A Minimum-Delay-Difference Method for Mitigating Cross-Traffic Impact on Capacity Measurement

Edmond W. W. Chan[‡], Xiapu Luo[§], and Rocky K. C. Chang[‡]
Department of Computing[‡]
The Hong Kong Polytechnic University
{cswwchan|csrchang}@comp.polyu.edu.hk
College of Computing[§]
Georgia Institute of Technology
csxpluo@cc.gatech.edu

ABSTRACT

The accuracy and speed of path capacity measurement could be seriously affected by the presence of cross traffic on the path. In this paper, we propose a new cross-traffic filtering method called minimum delay difference (MDDIF). Unlike the classic packet-pair dispersion techniques, the MDDIF method can obtain accurate capacity estimate from the minimal possible delay of packets from different packet pairs. We have proved that the MDDIF method is correct and that it takes less time to obtain accurate samples than the minimum delay sum (MDSUM) method. We also present analytical and measurement results to evaluate the MDDIF method and to compare its performance with the MDSUM method.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations; C.4 [**Performance of Systems**]: Measurement Techniques

General Terms

Measurement, Experimentation, Performance

Keywords

Network capacity, Bottleneck bandwidth, Non-cooperative measurement, Packet-pair dispersion, Packet delay

1. INTRODUCTION

Knowing network capacity is useful for many network applications to improve their performance. Network capacity (a.k.a. bottleneck bandwidth) refers to the smallest transmission rate of a set of network links, forming a network path from a source to a destination [5]. Measuring capacity is, however, a challenging task in practice, because the accuracy and speed can be adversely affected by cross traffic, packet loss and reordering events, packet sizes, time resolution supported by measurement endpoints, probing methods, and others.

Existing capacity measurement tools are mostly active methods which are based on two main approaches: variable packet size and packet dispersion. The focus of this paper is on the latter approach, in particular, the packet-pair dispersion (PPD) technique that sends a pair of back-to-back probe packets to measure their dispersion. The PPD technique could be conducted in a cooperative (e.g., [5, 10]) or non-cooperative manner (e.g., [25, 14, 4]).

The accuracy and speed of the PPD technique, however, could be seriously degraded by the cross traffic present on the path under measurement. Therefore, the basic PPD technique is usually augmented by a component to filter measurement samples that have been biased by cross traffic. A notable example is the minimum delay sum (MDSUM) method first introduced to CapProbe [10]. The MDSUM method filters out packet pairs that do not meet a minimum delay sum condition.

However, the existing cross-traffic filtering techniques still suffer from slow speed and a large overhead. Applications, such as determining optimal software download rates, forming peer-to-peer networks, and establishing multicast trees, will benefit from a fast estimation of network capacity [25]. Moreover, injecting a large amount of probe traffic unnecessarily not only prolongs the estimation process, but also affects the normal traffic and introduces additional processing burdens to both the measuring and remote nodes.

In this paper, we propose a new technique called minimum delay difference (MDDIF). Unlike the MDSUM method that admits a packet pair as the basic unit for capacity measurement, the MDDIF method admits a packet as a basic unit. The MDDIF method obtains minimal possible delay of a first probe packet and a second probe packet, but these two packets do not necessarily belong to the same packet pair. By exploiting

CoNEXT'09, December 1–4, 2009, Rome, Italy.

useful information in a single packet (which is discarded by the MDSUM method), the MDDIF method requires less time to obtain accurate capacity estimates and has very low computation and storage costs.

In §2, we first summarize the existing cross-traffic filtering techniques. In §3, we present the model and assumptions used throughout this paper and review the classical PPD technique. We then introduce the MDDIF method in §4 and compare its performance with the MDSUM method based on their first passage times in §5. In §6, we further evaluate the MDDIF method's performance based on Internet and testbed experiment results. We finally conclude this paper in §7.

2. RELATED WORK

There are some apparent similarity between the VPS techniques [9, 18, 6] and the MDDIF method in terms of requiring minimal possible packet delay. The VPS techniques require the minimal possible delay for a sequence of variable-sized packets; the MDDIF method, however, requires the minimal possible delay only for a pair of packets with the same size. As a result, the MDDIF method achieves a faster capacity estimation with a lower storage requirement.

Previous works [24, 22, 5, 25, 10, 4] using packet dispersion for capacity measurement compute capacity estimates by measuring the PPDs based on the inter-arrival time of the two packets or the difference between the two packets' delay. Although the MDDIF method also involves sending packet pairs, it does not measure packet pairs' PPDs. Instead, it measures the minimal possible packet delay for the first and second packets.

Many cross-traffic filtering techniques have been proposed in the past. Carter and Crovella propose Bprobe [24] which filters inaccurate estimates using union and intersection of packet-pair measurements with different packet sizes. Lai and Baker [15] use a kernel density estimation method to filter capacity estimates. Pásztor and Veitch [22] analyze several types of components embedded in the packet-pair dispersion and select the capacity mode from the high-resolution histogram. Kapoor et al. [10] propose the use of the minimum of packet-pair delay sum to filter distorted dispersion samples.

The packet train dispersion (PTD) technique, on the other hand, performs capacity measurement with the dispersion of a burst of back-to-back probe packets. Pathrate [5] uses both PPD and PTD for capacity measurement. DSLprobe [4] exploits the PTD technique for capacity measurement and various methods to remove any cross-traffic interfered spacing between adjacent packets in the packet train. A most notable problem with the packet train technique is that as the trunk length increases, it is more likely for the packet train to be affected by cross traffic.

3. MODEL AND PRELIMINARIES

3.1 The model

This paper considers the capacity measurement scenario in Figure 1. A local endpoint measures the path capacity by dispatching a sequence of packet pairs (two back-to-back packets) to a remote endpoint. Each probe packet elicits a response packet from the remote endpoint. The (round-trip) network path under the measurement starts from and ends at the local endpoint, consisting of n (where $n \geq 2$) hops. The first m hops (where $1 \leq m < n$) belong to the forward path and the remaining $n - m$ hops to the reverse path. The probe packets travel on the forward path; the response packets travel on the reverse path.

Each hop consists of a (local, remote, or forwarding) node and its outgoing link. We use $H^{(h)}$ $(1 \leq h \leq n)$ to denote the h^{th} hop which transmits packets to the outgoing link with a rate of $C^{(h)}$ bits/second. For convenience, we label the hops on the path sequentially. Therefore, the local endpoint belongs to $H^{(1)}$, whereas the remote endpoint belongs to $H^{(m+1)}$. The figure also shows a bottleneck link on the forward path which belongs to a hop denoted by $H^{(h_f)}$. If there are more than one bottleneck hop on the forward path, $H^{(h_f)}$ is referred to the one with the largest h_f. The above applies similarly to the reverse path, where the bottleneck link belongs to a hop denoted by $H^{(h_r)}$.

There are three types of path capacity metrics: *forward-path capacity* (denoted by $C_f^{(n)}$), *reverse-path capacity* (denoted by $C_r^{(n)}$), and *round-trip capacity* (denoted by $C_b^{(n)}$), where

$$\begin{aligned} C_f^{(n)} &\equiv C^{(h_f)} = \min_{1 \leq h \leq m} C^{(h)}. \\ C_r^{(n)} &\equiv C^{(h_r)} = \min_{m+1 \leq h \leq n} C^{(h)}. \\ C_b^{(n)} &= \min\{C_f^{(n)}, C_r^{(n)}\}. \end{aligned}$$

3.2 The assumptions

Unless stated otherwise, we adopt the following assumptions in this paper:

1. Both the forward and reverse paths are unique and do not change during the measurement.
2. The forwarding node in each hop is a store-and-forward device using a FIFO queue.
3. Each probe packet elicits a single response packet from the remote endpoint with negligible delay.
4. All probe and response packets are received successfully. Combining with (1)-(3) also implies that the probe packets arrive at the remote endpoint in the original order, and the response packets arrive at the local endpoint in the original order.

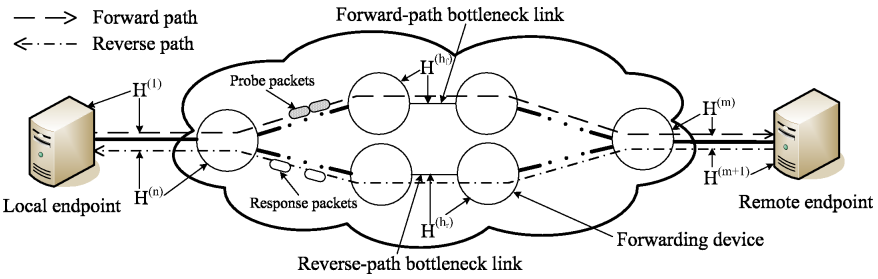


Figure 1: The capacity measurement scenario considered in this paper.

5. The processing delay introduced by the forwarding nodes is small compared with the packet-pair dispersion and therefore negligible.
6. The packet pairs are sufficiently spaced out that a first probe packet is never queued behind the preceding packet pair, and a first response packet is never queued behind the preceding response packets.

