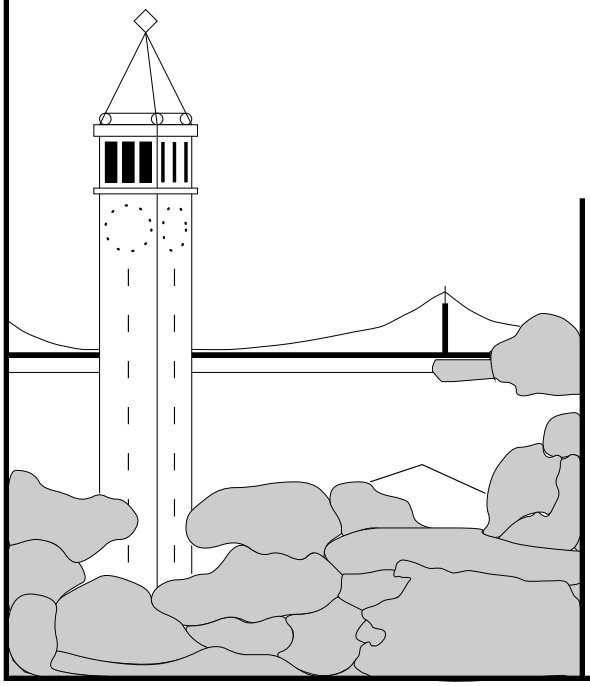
# SCONE: A Tool to Estimate Shared Congestion Among Internet Paths

*Weidong Cui, Sridhar Machiraju, Randy H. Katz, Ion Stoica*
*EECS Department, University of California, Berkeley*



**Report No. UCB/CSD-4-1320**

May 2004

Computer Science Division (EECS)
University of California
Berkeley, California 94720

# SCONE: A Tool to Estimate Shared Congestion Among Internet Paths

Weidong Cui, Sridhar Machiraju, Randy H. Katz, Ion Stoica
EECS Department, University of California, Berkeley

Technical Report UCB//CSD-04-1320

**Abstract**

It is well-known that the use of *path diversity*, i.e., the use of multiple end-to-end paths can improve the performance of applications such as multimedia streaming and Voice over IP. Leveraging path diversity requires applications to choose *independent* paths, i.e., paths that do not share *Points of Congestion* (PoCs). Although independent paths may not be available, performance can often be improved by selecting paths with the least amount of shared congestion. In this paper, we propose *Shared CONgestion Estimator* (SCONE), a tool that provides an estimate of shared congestion between two paths. SCONE sends probe flows along each of these paths and calculates the fraction of drops appearing in correlated bursts as the estimate of shared congestion. The assumption here is that correlated bursts typically occur at shared PoCs. Previously proposed path selection techniques only determine if two paths share a PoC or not. Also, previous techniques require that the two paths form one of only $2$ topologies, while SCONE can work with two additional topologies relevant to applications such as media streaming for a single source/destination pair over multiple paths. We used both *ns*-2 simulations and wide area experiments on PlanetLab to evaluate SCONE. We measured SCONE's ability to correctly estimate the fraction of drops that occurred at shared PoCs. We found that the absolute estimation error of SCONE is no more than $0.2$ in $95\%$ of simulation experiments. In $80\%$ of the wide area experiments, SCONE has an absolute estimation error less than $0.3$. We also show that the overhead of probe traffic can be avoided by passively observing application traffic such as multimedia streams which send adequate amounts of traffic.

## 1 Introduction

Congestion on paths in today's Internet is known to be bursty in nature[FJ93, YMKT99, ZDPS01]. When applications such as multimedia streaming experience bursty losses, their performance is adversely impacted. The use of multiple end-to-end paths, also referred to as *path diversity*, has recently [AWTW02, NZ03, RKB00] emerged as an effective method to limit the effects of congestion on any one of the paths. For instance, Nguyen et al. [NZ03] show that a suitably encoded multimedia stream split over two overlay paths between the same source and destination exhibits lesser variation in playback quality than a stream over a single path.

The simplest path selection mechanism to exploit path diversity is to choose link-disjoint paths (using *traceroute*) or *independent* paths, i.e., paths that do not share a *Point of Congestion (PoC)*[1]. However, independent paths may not exist. For instance, any two overlay paths between the same source and destination [NZ03] may share PoCs among the common last mile links near the source and destination. In the absence of independent paths, path diversity is best exploited by selecting paths with the least amount of shared congestion. However,

[1]We define a PoC to be a router that is dropping packets

previously proposed techniques [HBB00, KB01, RKT00] only detect the existence of shared PoCs. In this paper, we present *Shared CONgestion Estimator (SCONE)*, a tool that enables more practical path selection mechanisms by measuring shared congestion between two paths.

Given two paths, SCONE sends a probe flow along each path and calculates the fraction of drops of each flow that are likely caused at shared PoCs. SCONE provides these as estimates of shared congestion between the two paths. To compute these estimates, SCONE assumes that drops in correlated bursts occur at shared PoCs. SCONE employs the Expectation-Maximization (EM) [DLR77] algorithm with Gaussian mixture model to classify packet drops of each flow into bursts. To determine correlated bursts, SCONE approximates the times at which packets traversed shared links (and PoCs, if any) using their sending and receiving times. The approximation depends on the topology formed by the pair of paths. The topology could be one of the four shown in Figure 1. We believe that most applications that leverage path diversity with two paths use only one of these topologies.

We evaluate SCONE using *ns-2* simulations and wide area experiments on PlanetLab [Pla03], a global overlay network. The wide area experiments use a novel *concatenated IP paths* method to evaluate the performance of SCONE over real IP paths without knowledge on PoCs along these paths. We measured SCONE's ability to correctly estimate the fraction of drops that occurred at shared PoCs. We found that the absolute estimation error of SCONE is no more than $0.2$ in $95\%$ of simulation experiments. In $80\%$ of wide area experiments, SCONE has an absolute estimation error less than 0.3. We also show that the overhead of probe traffic can be avoided by passively observing application traffic such as multimedia streams which send adequate amounts of traffic.

Prior work related to choosing independent paths [RKT00, KB01, HBB00] used the fact that two flows through the same PoC will see correlated losses and/or delay. However, they considered only $2$ of the $4$ topologies that SCONE can work with and cannot be used with applications such as the multipath multimedia streaming application [NZ03]. Also, all of them only detect shared congestion and do not provide any estimate of shared congestion which is not useful for many applications. The techniques presented in [RKT00, HBB00] use active probes. As we show in this paper, SCONE can easily work by passively observing typical multimedia streams. Only limited evaluations of all these techniques over true wide-area paths has been done. One of our strengths is the detailed evaluations that we performed of SCONE.

This paper is organized as follows. In Section 2, we use prior work to motivate the target topologies using various applications and briefly describe how they may use SCONE. In Section 3, we explain our goals and assumptions, and describe the design of SCONE. We describe our simulation experiments in Section 4 and the wide area experiments in Section 5. We present our conclusions and future directions in Section 6.

## 2 Related Work

### 2.1 Applications and Target Topologies

In this paper, we consider the estimation of shared congestion between two paths. Generalizing this to multiple paths is a subject of future work. The two paths may form one of $4$ topologies shown in Figure 1. We believe that most applications that leverage path diversity use paths that form one of these topologies. We now describe motivating applications for each of these topologies.

