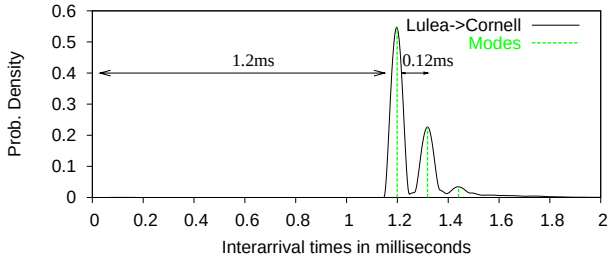
Figure 4b demonstrates a rare case where the PDF contains evidence of a congested link *upstream* of the minimum-capacity link. The flow traverses an upstream highly congested 100 Mb/s bottleneck and then a downstream 10 Mb/s bottleneck. The downstream bottleneck erases the first few spikes, piling up their probability at 1.2 ms, but the tail of 0.12 ms EMGs from the highly-congested 100 Mb/s link is long enough that a second spike remains.

Figure 4c shows an interesting three-bottleneck structure. The minimum-capacity bottleneck is a 380 Kb/s link, which is apparent from the envelope's peak. The envelope is modulated by EMGs of around 1.2 ms, revealing a 10 Mb/s link. If we then look closely around one of these modes, we see smaller modes equally spaced at intervals of 0.08 ms, revealing a downstream 155 Mb/s link.

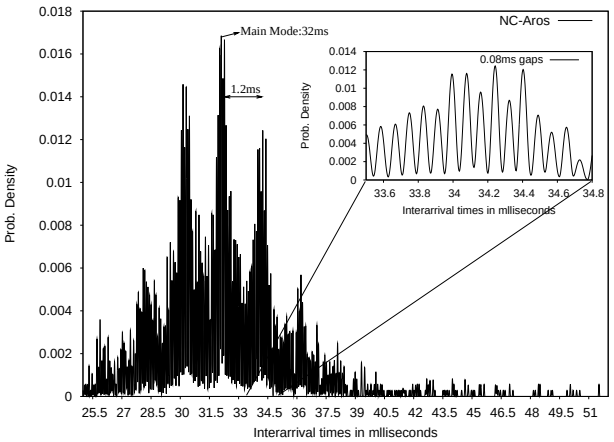
As more bottlenecks leave their fingerprints on the flow's interarrivals, it becomes harder to disentangle their marks. 2 bottlenecks are relatively easy to identify, but we've never seen more than 3. We



(a) Upstream congested T1 and downstream 100 Mb/s.



(b) Upstream highly congested 100 Mb/s and downstream 10 Mb/s.



(c) Three bottlenecks. The envelope peaks at 32 ms, indicating an upstream 380 Kb/s link; 1.2 ms EMGs correspond to the 10 Mb/s downstream link; and 0.08 ms EMGs in the zoomed figure show a 155 Mb/s bottleneck.

Figure 4—Some interarrival PDFs with equally-spaced mode gaps.

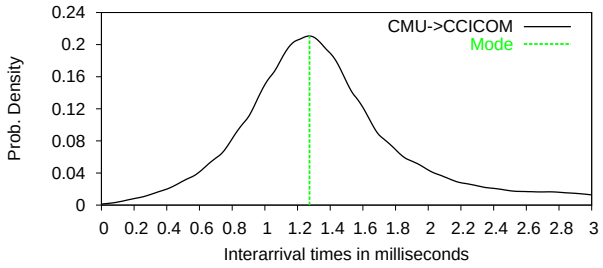
cannot confidently tell the maximum number of detectable bottlenecks in a single PDF, but without additional information, it will be difficult to identify more than 3 bottlenecks.

3.3 Ack Interarrivals

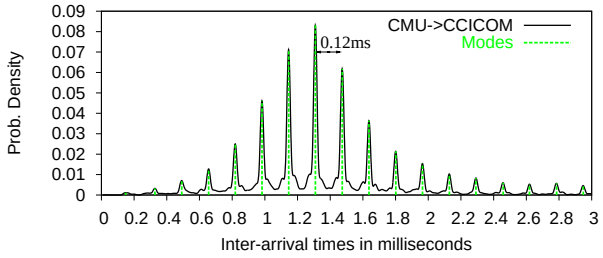
Thus far, we've created PDFs from data packet interarrivals, using traces collected downstream of any bottlenecks. This is useful when we have control of the receiver or some observation point close to the receiver. However, when the trace is taken at the sender side, the ack stream holds whatever information can be recovered; and when the observation point is in the middle of the network, both data and ack interarrivals should be studied to discover bottlenecks upstream and downstream of the observation point.

