Congestion Location Detection: Methodology, Algorithm, and Performance

Shao Liu[1], Mung Chiang[1], Mathias Jourdain[2], Jin Li[3], and Phil A. Chou[3]
[1]Princeton University, [2] Microsoft Corporation, [3] Microsoft Research

***Abstract*—Can an end-host running multiple TCP sessions detect not just the occurrence, but also the location of congestion? This paper answers this question through new analytic results on the two underlying technical difficulties: synchronization effects of loss and delay in TCP and distributed hypothesis testing using only local loss and delay data, as well as practical algorithm development and extensive simulations. It presents a Congestion Location Detection algorithm that effectively allows an end host to distributedly detect whether congestion happens in the local access link or in more remote links. This further enables the practical usage of low-priority congestion control protocols.**

I. Introduction

Congestion control relies on the ability to detect the *occurrence* of congestion. What if an end host can also detect the *location* of congestion, at least to the degree of detecting whether congestion happens on a *local* access link shared only by TCP sessions from itself or on a more *remote* link? The ability to do Congestion Location Detection would provide an answer to practical questions like the following one.

A. Motivation

Consider two types of Internet applications: 1) high-priority applications that are quality of service (QoS) sensitive, such as real-time media streaming, instant messaging, web browsing, and 2) low-priority applications that are QoS insensitive, such as peer-to-peer file sharing, FTP file download, software updates, database synchronization, and file server backup. A number of proposals have been developed to emulate a low-priority service by end-to-end congestion control (e.g. TCP-Nice [1], TCP-LP [2], BATS [3], BITS [4], 4CP [5]). Some have already been used in end systems, e.g., Windows uses BATS and Linux uses TCP-LP for automatic software update. These low-priority flows give up network bandwidth when the network is congested, and thus benefit high-priority flow.

An issue that hinders the wide deployment of the end-to-end low-priority TCP protocols is that the low-priority flow gives up bandwidth whenever the network is congested, no matter where the congested link is. If the congested link is a local broadband link, e.g., a DSL or Cable Modem link, from home gateway to central office, the aggressive back off of the low-priority applications during congestion benefits the high-priority applications of the same end host. On the other hand, if the congested link is a remote link, either in the Internet core or at the server side, the back off of the low-priority applications only benefits high-priority flows competing for that link, which are most probably flows from other users. This altruism behavior is not desirable for most low-priority applications.

One way to solve the incentive issue for low-priority TCP deployment is to provide a mechanism that detects the location of the congestion, or more specifically, to determine whether the congested link is a local link shared only by all flows from a home, or a remote link where the flows from the same home constitute just a small proportion of the traffic.

Similarly, congestion location detection can allow an ISP to decide if the links next to a server is often the congestion location rather than links farther away from the server, all without using extra bandwidth for probing and measurement.

B. Challenges

Congestion location detection is a very challenging problems due to the following reasons:

- It is intractable to solve the Congestion Location Detection problem using traditional estimation and detection techniques, e.g., [6]. If we think of it as a hypothesis testing problem, we have to calculate the statistics of each individual hypothesis (which is complex by its own), and work through a gigantic number of hypotheses.
- We cannot send probing packets or rely on router supports. Sending constant stream of probing packets causes too much overhead, and is impractical in most home scenario. If we only send probing packets after the occurrence of congestion, it lead to inaccurate location detection.
- Without router support, the only congestion related signals to end applications are packet losses and delays. If packet losses were completely synchronized, i.e., all home flows passing a link see packet losses if the link is congested, then this problem would have been trivial. In reality, the packet loss pattern is partially synchronized in general [7]. There are no established results on the number of flows seeing loss when the shared link is congested.
- Packet delay cannot give sufficient information on congestion location detection, either. Packet delay measurements are buried with noise [8], and sometimes can be heavily polluted [9]. There may be extreme or oscillatory delay samples within one individual flow, and outliers among the delay statistics of all flows.

Given the enormous challenges above, we ask for less in the problem of Congestion Location Detection: can an end host use only local loss and delay information to detect if

congestions happen in a local access link shared only by TCP sessions from itself or in some more remote links? Even this detection problem is extremely challenging. Indeed, we can draw an analogy with the much more extensively studied problem of Congestion Occurrence Detection (COD), where the underlying reasoning is that events such as 3 duplicated ACK packets imply packet loss, which in turn implies the occurrence of congestion (somewhere in the network). Neither implication relationships is always true, but the resulting design of TCP has been working well enough. In our approach to the more difficult problem of Congestion Location Detection (CLD), the underlying reasoning is that similarities of loss and delay behaviors across multiple TCP sessions running at the same end host imply synchronization of congestion across the sessions, which in turn implies that congestion happens close to the end host since the sessions will share less and less common paths farther away they go from the end host. Again, neither implication relationships is always true. In fact, the "events" in CLD is much more fuzzily defined than those in COD, and the implication relationships are much harder to quantify probabilistically in CLD than those in COD.

C. Main Contributions

There are two major contributions in terms of methodology: 1) the first comprehensive study of the synchronization behavior of packet loss and delay among multiple TCP sessions, and 2) a distributed hypothesis testing theory for TCP. These are important and under-explored problems in their own right, and they together lead to the design and performance analysis of an algorithm for CLD using local packet loss and delay information. Through both analytic results and extensive simulations, we show that our CLD algorithm can effectively allow an end host to distributedly detect whether congestion happens in the local access link or in more remote links.

We first introduce the CLD algorithm in Section II. We then study the synchronization of flow loss events in Section III, study the synchronization of delay increase and describe the distributed hypothesis testing in Section IV. Using the analysis of both loss and delay, we justify the CLD parameter configuration and analyze the detection accuracy in Section V. We finally provide extensive *ns-2* simulation results to test the performance of our algorithm in Section VI, and conclude this paper in Section VII. Without explicit explanation later, all proofs of our analytical results are given in the Appendix.

II. The Congestion Location Detection Algorithm

The following key ideas will be used in the construction of our CLD algorithm, whose description will be provided in this section and development presented in Sections III and IV.

Each end host sends multiple TCP flows, e.g., from the same home. Whenever one flow sees a packet loss, we consider a congestion event occurred in the network, and trigger the CLD algorithm, which is based on the following ideas: (1) If many flows "see" congestion (as defined later this section), then the local link is the congested link. If the congested link is remote, it is very rare that many flows from this home pass the same congested remote link and see congestion synchronously, as it is unlikely for different remote links to be congested at the same time. (2) If there are only a small number of flows seeing congestion, we need to make location detection based queueing delay patterns. If the local link is congested, typically most flows will experience high delays. Furthermore, as the queueing delays of all flows are caused by the same link, queueing delay increases for most flows should be of a similar level. If the delay patterns satisfy the above conditions, we declare that the congestion is local, otherwise, we consider the congestion to be remote. (3) Delay measurements tend to be polluted and some out of normal range. These outlier samples should not be used when we compute delay statistics among all flows. If there are too many outliers, it simply means that the queuing delay increases differently among the flows, and this is an indication that the congested link is remote.

With these ideas, we now describe the (CLD) algorithm in details. Using both loss and delay, CLD periodically queries loss and delay information of all TCP flows from the transport layer. This query can be done by kernel level modifications, or can be obtained through the existing TCP-E-STATS-MIB (Extended Statistics Management Information Base) module [10]. The loss and delay query is straight forward at senders, and at receiver side, the mechanism in [11] is used. The CLD algorithm updates the total number of loss events, denoted by L, of all flows, and the moving average of queueing delay over the last query period, denoted by q, of all flows. By default, one query period lasts 0.1 second, and a longer query period is also allowed. It is possible that one congestion event spans over multiple consecutive query periods. If that occurs, CLD will combine the statistics over all these consecutive query periods and generate one single set of loss and delay counters. Therefore, the more frequent CLD queries, the more accurate and quick the detection. It is also possible that a query period is much longer than the duration of a congestion event, and packet loss occurs at the very beginning of this query period. If that occurs, most delay samples are taken after the congestion is alleviated, and the moving average q is much less than the actual queueing delay caused by this congestion event. To avoid such underestimation, once we see loss event spanning over only one query period, we compare the average delays of the current and previous query periods, and set the delay counter to be the maximum of the two. Suppose one home has altogether N flows, indexed by $i = 1, 2, \cdots, N$, we add subscript i to each variable we introduced above, e.g., L_i, q_i, etc. If none of the flows sees an increase in its L_i value compared with the previous query, then there is no congestion in the network during the last period. Otherwise, congestion occurs during the last (one or many) query period, and we use the following three modules or steps to detect congestion location.

The flow chart and the pseudo code of the CLD algorithm described below are shown in Figure 1 and Figure 2; and the parameters, counters and state variables are summarized in Table I and Table II.

1) Quick Detection (QD) Module: For each flow i, we say it "sees" congestion if either it experiences packet losses, or q_i is larger than a threshold. As shown in Table II, we maintain four state variables, $\hat{\mu}_{LC}, \hat{\mu}_{RC}, \hat{\sigma}_{LC}$ and $\hat{\sigma}_{RC}$, which

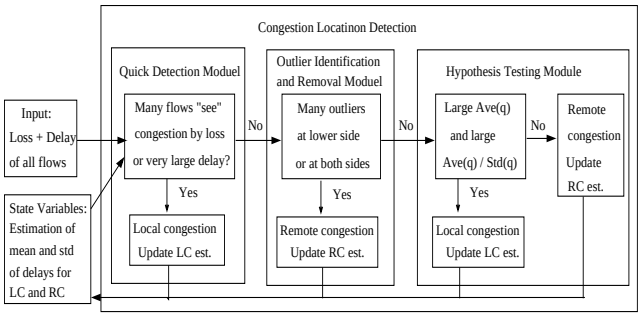

Fig. 1. **Flow chart of the CLD algorithm**.