Assumptions (1)-(2) are reasonable and have been adopted in previous works [3, 24, 15, 5, 7]. Assumptions (3)-(5) are required to ensure that the packet-pair dispersion is not biased by the remote endpoint's processing delay, packet loss and reordering events, and forwarding nodes' processing delay. Finally, assumption (6) is valid for adequately spaced packet pairs.

3.3 Preliminaries

This section provides preliminary results for deriving the main results in the next two sections. These preliminary results are not new as they appeared in previous works, such as [16, 22, 14].

In a capacity measurement session, a local endpoint dispatches a sequence of packet pairs $P_i,\ i = 1, 2, \ldots$. Now consider any packet pair $\{p_{j-1}, p_j\}$ in the sequence, where $j = 2i$ indicates the position of the second packet in P_i. We also let S_f and S_r be the sizes of the probe and response packets in bits, respectively. Due to assumption (3), it is convenient to regard the first response packet as the first probe packet "bounced back" from the remote endpoint and similarly for the second response packet. Therefore, we also use p_{j-1} and p_j to refer to the first and second response packets, respectively, but S_f and S_r are generally different.

Individual packet delay Let $d_j^{(h)}$ be the time interval that p_j spends on the first h hops. As illustrated in Figure 2, $d_j^{(h)}$ can be defined recursively by

$$d_j^{(h)} = d_j^{(h-1)} + (w_j^{(h)} + X^{(h)} + T^{(h)})\ for\ h \geq 1, \quad (1)$$

and $d_j^{(0)} = 0$ [16]. The delay at $H^{(h)}$ comprises a queueing delay ($w_j^{(h)}$), a constant transmission delay of $X^{(h)}$ ($X^{(h)} = S_f/C^{(h)}$ for $1 \leq h \leq m$ and $X^{(h)} = S_r/C^{(h)}$ for $h > m$), and a constant delay ($T^{(h)}$) for propagating the packet to the next hop. The expression of $d_{j-1}^{(h)}$ for p_{j-1} is the same as Eqn. (1) after updating the subscripts.

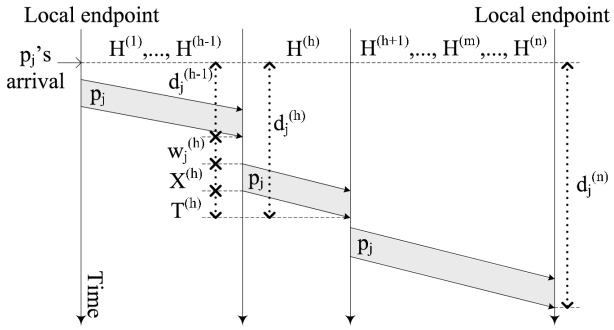


Figure 2: The delay components for p_j to traverse the first h hops.

Packet-pair dispersion and capacity The (round-trip) *packet-pair dispersion* (PPD) for $\{p_{j-1}, p_j\}$, denoted by $\delta_{j-1,j}^{(n)}$, is given by [22]

$$\delta_{j-1,j}^{(n)} = d_j^{(n)} - d_{j-1}^{(n)} + \tau_{j-1,j}^{(1)}, \quad (2)$$

where $\tau_{j-1,j}^{(1)}$ is the inter-arrival time for p_{j-1} and p_j at $H^{(1)}$. Without loss of generality, we let $\tau_{j-1,j}^{(1)} = 0$.

If the PPD is not affected by the cross traffic on the path, the unbiased PPD is given by [14]

$$\delta_{j-1,j}^{(n)} \equiv X^{(h_b)} = \max\left\{X^{(h_f)}, X^{(h_r)}\right\}, \quad (3)$$

where $h_b = h_f$ if $X^{(h_f)} > X^{(h_r)}$ and $h_b = h_r$ if $X^{(h_f)} \leq X^{(h_r)}$. Since $X^{(h_f)} = S_f/C_f^{(n)}$ and $X^{(h_r)} = S_r/C_r^{(n)}$, the path capacity computed based on $\delta_{j-1,j}^{(n)}$ can give $C_f^{(n)}$ ($= S_f/X^{(h_b)}$) and $C_r^{(n)}$ ($= S_r/X^{(h_b)}$). For the case of $S_f = S_r$, $C_b^{(n)} = S_f/X^{(h_b)} = S_r/X^{(h_b)}$.

4. MITIGATING CROSS-TRAFFIC INTERFERENCE

Previous studies [19, 23] have shown that the PPD could be distorted by two types of cross traffic. The first type is the traffic already existing in a forwarding node located after the bottleneck link when the

first probe/response packet arrives. This cross traffic could delay the first packet to the extent that the PPD is compressed, causing a capacity overestimation. The second type is the traffic intervening the first and second probe/response packets in a forwarding node. This cross traffic could increase the second packet's queueing delay, thus causing a capacity underestimation [5].

4.1 Minimum delay sum

A minimum delay sum (MDSUM) method is proposed to remove distorted PPDs for, such as, CapProbe [10] and AsymProbe [14]. The basic idea is that if any packet in a packet pair is interfered by cross traffic, additional packet delay will be introduced; therefore, a sum of the two packets' delay (i.e., a delay sum) will also increase. To implement this idea, CapProbe dispatches a sequence of packet pairs until a P_i satisfies the MDSUM conditions: Eqn. (4) holds for P_i, and the left and right hand sides in Eqn. (4) remain unchanged for the next I consecutive packet pairs ($I = 40$ suggested in [10]). CapProbe then uses P_i's PPD to compute the path capacity.

$$\min_i \left\{ d_{2i-1}^{(n)} + d_{2i}^{(n)} \right\} = \min_i \left\{ d_{2i-1}^{(n)} \right\} + \min_i \left\{ d_{2i}^{(n)} \right\}. \quad (4)$$

The drawback of the MDSUM technique is that it considers only the packet pair whose packets are *both* unaffected by cross traffic. It therefore discards all other packet pairs, including those in which *only* a single packet has been interfered. As a result, useful information in those packet pairs is not fully utilized to speed up the measurement process. This observation leads us to propose a new filtering technique that is based on the minimum delay of a single packet (instead of a packet pair), to be discussed next.

4.2 Minimal possible packet delay

Our new filtering technique is based on the notion of *minimal possible packet delay* (or minDelay) defined as:

DEFINITION 1. *A minimal possible packet delay is the delay experienced by a packet in a packet pair for which both the probe packet and the elicited response packet do not encounter any cross-traffic-induced queueing delay on the path, including*

1. *Type-H queueing delay: the queueing delay caused by the cross traffic present at the "head" of the queue upon the first packet's arrival, and*

2. *Type-I queueing delay: the queueing delay caused by the intervening cross traffic between the first and second packets in a packet pair.*

In the following we consider a packet pair $\{p_{j-1}, p_j\}$. It is not difficult to see that p_{j-1}'s minDelay can be obtained iff the probe packet and the elicited response packet are not queued (behind type-H cross traffic) at all hops of the path. Therefore,

PROPOSITION 1. *(The first packet's minDelay) The necessary and sufficient conditions for $d_{j-1}^{(n)}$ being a minDelay are $w_{j-1}^{(h)} = 0,\ h = 1, \ldots, n$.*

However, obtaining the conditions for a second packet's minDelay is more involved. We first derive in Prop. 2 general expressions for $w_j^{(h)}$ and $\delta_{j-1,j}^{(h)}$ which take into account the two types of cross traffic. Figure 3 illustrates the two scenarios for which their PPDs are not the same.

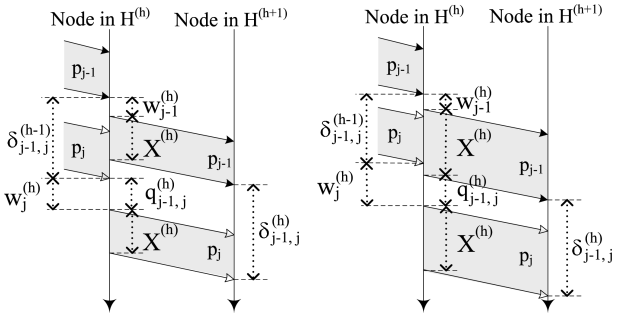
PROPOSITION 2. *At $H^{(h)}$, p_j's queueing delay is given by Eqn. (5), and $\{p_{j-1}, p_j\}$'s PPD is given by Eqn. (6).*

$$w_j^{(h)} = \left(w_{j-1}^{(h)} + X^{(h)} - \delta_{j-1,j}^{(h-1)} \right)^+ + q_{j-1,j}^{(h)}, \quad (5)$$

where $(x)^+ = \max\{0, x\}$, and $q_{j-1,j}^{(h)}$ is p_j's type-I queueing delay at $H^{(h)}$.

$$\delta_{j-1,j}^{(h)} = \begin{cases} X^{(h)} + q_{j-1,j}^{(h)}, & \text{if } w_{j-1}^{(h)} + X^{(h)} \geq \delta_{j-1,j}^{(h-1)}, \\ \delta_{j-1,j}^{(h-1)} - w_{j-1}^{(h)} + q_{j-1,j}^{(h)}, & \text{otherwise.} \end{cases} \quad (6)$$

PROOF. It is straightforward to obtain $w_j^{(h)}$ and $\delta_{j-1,j}^{(h)}$ directly from Figures 3(a)-3(b). Alternatively, $w_j^{(h)}$ can be derived from the Lindley's recurrence equation [13]. □



(a) $w_{j-1}^{(h)} + X^{(h)} < \delta_{j-1,j}^{(h-1)}$. (b) $w_{j-1}^{(h)} + X^{(h)} \geq \delta_{j-1,j}^{(h-1)}$.