Prior work in multimedia streaming [AWTW02, LSG01, NZ02, NZ03] showed that suitably encoded streams on multiple paths are much more resistant to congestion. In [AWTW02, NZ02], a client receives multimedia streams from multiple content servers. In this case, servers must be chosen such that the resulting paths to the client share as little congestion as possible. Any two paths from different servers to the client form the $Y$ topology shown in Figure 1(a). Here, $S_1, S_2$ represent the sources and $R_1, R_2$ represent the destinations of the

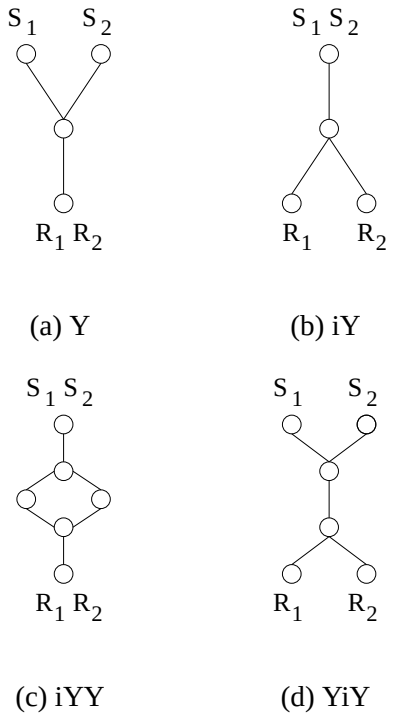



Figure 1: Topologies Created by Applications

two paths. The $iY$ topology shown in Figure 1(b) is useful in minimizing the time taken to distribute content [BLMR98]. In [NZ03], multimedia streams are sent using multiple overlay paths from the same source to the same destination. In this case, it is desirable to limit the congestion shared by the paths. Any two paths using an additional overlay node form the $iYY$ topology shown in Figure 1(c)). Determining the amount of shared congestion between two arbitrary paths (the $YiY$ topology) is useful in reducing the probing overhead [CBK03] of overlay networks such as RON [ABKM01].

The above applications use estimates of shared congestion to improve performance metrics such as playback quality and probing overhead. Certain other applications may use estimates of shared congestion to minimize undesirable or unfair competition to resources. Typically, this happens when an application has multiple flows traversing the same PoC. For instance, consider a client using parallel downloads [RKB00]. The various paths used by the client form the $Y$ topology shown in Figure 1(a). Parallel downloads that use traverse the same PoC would grab an unfair portion of the bandwidth at the PoC. Similarly, the paths from a content server to various mirror servers form an $iY$ topology. It is desirable that these paths share as little congestion as possible to prevent unfair competition between the flows to various mirror servers.

### 2.2 Techniques

Our work uses the observations made by previous studies on Internet path characteristics [JS00, ZDPS01, YMKT99]. These studies show that droptail queues used in most Internet routers lead to periods of bursty loss. Prior work on shared congestion [HBB00, KB01, RKT00] attempted to only detect the existence of shared PoCs and did not provide any estimate of shared congestion. Also, they only considered the $Y$ and $iY$ topologies. To our knowledge, our technique is one of the first ones to provide a solution to estimating shared congestion with the $YiY$ and $iYY$ topologies. A solution based on wavelet analysis of delay is presented in

[KKS+03]. However, their evaluation is mostly restricted to simulations.

In [RKT00], Rubenstein *et al.* use cross and auto correlation of delay and loss of Poisson probes to detect a shared PoC. They prove that the cross correlation of delay or loss is greater than the autocorrelation when two flows share a PoC. An implicit assumption made by them is that arrival times of Poisson probes in the queue at a shared PoC still follow a Poisson distribution. However, this need not be true, especially if there is significant jitter before the shared PoC. Also, their solution cannot be used when there are multiple PoCs on any of the paths. They provide limited experimental evidence to support their solution. Harfoush *et al.* [HBB00] use loss probabilities of single-packet and packet-pair probes (so-called conditional Bayesian probing) to identify shared losses. They evaluate their scheme with extensive simulations using a 2-hop topology. They do not provide experimental evidence to support their scheme.

Katabi and Blake [KB01] measure Renyi entropy of packet inter-arrival times of passive traffic to infer shared PoCs and then cluster flows which share the same PoC. In other words, they consider only the $Y$ topology. In [PQW03], Padmanabhan *et al.* study the use of various sampling methods to characterize lossy links to a server from clients. They use passively observed client-server traffic to identify lossy link(s) from the client to the server. Tomography-based solutions to identifying lossy links and loss rates [PQW03] use much larger datasets than we assume and may be considered complementary to our work. In [YF02], Younis *et al.* use in-band delay measurements to cluster clients sharing the same congested route to a server.

## 3 Design

Having motivated how the various topologies arise in the context of various applications, we start this section by describing our goals and assumptions. Then, we discuss technical challenges to achieve them. Finally, we describe SCONE and how it tackles each of these challenges.

### 3.1 Goals and Assumptions

Given two paths, our goal is to send a probe flow along each of them and to estimate, for each flow, the fraction of drops caused at shared PoCs. The two paths form one of the $4$ topologies shown in Figure 1. The probe flows may consist of actively sent traffic or passively observed application traffic. We assume that drops at PoCs are bursty due to the droptail nature of most routers and TCP congestion control. We also assume no network support; the only data available to calculate our estimates are the sending and receiving times along with loss information. Path properties such as loss and delay have been observed to be stationary (on the order of a few minutes) [ZDPS01]. This justifies our approach of using estimates based on past behavior to select paths for future use. Also, in the absence of admission control, past behavior is the best available indicator of future behavior.

### 3.2 Challenges

The occurrence of bursty packet drops implies that packets of two flows traversing a PoC at roughly the same time are usually either both dropped or not dropped [RKT00]. Hence, we estimate the fraction of shared drops by assuming that correlated drops occurred at shared PoCs. Here, we define correlated drops to be drops that occurred within some time $\delta$ of each other. We introduce $\delta$ only for developing our technique. It is not required in our final technique. To use this approach, three technical challenges need to be addressed.

- We need to determine correlated drops using only the sending and receiving times.
- Not all packets traversing a PoC during a loss period are dropped. This leads to false negatives.

- A loss period at a PoC that is not shared may persist long enough to coincide with a loss period from another PoC. Thus, the above approach could also lead to false positives.

### 3.3 Shared CONgestion Estimator

Next, we describe *Shared CONgestion Estimator (SCONE)*, our tool to estimate shared congestion between two paths and explain how it tackles each of the above challenges. We assume that the probe flows, $F_1, F_2$ used by SCONE consist of periodically sent UDP packets of size $S$ bytes. We use $t$ to denote the time between consecutive probe packets. We denote $\mu_{actual,i}$ to be the actual fraction of packet drops of flow $F_i$ at shared PoCs and $\mu_i$ to be the estimate computed by SCONE. For convenience, we sometimes drop the subscript $i$ to refer to either of the two flows. Table 1 summarizes all notations that we use in this paper.

#### *3.3.1* Determining Correlated Drops

The basic idea behind determining correlated drops is simple. Consider a hypothetical network in which all PoCs are clock synchronized. Also, assume that these PoCs timestamp packets that are dropped and provide the packet identities and timestamps to the source. Dropped packets with timestamps that are within $\delta$ of each other are correlated. In reality, we need to approximate these timestamps using the sending and receiving times only. Since the sources in the $iY$ topology (see Figure 1(b)) are co-located, their clocks are synchronized. Also, the delay from each source to any shared link (or PoCs) is the same. Hence, the sending times of each source differ from the hypothetical timestamps at a shared PoC by the same amount. Thus, the sending times can be used to defi ne correlated drops. Similarly, in the $Y$ topology, the receiving times of each destination differ from the hypothetical timestamps at a shared PoC by the same amount. Since receiving times of dropped packets do not exist, we intrapolate the receiving times of other packets to assign receiving times to dropped packets. For instance, consider a single dropped packet sent between two packets which are received at $t_1$ and $t_2$. The dropped packet is assigned an intrapolated receiving time of $(t_1 + t_2)/2$. The intrapolated receiving times approximate the hypothetical timestamps and can be used to defi ne correlated drops. Packets in the $iYY$ topology (see Figure 1(c)) may be dropped by PoCs shared near the sources or destinations. Hence, two drops are correlated if they are correlated using the sending times or intrapolated receiving times.