Parameter	κ_{QD}	κ_{low}	κ_{both}	β	g
Default Value	0.8	0.2	0.3	0.9	2/3/5
Module of usage	QD	OIR	OIR	QD	OIR
Section of discussions	IV-C/D	IV-B	IV-B	V	IV-B

TABLE I
SUMMARY OF CLD PARAMETER SETTINGS.

are estimations of the mean and standard deviation of delays among N flows at one congestion event, for local and remote congestions. The detailed estimation mechanism is explained in Section IV-C. With these estimations, we set the threshold to be $\beta \times \hat{\mu}_{LC}$, where β is a parameter with default value of 0.9. If more than $\kappa_L N$ flows "see" congestion, where κ_L is a parameter with default value of 0.8, we detect local congestion directly, update $\hat{\mu}_{LC}$ and $\hat{\sigma}_{LC}$, and the algorithm ends. Otherwise, we enter the next module.

2) Outlier Identification and Removal (OIR) Module: We use Hampel's identifier to identify and remove outliers. For details of the identifier, see Section IV-B. If the number of the two-side outliers exceeds $\kappa_{both}N$, where κ_{both} is a parameter with default value of 0.3, then the delay samples are too diversified, and we consider the congestion to be remote. Furthermore, if it is a local congestion, most delays should have a large queueing delay, i.e., there should be no or very few lower side outliers. If the number of lower side outliers exceed $\kappa_{low}N$, where $\kappa_{low} = 0.2$ is a parameter, then there are too many flows with low delay for the congestion to be local. For either case, we detect remote congestion, update $\hat{\mu}_{RC}$ and $\hat{\sigma}_{RC}$, and the algorithm ends. Otherwise, we enter the next module.

3) Hypothesis Testing (HT) Module: We compute the mean and standard deviation of the inlier samples, and denote them by μ_q and σ_q, respectively. From hypothesis testing analysis in Section IV-C, we check whether $\mu_q > \alpha_1$ and $\mu_q/\sigma_q > \alpha_2\sqrt{(N-1)/N}$, where α_1 and α_2 are two thresholds that are functions of $\hat{\mu}_{LC}, \hat{\mu}_{RC}, \hat{\sigma}_{LC}$ and $\hat{\sigma}_{RC}$. For detailed relationship, see equation (23) or Figure 2. If both conditions satisfy, then the queueing delays of all inliers are sufficiently large and close, which is a pattern of local congestion, and we detect local congestion, update $\hat{\mu}_{LC}$ and $\hat{\sigma}_{LC}$. Otherwise, we detect remote congestion, and update $\hat{\mu}_{RC}$ and $\hat{\sigma}_{RC}$.

	Queried Counters		Estimated Statistics/State Variables			
Name	L_i	q_i	$\hat{\mu}_{LC}$	$\hat{\mu}_{RC}$	$\hat{\sigma}_{LC}$	$\hat{\sigma}_{RC}$
Meaning	Loss	Delay	Estimation of mean and std of $\{q_i, \forall i\}$ for local and remote congestion			
Module	QD	ALL	QD+HT	HT	HT	HT
Section of Discussion	III	IV	IV-C			

TABLE II
SUMMARY OF COUNTERS AND STATE VARIABLES OF CLD ALGORITHM.

Congestion Location Detection Algorithm

Parameters: $\kappa_{QD} = 0.8$, $\kappa_{low} = 0.2$, $\kappa_{both} = 0.3$, $\beta = 0.9$, $g = 5$ if $N \leq 5$, $g = 3$ if $5 < N \leq 10$, and $g = 2$ otherwise.

Input: ΔL_i, q_i, $\forall i = 1, 2, \cdots N$.

State Variables: $\hat{\mu}_{LC}, \hat{\mu}_{RC}, \hat{\sigma}_{LC}, \hat{\sigma}_{RC}$.

if $\Delta L_i = 0, \forall i = 1, 2, \cdots, N$
  Output: "No Congestion".
  goto :END
end if
if $N = 1$
  Output: "Congestion, But No Location Detection".
  goto :END
end if
Enter Quick Detection Module:
$Count = \sum_{i=1}^{N} \mathbb{1}_{(\Delta L_i > 0 \text{ or } q_i > \beta\hat{\mu}_{LC})}$
if $Count > \kappa_{QD} N$
  Output: "Local Congestion". Update $\hat{\mu}_{LC}$ and $\hat{\sigma}_{LC}$.
  goto :END
end if
Enter Outlier Identifier Module:
$med \leftarrow median(q_i, i \in 1, \cdots N)$
$mad \leftarrow median(|q_i - med|, i \in 1, \cdots N)/0.6745$
$O_L \leftarrow \sum_{i=1}^{N} \mathbb{1}_{q_i < med - g \times mad}$
$O_U \leftarrow \sum_{i=1}^{N} \mathbb{1}_{q_i > med + g \times mad}$
if $O_L > \kappa_{low} N$ or $O_L + O_U > \kappa_{both} N$
  Output: "Remote Congestion". Update $\hat{\mu}_{RC}$ and $\hat{\sigma}_{RC}$.
  goto :END
end if
Enter Hypothesis Testing Module:
$\mu_q \leftarrow mean(q_i : q_i \in [med - g \times mad, med + g \times mad])$
$\sigma_q \leftarrow std(q_i : q_i \in [med - g \times mad, med + g \times mad])$
$\alpha_1 \leftarrow \frac{1}{2}(\hat{\mu}_{LC} + \hat{\mu}_{RC})$ and $\alpha_2 \leftarrow \frac{1}{2}\left(\frac{\hat{\mu}_{LC}}{\hat{\sigma}_{LC}} + \frac{\hat{\mu}_{RC}}{\hat{\sigma}_{RC}}\right)$
if $\mu_q > \alpha_1$ and $\mu_q/\sigma_q > \alpha_2$
  Output: "Local Congestion". Update $\hat{\mu}_{LC}$ and $\hat{\sigma}_{LC}$.
else
  Output: "Remote Congestion". Update $\hat{\mu}_{RC}$ and $\hat{\sigma}_{RC}$.
end if
:END

Fig. 2. Pseudo code for the Congestion Location Detection Algorithm. The mechanism of updating $\hat{\mu}_{LC}$, $\hat{\sigma}_{LC}$, $\hat{\mu}_{RC}$ and $\hat{\sigma}_{RC}$ is explained in Section IV-C.

III. SYNCHRONIZATION OF PACKET LOSS PATTERN

From this section to Section V, we justify our algorithm, study the parameter configuration, and analyze the accuracy of detection. We focus on packet loss in this section, study

delay in Section IV, and derive the detection accuracy and parameter configuration guideline in Section V. The main analytical results are summarized in Table III.

Packet loss is used in the Quick Detection Module. If most flows see congestion either by packet loss or very large delay, local congestion is detected. To study the parameter setting of κ_{QD} and analyze the false local detection probabilities, we need to answer the following problem on the synchronization level of packet losses: Assume a link shared by N TCP flows is congested, and let H be the number of flows that experience a loss event (one or more packet losses from this congestion), what is the statistics of H?

Suppose during one congestion, altogether M packets are dropped at the link. To derive the statistics of H, we must know the statistics of M first. Therefore, the synchronization problem is divided to the following two subproblems: 1) What is the statistics of M? and 2) what is the statistics of H after we know the statistics of M? We study the two subproblems one by one.

A. Total Number of Dropped Packets (M)

1) Homogeneous RTT Users: Consider a link with buffer size B and capacity C, shared by N users with homogeneous RTT $T = D+q$, where D and q are propagation and queueing delays, respectively. Suppose flow i has quantized integer value window size W_i, and let $S := \sum_{i=1}^{N} W_i$ be the sum of all window sizes, then totally there are S packets outgoing in the network pipe, either along the network path, or in the buffer of the bottleneck link router. Consider a congestion event, which starts when the queue becomes full and the next arrival packet has to be dropped, and ends when one flow sees packet losses, backs off its window size, and the total arrival rate at the link drops to below its capacity. We call the duration of a congestion event a *congestion alleviation period*, denoted by T_a. We further denote by W_i^- and W_i^+ the window sizes of flow i right before and after the congestion event, and correspondingly, we have S^- and S^+. Note that T_a is the sum of the time it takes for the loss propagate to the sender, and the time it takes for the effect of throughput reduction to propagate to the router. Therefore, T_a does not depend on the split of propagation delay between forward and backward delays, or the location of the router; it only depends on D, q, and the queueing policy. For simplicity of analysis, we let the forward delay be zero and the backward delay equal to D, as depicted in the left figure of Figure 3. The result derived from this simplified model also holds for general cases. We first have the follow result for the simplest homogeneous RTT, Droptail, and randomness free case:

Proposition III.1. *If a Droptail router is shared by N flows with identical RTT, and if there is no randomness caused by bursty packet arrival process or background traffic, then $M = N$.*

This result explains the global synchronization phenomena, which states that if N flows have identical RTTs and if the network has no randomness, then all or most flows experience packet losses and back off window sizes synchronously [12],

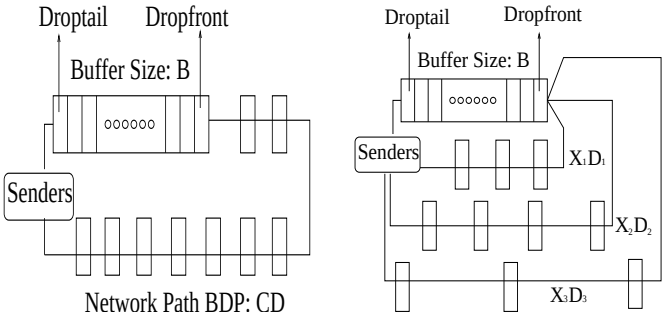

Fig. 3. **Homogeneous RTT users (left) Vs heterogeneous RTT users (right)**.

[13]. However, it is widely observed that global synchronization is actually very rare in reality [14], since in real-world networks, RTTs of all flows are not identical (even if users are considered to have homogeneous RTT, they usually have slight RTT differences), the packet arrival process is random, and typically there are background traffics in reality [15]. For those realistic cases, M is no longer a constant, and global synchronization is avoided. Furthermore, if "drop from front" option (Dropfront) [16] is selected, M is not a constant either. All these analysis are validated by extensive *ns-2 simulations* in Section VI.

We have performed *ns-2* simulations showing that M trajectory is constant for simplest case, but even with small background traffic or slight RTT differences, randomness is brought into the system and M shows fluctuations. Furthermore, if "drop from front" option (Dropfront) [16] is selected, M is not a constant either. All these results are demonstrated in Figure 4.