Ack interarrival PDFs contain more information than data interarrival PDFs, but they also have a higher level of noise. The major differences between the two PDFs are:

(a) *Forward- and reverse-path bottlenecks.* If every data packet generated an ack, and ack spacing was undisturbed by the network, then sender-side ack interarrivals would exactly equal the receiver-side data packet interarrivals. Of course, the world is more complicated than this. Acks also traverse the network, where their interar-



(a) Main mode at around 1.2ms shows the 10Mb/s CMU link



(b) Gaps of 0.12ms show the 100Mb/s CCICOM link

Figure 5—The data from Figure 4b at two different resolutions.

rival times pick up a record of any bottlenecks on the reverse path. This record is superimposed on the record of forward-path bottlenecks generated by the data packets. We cannot tell whether a specific bottleneck is on the forward or reverse path unless we examine the data interarrivals as well.

(b) *Noise.* Ack PDFs are much noisier than data-packet PDFs. Data packets are mostly 1500 bytes long; thus, they reinforce the EMG structure created by cross-traffic bursts (modes spaced by 1500-byte packets' transmission time) even if they arrive at the bottleneck queue back-to-back. 40-byte acks, on the other hand, do not reinforce the mode structure when they arrive back-to-back.

(c) *Delayed acks.* In many ack PDFs, the biggest spike is at twice the transmission time of the 1500 bytes packet on the minimum capacity link. This is caused by delayed acks, where the receiver generates one ack for roughly every second data packet.

Examination of many ack PDFs shows that EMG can be applied to ack interarrivals, but with lower accuracy than data packet interarrivals. In §5.3, we quantify the difference.

4 MULTIQ: AUTOMATING EMG

The `multiQ` passive bottleneck detection tool automates the EMG capacity detection technique. It takes as input a `tcpdump` trace, or a set of interarrivals obtained some other way, and automatically discovers and estimates the capacity of the bottlenecks traversed by certain flows specified by the user.

Automating multiple bottleneck discovery is tricky: it requires interpreting the interarrival PDF to extract the relevant information and ignore the noise. To do this, `multiQ` analyzes the interarrival PDF at a *progression of resolutions* corresponding to a known set of common link speeds. To demonstrate this, Figure 5 plots the CMU-to-CCICOM data from Figure 2b at two different resolutions. At the lower resolution, we see one large mode in the distribution, which corresponds to the upstream lower-capacity bottleneck. As we increase the resolution, the large mode becomes fractured into smaller spikes corresponding to the higher-capacity bottleneck.

Figure 6 shows the `multiQ` procedure in pseudocode. At each resolution, starting with the highest resolution, `multiQ` constructs a kernel density estimate of the PDF[1] and scans it for modes, which

1. Compute flow interarrivals from trace file
2. Set *scale* := 10 μs
3. While *scale* $<$ 10,000 μs:
4. Compute kernel PDF estimate with width = *scale*
5. Find the modes
6. If there's only one mode, at M:
7. Output a capacity of $(1500 \times 8/M)$ Mb/s
8. Exit
9. Compute the mode gaps
10. Compute the PDF of the gaps
11. Set G := the tallest mode in the gap PDF
12. If the probability in $G > 0.5$:
13. Output a capacity of $(1500 \times 8/G)$ Mb/s
14. Increment *scale*

Figure 6—Pseudocode for `multiQ`.

are defined as local maxima with statistically-significant dips.[2] The gaps between these modes are computed. Then, `multiQ` finds the probability distribution of the gaps themselves. A mode in the gap's PDF corresponds to a highly repeated gap length—the hallmark of a congested link. If `multiQ` finds a significantly dominant mode in the gap distribution at the current resolution, it decides that mode represents the transmission time of 1500 bytes on some bottleneck, and outputs that bottleneck's capacity. If there is no dominant gap at the current resolution, `multiQ` decreases the resolution by increasing the kernel width, which is similar to the bin width of a histogram, and repeats the procedure.