Determining correlated drops of paths forming the $YiY$ topology is more complicated. Consider hypothetical PoCs in the $YiY$ topology (see Figure 1(d)) that timestamp dropped packets as before. Denote $d_i$ to be the one-way delay from the source $S_i$ of flow $F_i$ to an arbitrary shared PoC. Also, assume that $S_1$ is synchronized with the clock at each PoC and that $S_2$ lags this clock by $\Delta$. Now, a packet of flow $F_i$ with a timestamp of $\tau_i$ would have been sent at time $\tau_i - d_i$ of the clock at each PoC. The local sending time of the packet of flow $F_1$ would be $s_1 = \tau_1 - d_1$. The local sending time of the packet of flow $F_2$ would be $s_2 = \tau_2 - d_2 - \Delta$. By our defi nition, $|\tau_1 - \tau_2| \leq \delta$ for correlated drops. Hence, the following also holds for correlated drops:

$$|s_2 - s_1 + (d_2 - d_1 + \Delta)| \leq \delta \tag{1}$$

We refer to the quantity, $(d_2 + d_1 - \Delta)$ as the *synchronization lag (synclag)*. Figure 2 illustrates synclag using two sources $S_1$ and $S_2$ of a $YiY$ topology. The packet sent at time $0$ of the local clock at $S_2$ traverses the shared PoC at around the same time as the packet sent at time $2$ of the local clock at $S_1$. Therefore, the synclag of flow $F_2$ (with respect to flow $F_1$) is about $2 - 0 = 2$. The above analysis is valid even without our assumption that the clock at $S_1$ is synchronized with hypothetical clocks at shared PoCs. This is because the defi nition of synclag involves the difference in times at $S_1$ and $S_2$ only.

To estimate the synclag, we assume that it remains constant during the lifetime of the probes. Note that the difference between the times at two sources can be bounded by the round trip time (RTT) between the sources

Table 1: Notation used in this paper

| Notation | Comments |
|---|---|
| $\mu$ | estimated fraction of packet drops at shared PoCs of any flow |
| $\hat{\mu}$ | actual fraction of packet drops at shared PoCs of any flow |
| $\mu_i$ | estimated fraction of packet drops at shared PoCs of flows along path $i(i = 1, 2)$ |
| $\hat{\mu_i}$ | actual fraction of packet drops at shared PoCs of flows along path $i(i = 1, 2)$ |
| $\mu_{min}$ | lower bound on the fraction of packet drops at shared PoCs of any flow |
| $\mu_{max}$ | upper bound of the fraction of packet drops at shared PoCs of any flow |
| $F_i$ | probe flow $i(i = 1, 2)$ |
| $f$ | overlap fraction |
| $b$ | burst interval |
| $t$ | interval between successive probe packets |
| $s$ | size of probe packets |
| synclag | synchronization lag |
| CCC | cross correlation coeffi cient |

by letting them exchange clock information. Also, the difference of one-way delay to shared links is bounded by the maximum RTT from the sources to destinations. Hence, the synclag is bounded by $2RTT_{max}$. A correct estimate of synclag is likely to correlate shared drops than a wrong estimate. Referring to Figure 2, assume that a bursty loss period causes packets $2, 3$ sent by $S_1$ and $0, 1$ sent by $S_2$ to be dropped. The correct estimate of synclag would correlate all these drops. A wrong estimate would correlate at most $1$ drop per flow. In other words, the correct value of synclag is likely to lead to the maximum number of correlated drops. Therefore, we try various values of synclag and choose the one that leads to maximum number of correlated drops. Since $|synclag| < 2RTT_{max}$, we need to try $2RTT_{max}/t$ values of synclag with a probing period of $t$. In practice, the number of values is not more than $100$ assuming $RTT_{max}$ of $0.5s$ and a probing period of $10ms$.

We make three points on using the above method to estimate synclag.

- If two flows do not share any PoC, the above process could sometimes lead to a signifi cant number of drops being correlated falsely. This is a disadvantage of the above scheme. But we observe this in very few cases in our experiments.
- One component of the synclag is the difference between the clocks at the two senders. NTP [Mil92] can be used to eliminate this. However, the delay to shared links cannot be measured using end-host measurements alone. Hence, for the $YiY$ topology, we must use the described algorithm to calculate synclag even when the end-hosts are perfectly synchronized.
- The method to calculate the synclag in the $YiY$ topology assumes that the shared links form one contiguous segment and that the times at which all shared PoCs are traversed can be approximated by shifting

the sending times by the same value. In topologies where the shared links do no all form a contiguous segment (e.g., the $iYY$ topology), our method of calculating synclag cannot be used.

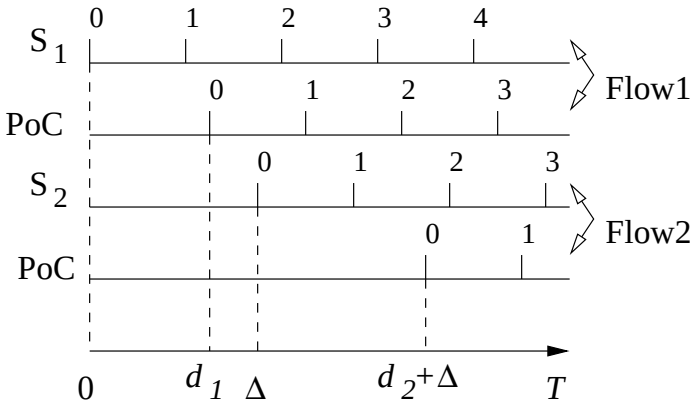



Figure 2: An illustration of synclag ($= \Delta + d_2 - d_1$) of $F_2$ with respect to $F_1$ when the time at two sources $S_1$ and $S_2$ differs by $\Delta$ and have a delay of $d_1$ and $d_2$ to the shared PoC. A packet sent at time $t$ by $S_2$ traverses shared links at around the same time as a packet sent at time by $t + synclag$ by $S_1$.

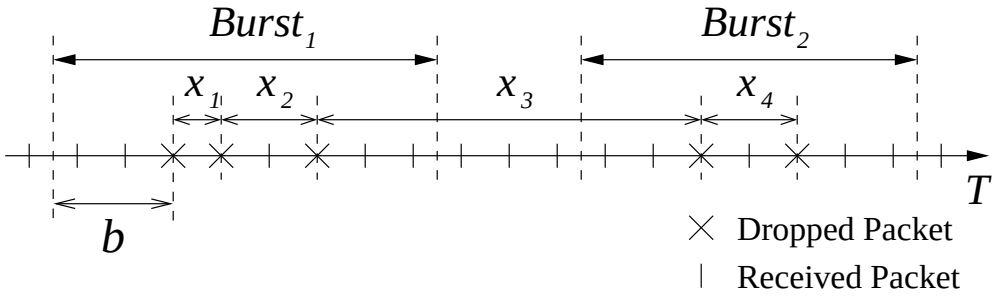



Figure 3: Illustration of SCONE's Classifi cation of Bursts

### *3.3.2* **Clustering Drops into Bursts**

Counting correlated drops may lead to false negatives. This is because all packets in a bursty loss period may not be dropped. Hence, we take the approach of classifying drops into bursts and counting the number of drops in correlated bursts. We classify two drops of a flow to belong to the same burst if and only if they were sent within a time $b$ of each other. We refer to $b$ as the *burst interval* and consider the burst to have lasted from time $b$ before its fi rst observed drop to $b$ after its last observed drop. This is shown in Figure 3.

To determine $b$, we observe that the time between two consecutive drops would either be the time between two drops in a burst ($x_1, x_2, x_4$ in Figure 3) or the time between drops belonging to consecutive bursts($x_3$ in Figure 3). Hence, we expect the times between consecutive drops to form two clusters. We assume that these two clusters are Gaussian and use the Expectation-Maximization (EM) algorithm [DLR77] with Gaussian mixture model to determine these clusters. Since the latter can be assumed to be much larger on average, the $95^{th}$ percentile of the Gaussian distribution with smaller mean is taken to be $b$.

### *3.3.3* Defining Correlated Bursts

Having classified drops of the $2$ flows into bursts, we need to define correlated bursts, i.e., those that are likely to have occurred at the same PoC. Requiring two bursts to have occurred during the exact time interval may be too strict. Requiring them to just overlap could lead to false positives. Therefore, we define two bursts to be correlated if and only if the overlap of the two bursts consists of more than some fraction $f$ of the drops of *each* burst. We call $f$ the *overlap ratio*. For instance, in Figure 4 a burst of $4$ drops overlaps with a burst of $2$ drops. The overlap contains all the drops of $F_2$'s burst and $3$ out of $4$ drops of $F_1$'s burst. Hence, these bursts would be considered correlated if $f \leq 0.75$ and not, otherwise. Thus, SCONE calculates $\mu_1(\mu_2)$ to be the fraction of drops of $F_1(F_2)$ that belong to correlated bursts as defined above. Note that $\mu_1$ and $\mu_2$ need not have the same value since each flow could traverse other PoCs, too. As mentioned earlier, our use of correlated bursts removes the need to use the parameter $\delta$.