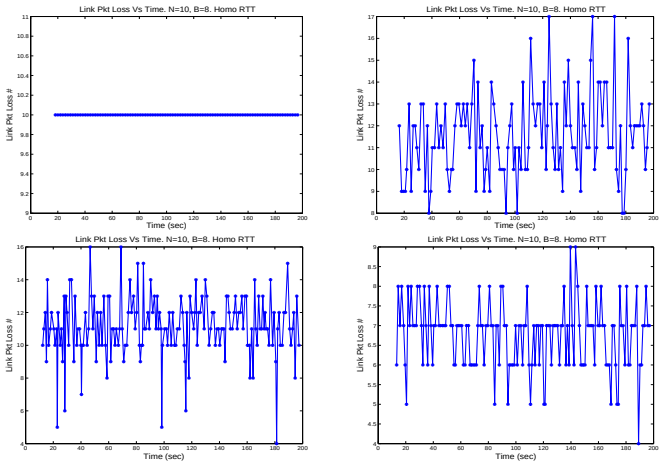

Fig. 4. **Global synchronization and its avoidance** Top left: N identical RTT TCP flows pass a Droptail queue without background traffic, where $N = 10$, propagation delay $D = 60$, capacity $C = 20$ Mbps, and buffer size $B = 160$ packets. Top right: same as top left, but with background traffic: we add $N = 10$ Pareto ON-OFF UDP flows, each with a rate of 100 kbps. Left bottom: same as top left, but with slight RTT differences: $D_i = 60 + (i-1) \times 0.4, \forall i = 1, 2, \cdots, 10$. Right bottom: same as top left, but use Dropfront queue. From this figure, M is constant in the radical case, but becomes a variable if there is background traffic, or if the RTTs are not exactly the same, or Dropfront queue is used.

With randomness brought in, we have the following result:

Proposition III.2. *If a link shared by N homogeneous RTT*

users is congested, then M, the number of packet dropped at this congestion event, is a random variable, and $E[M] = \eta N$, where $\eta = 1$ for Droptail, $\eta = D/T = D/(D+q) = CD/(CD+B)$ for Dropfront. Furthermore, $Std(M)$ is also proportional to N.

The reason that Dropfront queue reduces $E[M]$ is that, with a packet at the front of queue dropped, its sender realizes the congestion occurrence earlier than if the packet at the end of queue is dropped, and the congestion event lasts a shorter time. Consider the congestion alleviation period T_a: $E[T_a] = T$ for Droptail, but $E[T_a] = D$ for Dropfront, since the propagation of loss does not need to wait at the queueing delay. This advantage of Dropfront has been qualitatively stated in [16], but not quantitatively studied before.

2) Heterogeneous RTT Users: We next consider heterogeneous users case, where user i has propagation delay D_i and RTT T_i, as shown in the right figure of Figure 3. Suppose there are B_i packets in the buffer from flow i, and flow i has throughput x_i, then $W_i = x_i D_i + B_i$. During each congestion event, we have $\sum_{i=1}^{N} x_i = C$, $\sum_{i=1}^{N} B_i = B$, $q = B/C = B_i/x_i$, and $T_i = D_i + q, \forall i$.

The key difference between heterogeneous and homogeneous RTT flows is that, a congestion event lasts a fixed duration for homogeneous RTT case, but has a variable length for heterogeneous RTT case: the length could be any value between the minimum and maximum of all RTTs (or propagation delays for Dropfront), depending on the packet loss pattern. Consider a simple example of two flows: flow 1 has 10 milliseconds (ms) delay (RTT for Droptail and propagation delay for Dropfront) and flow 2 has 12 ms delay. When congestion occurs, multiple packets are dropped. If the first dropped packet belongs to flow 1, then $T_a = 10$ ms. However, if the first dropped packet belongs to flow 2, then there are two possibilities: if there is a flow 1 packet dropped within 2 ms after the first packet drop (say, after $\psi < 2$ ms), then $T_a = 10 + \psi$ ms, as the effect of throughput reduction from flow 1 propagate to the link first; on the other hand, if there is no flow 1 packet loss, or all flow 1 packet drops occur more than 2 ms later than the first flow 2 packet drop, then $T_a = 12$ ms. Therefore, T_a could be any value between 10 ms and 12 ms, and the distribution of T_a is very complicated, depending on the modeling of packet arrival process and the quantization of window size of each flow. We make the following assumption on the packet loss pattern [7]:

Assumption 1. *The probability of each dropped packet belongs to flow i, is $B_i/B = x_i/C$.*

This assumption comes from the following reasoning: the probability that a random packet in the queue or a random incoming packet belongs to flow i is $B_i/B = x_i/C$, so is the probability for the dropped packet. From Assumption 1, we can prove the following result:

Proposition III.3. *For heterogeneous RTT case, still $E[M] = N$ for Droptail queue, and for Dropfront queue,*

$$\eta_{min} N \le E[M] \le \eta_{max} N\,, \qquad (1)$$

where

$$\eta_{min} := \min_i D_i/T_i \text{ and } \eta_{max} := \max_i D_i/T_i\,. \qquad (2)$$

Furthermore, $Std(M)$ is also proportional to N, and $Std(M)$ for heterogeneous RTT case is much larger than that for homogeneous RTT case.

The reason that heterogeneous RTT has higher variance than homogeneous is that the distribution of M depends on the distribution of T_a, while T_a has high variance for heterogeneous RTT case than homogeneous RTT case.

From Proposition III.2 and III.3, both $E[M]$ and $Std(M)$ are always proportional to N, no matter homogeneous or heterogeneous RTT, no matter Droptail or Dropfront queue, and thus we can always write

$$E[M] = \eta N \quad \text{and} \quad Std(M) = \gamma N\,, \qquad (3)$$

where $\eta = 1$ for Droptail queue, $\eta = D/T$ for homogeneous RTT with Dropfront, $\eta \in [\eta_{min}, \eta_{max}]$ for heterogeneous RTT with Dropfront, and η_{min} and η_{max} are defined in (2). As for γ, its value cannot be analytically derived, but can be empirically studied through simulations, and we know that $\gamma \ll 1$ for homogeneous RTT case and $\gamma \lessapprox 1$ for heterogeneous RTT case.

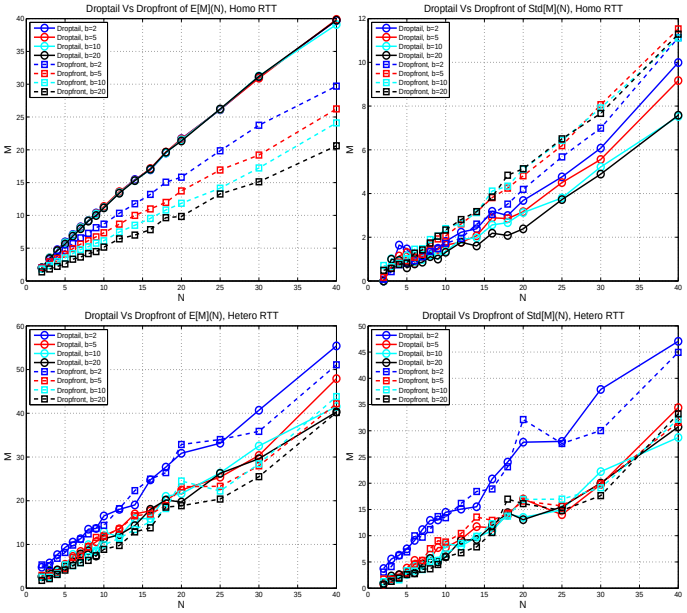

Fig. 5. $E[M]$ **and** $Std(M)$ **as functions of** N, B**.** Top row: homogeneous RTT. Bottom row: heterogeneous RTT. Left column: $Ave(M)$. Right column $Std(M)$. For all simulations, we vary N from 2 to 40, set $C = 2N$ Mbps, set $B = 2bN$ packets and vary b from 2 to 20, set $D_1 = 60$ Mbps, and set $D_N = 140$ ms for heterogeneous RTT case.

B. Number of Flows Seeing Loss (H)

We now study the first and second order statistics of H by first considering the conditional statistics of H given M, then extending to unconditional statistics of H.

Proposition	III.2	III.3	III.4	III.5	IV.1	V.1
Section of Discussions	III-A1	III-A2	III-B1	III-B2	IV-C	V
Results on	$E[M]$ and $Std(M)$		$E[H\|M]$ and $Std(H\|M)$	$E[H]$ and $Std(H)$	Hypothesis Testing	Detection Accuracy
Major Equations	(3)		(5-7)	(10-14)	(22)	

TABLE III
SUMMARY OF ALL ANALYTICAL RESULTS.

	Homogeneous RTT, Droptail	Homogenous RTT, Dropfront	Heterogenous RTT, Droptail	Heterogenous RTT, Dropfront
$E[M]$	N	$\frac{D}{T}N$	N	ηN, where $\min_i \frac{D_i}{T_i} \le \eta \le \max_i \frac{D_i}{T_i}$
$Std(M)$	γN, where $\gamma \ll 1$	γN, where $\gamma \ll 1$	γN, where $\gamma \lessapprox 1$	γN, where $\gamma \lessapprox 1$
$E[H]$	$0.21N \le E[H] \le 0.632N$	see right, with $\alpha = 1$	see right, with $\eta = 1$	$N(1-e^{\eta/(2\alpha)} - \frac{e^{-\eta}}{2}) \le E[H] \le N(1-e^{\eta})$
$\frac{Std(H)}{Std(M)}$	$\gtrapprox 0.271$	$\gtrapprox \min(\frac{e^{-\frac{\eta}{2}}}{2}, 2e^{-2\eta})$	see right, with $\eta = 1$	$\gtrapprox \min(\frac{1}{2\alpha_{max}} e^{-\frac{\eta}{2\alpha_{max}}}, \frac{2}{\alpha_{min}} e^{-\frac{2\eta}{\alpha_{min}}})$

TABLE IV
SUMMARY OF RESULTS ON M AND H.