Figure 3: Two scenarios for deriving the queueing delay and packet-pair dispersion at $H^{(h)}$.

We next consider the following three lemmas which will be used to prove the main results for p_j's minDelay in Prop. 3. Lemma 1 addresses the effect of type-I cross traffic on p_j's delay, whereas Lemmas 2-3 address that of type-H cross traffic. We let $w_{j,H}^{(h)}$ be the type-H queueing delay experienced by p_j at $H^{(h)}$.

LEMMA 1. *For $d_j^{(n)}$ to be free from type-I queueing delay, $q_{j-1,j}^{(h)} = 0$ for $h = 1, \ldots, n$.*

PROOF. It is clear from Eqn. (5) that $w_j^{(h)}$ does not include type-I queueing delay iff $q_{j-1,j}^{(h)} = 0$. Therefore, type-I cross traffic does not contribute to $d_j^{(n)}$ iff $q_{j-1,j}^{(h)} = 0,\ \forall h$. □

For the next two lemmas, Lemma 1 is assumed true, and we consider two types of hops: $H^{(h)}$ is a *local bottleneck hop* (LBH) if $X^{(h)} \geq \delta_{j-1,j}^{(h-1)}$ and a non-LBH, otherwise. In the absence of type-I cross traffic, p_{j-1} and p_j will be sent back to back on an LBH, but the two packets may be sent with a time gap on a non-LBH.

LEMMA 2. *Considering that Lemma 1 holds and $H^{(h)}$ is a non-LBH, p_j does not experience type-H queueing delay at $H^{(h)}$ iff $w_{j-1}^{(h)} \leq \delta_{j-1,j}^{(h-1)} - X^{(h)}$.*

PROOF. Note that for a non-LBH,

$$w_{j,H}^{(h)} = \left(w_{j-1}^{(h)} + X^{(h)} - \delta_{j-1,j}^{(h-1)}\right)^+. \quad (7)$$

Eqn. (7) shows that $w_{j-1}^{(h)} \leq \delta_{j-1,j}^{(h-1)} - X^{(h)}$ is the only condition for $w_{j,H}^{(h)} = 0$. □

LEMMA 3. *Consider that Lemma 1 holds and $H^{(h)}$ is an LBH.*

(i) For $h = 1$, p_j does not experience type-H queueing delay at $H^{(1)}$ iff $w_{j-1}^{(1)} = 0$.

(ii) For $h > 1$, given that $H^{(h)}$ is preceded immediately by s ($0 \leq s \leq h-2$) adjoining non-LBHs and then an LBH, p_j does not experience type-H queueing delay at $H^{(h)}$ iff $w_{j-1}^{(h-k)} = 0$ for $k = 0, \ldots, s$.

PROOF. For case (i), note that $H^{(1)}$ is an LBH, because $X^{(1)} > \delta_{j-1,j}^{(0)} \equiv \tau_{j-1,j}^{(1)} = 0$. From Eqn. (5), $w_j^{(1)} = w_{j-1}^{(1)} + X^{(1)}$; therefore, it is required that $w_{j,H}^{(1)} \equiv w_{j-1}^{(1)} = 0$.

For case (ii), we first consider the case of $s = 0$ (i.e., $H^{(h-1)}$ is an LBH). From Eqn. (6), $\delta_{j-1,j}^{(h-1)} = X^{(h-1)}$, and from Eqn. (5),

$$\begin{aligned} w_j^{(h)} &= w_{j-1}^{(h)} + X^{(h)} - \delta_{j-1,j}^{(h-1)}, \\ &= w_{j-1}^{(h)} + X^{(h)} - X^{(h-1)} \geq 0. \end{aligned}$$

Therefore, it is required that $w_{j,H}^{(h)} \equiv w_{j-1}^{(h)} = 0$.

For case (ii) with $s > 0$, note that $h > 2$. Since $H^{(h-1)}$ to $H^{(h-s)}$ are non-LBHs and the second packet's delay at these hops satisfy Lemma 2, from Eqn. (6),

$$\delta_{j-1,j}^{(h-k)} = \delta_{j-1,j}^{(h-k-1)} - w_{j-1}^{(h-k)} \ \textit{for}\ k = 1, \ldots, s. \quad (8)$$

By repeatedly substituting Eqn. (8) into $w_j^{(h)}$,

$$\begin{aligned} w_j^{(h)} &= w_{j-1}^{(h)} + X^{(h)} - \delta_{j-1,j}^{(h-1)}, \\ &= \sum_{k=0}^{s} w_{j-1}^{(h-k)} + X^{(h)} - X^{(h-s-1)} \geq 0. \quad (9) \end{aligned}$$

From Eqn. (9), $w_{j,H}^{(h)} \equiv \sum_{k=0}^{s} w_{j-1}^{(h-k)}$. Therefore, $w_{j-1}^{(h-k)} = 0$ for $k = 0, \ldots, s$ yields $w_{j,H}^{(h)} = 0$. □

PROPOSITION 3. *(The second packet's minDelay) The necessary and sufficient conditions for $d_j^{(n)}$ being a minDelay are:*

$$\begin{aligned} &(i) \quad q_{j-1,j}^{(h)} = 0,\ h = 1, \ldots, n,\ \textit{and} \\ &(ii) \quad w_{j-1}^{(h)} = 0,\ h = 1, \ldots, h_b,\ \textit{and} \\ &(iii) \quad w_{j-1}^{(h)} \leq \delta_{j-1,j}^{(h-1)} - X^{(h)},\ h = h_b + 1, \ldots, n. \end{aligned}$$

When $h_b = n$, condition (iii) is not needed.

PROOF. Condition (i) is required because of Lemma 1.

For conditions (ii) and (iii), we first consider $H^{(h_b)}$. With $q_{j-1,j}^{(h)} = 0,\ \forall h$, it is not difficult to see that $X^{(h_b)} \geq \delta_{j-1,j}^{(h_b-1)}$, because $\delta_{j-1,j}^{(h_b-1)} \leq \max_{\forall h}\{X^{(h)}\}$. Therefore, $H^{(h_b)}$ is an LBH. According to Lemma 3, if all the hops between $H^{(h_b)}$, where $h_b > 1$, and $H^{(1)}$ are non-LBHs, condition (ii) must hold. Even if there are one or more LBHs between them, condition (ii) still holds, because the same argument can be applied to each segment of adjoining non-LBHs.

For condition (iii), all the hops after $H^{(h_b)}$, if any, must be non-LBHs for $d_j^{(n)}$ being a minDelay. To see why, assume that $H^{(h_a)}$, where $h_b < h_a \leq n$, is an LBH. Moreover, by setting $h_a - h_b = s + 1$ ($s \geq 0$), we could apply Lemma 3 and Eqn. (9) to $w_j^{(h_a)}$:

$$\begin{aligned} w_j^{(h_a)} &= \sum_{k=0}^{s} w_{j-1}^{(h_a-k)} + X^{(h_a)} - X^{(h_b)}, \\ &= X^{(h_a)} - X^{(h_b)}. \end{aligned}$$

Since $X^{(h_a)} < X^{(h_b)}$, $w_j^{(h_a)} < 0$ which contradicts that $w_j^{(h_a)} \geq 0$ for $H^{(h_a)}$ being an LBH. Therefore, $H^{(h_a)}$ must be a non-LBH. By applying Lemma 2 to each hop after $H^{(h_b)}$, we obtain condition (iii). □

4.3 Minimum delay difference

We propose a new cross-traffic filtering method called *minimum delay difference* (MDDIF) which exploits the minDelay of the packet pairs for capacity estimation. Prop. 4 shows that the unbiased PPD in Eqn. (3) can be obtained by the difference between a second packet's minDelay and a first packet's minDelay, and the two packets do not belong to the same packet pair.