When run on data-packet traces, `multiQ` estimates the capacity of bottlenecks upstream from the observation point. To estimate bottleneck capacities downstream of the observation point, it needs access to ack traces. When analyzing ack interarrival PDFs, `multiQ` uses a slightly different procedure to deal with the first mode in the PDF: a large spike close to zero is a sign of compressed acks and should be ignored, whereas a spike located at twice the repeated gap in the PDF is a sign of delayed acks and corresponds to the transmission time of 3000 bytes on the bottleneck link. EMG estimation is less robust on ack traces than data-packet traces, so the current version of `multiQ` does not try to discover bottlenecks with capacity higher than 155 Mb/s when run on ack traces.

The EMG technique relies on cross-traffic burst structure, which depends on packet size distribution. If 1500 bytes stops being the dominant large-packet mode, this technique will fail. Fortunately, this distribution appears to be changing towards further emphasis of the 40-byte and 1500-byte modes; for instance, compare Claffy's packet size distributions from 1998 and 2001 [3, 22].

5 VALIDATION

We evaluate the accuracy of `multiQ` using 10,000 experiments over 400 diverse Internet paths from the RON overlay network, and compare it both with known topology information and two other capacity measurement tools, Pathrate and Nettimer. Our results show:

- When measuring minimum-capacity bottlenecks, `multiQ` is as accurate as Pathrate, an active tool, with 85% of its measurements within 10% of their true value. `multiQ` is 11% more accurate than Nettimer when both tools are given access only to receiver-side traces (data packet interarrivals). Nettimer needs access to

[1]Kernel density estimation is a standard method for constructing an estimate of a PDF from measurements of the random variable; the flat bins of a histogram would prevent precise mode identification at low resolutions. We use the quartic kernel density function [21].

[2]A significant dip [21] is defined as one in which the dips on either side of a local maximum drop by more than the standard deviation of the kernel density estimate at the local maximum. The standard deviation is given by $StdDev(g(x)) = \sqrt{g(x) \times R(K)/nh}$, where $g(x)$ is the estimate at point x, $R(K)$ is the roughness of the kernel function, n is the number of points, and h is the kernel's width.

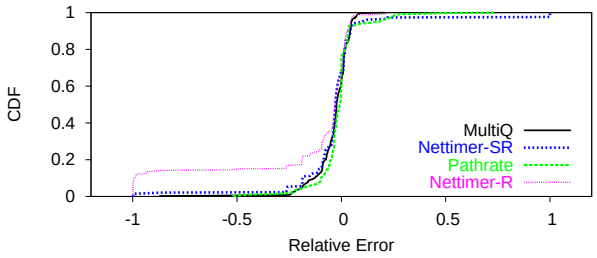


Figure 7—Comparison of the accuracy of MultiQ, Nettimer and Pathrate. Graphs show the CDF of the relative error. MultiQ and Nettimer-R require only receiver-side traces, while Nettimer-SR requires both receiver- and sender- side traces.

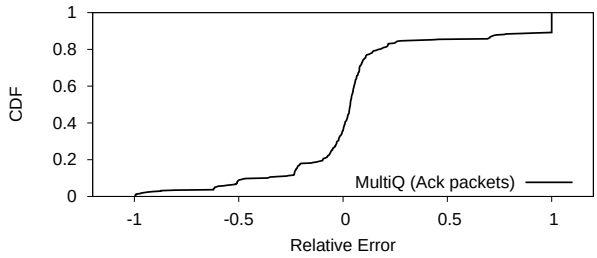


Figure 8—Accuracy of capacity estimates based on ack interarrivals.

both receiver- and sender-side logs to achieve accuracy comparable to `multiQ`.

- On sender-side ack traces, 70% of `multiQ`'s measurements are within 20% of their correct value.
- `multiQ` automatically detects 64% of the non-minimum-capacity bottlenecks ("tight" links); misses 21%, though a human could detect them visually on an interarrival PDF using our EMG technique; and mislabels 15%.
- `multiQ`'s average error is highly independent of flow size for flows larger than 50 packets (discussion omitted for space; see [8]).