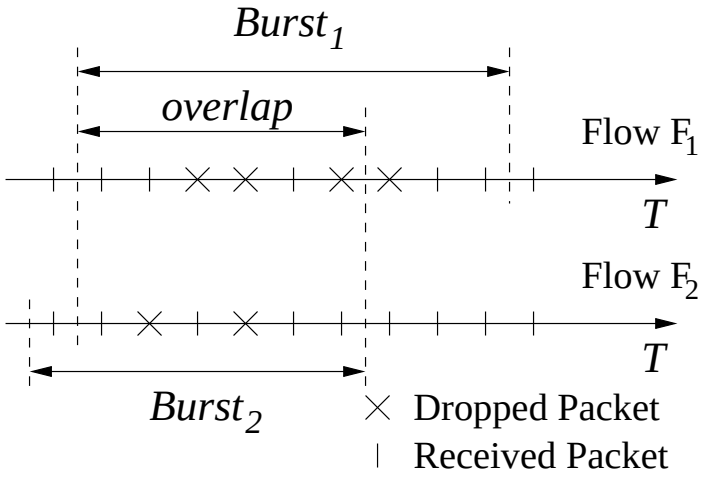



Figure 4: Motivating and defining overlap ratio

## 4 Simulation Experiments

In this section, we present the results of our simulation experiments. First, we describe our simulation setup which is similar to the setup used in [RKT00]. Then, we present the results of experiments with various topologies. Finally, we use multimedia traces to demonstrate that SCONE can replace active probes with passively observed flows that send adequate traffic.

### 4.1 Methodology

We ran *ns-2* simulations with each of the $4$ target topologies. Each link in Figure 1 was replaced by $2-3$ links in the simulated topology. This was done to better simulate wide-area paths. The one-way delay along each simulated link was chosen uniformly at random between $3ms$ and $7ms$. Each flow had at most $2$ shared PoCs and $2$ non-shared PoCs. The capacity of a PoC was chosen uniformly at random between $20$ and $40$Mbps. This ensured that we simulated various values of shared congestion. The capacity of the other links was $1000$Mbps. Each PoC had cross traffic consisting of $10-15$ flows. At least $75\%$ of these flows were TCP flows. The rest were exponential "on-off" CBR flows with a maximum rate of $300$Kbps and average on and off times of $0.5s$. Each experiment had two probe flows whose sending and receiving times were used by SCONE to calculate $\mu_1$, $\mu_2$, estimates of shared congestion for the two paths. We count each of these estimates as one datapoint in our plots. Each simulation ran for $320s$ where probe flows start at $20s$ and background flows start at $0s$.

Unless otherwise specified, we used probe flows sending UDP packets of size $40$ bytes every $10ms$ in both *ns-2* simulations and wide-area experiments. We repeated experiments for each set of parameters $100$ times using random initial seeds.

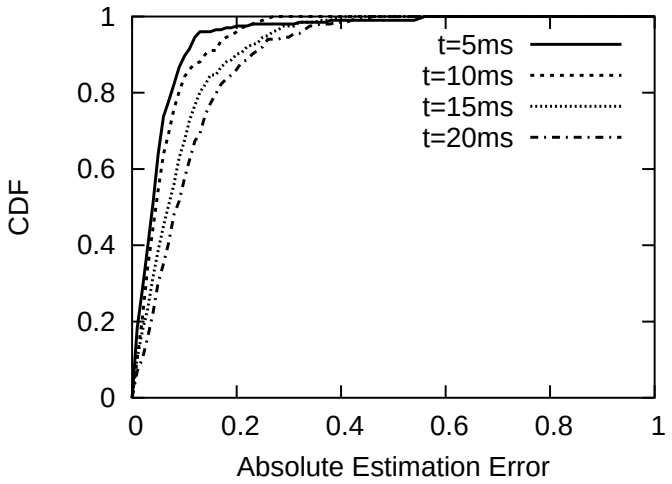



Figure 5: Plot of the CDF of the absolute estimation error ($|\mu - \mu_{actual}|$) for various probing rates in the case of $iYY$ topology using an overlap ratio of $0.7$.

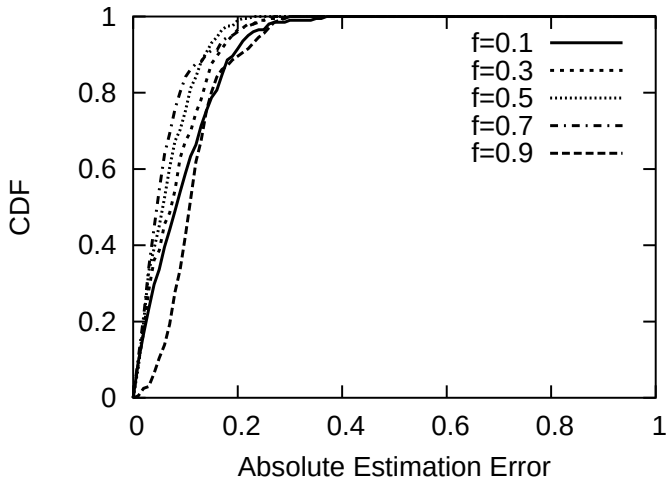



Figure 6: Plot of the CDF of the absolute estimation error ($|\mu - \mu_{actual}|$) for various overlap ratios in the case of $iYY$ topology using a probing rate of $10ms$.

### 4.2 Parameter Selection

We conducted the first set of simulations to explore parameter selection. We did this for the only two parameters that SCONE uses, probing period and overlap ratio. The results of varying probing rates with an overlap ratio of $0.7$ in the $iYY$ topology in Figure 5. In this figure, we plot the CDF of the absolute value of estimation error (the difference between SCONEs' estimate and the actual fraction of shared drops) obtained with SCONE for probing periods of $5$,$10$,$15$ and $20ms$. We did not use larger probing periods since we observed that many bursty loss periods lasted no longer than $20ms$ and many shared losses were not correlated. This plot shows that there was a gradual decrease in accuracy with increasing probing periods. The relatively large difference between probing periods of $10ms$ and $15ms$ indicates that $10ms$ provides the best tradeoff between overhead and accuracy. We obtained similar results for other topologies and overlap ratios. In absolute terms, the performance with a $10ms$ probing period is that the estimation error is no more than than $0.2$ in $95\%$ of the experiments.

We also conducted experiments to evaluate the effect of overlap ratio, $f$, on SCONE's performance. Figure 6 plots the CDF of the absolute estimation error for overlap ratios from $0.1$ to $0.9$ for a probing period of $10ms$. This plot shows that a large overlap ratio of $0.9$ and a small overlap ratio of $0.1$ both perform poorly. The reason is that $f = 0.9$ results in underestimation of $\mu_{actual}$ while $f = 0.1$ results in overestimation. We also see that an overlap ratio from $0.5$ to $0.7$ provides the best results. Based on the above discussion, we recommend the use of $0.7$ as the overlap ratio and $10ms$ to be an appropriate probing period. Our results show that deviations from these values degrade performance smoothly and slowly.