1) Conditional statistics of H given M: From Assumption 1, we know that, at one particular congestion event, the probability that each dropped packet belonging to flow i is λ_i, where $\lambda_i = B_i/B = x_i/C$ and $\sum_{i=1}^N \lambda_i = 1$. Define $A_i = 1$ if flow i sees packet losses, and $A_i = 0$ otherwise, then we have $H = \sum_{i=1}^N A_i$. Given $M = m$, we have

$$A_i = \begin{cases} 1 & \text{w.p.} \quad 1-(1-\lambda_i)^m \,, \\ 0 & \text{w.p.} \quad (1-\lambda_i)^m \,. \end{cases}$$

Let $f(m) := E[H|H=m]$ and $g(m) := Var(H|M=m)$, then it is easy to see that $f(m) = \sum_{i=1}^N P(A_i = 1)$. For variance, it is not that straight-forward: since $\sum_{i=1}^N A_i \ge 1$, A_i's are not independent, and thus $g(m)$ depends on $Cov(A_i, A_j), \forall i \ne j$. However, as N and m become large, the probability that $A_i = 0, \forall i$ is very small even if we assume that they are independent. Therefore, we can ignore the correlation of A_i's, and assume $g(m) \approx \sum_{i=1}^N Var(A_i)$. We further define

$$\begin{cases} \xi_{min} & := & N \min_i \lambda_i & = & N \min_i x_i/C \,, \\ \xi_{max} & := & N \max_i \lambda_i & = & N \max_i x_i/C \,. \end{cases} \tag{4}$$

We have the following result:

Proposition III.4. *For the conditional expectation, we have*

$$f(m) = \sum_{i=1}^N (1-(1-\lambda_i)^m) \,, \tag{5}$$

and

$$N(1-(1-\frac{\xi_{min}}{N})^m) \le f(m) \le N(1-(1-\frac{1}{N})^m) \,. \tag{6}$$

Furthermore, for conditional variance, we have

$$g(m) \approx \sum_{i=1}^N (1-\lambda_i)^m (1-(1-\lambda_i)^m) \,. \tag{7}$$

2) Unconditional Statistics of H: For unconditional statistics, we have the following equations:

$$\begin{cases} E[H] & = & E[f(M)] \,, \\ Var(H) & = & Var(E[H|M]) + E[Var(H|M)] \\ & = & Var(f(M)) + E[g(M)] \,. \end{cases} \tag{8}$$

From Jensen's inequality, $E[f(M)] \le f(E[M])$, since $f(m)$ is concave over m. As $E[f(M)]$, $Var(f(M))$ and $E[g(M)]$ appear in (8), and their exact formula are unknown, we choose the Taylor expansions for the moments of functions of random variables [17]. Then, we have the following approximations:

$$\begin{cases} E[f(M)] & \approx & f(E[M]) + \frac{f''(E[M])}{2} Var(M) \,, \\ E[g(M)] & \approx & g(E[M]) + \frac{g''(E[M])}{2} Var(M) \,, \\ Var(f(M)) & \approx & (f'(E[M]))^2 Var(M) \,. \end{cases} \tag{9}$$

Plugging the results of Proposition III.2, Proposition III.3 and Proposition III.4 to (8) and (9), we have the following result:

Proposition III.5. *The unconditional expectation of H has the following bounds:*

$$N(1-e^{-\eta\xi_{min}} - \gamma^2 \frac{e^{-\eta}}{2}) \le E[H] \le N(1-e^{-\eta}) \,, \tag{10}$$

where η and γ are defined in (3), and ξ_{min} is defined in (4). Numerically, for all case, we have the following upper bound:

$$E[H] \le N(1-e^{-1}) \approx 0.632N \,. \tag{11}$$

and for the special homogeneous RTT flows and Droptail queue case, we have the following lower bound:

$$E[H] \ge 0.21N \approx N(1-e^{-\frac{1}{2}} - \frac{e^{-1}}{2}) \tag{12}$$

Furthermore, for the standard deviation of H,

$$\frac{Std(H)}{Std(M)} \gtrapprox \min(\xi_{min} e^{-\eta\xi_{min}}, \xi_{max} e^{-\eta\xi_{max}}) \,, \tag{13}$$

where ξ_{min} and ξ_{max} are defined in (4). For the special case of homogeneous RTT flows and Droptail queue, we have the following numerical results:

$$\begin{array}{rcl} Std(H) & \gtrapprox & \min(\frac{e^{-\frac{1}{2}}}{2}, 2e^{-2}) Std(M) \\ & \approx & 0.271 Std(M) \approx 0.271 \cdot \gamma \cdot N \,. \end{array} \tag{14}$$

The results on M and H from all above analysis are summarized in Table IV, and α_{min} and α_{max} in Table IV are defined as

$$\alpha_{max} := \frac{1}{N} \sum_{j=1}^N \frac{T_{max}}{T_j} \quad \text{and} \quad \alpha_{min} := \frac{1}{N} \sum_{j=1}^N \frac{T_{min}}{T_j} \,. \tag{15}$$

Note that the most important implications of M and H analysis are: 1) $E[H]$ is proportional to N, which means that we can get a rough idea on N if we do not know N but can observe H; and 2) $Std(H)$ is also proportional to N, and thus $Std(H)/E[H]$ does not diminish as $N \to \infty$, which means that if we only use loss to detect congestion location, the false detection probability does not go to 0 as $N \to \infty$.

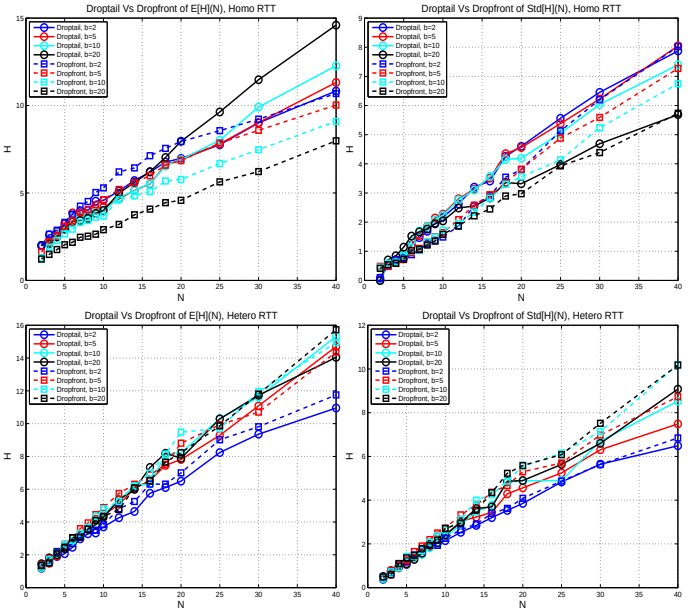

Fig. 6. $E[H]$ **and** $Std(H)$ **as functions of** N **and** B. Top row: homogeneous RTT users. Bottom row: heterogeneous RTT users. Left column: $E[M]$. Right column: $Std(M)$. Simulation setups are the same as those in Figure 5

C. Simulation Validations on M and H Analysis

We have performed extensive *ns-2* simulations to validate our analysis of M and H, for both homogeneous and heterogeneous RTT users, for both Droptail and Dropfront queue. For our simulations, we pick a dumbbell network topology, vary N from 2 to 50, vary C from 5 to 200 Mbps, vary B from 10 to 1000 packets, vary background traffic volume, and test both Droptail and Dropfront. For homogeneous RTT case, we choose slightly different RTTs to introduce randomness by setting $D_i = D_1 + (i-1) \times 0.4$, and vary D_1 from 60 ms to 220 ms. For heterogeneous RTT case, we set $D_i = D_1 + (i-1) \times D_N/(N-1)$, and vary the ratio between D_N/D_1 from 2 to 5. For each simulation corresponding to a particular combination of N, C, B, D_1, D_N, background traffic volume, and queueing policy, we run 400 seconds, and for each congestion event, we measure M at the router and measure H at the senders. After each simulation finishes, we compute $Ave(M)$, $Std(M)$, $Ave(H)$ and $Std(H)$ over all congestion events. Due to space limitation, we cannot show all the simulation results, and for some examples, we plot M in Figure 5 and H in Figure 6. The simulation results in these figures validate our analysis on M and H: 1) $E[M]$, $Std(M)$, $E[H]$, and $Std(H)$ are all proportional to N; 2) For Droptail queue, $E[M]/N \approx 1$, and for Dropfront queue, $E[M]/N < 1$, is a decreasing function of buffer size B (large B means small D/T ratio), and is less dependent on B as D becomes large; 3) $E[H]/N$ is between 0.632 and 0.21; 4) $Std(M)$ for heterogeneous RTT is much larger than that of homogeneous RTT case; and 5) the ratio of $Std(H)/Std(M)$ is larger than 0.271, and this ratio is smaller for heterogeneous RTT than that for homogeneous RTT. We further add background traffics which are Pareto On-OFF UDP flows, and vary their volumes. The result is plotted in Figure 7, which shows that the result on M and H are independent of background traffics.

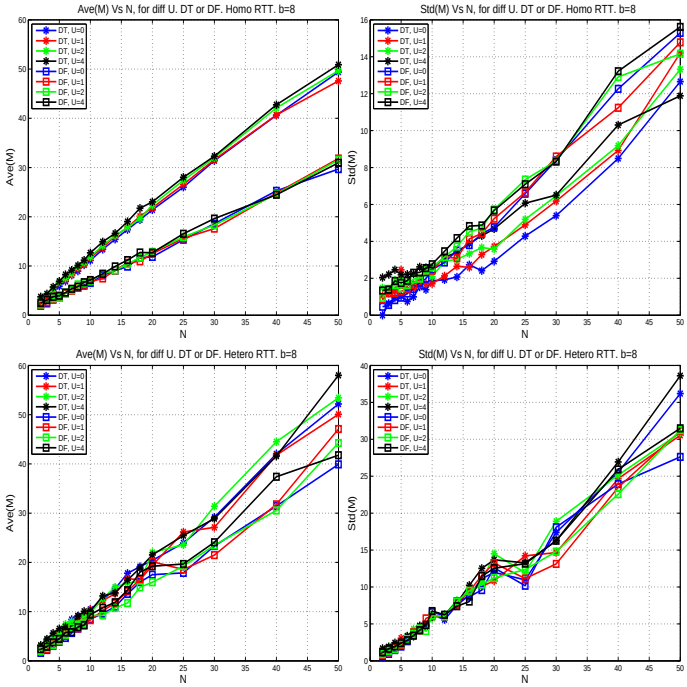

Fig. 7. $E[M]$ **and** $Std(M)$ **as functions of** N **and background traffic.** Top row: homogeneous RTT. Bottom row: heterogeneous RTT. Left column: $E[M]$. Right column: $Std(M)$. For all simulations, $D_1 = 60$ ms, $C = 2N$ Mbps, $B = 16N$ packets.