PROPOSITION 4. *A sequence of packet pairs $\{p_{2i-1}, p_{2i}\}$, $i = 1, 2, \ldots$, is dispatched by a local endpoint to measure the capacity of an n-hop path ($n \geq 2$). Moreover, $d_{2k-1}^{(n)}$ (for the first packet in P_k) and $d_{2l}^{(n)}$ (for the second packet in P_l) are minDelays, where $l \neq k$. Then,*

$$d_{2l}^{(n)} - d_{2k-1}^{(n)} = \max\left\{\frac{S_f}{C_f^{(n)}}, \frac{S_r}{C_r^{(n)}}\right\}. \quad (10)$$

PROOF. First of all, from Eqn. (1),

$$d_{2l}^{(n)} - d_{2k-1}^{(n)} = w_{2l}^{(n)} - w_{2k-1}^{(n)} + d_{2l}^{(n-1)} - d_{2k-1}^{(n-1)}. \quad (11)$$

Using $w_{2k-1}^{(n)} = 0$ (from Prop. 1), Eqn. (5) for $w_{2l}^{(n)}$, and $q_{2l-1,2l}^{(n)} = 0$ (from Prop. 3(i)), Eqn. (11) becomes

$$\begin{aligned} d_{2l}^{(n)} - d_{2k-1}^{(n)} &= d_{2l}^{(n-1)} - d_{2k-1}^{(n-1)} + \\ &\quad \left(w_{2l-1}^{(n)} + X^{(n)} - \delta_{2l-1,2l}^{(n-1)}\right)^+. \quad (12) \end{aligned}$$

We now use mathematical induction on n for the proof.
The base case: $n = 2$ (i.e., $h_f = 1$ and $h_r = 2$) By applying Eqn. (12) recursively for $n = 2$, we obtain

$$d_{2l}^{(2)} - d_{2k-1}^{(2)} = \sum_{h=1}^{2} \left(w_{2l-1}^{(h)} + X^{(h)} - \delta_{2l-1,2l}^{(h-1)}\right)^+. \quad (13)$$

Note that $\delta_{2l-1,2l}^{(0)} \equiv \tau_{2l-1,2l}^{(1)} = 0$. Moreover, $H^{(h_b)}$ is either $H^{(1)}$ (the forward-path hop) or $H^{(2)}$ (the reverse-path hop):
Case 1 ($h_b = h_f = 1$): Since $d_{2l}^{(2)}$ is a minDelay, $w_{2l-1}^{(1)} = 0$ (from Prop. 3(ii)) and $\left(w_{2l-1}^{(2)} + X^{(2)} - \delta_{2l-1,2l}^{(1)}\right)^+ = 0$ (from Prop. 3(iii)). Eqn. (13) therefore becomes

$$d_{2l}^{(2)} - d_{2k-1}^{(2)} = X^{(1)} = S_f/C^{(1)},$$

which is the same as Eqn. (10) for $n = 2$.
Case 2 ($h_b = h_r = 2$): Since $d_{2l}^{(2)}$ is a minDelay, $w_{2l-1}^{(1)} = w_{2l-1}^{(2)} = 0$ (from Prop. 3(ii)) and $\delta_{2l-1,2l}^{(1)} = X^{(1)}$. Eqn. (13) therefore becomes

$$d_{2l}^{(2)} - d_{2k-1}^{(2)} = X^{(2)} = S_r/C^{(2)},$$

which is the same as Eqn. (10) for $n = 2$.
The inductive step: Assuming that Eqn. (10) holds for $n \geq 2$, we prove that Eqn. (10) also holds for $n+1$. By substituting Eqn. (10) (the inductive hypothesis for n) into Eqn. (12) for $n+1$, we have

$$\begin{aligned} d_{2l}^{(n+1)} - d_{2k-1}^{(n+1)} &= \left(w_{2l-1}^{(n+1)} + X^{(n+1)} - \delta_{2l-1,2l}^{(n)}\right)^+, \\ &\quad + \max\left\{S_f/C_f^{(n)}, S_r/C_r^{(n)}\right\}. \quad (14) \end{aligned}$$

There are two cases to consider: h_b remains the same, and $h_b = n+1$. Note that $H^{(n+1)}$ introduces a new link to the reverse path.
Case 1 ($h_b < n+1$): Since $d_{2l}^{(n+1)}$ is a minDelay, applying $w_{2l-1}^{(n+1)} \leq \delta_{2l-1,2l}^{(n)} - X^{(n+1)}$ (from Prop. 3(iii)) to Eqn. (14) yields

$$\begin{aligned} d_{2l}^{(n+1)} - d_{2k-1}^{(n+1)} &= \max\left\{S_f/C_f^{(n)}, S_r/C_r^{(n)}\right\}, \\ &= \max\left\{S_f/C_f^{(n+1)}, S_r/C_r^{(n+1)}\right\}. \quad (15) \end{aligned}$$

Case 2 ($h_b = n+1$): Since $d_{2l}^{(n+1)}$ is a minDelay, we have $w_{2l-1}^{(h)} = 0,\ \forall 1 \leq h \leq n+1$ (from Prop. 3(ii)). Accordingly, both $d_{2l}^{(n)}$ and $d_{2l-1}^{(n)}$ are also minDelays; therefore, $d_{2l}^{(n)} - d_{2l-1}^{(n)} = \max\left\{S_f/C_f^{(n)}, S_r/C_r^{(n)}\right\}$ (the inductive hypothesis). Substituting $w_{2l-1}^{(n+1)} = 0$ and $\delta_{2l-1,2l}^{(n)} = d_{2l}^{(n)} - d_{2l-1}^{(n)}$ (from Eqn. (2)) into Eqn. (14) yields

$$d_{2l}^{(n+1)} - d_{2k-1}^{(n+1)} = X^{(n+1)} = S_r/C^{(n+1)},$$

which is the same as Eqn. (15). □

5. A FIRST-PASSAGE-TIME ANALYSIS OF THE MDDIF AND MDSUM METHODS

In this section, we analyze and compare the MDDIF and MDSUM methods based on their first passage times. The MDDIF method's *first passage time* (FPT) is defined as the first time (in terms of the number of packet pairs sent) to obtain the two minDelays. On the other hand, the MDSUM method's FPT is defined as the first time to obtain the minimum delay sum (which is equal to the sum of the two minDelays). Therefore, a smaller FPT results in a faster measurement. Moreover, we consider a multi-hop path scenario, instead of a single-hop model considered in [10].

5.1 The first passage time

Let $X_i, i \geq 1$, be a sequence of independent and identically distributed (i.i.d.) Bernoulli random variables with parameter p_X (probability for $X_i = 1$) for the minDelay event of p_{2i-1} (the first packet in P_i). $X_i = 1$ if $d_{2i-1}^{(n)}$ is a minDelay and $X_i = 0$, otherwise. Similarly, $Y_i, i \geq 1$, is a sequence of i.i.d. Bernoulli random variables with parameter p_Y for the minDelay event of p_{2i} (the second packet in P_i). $Y_i = 1$ if $d_{2i}^{(n)}$ is a minDelay and $Y_i = 0$, otherwise. Moreover, the sequence of the joint random variables (X_i, Y_i) are i.i.d. with a joint probability density function (pdf) $p_{XY}(x, y)$. Note that X_i and Y_i are generally not independent.

The MDDIF method's FPT is given by

$$T_{DIF} = \inf\{i : SX_i > 0 \ and \ SY_i > 0\}, \quad (16)$$

where $SX_i = \sum_{k=1}^{i} X_k$ and $SY_i = \sum_{k=1}^{i} Y_k$. To obtain the pdf for T_{DIF}, we consider another sequence of random variables $Z_i, i \geq 1$, for which

$$Z_i = \begin{cases} 0, & \text{if } SX_i = 0 \text{ and } SY_i = 0, \\ 1, & \text{if } SX_i = 0 \text{ and } SY_i > 0, \\ 2, & \text{if } SX_i > 0 \text{ and } SY_i = 0, \\ 3, & \text{if } SX_i > 0 \text{ and } SY_i > 0, \end{cases}$$

is a time-homogeneous Markov chain with transient states 0, 1, and 2, and an absorbing state 3. Denote the stationary transition probabilities by $p_{mn} = P[Z_{i+1} = n | Z_i = m],\ m, n = 0, 1, 2, 3$, and the transition prob-

ability matrix $\mathbf{P}$ of the Markov chain by

$$\begin{aligned}\mathbf{P} &= [p_{mn}],\\ &= \left[\begin{array}{c|c}\mathbf{Q} & \mathbf{A}\\ \hline 0 & 1\end{array}\right],\\ &= \left[\begin{array}{ccc|c} p_{XY}(0,0) & p_{XY}(0,1) & p_{XY}(1,0) & p_{XY}(1,1)\\ 0 & 1-p_X & 0 & p_X\\ 0 & 0 & 1-p_Y & p_Y\\ \hline 0 & 0 & 0 & 1\end{array}\right],\end{aligned}$$

where $\mathbf{Q}$ is for the transitions among the three transient states, whereas $\mathbf{A}$ is for the transitions from the transient states to the absorbing state. Since the MDDIF method starts from state 0, the initial probability vector for the first three (transient) states is given by $\pi_0 = [1\ 0\ 0]$. From [20],

$$P[T_{DIF} = i] = \pi_0 \mathbf{Q}^{i-1}\mathbf{A}. \tag{17}$$

To determine the expectation of the FPT, we obtain $\mathbf{t} = [t_0, t_1, t_2]^T$, for which $t_k,\ k = 0, 1, 2$, is the expected number of steps taken prior to reaching the absorbing state, given that the chain begins from state k. From [12], $\mathbf{t} = (\mathbf{I} - \mathbf{Q})^{-1}\mathbf{c}$, where $\mathbf{I}$ is an identity matrix, $\mathbf{c} = [1\ 1\ 1]^T$, and $(\mathbf{I} - \mathbf{Q})^{-1}$ is the fundamental matrix of the Markov chain. We therefore have

$$\begin{aligned}&E[T_{DIF}] = \pi_0\mathbf{t},\\ &= \left(\frac{1}{1-p_{XY}(0,0)}\right)\left(1 + \frac{p_{XY}(0,1)}{p_X} + \frac{p_{XY}(1,0)}{p_Y}\right).\end{aligned} \tag{18}$$

On the other hand, the MDSUM method's FPT is defined as

$$T_{SUM} = \inf\{i : X_i = 1 \textit{ and } Y_i = 1\}. \tag{19}$$

Therefore, T_{SUM} is a geometrically distributed random variable with parameter $p_{XY}(1,1)$.