5.1 Experimental Methodology

Ideally, we'd like to have information about all the capacities and loss rates along a large number of heterogeneous paths that form a representative cross-section of the network. This is inherently difficult on the Internet, of course, but we have tried to evaluate our tool on as representative a network as possible. We use the RON overlay network [18], whose 22 geographically-distributed nodes have a diverse set of access links and ISPs on both the commercial Internet and Internet2. 9 nodes have 100 Mb/s uplinks, 6 have 10 Mb/s, 3 have T1, and 4 have DSL. Of RON's 462 heterogeneous paths, only 25% use Internet2.

We compare the capacity tools' estimates for each RON path against that path's "true" bottleneck capacity. The "true" capacity was determined by contacting each node's hosting site and obtaining a list of all their access links and the capacities of the local networks to which the nodes are connected. RON nodes not on Internet2 have low-speed access links ranging from DSL to 10 Mb/s, and hence are unlikely to encounter a lower-capacity link on the Internet backbone. For nodes in Internet2, we additionally obtained information about *all* Internet2 links on the relevant paths.

To verify the consistency of these "true" capacities, we ran all three capacity measurement tools and a number of `ttcp` and UDP flows of varying rates on each path. If a path's results pointed out an inconsistency—for example, if `ttcp` or UDP obtained more bandwidth than the "true" capacity—then we eliminated the path from our experiments. Only 57 out of 462 paths needed to be eliminated.

We also analyzed the errors in interarrivals of successive packets computed from `tcpdump` timestamps. A data set collected at RIPE containing timestamps from both DAG hardware and `tcpdump` [23] indicates that errors in interarrivals are only a few μs. Such small errors should not affect our results.

5.2 Minimum Capacity Estimation

We first evaluate `multiQ`'s minimum capacity estimation, and compare it with the results of two other capacity measurement tools—Pathrate, which is active, and Nettimer, which is passive. We first conduct a 2 minute run of `ttcp` and collect traces at both endpoints, which serve as data sets for `multiQ` and Nettimer. Immediately thereafter, we run Pathrate on the same path and compute its estimate; we use the average of Pathrate's high and low estimates. This procedure is repeated five times, and we report the average. Finally, the same set of experiments is run both at day and night, to compensate for time-of-day patterns. We plot the *relative error* ξ for each capacity estimate C_e; this is defined as $\xi = \frac{C_e - C_t}{C_t}$, where C_t is the path's "true" capacity.

Figure 7 shows the cumulative distribution function (CDF) of the relative errors of `multiQ`, Nettimer, and Pathrate estimates on the 405 RON paths with reliable "true" capacities. Nettimer has two lines: Nettimer-SR uses both sender- and receiver-side traces, while Nettimer-R uses only receiver-side data-packet traces. `multiQ` uses only receiver-side traces. Ideally, the CDF should be a step function at 0, meaning that all experiments reported the "true" capacity. A negative relative error indicates an underestimate of capacity, whereas a positive relative error indicates an overestimate.

Our results show that `multiQ`, though passive, is as accurate as the active tool Pathrate. In particular, 85% of the measurements are within 10% of their "true" value. `multiQ` is more accurate than Nettimer if both tools are given only receiver-side traces. In this case, only 74% of Nettimer estimates are within 10% of the actual values. Nettimer achieves an accuracy comparable to `multiQ` only when given access to both receiver- and sender-side traces. All three tools are biased towards underestimating the capacity.

Next, we look more closely at the errors exhibited by each tool. `multiQ` errors are caused mainly by over-smoothing in the iterative procedure for discovering mode gaps. This can flatten the modes and prevent accurate gap computation. Pathrate's logs indicate that its errors happen when the interarrival PDF exhibits many modes; the correct bottleneck capacity is usually one of those modes, but Pathrate picks a different mode as the bottleneck capacity. When Nettimer made errors, we found that often the path has low RTT (< 16 ms). The tool mistakes the RTT mode in the inter-arrival PDF for the transmission time over the bottleneck. The effect is most pronounced when Nettimer is operating with only traces at the receiver side; when it has both traces, we theorize that it can estimate the RTT and eliminate the corresponding mode.