IV. Synchronization of Delay Increase

A. Modeling on Queueing Delays

Traditional queueing theory results cannot be directly applied to TCP flows, as feedbacks play an important role and the packet arrival process of TCP flows is unknown. There has been several modeling work [18]–[20] and measurement studies [8], [9], [21], [22] on TCP RTTs, but there is no widely acceptable modeling for TCP RTTs from the measurement studies, and the following properties of TCP RTTs are observed: 1) the variation of TCP RTTs can be very large, where the variation can be either inter-session, or intra-session; 2) there might be extreme values of RTT samples, which have abnormally large RTTs. Therefore, we model the measured queueing delay to be sum of the real queueing delay and a noise term. As the instant noisy delay samples fluctuate significantly, a common technique is to compute the moving average of sample delays over a certain period, like one RTT or one query period, and call this average value the current delay. Assume that the distributions of delays within one moving window do not change and the queueing delay

sequence is Martingale, and assume that the moving window contains sufficiently large number of samples, from Martingale Central Limit Theorem, we can model the current delays to be Gaussian random variables:

$$q_i = n_i + s_i \,, \tag{16}$$

where $n_i \sim \mathcal{N}(\mu_n, \sigma_n)$ is a Gaussian noise, n_i's of all flows are i.i.d., and s_i is the real queueing delay signal, which is zero if the flow does not experience congestion, and $s_i \sim \mathcal{N}(\mu_i, \sigma_i)$ otherwise.

If one delay sample is abnormally large, there is a third term beyond real queueing delay and Gaussian noise:

$$q_i = n_i + s_i + \omega_i \,, \tag{17}$$

where ω_i is a random variable whose expectation is much larger than μ_i and μ_n. We assume that the extreme values are very rare, and at one query periods, at most a small proportion of flows yield extreme values.

We now study how to use $\{q_i, \forall i\}$ to detection congestion location. Assume all q_i's have the same distribution, congestion location detection becomes a hypothesis testing problem. As we will show in Section IV-C, we can use the sample mean and standard deviation to do hypothesis testing. However, q_i's are not of the same distribution in general: if the congestion is remote, some flows have $s_i \sim \mathcal{N}(\mu_i, \sigma_i)$ and some flows have $s_i = 0$; even if the congestion is local, there might be extreme values as in (17). We model the sample set to consist of inliers and outliers, where the inliers have the same distribution and the outliers have different distributions. With this model, we divide the location detection using delay into two modules: Outlier Identification and Removal Module and Hypothesis Testing Module.

B. Outlier Identification and Removal

There has been a lot of algorithms for outlier identification and removal, like Chauvenet's criterion [23], Grubbs' test [24], Peirce's criterion [25], Hampel's identifier [26], [27], etc. Some use functions of sample mean and variance as break down points, and some use functions of quartiles, medians, and median absolute deviations (MAD) as break down points. Major metrics for outlier identifer are *masking* (miss) and *swamping* (false positive) probabilities [27]. It is widely known that identifiers based on sample mean and variance are more prone to the masking effect than those based on medians, MADs, and quartiles, as extreme outlier samples alter the sample mean and standard deviation significantly [27]. Therefore, we pick Hampel's identifier and modify it to accommodate our special case that "swamping" effect is not bad as long as the number of false positives is not too large.

1) Hampel's Identifier: We now briefly introduce Hampel's Identifer. Suppose we have N samples, where $N - k$ samples are regular observations (inliers) with common distribution $\mathcal{N}(\mu, \sigma)$, and k non-regular observations (outliers). Neither μ, σ nor k is known. Denote the samples by $\mathbf{X}_N = (X_1, \cdots, X_N)$, denote by $med(\mathbf{X}_N)$ the median of $\mathbf{X}_N$, and let $mad(\mathbf{X}_N)$ be the normalized median of the absolute sample deviation from the median:

$$mad(\mathbf{X}_N) := \frac{1}{0.6745} median(|X_i - med(\mathbf{X}_N)|, \forall i) \,,$$

where $1/0.6745$ is a renormalization factor to make the MAD value Fisher consistent, which means that $med(\mathbf{X}_N)$ and $mad(\mathbf{X}_N)$ are unbiased estimations of μ and σ [26]. Using median and MAD to estimate μ and σ is less sensitive to extreme outlier values than using sample mean and standard deviation. Hampel's identifier states that: X_i is an outlier if

$$|X_i - med(\mathbf{X}_N)| \geq g(N, \alpha_N) mad(\mathbf{X}_N) \,, \tag{18}$$

where $g(N, \alpha_N)$ is a critical value that is picked such that even if all the samples are inliers, the swamping probability is upper bounded by $\alpha = 1 - (1 - \alpha_N)^N$. Here, α typically picks values like 0.01, 0.05 or 0.1. The critical value $g(N, \alpha_N)$ can be analytically derived for some special values of N and α, or computed by Monte Carlo method in general [26], [27].

2) Our Modification: The standard Hampel's identifier picks the critical value to upper bound the swamping probability. For our problem, we actually allow the existence of swamps (false outlier detection), as long as the number of swamps is upper bounded. Recall our detection rule: we detect remote congestion only if the number of two-side outliers exceeds $\kappa_{both} N$, or the number of lower-side outliers exceeds $\kappa_{low} N$. This suggests that we should pick a different critical value:

$$g(N, \kappa_{both}, \kappa_{low}, \alpha) = \max(g_1(N, \kappa_{both}, \alpha), g_2(N, \kappa_{low}, \alpha)) \,,$$

where $g_1(\cdot)$ and $g_2(\cdot)$ are such that:

$$\begin{cases} P\left((\sum_{i=1}^N \mathbb{1}_{|q_i - med| > g_1(N, \kappa_{both}, \alpha) mad}) > \kappa_{both} N \right) < \alpha \,, \\ P\left((\sum_{i=1}^N \mathbb{1}_{q_i < med - g_2(N, \kappa_{low}, \alpha) mad}) > \kappa_{low} N \right) < \alpha \,. \end{cases} \tag{19}$$

We call the two probabilities in (19) "strong two-side swamping probability" and "strong lower-side swamping probability".

We have used Monte Carlo method to find $g_1(N, \kappa_{both}, \alpha)$ and $g_2(N, \kappa_{low}, \alpha)$ curve for $\kappa_{both} = 0.3$, $\kappa_{low} = 0.2$, and $\alpha = 0.01$, 0.05, or 0.1. The results are shown in the left plot of Figure 8. From the plot, we see that if we set $g = 5$ if $N \leq 5$, $g = 3$ if $5 < N \leq 10$, and $g = 2$ otherwise, then none of the two strong swamping probabilities exceed 0.05. We further verify that conclusion by fixing g value by the above rule, and use Monte Carlo method to find the two strong swamping probabilities. The results are shown in the right plot of Figure 8. From the analysis and the Monte Carlo results, if the congested link is local, the false remote probability is upper bounded by 0.05 for small N values and is negligible for large N values.

C. Hypothesis Testing with Delay Statistics

We consider the outcome after the OIR Module. For local congestion, extreme values are removed as outliers, and the inliers have identical distribution $\mathcal{N}(\mu_{LC}, \sigma_{LC})$. For remote congestion, suppose N_c flows are congested. If $\kappa_{both} N <$

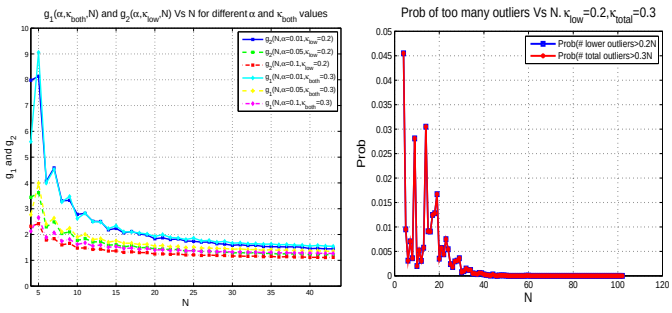

Fig. 8. $g(N, \kappa_{both}, \kappa_{low}, \alpha)$ **and** $P($**Number Outliers** $> \kappa N)$ **versus** N. Left: $g_1(N, \kappa_{both}, \alpha)$ and $g_2(N, \kappa_{low}, \alpha)$ versus N, for $\alpha = 0.01$, 0.05, or $\alpha = 0.1$, with $\kappa_{both} = 0.3$ or $\kappa_{low} = 0.2$. Right: The probability that the number of lower outliers exceeds $\kappa_{low}N$ and the probability that the number of both side outliers exceeds $\kappa_{both}N$ versus N, for fixed g choice: $g = 5$ if $N \leq 5$, $g = 3$ if $5 < N \leq 10$ and $g = 2$ otherwise.

$N_c < (1 - \kappa_{low})N$, then remote congestion is detected due to many outliers. If $N_c > \kappa_{QD}N$, or $N_c > (1 - \kappa_{low})N$, then there are too many congested flows to distinguish remote from local congestion: either it is detected as local congestion at the QD module, or the non-congested samples are removed as outliers, and the inliers yield a local congestion pattern. If $N_c < \kappa_{both}N$, then the large delay samples are removed, and the inliers have identical distribution $\mathcal{N}(\mu_n, \sigma_n)$. Therefore, after the OIR Module, if $N_c < \kappa_{both}N$, we translate the location detection to the following two hypotheses testing problem:

$$\begin{array}{ll} H_0: & q_i \sim \mathcal{N}(\mu_n, \sigma_n)\,, \forall i \in I\,, \\ H_1: & q_i \sim \mathcal{N}(\mu_{LC}, \sigma_{LC})\,, \forall i \in I\,, \end{array} \tag{20}$$

where I is the set of inliers, H_0 represents remote congestion, and H_1 represents local congestion.