### 4.3 Target Topologies

We ran experiments to measure the performance of SCONE with each of the four target topologies, iY, Y, iYY and YiY. These are plotted in Figures 4.2, 4.2, 4.2 and 4.2 respectively. We plot the error of SCONE's

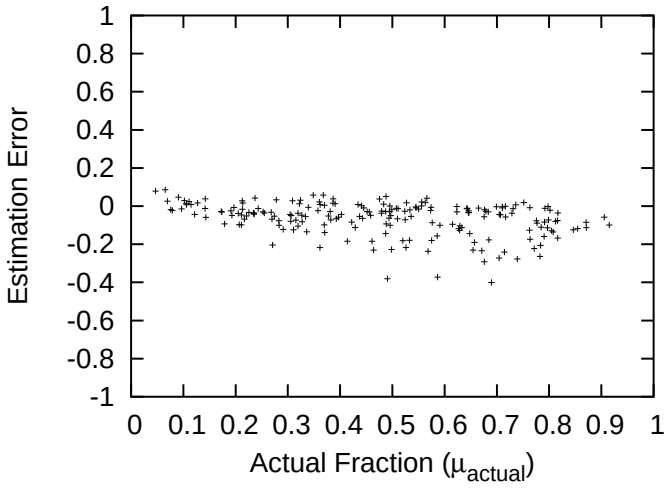



Figure 7: iY Topology

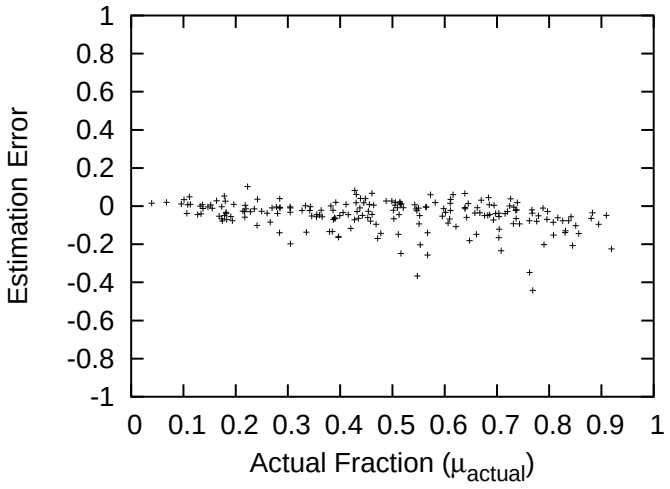



Figure 8: Y Topology

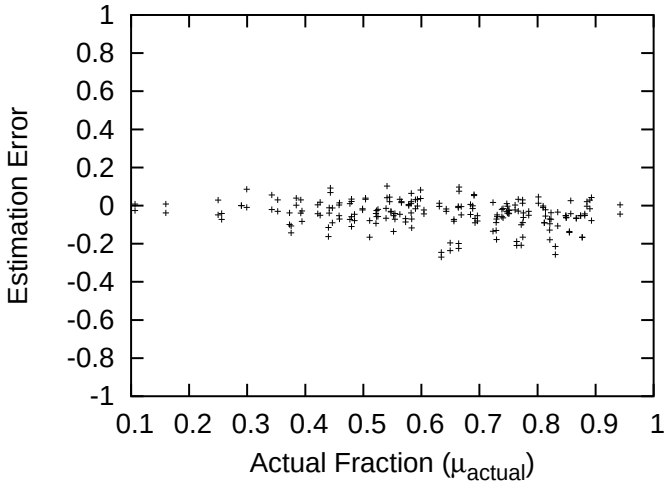



Figure 9: iYY Topology

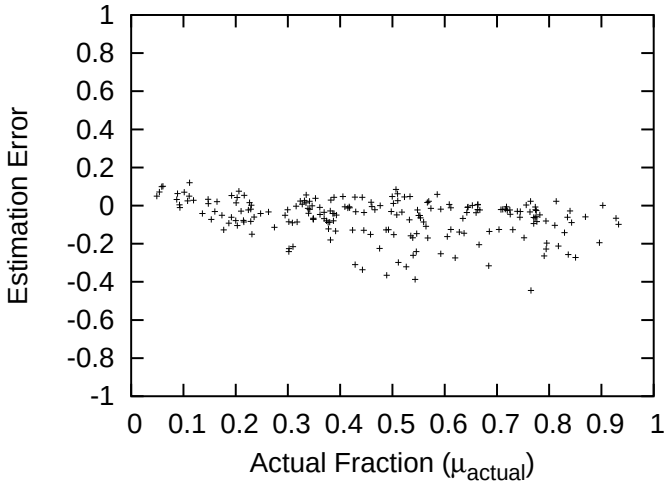



Figure 10: YiY Topology

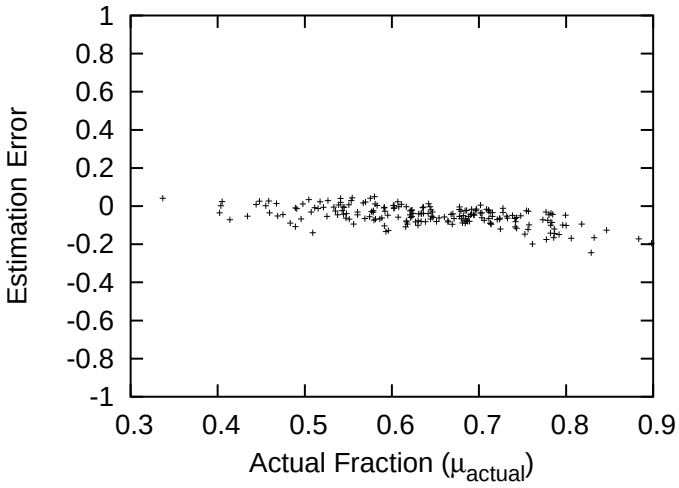



Figure 11: Passively observed flows in the $iYY$ topology.

estimates versus the actual fraction of shared drops in each of these plots. These plots provide us with two key observations. These show that, for each topology, our experiments included values of shared congestion from $0$ to $1$. These plots also show that, for each topology, the absolute estimation error is no more than $0.2$ in $95\%$ experiments.

### 4.4 Passive Probes

Probe flows sending $40$ byte packets every $10ms$ generate a relatively large packet overhead of $100$ packets/sec. We conducted experiments to investigate if SCONE can use passively observed application traffic as probe flows. These experiments simulated a source using an overlay to split an MPEG stream over two overlay paths to the same destination [NZ03]. In the absence of a standard multiple-description coding, we simply sent odd packets of the MPEG stream on one path and even packets on another path. Figure 11 plots the results of using SCONE with such multimedia traffic. This plot shows that SCONE's performance is almost the same as obtained with active probes, i.e., the absolute estimation error is not more than $0.2$ in $95\%$ experiments. The multimedia streams we used sent one packet every $12ms$ on each path which is close to the probing period we used. In general, most multimedia streams send one frame every $30ms$ and each frame consists of multiple packets. All such streams which send one packet every $10ms$ can be passively observed by SCONE. Note that application traffic uses varying packet sizes. A focus of future work is to quantify the dependence of variable-sized probe packets on SCONE's performance.

## 5 Wide-area Experiments

To evaluate SCONE on a large and diverse set of IP paths, we used PlanetLab [Pla03], a global overlay network with over $300$ nodes at more than $140$ sites. To measure the accuracy of the estimates computed by SCONE for two paths between PlanetLab nodes, we needed to know the number of drops on all shared IP links of the paths. This was clearly not possible without extensive network support. Hence, we used a novel method of concatenating IP paths to verify the accuracy of SCONE without such information. In this section, we first explain our method of concatenating IP paths. Then, we describe our experimental setup and datasets. Finally, we provide the results of our experiments.

### 5.1 Concatenated IP paths Method

We used a novel method of concatenating IP paths since we had no information on shared congestion between two arbitrary IP paths between Planetlab nodes. We explain our method of concatenated IP paths for the $YiY$ topology shown in Figure 1(d). We constructed the $YiY$ topology by choosing $6$ random PlanetLab nodes for each of $S_1$, $S_2$, $R_1$, $R_2$, $M_1$ and $M_2$. We used SCONE to start probe flows from $S_1$ and $S_2$. Instead of sending them on direct IP paths to $R_1$ and $R_2$, we used application-level routers at each node that sent packets from $S_1$ ($S_2$) to $M_1$, $M_1$ to $M_2$ and $M_2$ to $R_1$ ($R_2$). Thus, we created two paths, each consisting of $3$ concatenated IP paths, that formed the $YiY$ topology. These two paths shared all PoCs on the IP path between $M_1$ and $M_2$. Hence, packets received by $M_1$ and not received by $M_2$ were drops at shared PoCs. However, the IP paths from $S_1$ ($M_2$) to $M_1$ ($R_1$) and $S_2$ ($M_2$) to $M_1$ ($R_2$) were likely to have last mile IP links near $M_1$ ($M_2$) in common. A PoC among these links was a shared PoC whose drops were not caused on the IP path from $M_1$ to $M_2$. Hence, including these drops would give us an upper bound $\mu_{max}$ on $\mu_{actual}$ of each path. Excluding these drops would give us a lower bound $\mu_{min}$ on $\mu_{actual}$. Note that $\mu_{max}$ is trivially $1$ for the above setup. Figure 12 shows an experimental setup involving $8$ PlanetLab nodes for which $\mu_{max}$ was not $1$ whenever drops occur on the overlay link from $S_i$ to $N_i$. *Traceroutes* along the the IP paths from $S_1$ to $N_1$ and $S_2$ to $N_2$ can be used to make sure that they do not share any link and hence, do not share any PoC. Note that, the other topologies can also be generated by concatenating appropriately created IP paths. As shown in Figure 12 for the $YiY$ topology, these