Besides showing that $E[T_{DIF}] < E[T_{SUM}]$, Prop. 5 also states the main idea of the MDDIF method. The necessary and sufficient condition for the MDDIF method to obtain capacity estimates faster than the MDSUM method is when it is possible to find the two minDelays from different packet pairs (i.e., $p_{XY}(0,1) > 0$ and $p_{XY}(1,0) > 0$).

PROPOSITION 5. *$E[T_{DIF}] < E[T_{SUM}]$ iff $p_{XY}(0,1) > 0$ and $p_{XY}(1,0) > 0$.*

PROOF. By using Eqn. (18) and $E[T_{SUM}] = 1/p_{XY}(1,1)$, we compute the relative gain of $E[T_{DIF}]$ as

$$\Psi = \frac{E[T_{SUM}] - E[T_{DIF}]}{E[T_{SUM}]} = \frac{\sigma}{\sigma + \xi}, \tag{20}$$

where

$$\sigma = p_{XY}(0,1)p_{XY}(1,0)(p_X + p_Y), \tag{21}$$

$$\xi = p_{XY}(1,1)[p_{XY}(0,1)(p_X + p_Y) + p_X^2]. \tag{22}$$

Assume that $p_{XY}(0,1) > 0$ and $p_{XY}(1,0) > 0$. Since $p_X = p_{XY}(1,0) + p_{XY}(1,1)$, $p_{XY}(1,0) > 0$ implies $p_X > 0$. Similarly, $p_{XY}(0,1) > 0$ implies $p_Y > 0$. From Eqn. (21), $\sigma > 0$. Moreover, $0 \le p_{XY}(1,1) < 1$ due to the law of total probability, and it is easy to see that $0 < [p_{XY}(0,1)(p_X + p_Y) + p_X^2] < 1$. Therefore, $0 \le \xi < 1$ from Eqn. (22), and as a result, $0 < \Psi \le 1$.

In the other direction, assume that $\Psi > 0$. From Eqns. (20)-(21), $p_{XY}(0,1)p_{XY}(1,0)(p_X + p_Y) > 0$ which is equivalent to $p_{XY}(0,1) > 0$ and $p_{XY}(1,0) > 0$. □

Prop. 6 shows that the MDDIF method does not have the speed advantage for $h_b = n$. Since $H^{(n)}$ is a reverse-path hop, the MDDIF and MDSUM methods give the same expected FPTs for measuring $C_r^{(n)}$. Nevertheless, the MDDIF method's speed advantage may still be retained for measuring $C_f^{(n)}$ if S_f and S_r are selected such that $h_b = h_f$. This could be done for $C_r^{(n)} \ge C_f^{(n)}$ (e.g., by choosing $S_f = S_r$), and for $C_r^{(n)} < C_f^{(n)}$ if it is feasible to achieve $S_f/S_r > C_f^{(n)}/C_r^{(n)} > 1$ (see the discussion for Eqn. (3)).

PROPOSITION 6. *$E[T_{DIF}] = E[T_{SUM}]$ for $h_b = n$.*

PROOF. The event of $X_i = 0$ and $Y_i = 1$ is not possible (i.e., $p_{XY}(0,1) = 0$) for this scenario. Since p_{2i-1} is not a minDelay, $w_{2i-1}^{(h)} \neq 0$ for some h (from Prop. 1). Thus, it is not possible for p_{2i} being a minDelay, because $w_{2i-1}^{(h)} = 0$ is required for all h (according to Prop. 3). As a result, $\sigma = 0$ in Eqn. (21); thus, $\Psi = 0$. □

5.2 A first-passage-time analysis for $h_b = n - 1$

To quantify the MDDIF method's speed advantage for $h_b \neq n$, we analyze the case of $h_b = n - 1$ here. We model the node in $H^{(h)}$ as a single FIFO queue with unlimited buffer. The packet inter-arrival times for the cross traffic to the queue at $H^{(h)}$ (denoted by $A^{(h)}$) are exponentially distributed with rate $\lambda^{(h)}$. This assumption is based on the previous study that the cross traffic distribution is reasonably represented by the Poisson process on sub-second timescales [11]. The inter-arrival process for the packet pairs to the queue is also exponential; therefore, they take a random look at the state of the queue. Since the packet pairs do not generate a significant load to $H^{(n)}$, the average packet arrival rate $\lambda^{(h)}$ is retained. The packet service time at $H^{(h)}$ (denoted by $B^{(h)}$) is a random variable which depends on the packet size (denoted by $S^{(h)}$) distribution and $C^{(h)}$. To make the analysis simple, we assume that $B^{(h)}$ is an exponential random variable with $\mu^{(h)} = 1/E[B^{(h)}] = C^{(h)}/E[S^{(h)}]$ being the packet service rate at $H^{(h)}$. As a result, each node is modeled as a classic M/M/1 queue.

5.2.1 Computing the probabilities

In this section we derive analytical expressions for the probabilities in Eqn. (18) for the MDDIF method and $p_{XY}(1,1)$ for the MDSUM method. We also note that it is sufficient to obtain expressions for $p_{XY}(1,1)$, p_X, and p_Y, because they can be used to obtain other probabilities.

Computing p_X We again consider $\{p_{j-1}, p_j\}$. Since p_j will not affect p_{j-1}, p_X is the probability that all nodes on the path are empty upon p_{j-1}'s arrival. The empty probability is given by $1-\rho^{(h)}$ for $H^{(h)}$ [20]. By applying an independence assumption for the nodes,

$$p_X = \prod_{h=1}^{n} \left(1-\rho^{(h)}\right), \tag{23}$$

where $\rho^{(h)} = \lambda^{(h)}/\mu^{(h)}$.

Computing $p_{XY}(1,1)$ Same as the last case, p_{j-1} arrives at an empty node in $H^{(h)}$ with probability $1-\rho^{(h)}$. Given that a period of t has been passed since the last cross-traffic packet arrival upon p_{j-1}'s arrival at $H^{(h)}$, the probability that p_j will not be delayed by the intervening cross traffic between p_{j-1} and p_j is given by $P[A^{(h)} > t + \delta_{j-1,j}^{(h-1)} | A^{(h)} > t]$. By the memoryless property of an exponential distribution, this conditional probability is given by $P[A^{(h)} > \delta_{j-1,j}^{(h-1)}] = e^{-\lambda^{(h)}\delta_{j-1,j}^{(h-1)}}$. Hence,

$$p_{XY}(1,1) = \prod_{h=1}^{n} \left(1-\rho^{(h)}\right) e^{-\lambda^{(h)}\delta_{j-1,j}^{(h-1)}}. \tag{24}$$

Computing p_Y Since $h_b = n-1$, we consider two subpaths for the analysis: (i) $\{H^{(1)}, \ldots, H^{(n-1)}\}$ and (ii) $\{H^{(n)}\}$. Let p'_Y be the probability that p_j's delay on subpath (i) is a minDelay and p''_Y the probability that p_j's delay on subpath (ii) is a minDelay. Therefore, $p_Y = p'_Y p''_Y$.

For subpath (i), due to Prop. 3(i)-(ii), both p_{j-1} and p_j do not experience queueing delay on the subpath. Therefore, p'_Y is the same as Eqn. (24) except for the last hop.