This two hypotheses testing is a composite problem, as both $\theta_n := (\mu_n, \sigma_n)$ and $\theta_{LC} := (\mu_{LC}, \sigma_{LC})$ are unknown. We need to identify the parameter regions Γ_0 and Γ_1 under the two hypothesis. We make the following assumption:

Assumption 2. *$\mu_{LC} \gg \mu_n$, $\sigma_n \sim \mu_n$, and $\sigma_{LC} \ll \mu_{LC}$.*

From Assumption 2, we have the following modeling on the parameter region:

$$\begin{cases} \Gamma_1 &= \{\mu > \alpha_1 \text{ and } \mu > \alpha_2\sigma\}\,, \\ \Gamma_0 &= \{\mu < \alpha_1 \text{ or } \mu < \alpha_2\sigma\}\,, \end{cases} \tag{21}$$

where α_1 and α_2 are constant threshold values.

It is easy to verify that there is no UMP solution for this composite hypothesis testing problem, and the LMP approach does not apply either. However, we can use the general likelihood ratio (GLR) approach [6], which gives the following result:

Proposition IV.1. *Given a set of inlier delay samples $\{q_i, \forall i \in I\}$, denote by μ_q and σ_q the sample mean and standard deviation. Then, the optimal detection rule from GLR with equal cost of false local and remote detections is:*

$$\begin{cases} \mu_q > \alpha_1 \text{ and } \frac{\mu_q}{\sigma_q} > \sqrt{\frac{N-1}{N}}\alpha_2 \Rightarrow H_1\,, \\ \mu_q \leq \alpha_1 \text{ or } \frac{\mu_q}{\sigma_q} \leq \sqrt{\frac{N-1}{N}}\alpha_2 \Rightarrow H_0\,. \end{cases} \tag{22}$$

With the detection rule at hand, we only need to find the proper α_1 and α_2 values. For the first few congestion events (initial stage), we set α_1 to be half of the maximum experienced queueing delay, and set α_2 to be $1/2$ (this number comes from Assumption 2). For each congestion event, we compute the mean of the queueing delays over the samples between the 25% percentile and the 75% percentile, denote it by $\hat{\mu}_k$, compute the sample standard deviation over all inlier samples, and denote it by $\hat{\sigma}_k$, where k indexes this congestion event. As k goes by, we average $\hat{\mu}_k$ and $\hat{\sigma}_k$ over all past local (respectively, remote) congestion events to estimate μ_{LC} and σ_{LC} (respectively, μ_n and σ_n), and denotes the estimations by $\hat{\mu}_{LC}$ and $\hat{\sigma}_{LC}$ (respectively, $\hat{\mu}_{RC}$ and $\hat{\sigma}_{RC}$). After we make several local and congestion detections, we set

$$\begin{cases} \alpha_1 &= (\hat{\mu}_{LC} + \hat{\mu}_{RC})/2\,, \\ \alpha_2 &= (\hat{\mu}_{LC}/\hat{\sigma}_{LC} + \hat{\mu}_{RC}/\hat{\sigma}_{RC})/2\,. \end{cases} \tag{23}$$

With the study of both loss and delay at hand, we derive the following result on CLD detection accuracy:

V. Configuration, Performance, and Discussions

In this section, we study the parameter configurations and detection accuracy of the CLD algorithm. We altogether have 5 parameters: g, κ_{both}, κ_{low}, κ_{QD}, and β. The first three parameters are discussed in Section IV-B, we now discuss the settings of κ_{QD} and β.

A. Discussions on κ_{QD}

Suppose a home has altogether N flows, and $N_c \leq N$ flows experience congestion. For local congestion, $N_c = N$ all the time, and for remote congestion $1 \leq N_c \leq N$. As N_c increases close to N, the end users can no longer distinguish remote congestion from local congestion. Typically, there is a threshold, denoted by N_{th}, such that once $N_c \geq N_{th}$, CLD cannot distinguish remote from local congestions. The parameter κ_{QD} determines this threshold N_{th}. If $N_c \geq \kappa_{QD}$, then local congestion detection might be made by the Quick Detection Module no matter the real congested link is local or remote. On the other hand, if $N_c < \kappa_{QD}N$, the probability that wildcard makes false local detection is negligible: both the probability of $H = N_c$ and the probability that $\kappa_{QD}N - H$ non-congested users experience large delays are very small, and their product is negligible. We wish to successfully distinguish local and remote congestion as long as $N_c < N/2$, so we require $\kappa_{QD} > 0.5$, and we set its default value to be 0.8. The dependence of N_c on κ_{QD} will be derived in Section V-C

B. Discussions on β

We now check β. Consider the probability that a non-congested TCP session sees queueing delay larger than $\beta\bar{\mu}_{LC}$, and call it false positive probability. Suppose our estimation $\bar{\mu}_{LC}$ is accurate, and $\bar{\mu}_{LC} = \mu_1 = \mu_s + \mu_1$. Then, we have

$$P_1 := P(\text{One false positive}) = 1 - \Phi(\frac{\beta\mu_s}{\sigma_n})\,,$$

where $\Phi(\cdot)$ is the CDF function of a standard normal distribution. From our analysis in Section IV, $\mu_s \gg \sigma_s > \sigma_n$, we know that P_1 is actually very small.

For a false detection, we need at least $N_d := (\kappa_{QD}N - H)$ false detections. From our analysis in Section III, we know that with high probability, H is less than $N/2$ even for local congestion, and H would be even smaller for remote congestion, so as an approximation study, we assume $N_d \gtrapprox (\kappa_{QD}-0.5)N$, and for our standard setting on κ_{QD}, $N_d \gtrapprox 0.3N$. With this assumption, we have

$$P(\text{false detection}) \leq 1 - \sum_{i=1}^{0.3N} \binom{n}{k} p_1^k (1-p_1)^{N-k}.$$

C. False Detection and Miss Probabilities

With all the above analysis on loss, delay, parameter configuration, we have the following result on the detection accuracy:

Proposition V.1. *Both the false remote and false local detection probabilities of the CLD algorithms approach* 0 *as* $N \to \infty$, *as long as* $N_c < \min(1-\kappa_{low}, \kappa_{QD})N$ *and* $\beta > \mu_n/\mu_{LC}$, *where* N_c *is the number of congested flows for a remote congestion, and* μ_n *and* μ_{LC} *are the expectations of noise term and local link queueing delay. Furthermore, for fixed* N *and parameter configuration, the accuracy improves as* B, *the local link buffer size, increases.*

Proposition V.1 gives both the parameter configuration guideline and asymptotical result on detection accuracy. From our simulations, we see that the performance of CLD is insensitive to parameter configurations as long as they are set within a wide reasonable range, and all over this range, CLD has accurate detection.

VI. NS-2 Simulations and Validation

We validated the performance of CLD algorithm by extensive *ns-2* simulations.

A. Network Topology and Simulation Scenario

We perform simulations for the following network scenario illustrated in Figure 9. As shown in the left plot, the network has three links L_1, L_2 and L_3, $N_1 + N_2$ persistent flows, 3 groups of on-off flows, which has the following on-off pattern: one cycle has 6 phases; all the three groups of flows are off at odd phases; one of the three groups are on at one even phase. The link capacities and the receiver window sizes are picked such that, each link is not congested if its corresponding on-off flows are off, and is congested if they are on. For this setup, L_1 mimics the home access local link, as all flows pass L_1, and L_2 and L_3 mimics remote links, as only part of flows pass them.

Since the real-world delay samples are buried with noises, which is not included in *ns-2*, we manually add noise into each RTT measurement, which is either Gaussian or uniformly distributed. For our simulations, We pick a variety of values for N_1 and N_2, link capacities C, router buffer size B, RTT T, noise level n, and check the performance of the CLD algorithm.

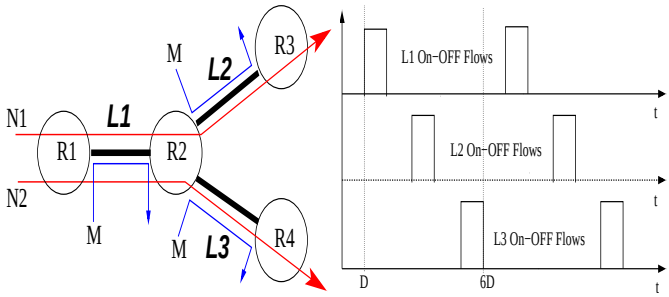

Fig. 9. **Simulations Scenario**. (Left) topology of simulations: three links, N_1 and N_2 two-hop flows, and M single-hop ON-OFF flows passing each individual link. (Right) Pattern of On-Off flows: second, fourth and sixth phases are on phases for L_1, L_2, and L_3 on-off flows, respectively.

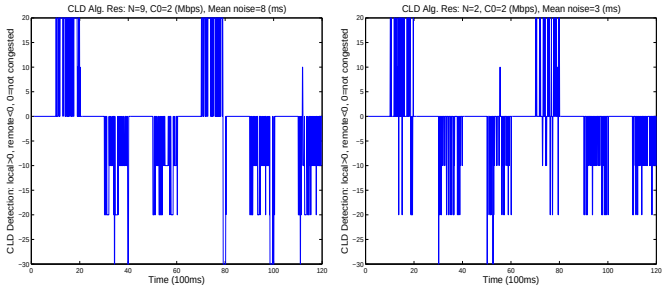

Fig. 10. **Examples of CLD Detection Result**. (Left) Detection of one simulation with $N_1 = N_2 = 9$ and noise level = 8 ms. The x-axis is time, and y-axis gives the detection result: 0 for no congestion, positive values for local congestion and negative values for remote congestion. (Right) same as left, but $N_1 = N_2 = 2$ and noise level is 3 ms.