Further, our experiments show that different tools can disagree on the capacity of a particular path, but can all be correct. In particular, Pathrate repeatedly reports capacities of 1 Mb/s for paths going to cybermesa, while Nettimer and `multiQ` estimate them as 10 Mb/s. Further investigation revealed that the differences are due to flow rate limits. The cybermesa access link capacity of 10 Mb/s is correctly estimated by Nettimer and `multiQ`, but Pathrate's relatively long trains of back-to-back packets trigger cybermesa's leaky bucket rate limit. They exceed the maximum burst size of the leaky bucket and become limited by the token rate, which is 1 Mb/s; TCP windows stay smaller than the bucket size, and so its packets are spaced by the actual link. This was confirmed by the site's owner.

5.3 Minimum Capacity Estimation Using Acks

Unlike existing tools, `multiQ` can obtain a reasonable capacity estimate using only a sender-side trace of ack interarrivals. Figure 8

shows the relative error of multiQ's minimum capacity estimation using ack interarrivals. The data comes from the experiments described in §5.1. Since acks contain information about both forward and reverse links, we define the true capacity C_t for sender-side multiQ measurements as the minimum of the forward and reverse paths' capacities. Sender-side ack interarrivals produce lower-quality results than receiver-side data packet interarrivals, but still, 70% of the measurements are within 20% of the "true" value.

5.4 Tight Links

We now evaluate multiQ's ability to discover non-minimum-capacity bottlenecks, or *tight links*. Since it is difficult to say with confidence what the tight links along a path in the Internet are, we limit our tests to Internet2 paths. Internet2 has a very low utilization [1], so any congestion should be at the edges, or access links, whose capacities we know. Also, because downstream narrow links tend to erase the effect of upstream bottlenecks from data-packet interarrivals (see §3.1), we limit this test to paths in which the downstream bottleneck capacity is larger than the upstream one.

To summarize the results, 64% of the experiments reported a tight link present on the path, defined as a non-minimum-capacity link within 20% of the actual tight link capcity. The relative error of these correct tight link capacities was 0.156; the standard deviation in error was 0.077. 15% of the experiments reported an incorrect tight link, and the rest (21%) reported only the minimum bottleneck.

6 Related Work

multiQ is related to prior work on capacity measurements and tight link discovery. Currently, Nettimer [10] is the main passive tool for discovering path capacity. Our work builds on its insights, but achieves higher accuracy and can discover multiple bottleneck capacities. Further, multiQ can discover bottleneck capacities from sender or receiver-side traces, whereas Nettimer requires the receiver-side trace to achieve any accuracy. Jiang and Dovrolis [7] describe a passive method based on histogram modes.

There are many active tools for measuring path capacity. Some of these tools try to find the capacities of all links along the path [12, 15]. Others, such as Pathrate, focus on the minimum capacity of a path [4]. The accuracy and the amount of generated traffic vary considerably from one tool to another. Being passive, our tool differs from active tools in its methodology and characteristics. Prior work that detects *tight* links has all been active to our knowledge [2, 12]. There are also tools for discovering the available bandwidth along a path [5, 6, 13, 19, 20], which all actively probe the network.

Much prior work used packet interarrival times to estimate link capacities. Keshav proposed the packet pair concept for use with Fair Queuing [9]. Packet pair is at the heart of many capacity and available bandwidth estimation methods, including ours. Cross traffic can cause errors in packet pair-based capacity estimates. Paxson observed that the distribution of packet-pair capacity measurements is multi-modal [17], and Dovrolis et al [4] show that the true capacity is a local mode of the distribution, often different from its global mode. It has also been noted that some of the modes in the interarrival PDF may be created by secondary bottlenecks or post-narrow links [4, 11, 14]. Various mechanisms to filter out the cross traffic effects were proposed, such as using the minimum dispersion in a bunch of packet pairs, using the global mode in the dispersion distribution [7, 10], and using variable size packet pairs [4].

7 Conclusions

multiQ is the first passive tool that can discover the capacity of *multiple* congested links along the path traversed by a single flow, and the first tool that effectively extracts capacity information solely from ack traces. It accomplishes this by extracting useful information from cross-traffic bursts, which previous capacity estimation tools considered to be noise, using the equally-spaced mode gaps technique. multiQ achieves accuracy comparable to Pathrate, an active capacity measurement tool, and can detect up to three bottleneck capacities along a single path.