The subpath (ii) consists of $H^{(n)}$ which is after $H^{(h_b)}$. Therefore, according to Prop. 3(iii), p_{j-1} must not be delayed by more than $\omega = \delta_{j-1,j}^{(n-1)} - X^{(n)} > 0$. Let $W^{(n)}$ be the random variable for p_{j-1}'s queueing delay at $H^{(n)}$. Based on the Pollaczek-Khinchin equation for an $M/M/1$ queue [20],

$$p_{W^{(n)}}(t) = \left(1-\rho^{(n)}\right)\left(\delta_0(t) + \lambda^{(n)} e^{-\mu^{(n)}(1-\rho^{(n)})t}\right), \tag{25}$$

where $\delta_0(t)$ is the Dirac delta function. Moreover, p_j does not encounter intervening cross traffic at $H^{(n)}$ (from Prop. 3(i)). The probability for this event, conditioned on the event that p_{j-1} has encountered a queueing delay of t, is given by the probability of the event $A^{(n)} > \delta_{j-1,j}^{(n-1)} - t$. Therefore, we can obtain p''_Y and p_Y:

$$\begin{aligned} p''_Y &= \int_0^{\omega} P[W^{(n)} = t, A^{(n)} > \delta_{j-1,j}^{(n-1)} - t]dt, \\ &= (1-\rho^{(n)}) e^{-\lambda^{(n)}\delta_{j-1,j}^{(n-1)}} \left[1 + \frac{\rho^{(n)}}{1-2\rho^{(n)}} \right. \\ &\quad \left. \times \left(1 - e^{-\mu^{(n)}(1-2\rho^{(n)})\omega}\right)\right], \end{aligned} \tag{26}$$

$$\begin{aligned} p_Y &= p'_Y p''_Y, \\ &= p_{XY}(1,1)\left[1 + \frac{\rho^{(n)}}{1-2\rho^{(n)}}\left(1 - e^{-\mu^{(n)}(1-2\rho^{(n)})\omega}\right)\right]. \end{aligned} \tag{27}$$

5.2.2 Analytical results

Using the analytical results from the last section for $h_b = n-1$, Figure 4 reports Ψ for $n=5$ with link capacities of $\{100, 75, 55, 40, 80\}$ Mbits/s and $S_f = \{240, 576, 1500\}$ bytes. Each sub-figure plots Ψ against a mean utilization ρ ($\rho^{(h)} = \rho$, $\forall h$) with a given mean (cross-traffic) packet size S_c ($E[S^{(h)}] = S_c$, $\forall h$). The results are in agreement with Prop. 5.

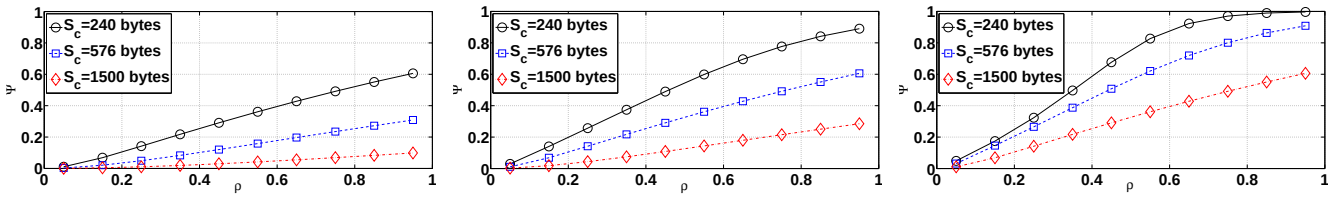
Figure 4 shows that the benefit of the MDDIF method increases with S_f and ρ, but decreases with S_c. As ρ increases, p_X, $p_{XY}(1,1)$, p_Y, $p_{XY}(0,1)$ all decrease (Eqns. (23) (24), (27), and (28)). That is, it is harder for both MDDIF and MDSUM methods to find valid samples as the intensity of cross traffic increases. However, the impact on the MDSUM method is much more serious, because it is required to obtain the minDelay for both packets from the same packet pair.

As for the impact of S_f and S_c, Figure 5(a) shows the distribution of $p_{XY}(1,1)$ with $S_c = [240, 1500]$ bytes, $S_f = S_r = [240, 1500]$ bytes, and $\rho = 20\%$. Notice that $p_{XY}(1,1)$ drops drastically as S_f increases and S_c decreases, causing the MDSUM method a longer time to find a valid capacity sample. This is because the PPD ($\delta_{j-1,j}^{(h_b)}$) increases with S_f and the probability for the cross-traffic packets intervening between the two packets increases for a small S_c. Although S_f and S_c also affect the MDDIF method in a similar fashion, the impact is less severe, because it can obtain the two minDelays from different packet pairs.

There is also a subtle relationship between S_f and S_c concerning $p_{XY}(0,1)$, which is illustrated in Figure 5(b). By inspecting Eqn. (27),

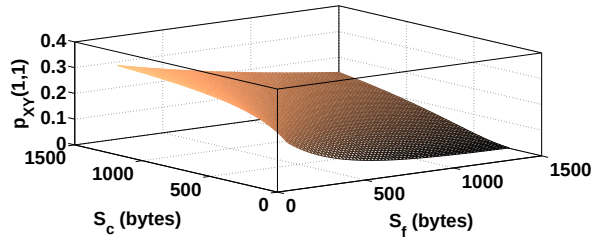
$$p_{XY}(0,1) = \frac{p_{XY}(1,1)\rho^{(n)}}{1-2\rho^{(n)}}\left(1 - e^{-\mu^{(n)}(1-2\rho^{(n)})\omega}\right). \tag{28}$$

Clearly, the likelihood of fulfilling Prop. 3(iii) increases with S_f, because the probability of p_j's queueing due to p_{j-1}'s decreases. Increasing S_f, however, can decrease the probability of fulfilling Prop. 3(i), because the increased dispersion can accommodate more cross-

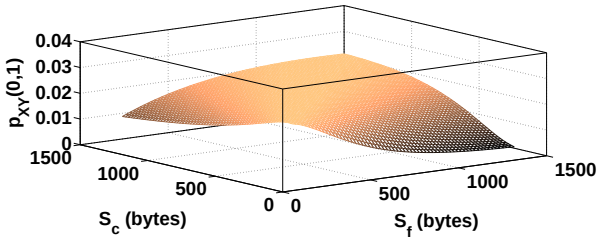


(a) $S_f = 240$ bytes. (b) $S_f = 576$ bytes. (c) $S_f = 1500$ bytes.

Figure 4: The relative gain of the expected first passage times for the MDDIF and MDSUM methods.



(a) $p_{XY}(1,1)$.



(b) $p_{XY}(0,1)$.

Figure 5: The values of $p_{XY}(1,1)$ and $p_{XY}(0,1)$ for $\rho = 20\%$, and different probe and cross-traffic packet sizes.

traffic packet arrivals between the two packets [5, 10]. Therefore, Figure 5(b) shows that $p_{XY}(0,1)$ peaks near $S_f = S_c$ but drops for $S_c > S_f$ and $S_f > S_c$.

Besides the speed advantage, the MDDIF method is also simpler than the MDSUM method. According to Eqn. (4), the MDSUM method is required to keep track of the minimum delay sum and the minDelay for the first and second packets, and performs a validation test for each measurement. Clearly, the MDSUM also needs to store the packet-pair dispersion sample responsible for the minimum delay sum. The MDDIF method, on the other hand, only needs to store two minDelays.

6. MEASUREMENT RESULTS

We have incorporated the MDDIF method into OneProbe [17] to measure forward-path, reverse-path, and round-trip capacity. OneProbe is a non-cooperative measurement tool using a probe of two back-to-back packets to measure multiple path-quality metrics. To measure the reverse-path capacity, OneProbe dispatches a specially crafted probe packet to elicit two back-to-back response TCP data packets (which deviates from assumption (3)). To measure the forward-path capacity, OneProbe dispatches a pair of probe TCP data packets, and each probe TCP data packet elicits a response TCP data packet. The MDDIF method is based on the RTT samples that measure the time between sending a probe packet and receiving the elicited response packet.

In the following, we present three sets of capacity measurement results using the OneProbe implementation. For the first set, we used a controlled testbed environment to evaluate the impact of cross traffic on the measurement accuracy and the FPTs for both the MDDIF and MDSUM methods. For the second and third sets, we used ADSL links (real and emulated) as the bottleneck links.

6.1 Testbed evaluation of the MDDIF and MDSUM methods

The testbed, shown in Figure 6, was configured with a 16-hop round-trip path ($n = 16$), consisting of a probe sender, a web server running Apache v2.2.3 as the remote node, four cross-traffic clients $X_1 - X_4$, and seven forwarding devices—three Linux 2.6.26 routers $R1 - R3$ and four store-and-forward Ethernet switches $S1 - S4$. $S1$ is a Gigabit switch, $S4$ is a 10 Mbits/s switch, and the others are 100 Mbits/s switches. Since we used $S_f = S_r$, $h_b = 10$ and $C_b^{(n)} = 10$ Mbits/s. We ran TC/Netem [8] in each router to emulate a fixed RTT of 300 milliseconds between the probe sender and web server. We found that the delay emulated by TC/Netem in each router was stable and similar to the results reported in [21].