B. Detection Accuracy

We plot the detection results for several representative settings with different network scenarios in Figure 10. For the figures, the x-axis is time, and y-axis is the detection result: 0 means no link is congested (no packet loss), positive values mean local congestion detection, and negative values mean remote congestion detection. For all these figures, in phase 2, most detections are local congestion, and in phase 4 and 6, most detections are remote congestion, and the miss and false alarm probabilities are both small.

We now check the detection result as N_1 and N_2 varies, or in another word, as the actual number of congested link, denoted by N_c varies. The result is shown in Figure 11. From

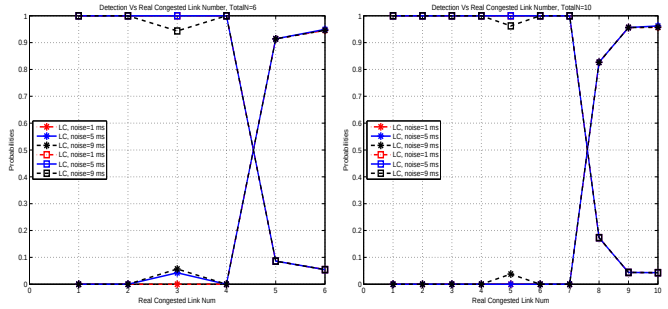

Fig. 11. **Detection Result Vs Number of Congested Links**. (Left) 6 total flows. (Right) 10 total flows. The x-axis is N_c and y-axis is the probability of local and remote congestion corresponding to N_c congested links. Since $\kappa_{both} = 0.3$, as long as $N_c \leq 0.7N$, the detection is very accurate.

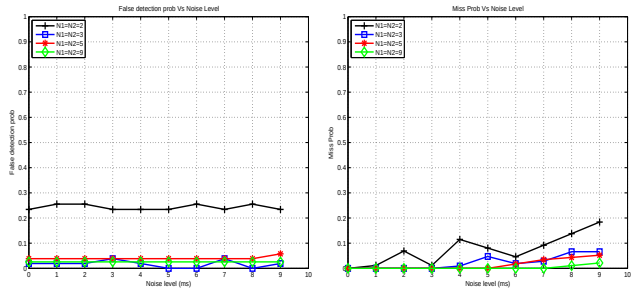

Fig. 12. **Detection Result Vs Noise Level**. The x-axis is the noise level, and the y-axis gives false detection probabilities. (Left) False local detection probability. (Right) False remote detection probability. As long as $N_1 = N_2 \geq 3$, the detection is accurate.

this figure, as long as $N_c < \min(1-\kappa_{low}, \kappa_{QD})N = 0.8N$, the detection is accurate. The right figure also shows that the detection is less accurate if $N_1 = N_2$. This is because for this special case, the OIR Module is less likely to remove outliers, and the hypothesis testing assumption of identical inlier distributions no long holds. Even though, the performance is still pretty accurate.

For the worse case of $N_1 = N_2 = N/2$, we test its performance against noise levels by varying N and n. The result is demonstrated in Figure 12, which shows that the detection is pretty accurate for the worst $N_1 = N_2$ case, and the false detection probabilities increase as n increases, decrease as N increases.

C. Robustness and Insensitivity

We then verify that CLD works for a large range of network scenarios. We vary D_1 from 60ms to 100 ms, set $D_i = D_1 + \delta(i-1)$ and vary δ from 0 to 5, set $B = bC$ and vary b from 5 to 20, set $C = Nc$ and vary from c from 1 to 5 Mbps, and for each of the above network scenario, we vary the queueing policy from Dropfront to Droptail. Our simulations show that CLD algorithm performs well for all these wide range of network scenarios, and the result for some examples are plotted in the left plot of Figure 13.

We further test the parameter sensitivity of CLD algorithm. CLD algorithm has 5 parameters: κ_{QD}, κ_{both}, κ_{low}, and β and g. The choice of g is given by the Hampel identifier analysis in Section IV. We now test the other parameter choices. We choose the same simulation scenario as that for Figure 11, with fixed $N_1 + N_2 = 10$ and varying $N_1 = 1, \cdots, 9$. For this fixed simulation scenario, we vary our parameter settings:

$$\begin{cases} \kappa_L = & 0.6 & 0.7 & 0.8 & 0.9 & & & \\ \kappa_{both} = & 0.2 & 0.25 & 0.3 & 0.35 & 0.4 & & \\ \kappa_{low} = & \frac{\kappa_{both}}{2} & \frac{2\kappa_{both}}{3} & \frac{3\kappa_{both}}{4} & \frac{4\kappa_{both}}{5} & & & \\ \beta = & 0.7 & 0.75 & 0.8 & 0.85 & 0.9 & 0.95 & 1.0 \end{cases} \quad (24)$$

We have all together $4 \times 5 \times 4 \times 7 = 560$ combinations of parameter settings, and we plot the detection results for all combinations in the right plot of Figure 13. From the figures, it is clear that κ_L, κ_{both} and κ_{low} do not affect the performance as long as they lie in their reasonable ranges. β value has slight influence on CLD detection result: if the number of flows sharing one remote link exceeds a threshold, CLD cannot distinguish it from a local congestion. This threshold is determined by β. If below this threshold, β has no other influence on the performance of CLD algorithm. From these simulations, we have shown that CLD is robust against network scenario variations and is insensitive to parameter setting choices, as long as they are chosen from a wide reasonable range.

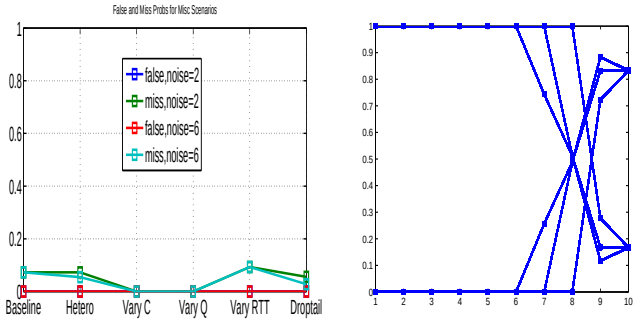

Fig. 13. **Robustness of the CLD Algorithm**. (Left) Robust against network scenario variations. For all simulations, we set $N_1 = N_2 = 6$, and vary the noise level to be 2 ms and 6 ms. For "Baseline", we pick Dropfront queue and homogeneous RTT users. For "Hetero", we choose heterogeneous RTT users. For "Vary C", "Vary Q", "Vary RTT", and "Vary Queue Policy", we pick alternative C, B, D values or pick Droptail. For all these variations, the performance of CLD is good. (Right) Robust against parameter settings. $\kappa_{QD} \in \{0.6, 0.7, \cdots, 0.9\}$, $\kappa_{both} \in \{0.2, 0.25, \cdots, 0.4\}$, $\kappa_{low} \in \{\kappa_{both}/2, 2\kappa_{both}/3, 3\kappa_{both}/4, 4\kappa_{both}/5\}$, and $\beta \in \{0.7, 0.75, \cdots, 1.0\}$. CLD has good performance for all these $4 \times 5 \times 4 \times 7 = 560$ combinations of parameter settings. Furthermore, the performance only depends on $\min(1-\kappa_{both}, \kappa_{QD})$, and as this value is fixed, the variation of other parameters makes negligible differences.

VII. Conclusion and Future Work

We proposed CLD, a user end Congestion Location Detection algorithm, that is crucial for the deployment of low priority TCP protocols. We described the algorithm in details, showed that it can accurately detect whether the congested link is local or remote, and validated our analysis with extensive simulations. In developing the CLD algorithm, we also proved a series of analytic results on two issues important in their own right: synchronization of packet loss and delay increase across TCP sessions and distributed hypothesis testing in TCP.

Acknowledgments

We are grateful to Osama Mazahir, Alvin Tan, Minghua Chen, and Vincent Poor for helpful discussions.

References

[1] A. Venkataramani, R. Kokku, and M. Dahlin, "TCP Nice: a mechanism for background transfer," in *Proc. OSDI'02*, 2002, pp. 329–343.

[2] A. Kuzmanović and E. Knightly, "TCP-LP: a distributed algorithm for low priority data transfer," in *Proc. IEEE Infocom 2003*, San Francisco, CA, USA, March 2003.

[3] P. Key, L. Massoulié, and B. Wang, "Emulating low-priority transport at the application layer: a background transfer service," in *Proc. ACM SIGMETRICS 2004*, 2004, pp. 118–129.

[4] Microsoft, "Background intelligent transfer service," description available at http://msdn.microsoft.com/en-gb/library/aa362827.aspx.

[5] S. Liu, M. Vojnović, and D. Gunawardena, "Competitive and considerate congestion control for bulk data transfers," in *Proc. IWQOS*, 2007.

[6] H. V. Poor, *An Introduction to Signal Detection and Estimation*. Springer Texts in Electrical Engineering, 1998.

[7] S. Liu, T. Başar, and R. Srikant, "TCP-Illinois: A loss and delay-based congestion control algorithm for high-speed networks," *Special Issues of Performance Evaluations*, June 2008.

[8] S. Biaz, "Is the round-trip time correlated with the number of packets in flight," in *In Proc. Internet Measurement Conference*, 2003, pp. 273–278.

[9] J. Aikat, J. Kaur, F. Smith, and K. Jeffay, "Variability in tcp round-trip times," Miami Beach, FL, October 2003.

[10] M. Mathis, J. Heffner, and R. Raghunarayan, "Tcp extended statistics mib," RFC 4898, available at ftp://ftp.rfc-editor.org/in-notes/rfc4898.txt.

[11] P. Key, L. Massoulie, G. Dinan, R. Black, and G. OShea, "Round trip time estimation," available at http://www.freepatentsonline.com/EP1744495.html.

[12] L. Zhang and D. Clark, "Oscillating behavior of network traffic: A case study simulation," *Internetwork: Research and Experience*, vol. 1, no. 2, 1990.

[13] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. on Networking*, vol. 1, no. 4, pp. 397–413, August 1993.

[14] T. V. Lakshman and U. Madhow, "The performance of tcp/ip for networks with high bandwidth-delay products and random loss."