The code for multiQ is available as a plugin module for Click, a modular software system for packet processing and router forwarding paths. Simple configurations can extract capacities from live packet flows, tcpdump traces, traces in other formats, and raw files of interarrival times. multiQ may be downloaded from http://nms.lcs.mit.edu/MNM/mm.html.

References

[1] Abilene. http://monon.uits.iupui.edu/.

[2] A. Akella, S. Seshan, and A. Shaikh. An Empirical Evaluation of Wide-Area Internet Bottlenecks. In *Proc. IMC*, October 2003.

[3] K. Claffy, G. Miller, and K. Thompson. The Nature of the Beast: Recent Traffic Measurements from an Internet Backbone, 1998. http://www.caida.org/outreach/resources/learn/packetsizes/.

[4] C. Dovrolis, P. Ramanathan, and D. Moore. Packet Dispersion Techniques and Capacity Estimation. *IEEE/ACM Trans. on Neworking*. Under submission.

[5] N. Hu and P. Steenkiste. Evaluation and Characterization of Available Bandwidth Techniques. *IEEE JSAC Special Issue in Internet and WWW Measurement, Mapping, and Modeling*, 2003.

[6] M. Jain and C. Dovrolis. Pathload: A Measurement Tool for End-to-End Available Bandwidth. In *Proc. Passive and Active Measurement Workshop*, March 2002.

[7] H. Jiang and C. Dovrolis. Source-Level IP Packet Bursts: Causes and Effects. In *Proc. IMC*, October 2003.

[8] S. Katti, D. Katabi, C. Blake, E. Kohler, and J. Strauss. M&M: A passive toolkit for measuring, tracking and correlating path characteristics. Technical Report 945, MIT CSAIL, 2004.

[9] S. Keshav. A Control-Theoretic Approach to Flow Control. In *Proc. ACM SIGCOMM*, September 1991.

[10] K. Lai and M. Baker. Nettimer: A Tool for Measuring Bottleneck Link Bandwidth. In *Proc. USENIX*, 2001.

[11] Kevin Lai and Mary Baker. Measuring Bandwidth. In *Proc. IEEE INFOCOM*, 1999.

[12] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. User Level Internet Path Diagnosis. In *Proc. ACM SOSP*, October 2003.

[13] B. Melander, M. Bjorkman, and P. Gunningberg. A New End-to-End Probing and Analysis Method for Estimating Bandwidth Bottlenecks. In *Proc. Global Internet Symposium*, 2000.

[14] A. Pasztor and D. Veitch. The Packet Size Dependence of Packet Pair Methods. In *Proc. 10th IWQoS*, 2003.

[15] pathchar. ftp://ee.lbl.gov/pathchar.tar.Z.

[16] V. Paxson. End-to-End Internet Packet Dynamics. *IEEE/ACM Trans. on Networking*, June 1999.

[17] V. E. Paxson. *Measurements and Analysis of End-to-End Internet Dynamics*. PhD thesis, Berkeley, 1997.

[18] Resilient Overlay Networks. http://nms.lcs.mit.edu/ron/.

[19] V. J. Ribeiro, M. Coates, R. H. Riedi, S. Sarvotham, and R. G. Baraniuk. Multifractal Cross Traffic Estimation. In *Proc. ITC Specialist Seminar on IP Traffic Measurement*, September 2000.

[20] V. J. Ribeiro, R. H. Riedi, R. G Baraniuk, J. Navratil, and L. Cottrell. pathChirp: Efficient Available Bandwidth Estimation for Network Paths. In *Proc. Passive and Active Measurement Workshop*, 2003.

[21] D. Scott. *Multivariate Density Estimation*. John Wiley, 1992.

[22] C. Shannon, D. Moore, and K. Claffy. Beyond Folklore: Observations on Fragmented Traffic. In *IEEE/ACM Trans. on Networking*, 2002.

[23] H. Uijiterwall and M. Santcroos. Bandwidth Estimations for Test Traffic Measurement Project, December 2003.