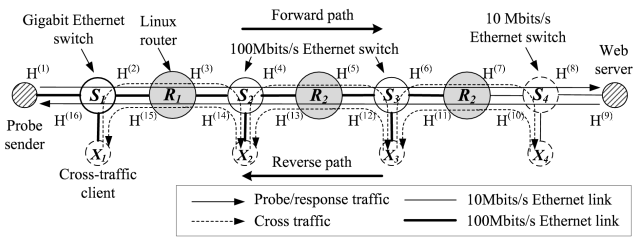


Figure 6: The testbed topology.

Each cross-traffic client generated forward-path (reverse-path) cross traffic to another cross-traffic client to the right (left) to emulate a loading rate of ρ on the corresponding path segment. Similar to [5], the cross-traffic packets had uniformly distributed sizes in the $[40, 1500]$ bytes range and Pareto inter-arrivals with a shape parameter $\alpha = 1.9$. We ran OneProbe from the probe sender to dispatch a sequence of L probes according to a Poisson distribution with a mean rate of 2Hz. The probe sender was equipped with a DAG 4.5 passive network monitoring card [1] to obtain the PPD and RTTs in microsecond resolution which was limited by the `pcap` header structure [2].

We conducted three sets of experiments with different cross-traffic loadings using the MDDIF and MDSUM methods with $S_f = S_r = 240$ bytes, and the results were plotted in Figures 7(a)-7(c). For each set, we conducted experiments for different values of L, ranging from 1 to 120. Moreover, for each L value, we repeated the experiments 50 times to obtain the mean and confidence intervals for $\hat{C}_b^{(n)}$ (an estimate of $C_b^{(n)}$) and Ψ. When T_{SUM} was undefined after sending L packet pairs, we let $T_{SUM} = L$ and computed $\hat{C}_b^{(n)}$ using the PPD of the packet pair with the smallest RTT sum.

For low cross-traffic loads, Figure 7(a) shows that the capacity estimates obtained by the two methods coincide for $L \geq 10$, and both were very accurate (MDDIF: 9.54 Mbits/s, MDSUM: 9.58 Mbits/s). Moreover, the figure shows that the MDDIF method has a clear speed advantage with $\Psi \approx 25\%$.

For higher cross-traffic load, Figure 7(b) shows that the measurement accuracy deteriorated for both methods. However, the impacts of the cross traffic on both methods were very different. For $L = 120$, the MDDIF method obtained 10.78 Mbits/s (7.8% error), whereas the MDSUM method 12.94 Mbits/s (29.4% error). Similar to the low cross-traffic case, the MDDIF method enjoyed a relative gain of about 25% on the measurement speed.

In the third set of experiments, we emulated a typical high-load downlink condition (e.g., downloading from a server) by deploying asymmetric cross-traffic loads of $\rho = 0.5$ for $X_3 \rightarrow X_2$ and $X_2 \rightarrow X_1$, and $\rho = 0.1$ for others. Figure 7(c) shows that both methods were sufficiently accurate for $L = 120$: the MDDIF method obtained 9.42 Mbits/s (5.8% error) and the MDSUM method 9.56 Mbits/s (4.4% error). Although the MDSUM method is slightly more accurate, its capacity estimates saw a higher variation when L is not large enough. Similar to the last two cases, the MDDIF method had a clear speed advantage, and the relative gain was also higher than that for the first two cases.

6.2 Measuring remote ADSL links

We deployed OneProbe to conduct both forward-path and reverse-path capacity measurement from a local measuring node connected to an 1 Gbit/s Ethernet link. A sequence of probes with a fixed sampling interval of 500 milliseconds was dispatched to a remote ADSL endpoint with a downlink speed of 8 Mbits/s and an uplink speed of 800 Kbits/s. Therefore, the forward path (from the measuring node to the ADSL node) contained the ADSL's downlink, whereas the reverse path (from the ADSL node to the measuring node) contained the ADSL's uplink. The ADSL downlink and uplink were also the bottleneck links on the forward path and reverse path, respectively. Both nodes were located in Hong Kong, and the forward path consisted of 11 hops.

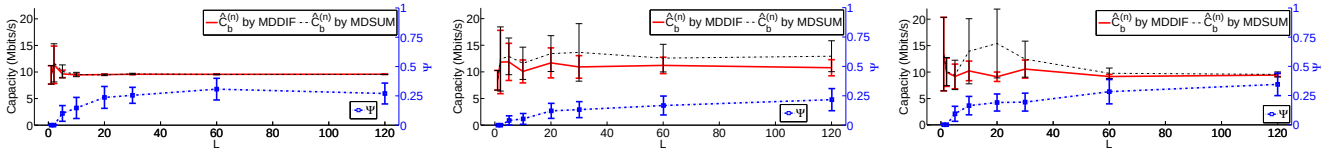
Same as the last section, the measuring node was equipped with a DAG 4.5 card to measure the RTTs and PPDs. The RTT (PPD) measurement was used for capacity estimation based on the MDDIF (MDSUM) method. Both the ADSL links and the cross traffic on the path could introduce interference to the RTT and PPD measurement which were obtained at the same time. We used $S_f = 1440$ bytes and $S_r = 90$ bytes (all packet sizes include the IP headers) for the forward-path measurement. According to Eqn. (3), this packet size setting ensures that the largest dispersion was introduced by the ADSL downlink. We used $S_f = S_r = 1440$ bytes for the reverse-path measurement.

Figures 8(a) and 8(c) report the PPDs for the forward-path and reverse-path capacity measurement, respectively. The ranges of the PPD measurement are $[0.01, 5.7]$ milliseconds for the forward path and $[14.7, 20.3]$ milliseconds for the reverse path. The corresponding ranges of the forward-path and reverse-path capacity estimates (denoted by $\hat{C}_f^{(n)}$ and $\hat{C}_r^{(n)}$) are $[2.215, 1272]$ Mbits/s and $[0.627, 0.865]$ Mbits/s, respectively. We also applied the approach in [4] to account for the layer-two overhead. Since each 1440-byte probe packet was carried by 30 ATM cells, each of which had 53 bytes, we scaled up the capacity estimates by a factor of 1.1 (1590/1440). By using the MDSUM method, we obtained $\hat{C}_f^{(n)} = 6.537$ Mbits/s after processing 140 packet pairs (for which the MDSUM conditions were fulfilled[1]) and $\hat{C}_r^{(n)} = 0.750$ Mbits/s after processing 63 packet pairs.

Figures 8(b) and 8(d), on the other hand, report the first and second probe packets' RTTs for the MDDIF method. The MDDIF method obtained fairly accurate results: $\hat{C}_f^{(n)} = 8.01$ Mbits/s and $\hat{C}_r^{(n)} = 0.776$ MBits/s after processing 106 and 22 packet pairs, respectively.

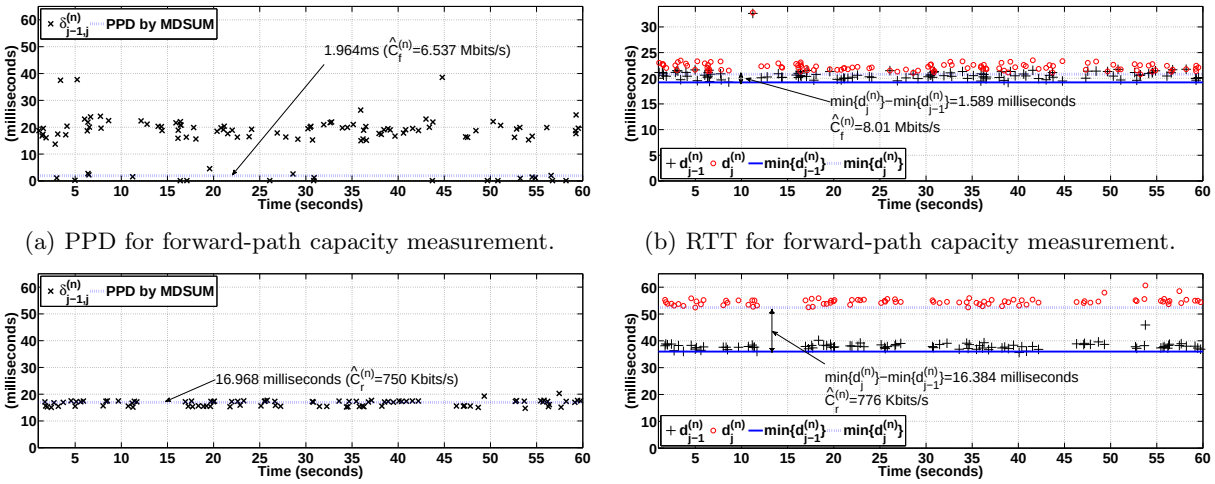
The above shows that the MDDIF method can resolve the PPD variability problem observed in an ADSL environment [4]. In particular, it was reported that the

[1]The MDSUM conditions are considered fulfilled when the difference between the left and right hand sides of Eqn. (4) is less than 1%. [10].



(a) $\rho = 0.1$ for all path segments. (b) $\rho = 0.5$ for all path segments. (c) $\rho = 0.5$ for $X_3 \rightarrow X_2$ and $X_2 \rightarrow X_1$; $\rho = 0.1$ for the rest.