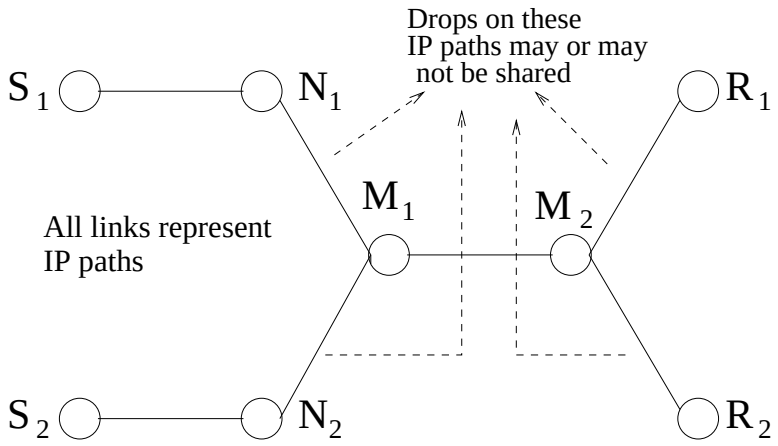



Figure 12: Experimental setup with concatenated IP paths for the $YiY$ topology; $\mu_{max}$ need not be $1$.

can be created such that $\mu_{max}$ need not always be $1$. The above method increases the average one-way delay and also introduces overhead at the application-level routers. We believe that these do not affect the validity of our results.

### 5.2 Experimental Setup

For each experiment, we chose *randomly* chosen nodes of Planetlab. Multiple nodes exist at most Planetlab sites. No two nodes located at the same site were chosen to be a part of the same experiment. In each experiment, We used flows with lifetimes of $600s$ periodically sending UDP packets of size $40$ bytes as our probes. The application-level routers, senders and receivers saved the local times at which packets were forwarded. We used this information to deduce the overlay links on which each packet drop occurred. We then compared the estimate $\mu$ computed by SCONE with the range $[\mu_{min}, \mu_{max}]$ of possible values. Note that SCONE computes its estimate using the traces from end-hosts only. Since Planetlab is a shared infrastructure not managed by us, our process would sometimes be blocked for signifi cant time intervals. Hence, after synchronizing the two flows, SCONE only considers those times during which both flows were sending packets. Note that $\mu_{min}$ and $\mu_{max}$ can be calculated with and without including such drops that were sent when one of the flows was inactive. For all our experiments, we verifi ed that these two values did not differ signifi cantly. We also collected *traceroutes* along IP paths to verify our assumption that certain IP paths do not share an IP address, and hence do not share a PoC (e.g., $S_1$ to $N_1$ and $S_2$ to $N_2$ in Figure 12). We conducted our experiments in March, June and July of $2003$. We used about $100$ experiments for $Y, iY, YiY$ topologies and about $500$ experiments for $iYY$ topologies. The asymmetry in the number of experiments was because we used the $iYY$ topology to fi ne-tune the design of SCONE before experimenting with the other topologies. We did not consider about $35\%$ of our experiments in which at least one path had less than $30$ drops ($0.05\%$ loss percentage).

### 5.3 Experimental Results

For each topology, we obtained experiments with various sizes of the interval $[\mu_{min}, \mu_{max}]$. Experiments with large sizes of the interval cannot be used to verify the accuracy of SCONEs' estimate, $\mu$. Hence, we consider only those experiments in which $\mu_{min}$ and $\mu_{max}$ do not differ by more than $0.1$. We show a scatterplot of the estimation error ($\mu - \mu_{min}$) versus $\mu_{min}$ with the $4$ topologies in Figures 13, 14, 15 and 16. We see that the estimates are mostly less than the actual amount of shared congestion. Also, the absolute estimation error is less than $0.2$ ($0.3$ if it is calculated as $\mu - \mu_{max}$) in about $80\%$ of the experiments for all the topologies. We plot