Figure 7: Round-trip capacity estimates for the MDDIF and MDSUM methods and the relative gain using $S_f = S_r = 240$ bytes.



(a) PPD for forward-path capacity measurement. (b) RTT for forward-path capacity measurement.

(c) PPD for reverse-path capacity measurement. (d) RTT for reverse-path capacity measurement.

Figure 8: Time series of the PPDs and RTTs for the ADSL-at-the-remote-node experiments.

inter-arrival time of adjacent packets under an ADSL environment can vary significantly even in the absence of cross traffic, and such variation could render the PPD techniques ineffective. The MDDIF method, however, does not suffer from this problem, because it neither obtains the PPD directly from adjacent packets nor requires achieving the minimal possible delays from the same packet pair.

6.3 Measuring local ADSL links

We conducted another set of capacity measurement experiments by "setting" the forward-path and reverse-path bottleneck links to the local endpoint's links. We achieved this by emulating two types of asymmetric link capacity—ADSL2 (upstream: 1 Mbits/s, downstream: 18 Mbits/s) and ADSL (upstream: 0.8 Mbits/s, downstream: 8 Mbits/s)—using a Click v1.6 router. We deployed OneProbe in three separate machines on our campus to measure the capacity of the paths to three PlanetLab nodes: KAIST (in Korea), UMASS (in US), and UNIBO (in Italy). Each machine targeted one of the PlanetLab nodes. Based on our knowledge, the forward-path and reverse-path capacity were limited by the emulated ADSL2/ADSL links.

Besides OneProbe, we also attempted to deploy AsymProbe [14], a cooperative measurement tool that implements the MDSUM method, for comparison purposes. Since our campus network blocked all incoming UDP packets used by AsymProbe, we implemented AsymProbe using OneProbe's two-packet probe and refer this implementation to as AProbe. OneProbe used $S_f = S_r = 1500$ bytes for reverse-path measurement and $S_f/S_r = 1500$ bytes/260 bytes for forward-path measurement, whereas AProbe used maximum/minimum packet sizes of 1500 bytes/260 bytes. Each machine conducted the OneProbe and AProbe measurement for every 15 minutes. Each tool obtained a capacity estimate by processing at most 200 packet pair samples with a fixed probing rate of 2Hz.

Table 1 shows the median capacity estimated by OneProbe and AProbe based on 24-hour measurement. Measurement results presented in the first two rows show that the capacity measurement obtained by OneProbe using the MDDIF method was very accurate. In particular, OneProbe could obtain accurate reverse-path estimates even when bottleneck link was in the last hop of the path. On the other hand, AProbe's reverse-path measurement was not accurate, because it could ob-

tain only the forward-path dispersion and therefore the estimates represent the lower bounds for the reverse-path capacity. Nonetheless, AProbe still obtained lower bound values for the two ADSL cases: $1500/260 \times 0.8 = 4.615$ Mbits/s and $1500/260 \times 1 = 5.769$ Mbits/s.

We repeated the experiments with a symmetric network link of 10 Mbits/s which, according to the reasons stated earlier, should be the bottleneck capacity. All other settings were unchanged. As shown in the third row of Table 1, OneProbe's and AProbe's results were close to 10 Mbits/s. We did not try a higher bandwidth, because we were no longer able to ensure that the bottleneck link was still located in our campus network.

Table 1: Median capacity (in Mbits/s) measured by OneProbe and AProbe.

Link Type	Tools	KAIST $\hat{C}_f^{(n)}$	KAIST $\hat{C}_r^{(n)}$	UMASS $\hat{C}_f^{(n)}$	UMASS $\hat{C}_r^{(n)}$	UNIBO $\hat{C}_f^{(n)}$	UNIBO $\hat{C}_r^{(n)}$
ADSL (Up = 0.8, Down = 8)	OneProbe	0.799	7.921	0.771	7.926	0.798	7.900
	AProbe	0.786	4.392	0.758	4.544	0.758	4.310
ADSL2 (Up = 1, Down = 18)	OneProbe	1.018	17.817	0.962	17.870	0.991	17.804
	AProbe	0.988	5.472	0.989	5.262	1.025	5.300
10 Mbits/s Ethernet Link	OneProbe	10.025	9.748	10.568	9.748	10.353	9.744
	AProbe	10.592	9.740	9.423	9.748	9.630	9.748

7. CONCLUSIONS

This paper introduced the minimum delay difference (MDDIF) method, a new cross-traffic filtering approach for capacity measurement. Unlike the existing packet-pair dispersion methods, the MDDIF method obtains the packet-pair dispersion from the minimal possible delay (minDelay) for a first probe packet and a second probe packet both of which generally belong to different packet pairs. We have proved that a difference of these two minDelays gives the packet-pair dispersion required for capacity estimation and that the MDDIF method is faster than the minimum delay sum (MDSUM) method. We also conducted testbed and Internet measurement experiments to compare the MDDIF and MDSUM methods.

Acknowledgments

We thank the four anonymous reviewers for their critical reviews and suggestions and Paolo Giaccone, in particular, for shepherding our paper. This work is partially supported by a grant (ref. no. ITS/152/08) from the Innovation Technology Fund and a grant (ref. no. H-ZL17) from the Joint Universities Computer Centre, both in Hong Kong.

8. REFERENCES

[1] endace. http://www.endace.com/.
[2] TCPDUMP/LIBPCAP public repository. http://www.tcpdump.org/.
[3] J. Bolot. End-to-end packet delay and loss behavior in the Internet. In *Proc. ACM SIGCOMM*, 1993.
[4] D. Croce, T. En-Najjary, G. Urvoy-Keller, and E. Biersack. Capacity estimation of ADSL links. In *Proc. ACM CoNEXT*, 2008.
[5] C. Dovrolis, P. Ramanathan, and D. Moore. Packet dispersion techniques and a capacity-estimation methodology. *IEEE/ACM Trans. Netw.*, 12(6), 2004.
[6] A. Downey. Using pathchar to estimate Internet link characteristics. In *Proc. ACM SIGCOMM*, 1999.
[7] K. Harfoush, A. Bestavros, and J. Byers. Measuring capacity bandwidth of targeted path segments. *IEEE/ACM Trans. Netw.*, 17(1), 2009.
[8] S. Hemminger. Network emulation with NetEm. In *linux.conf.au*, 2005.
[9] V. Jacobson. Pathchar: A tool to infer characteristics of Internet paths. ftp://ftp.ee.lbl.ogv/pathchar/.
[10] R. Kapoor, L. Chen, L. Lao, M. Gerla, and M. Sanadidi. CapProbe: A simple and accurate capacity estimation technique. In *Proc. ACM SIGCOMM*, 2004.
[11] T. Karagiannis, M. Molle, M. Faloutsos, and A. Broido. A nonstationary Poisson view of Internet traffic. In *Proc. IEEE INFOCOM*, 2004.
[12] J. Kemeny and J. Snell. *Finite Markov Chains*. Springer, 1976.
[13] L. Kleinrock. *Queueing Systems, Vol. 2: Computer Applications*. Wiley-Interscience, 1976.
[14] L. Chen, T. Sun, G. Yang, M. Sanadidi, and M. Gerla. End-to-end asymmetric link capacity estimation. In *Proc. IFIP Networking*, 2005.
[15] K. Lai and M. Baker. Measuring bandwidth. In *Proc. IEEE INFOCOM*, 1999.
[16] K. Lai and M. Baker. Measuring link bandwidths using a deterministic model of packet delay. *Proc. ACM SIGCOMM*, 2000.
[17] X. Luo, E. Chan, and R. Chang. Design and implementation of TCP data probes for reliable and metric-rich network path monitoring. In *Proc. USENIX Annual Tech. Conf.*, 2009.
[18] B. Mah. pchar: A tool for measuring Internet path characteristics. http://www.kitchenlab.org/bmah/Software/pchar/.
[19] J. Mogul. Observing TCP dynamics in real networks. In *Proc. ACM SIGCOMM*, 1992.
[20] R. Nelson. *Probability, Stochastic Processes, and Queueing Theory: The Mathematics of Computer Performance Modelling*. Springer, 1995.
[21] L. Nussbaum and O. Richard. A comparative study of network link emulators. In *Proc. CNS*, 2009.
[22] A. Pásztor and D. Veitch. The packet size dependence of packet-pair like methods. In *Proc. IWQoS*, 2002.
[23] V. Paxson. *Measurements and Analysis of End-to-End Internet Dynamics*. PhD dissertation, University of California Berkeley, 1997.
[24] R. Carter and M. Crovella. Measuring bottleneck link speed in packet-switched networks. *Performance Evaluation*, 1996.
[25] S. Saroiu, P. Gummadi, and S. Gribble. Sprobe: A fast technique for measuring bottleneck bandwidth in uncooperative environments. In *Proc. IEEE INFOCOM*, 2002.