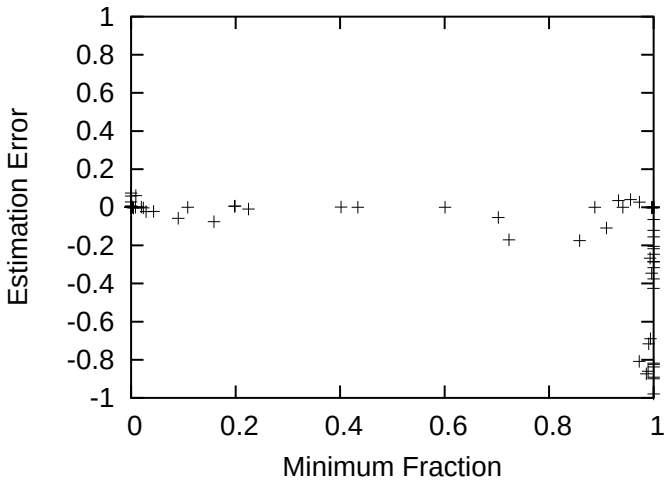



Figure 13: iY Topology

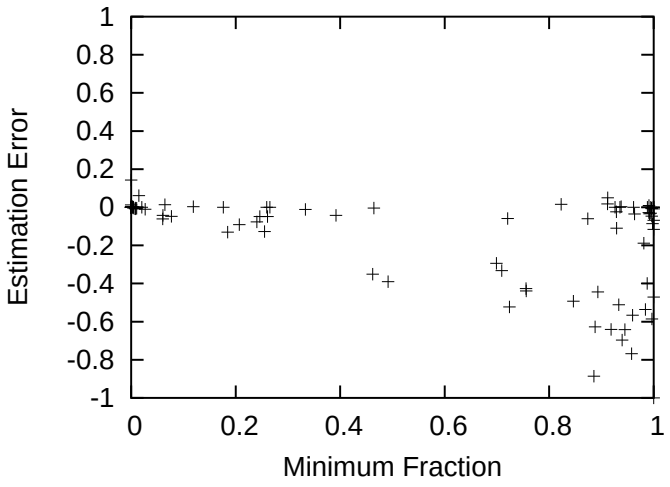



Figure 14: Y Topology

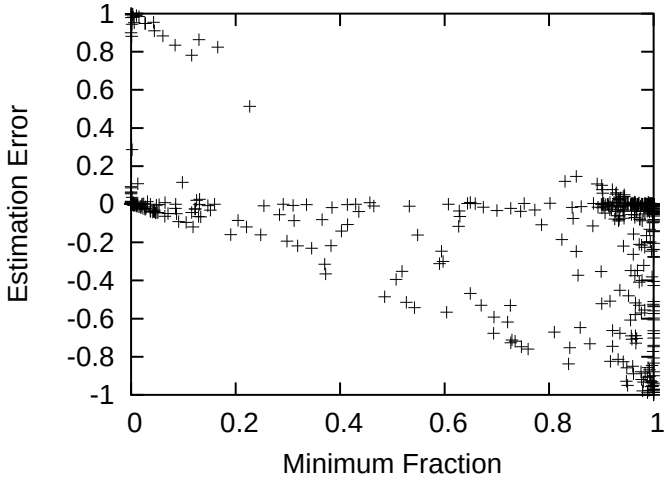



Figure 15: iYY Topology

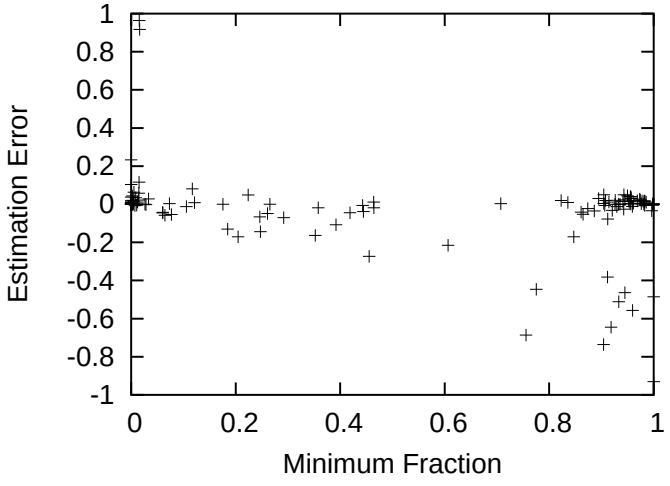



Figure 16: YiY Topology

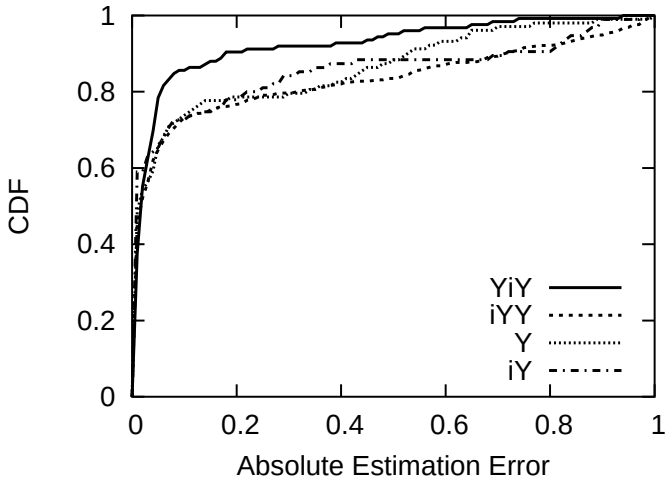



Figure 17: CDF of the Absolute Estimation Error for all Topologies

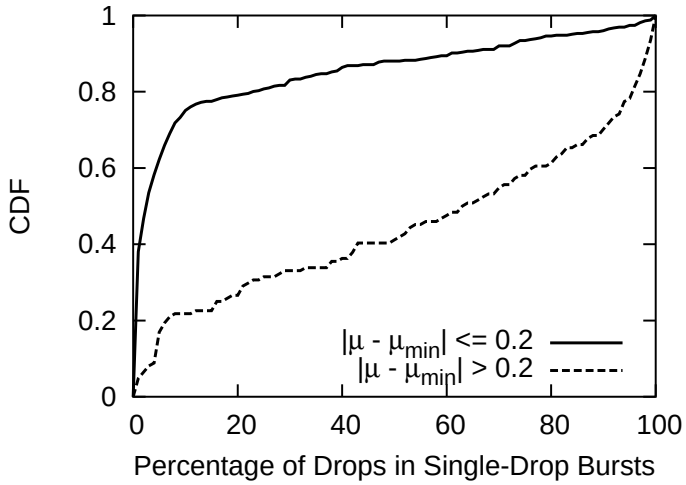



Figure 18: Demonstrating the Influence of Little Bursty Behavior on SCONE's performance.

the CDF of the absolute estimation error for each topology in Figure 17. The results with the $YiY$ topology are slightly better than with the other topologies. Since these experiments were conducted at a different time than the others, we believe that this difference is just because of random effects. We now discuss why some experiments had false positives (a large positive estimation error) and some others had false negatives (a large negative estimation error).

**False Positives:** In the above mentioned figures, we see a few cases of a large positive estimation error. For instance, in Figure 16, we notice two experiments with a $\mu$ close to $1$ and a $\mu_{min}$ close to $0$. For these experiments, we analyzed the traceroutes of the non-shared IP paths ($S_1$ to $N_1$ and $S_2$ to $N_2$ in Figure 12. We noticed that these paths actually had one or more common IP addresses in the traceroute corresponding to a trans-Atlantic link. Hence, we believe that the actual fraction of shared congestion is actually close to $1$ and hence, these experiments are actually not false positives.

**False Negatives Caused by Random Losses:** We noticed that SCONE failed for most experiments involving Planetlab nodes at Intel Research at Pittsburgh, Intel Research at Seattle and Stanford University. These three sites consistently exhibited no bursty loss behavior. Upon further investigation, we learned that the Intel sites were severely rate limiting traffic. In fact, with probing rates of $20KBps$, the probe traffic itself caused massive congestion. Rate limiting is frequently done using RED-like schemes in routers (e.g., Cisco IOS software [Cis03]). This would explain the consistently observed random drop behavior of these sites. All the results presented in this paper do not include experiments involving these three sites.

Many experiments involving the four topologies which had large negative estimation error exhibited random drops. Figure 18 illustrates this by plotting the percentage of drops that belonged to single-drop bursts. This is plotted for experiments with absolute estimation error less than $0.2$ and greater than $0.2$. About $80\%$ of the former and $20\%$ of the latter had less than $20\%$ of drops in single-drop bursts. If SCONE rejected all experiments that had more than $20\%$ of the drops in single-drop bursts, it would reject about $30\%$ of the experiments. However, among the remaining experiments, SCONE would have a success percentage of $95\%$. Whether SCONE can be augmented with mechanisms for the rejected experiments is a focus of ongoing work for us. We believe that random drops occur either due to the use of RED as the queuing discipline (instead of the droptail queuing discipline) or due to the occurrence of very short transient periods of congestion.

**Probing Period:** With our experiments, we showed that probing periods greater than $20ms$ were not suitable since some bursty periods did not last more than $20ms$. With our wide-area experiments, we explored lower probing periods. Specifically, with the $YiY$ topology, we sent probe flows of period $1ms$. Then, we used SCONE with these flows and with $10ms$ sub-flows extracted from them. We did this to compare the effects of probing rate on the same paths at the same time. The results of using SCONE with the $1ms$ flows and $10ms$ sub-flows are plotted in Figure 19. As we can see, very little is gained by using a probing period of $1ms$.

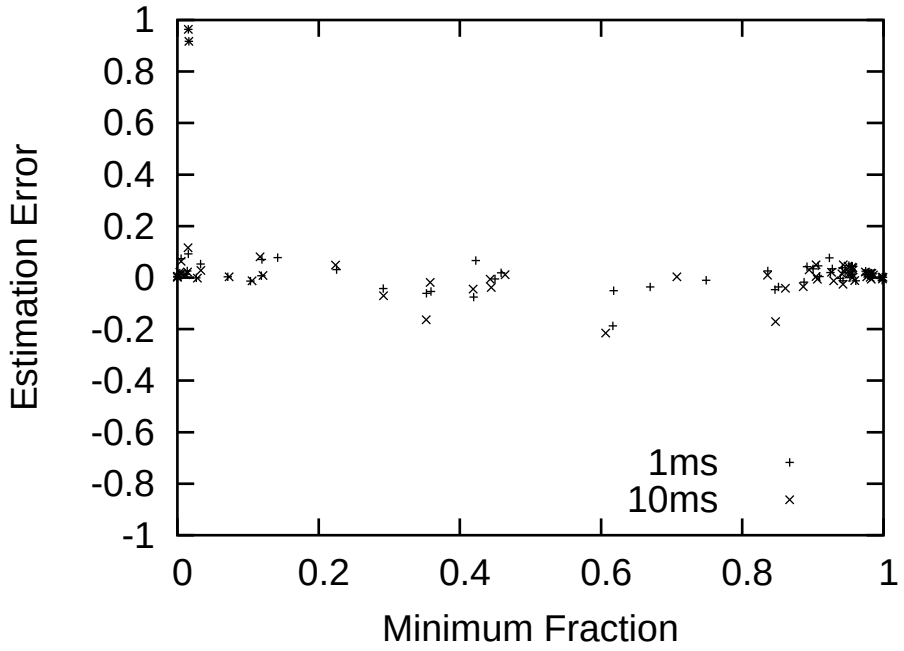



Figure 19: Scatterplot of SCONE's estimation error with probe flows having a period of $1ms$ and $10ms$ sub-flows extracted from these flows.

**Synclag:** For experiments involving the $YiY$ topology, we also verified that the estimated synclag value was close to the actual synclag which can be roughly computed based on the one-way delay to the shared IP path. In cases of two flows not sharing any PoC, the estimated synclag value was essentially random. Figure 20 shows the justification for calculating synclag as the value that maximizes the cross-correlation coefficient. In this figure, we show that the cross-correlation coefficient and the number of correlated drops are all maximized at the same assumed synclag value. We also verified that a synclag value of $500ms$ was indeed reasonable in this experiment.

One conclusion from our results is that $10ms$ is a good probing rate. Higher rates do not help improve the accuracy greatly but add to the overhead only. Also, SCONE for all the topologies provides an estimate at most $0.2$ ($0.3$) less than the $\mu_{min}$ ($\mu_{max}$) actual value in at least $80\%$ of the experiments. Estimates made by SCONE rarely exceeded the maximum value of shared congestion. As shown above, initial analysis of some experiments for which SCONE failed led us to believe that checking for random drops can be used to make SCONE very good at the expense of not being able to work with some IP paths.

## 6 Conclusions

Applications such as multimedia streaming can limit the effects of congestion on one path by using multiple paths that share the least amount of congestion. In this paper, we propose Shared CONgestion Estimator

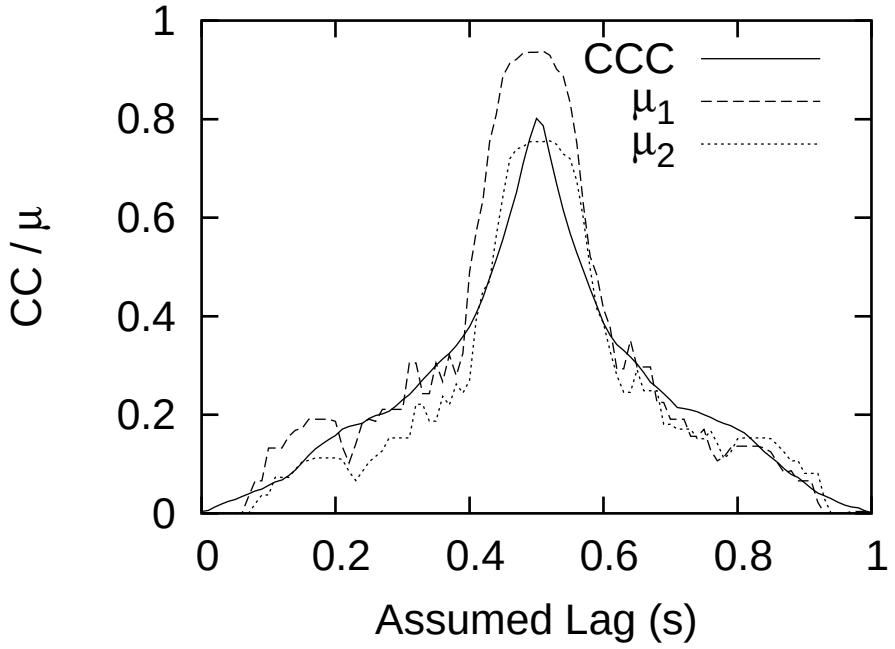



Figure 20: A Graph Plotting the Cross Correlation Coeffi cient and SCONE's Estimates with Various Values for Synclag

(SCONE), a tool that enables these applications to select such paths. SCONE is more powerful than previously known techniques which merely detect shared congestion. SCONE can work with $4$ topologies involving two paths, $2$ of which were not considered in prior work. Results of extensive simulations that we performed showed that, the absolute estimation error of SCONE is at most $0.2$ in $95\%$ of the experiments. We also showed that SCONE can replace active probe traffi c with passively observe application traffi c such as multimedia streams. The results of wide area experiments using PlanetLab showed that SCONE's absolute estimation error is at most $0.3$ in about $80\%$ of the experiments. We investigated the reasons for the larger error of the remaining experiments. We found that many of these paths exhibited random drops. We describe a preliminary method to detect the occurrence of such random drops which allows SCONE to achieve $95\%$ success. This comes at the expense of not being able to work with $30\%$ of the paths. A focus of future work for us is to enable SCONE to work with paths that exhibit random drops.

**References**

[ABKM01] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris. Resilient overlay networks. In *Proceedings of ACM SOSP'01*, 2001.

[AWTW02] J. Apostolopoulos, T. Wong, W. T. Tan, and S. Wee. On multiple description streaming with content delivery networks. In *Proceedings of IEEE INFOCOM'02*, July 2002.

[BLMR98] John W. Byers, Michael Luby, Michael Mitzenmacher, and Ashutosh Rege. A Digital Fountain Approach to Reliable Distribution of Bulk Data. In *Proc. of ACM SIGCOMM*, 1998.

[CBK03] Yan Chen, David Bindel, and Randy H. Katz. Tomography-based Overlay Network Monitoring. In *Proc. of the Internet Measurement Conference(IMC)*, 2003.

[Cis03] Cisco IOS Software. www.cisco.com/warp/public/732/Tech/car/, 2003.

[DLR77] A. Dempster, N. Laird, and D. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society*, pages 1–38, 1977.

[FJ93] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ ACM Transactions on Networking*, 1(4):397–413, 1993.

[HBB00] K. Harfoush, A. Bestavros, and J. Byers. Robust identifi cation of shared losses using end-to-end unicast probes. In *Proceeding of IEEE ICNP'00*, Osaka, Japan, October 2000.

[JS00] W. Jiang and H. Schulzrinne. Modeling of packet loss and delay and their effect on real-time multimedia service quality. In *Proceedings of NOSSDAV*, 2000.

[KB01] D. Katabi and C. Blake. Inferring congestion sharing and path characteristics for packet interarrival times. Technical report, MIT LCS, December 2001.

[KKS+03] Min Sik Kim, Taekhyun Kim, YongJune Shin, Simon S. Lam, and Edward J. Powers. A Wavelet-based Approach to Detect Shared Congestion. Technical report, University of Texas at Austin, Department of Computer Sciences, November 2003. Revised, March 2004.

[LSG01] Y. J. Liang, E. G. Steinbach, and B. Girod. Real-time voice communication over the internet using packet path diversity. In *Proceedings of ACM Multimedia*, pages 431–440, Ottawa, Canada, 2001.

[Mil92] D. L. Mills. RFC 1305: Network Time Protocol(v3), March 1992.

[NZ02] T. Nguyen and A. Zakhor. Distributed video streaming with forward error correction. In *Packet Video Workshop*, Pittsburgh PA, USA, 2002.

[NZ03] T. Nguyen and A. Zakhor. Path Diversity with Forward Error Correction (PDF) System for Packet Switched Networks. In *INFOCOM*, San Francisco, USA, 2003.

[Pla03] PlanetLab. http://www.planet-lab.org, 2003.

[PQW03] V. N. Padmanabhan, L. Qiu, and H. Wang. Server-based inference of internet performance. In *IEEE INFOCOM'03*, San Francisco, CA, USA, April 2003.

[RKB00] P. Rodriguez, A. Kirpal, and E. Biersack. Parallel-access for Mirror Sites in the Internet. In *INFOCOM'00*, 2000.

[RKT00] D. Rubenstein, J. F. Kurose, and D. F. Towsley. Detecting shared congestion of flows via end-to-end measurement. In *Proceedings of ACM SIGMETRICS'00*, June 2000.

[YF02] O. Younis and S. Fahmy. On effi cient on-line grouping of flows with shared bottlenecks at loaded servers. In *Proceeding of IEEE ICNP'02*, Paris, France, November 2002.

[YMKT99] M. Yajnik, S. B. Moon, J. F. Kurose, and D. F. Towsley. Measurement and modeling of the temporal dependence in packet loss. In *Proceedings of IEEE INFOCOM'99*, pages 345–352, 1999.

[ZDPS01] Y. Zhang, N. Duffi eld, V. Paxson, and S. Shenker. On the constancy of internet path properties. In *Proceedings of ACM SIGCOMM Internet Measurement Workshop*, November 2001.