[15] S. Ha, L. Le, I. Rhee, and L. Xu, "Impact of background traffic on performance of high-speed tcp variant protocols," *Comput. Netw.*, vol. 51, no. 7, pp. 1748–1762, 2007.

[16] T. V. Lakshman, A. Neidhardt, and T. J. Ott, "The drop from front strategy in tcp and in tcp over atm," in *IEEE INFOCOM*, March 1996.

[17] G. Casella and R. L. Berger, *Statistical Inference, 2nd ed.* Duxbury Press, 2002.

[18] S. Shakkottai, R. Srikant, N. Brownlee, A. Broido, and K. Claffy, "The rtt distribution of tcp flows in the internet and its impact on tcp-based flow control."

[19] S. H. Low, "A duality model of tcp and queue management algorithms," *IEEE/ACM Trans. Netw.*, vol. 11, no. 4, pp. 525–536, 2003.

[20] D. Gunawardena, P. Key, and L. Massouli, "Network characteristics: modelling, measurements and admission control," in *Proc. IWQoS*, 2003.

[21] H. Jiang and C. Dovrolis, "Passive estimation of tcp round-trip times," *ACM Computer Communication Review*, vol. 32, pp. 75–88, 2002.

[22] P. Sessini and A. Mahanti, "Observations on round-trip times of tcp connections," in *Proc. of SCS SPECTS*, Calgary, Canada, July 2006.

[23] J. R. Taylor, *An Introduction to Error Analysis*. Sausolito, California: University Science Books, 1997.

[24] F. Grubbs, "Procedures for detecting outlying observations in samples," *Technometrics*, vol. 11, no. 1, pp. 1–21, Febrary 1969.

[25] S. Ross, "Peirce's criterion for the elimination of suspect experimental data," *J. Engr. Technology*, 2003.

[26] F. R. Hampel, "The breakdown point of the mean combined with some rejectio rules," *Technometrics*, vol. 27, 1985.

[27] L. Davies and U. Gather, "The identification of multiple outliers," *J. American Statistical Association*, vol. 88, no. 423, pp. 782–792, 1993.

APPENDIX

A. Proof of Proposition III.1

Since the network pipe can hold at most $CD + B$ packets, it is easy to see that $S^- = CD + B$, and once S exceeds this value, packet drop occurs. As the router uses Droptail queue, the congestion alleviation period lasts exactly one RTT, and during this period, exactly W_i^- positive ACKs are feeded back to the senders. Here, for modeling simplicity, we ignore the triple duplicate acknowledgement mechanism of TCP, and assume that the negative ACK of a packet loss arrives at the sender in sequence with the positive ACKs. The difference between this simplified model and the triple duplicate acknowledgement mechanism is negligible if W_i^- is sufficiently large. With these W_i^- positive ACKs received before the sender realizes the network congestion, each individual flow increases its window size by 1, and we have

$$S^+ = \sum_{i=1}^{N} W_i^+ = \sum_{i=1}^{N} W_i^- + N = CD + B + N\,. \qquad (25)$$

As the congestion alleviation period lasts one RTT, the number of packet losses is

$$M = S^+ - (CD + B) = N\,. \qquad (26)$$

□

B. Proof of Proposition III.2

For the practical cases where M is a variable, we rewrite (25) as

$$S^- = \sum_{i=1}^{N} W_i^- = CD + B + \epsilon\,. \qquad (27)$$

where ϵ is a random variable representing the burstness of the packet arrival and effective capacity fluctuation due to background traffic, and we have $E[\epsilon] = 0$. Let V_i be the number of positive ACKs feeded back to flow i during the congestion alleviation period, we have

$$E[W_i^+ | V_i, W_i^-] = W_i^- + \frac{V_i}{W_i^-}$$

With the randomness caused by slight RTT difference and bursty packet arrival process, T_a is not a constant, but a variable, and we have $E[T_a] = T$ for Droptail and $E[T_a] = D$ for Dropfront. We also denote by ϖ the standard deviation of T_a. As T_a is a random variable, so is V_i, which is no longer exactly W_i^-. Since in average, W_i^- ACKS per RTT, we have $E[V_i|T_a] = W_i^- T_a/T$. Therefore, $E[V_i] = \eta W_i^-$, where $\eta = 1$ for Droptail, and $\eta = D/T = CD/(CD + B)$ for Dropfront. We then have

$$E[W_i^+ | W_i^-] = W_i^- + \frac{\eta W_i^-}{W_i^-} = W_i^- + \eta\,,$$

which means

$$\begin{aligned} E[M] &= \textstyle\sum_{i=1}^{N} E[W_i^+] - (CD + B) \\ &= E[S^-] + N\eta - (CD + B) = \eta N\,. \end{aligned} \qquad (28)$$

We now check the variance of M. Assume $Var[M|T_a]$ is negligible, then $Var(M) = N^2 Var(T_a) = N^2\varpi^2$, which means $Std(M) = N\varpi$. Even if $Var[M|T_a] \neq 0$, its effect does not scale with N^2, and for asymptotic analysis, its effect is always ignorable. The proportionality of $Std(M)$ and N is also validated by extensive *ns-2* simulations in Section III-C. □

C. Proof of Proposition III.3

Suppose all flows use the standard TCP, then, at steady-state, $E[x_i]$ is inversely proportional to T_i. From Assumption 1, we have

$$P_i := P(\text{dropped packet } \in i)\frac{x_i}{C} \approx \frac{1/T_i}{\sum_{j=1}^{N} 1/T_j} \qquad (29)$$

Next, we ignore the possibility that the flow of a later dropped packet has shorter RTT than the flow of the first dropped

packet and the throughput reduction of the later flow propagate faster than that of the first flow. With the simplification, we have $T_a \approx T_i$ (or D_i for Dropfront) if the first dropped packet belongs to i, and therefore,

$$E[T_a] \approx \begin{cases} \sum_{i=1}^{N} P_i T_i \approx \frac{N}{\sum_{j=1}^{N} \frac{1}{T_j}} & \text{for Droptail,} \\ \sum_{i=1}^{N} P_i D_i \approx \frac{\eta N}{\sum_{j=1}^{N} \frac{1}{T_j}} & \text{for Dropfront,} \end{cases} \quad (30)$$

where $\eta_{min} := \min_i D_i/T_i \leq \eta \leq \eta_{max} := \max_i D_i/T_i$. Following similar steps as the homogeneous RTT case, we have $E[M] = N$ for Droptail queue, and for Dropfront queue,

$$\eta_{min} N \leq E[M] \leq \eta_{max} N \,, \quad (31)$$

Furthermore, from our analysis on T_a, we know that $Var(T_a)$ is much larger than that for homogeneous RTT case. Therefore, $Std(M)$ is still proportional to N, but much larger than that for the homogeneous RTT case. □

D. Proof of Proposition III.4

The equalities of $f(m)$ and $g(m)$ are straight-forward from our modeling assumptions and approximations. We now prove the inequalities in (6).Define $h(x) := 1 - (1-x)^m$, then

$$f(m) = \sum_{i=1}^{N} h(\lambda_i) \,. \quad (32)$$

Since $h(x)$ is concave over x, from Jensen's inequality, we have

$$f(m) \leq N h(\sum_{i=1}^{N} \lambda_i/N) = N(1 - (1 - \frac{1}{N})^m) \,. \quad (33)$$

Furthermore, define $\lambda_{min} := \min_i \lambda_i = \min_i x_i/C$, define $\xi_i := N\lambda_i$ and $\xi_{min} := N\lambda_{min}$, then

$$f(m) \geq N(1 - (1-\lambda_{min})^m) = N(1 - (1 - \frac{\xi_{min}}{N})^m) \,. \quad (34)$$

So we have proved (6).

□

E. Proof of Proposition III.5

For the unconditional expectation of H, we have

$$E[H] = E_M[f(M)] \,.$$

As $f(m)$ is concave over m, from Jensen's inequality, we have

$$E[H] \leq f(E[M]) \leq N(1 - (1 - \frac{1}{N})^{E[M]}) \,.$$

From our analysis on M, we can write $E[M] = \eta N$ for all cases,, where $\eta = 1$ for Droptail, $\eta = D/T$ for Dropfront and homogeneous RTT flows, and $\eta \in [\eta_{min}, \eta_{max}]$ for Dropfront and heterogeneous RTT flows. Therefore, for all cases, we have

$$E[H] \leq N(1 - (1 - \frac{1}{N})^{\eta N}) \approx N(1 - e^{-\eta}) \,. \quad (35)$$

Since $\eta \leq 1$, we have $E[H] \leq\leq N(1 - e^{-1}) \approx 0.632N$.

We have found an upper bound for $E[H]$. Next we seek the lower bound for $E[H]$. From Taylor expansions for the moments of functions of random variables [17], we have the following approximation:

$$E[H] = E[f(M)] \approx f(E[M]) + \frac{f''(E[M])}{2} Var(M) \,. \quad (36)$$

From the formula of $f(m)$ in (5) and $E[M] = \eta N$, we have

$$\begin{aligned} f''(E[M]) &= -\sum_{i=1}^{N} (log(1-\lambda_i))^2 (1-\lambda_i)^{\eta N} \\ &\approx -\sum_{i=1}^{N} \lambda_i^2 (1-\lambda_i)^{\eta N} \,. \end{aligned} \quad (37)$$

It is easy to verify that $x^2(1-x)^{\eta N}$ is concave over x for any $0 < \eta \leq 1$. Then, from Jensen's inequality,

$$\begin{aligned} f''(E[M]) &\approx -\sum_{i=1}^{N} \lambda_i^2 (1-\lambda_i)^{\eta N} \\ &\geq -N \tfrac{1}{N^2} (1 - \tfrac{1}{N})^{\eta N} \approx \tfrac{1}{N^2} e^{-\eta} \\ &\approx -\tfrac{e^{-\eta}}{N} \,. \end{aligned} \quad (38)$$

From (3), $Var(M) = \gamma^2 N^2$. Plugging (6) and (38) into (36), we have

$$\begin{aligned} E[H] &\geq N(1 - (1 - \tfrac{\xi_{min}}{N})^{\eta N}) - \tfrac{e^{-\eta}}{2N} \gamma^2 N^2 \\ &\approx N(1 - e^{-\eta \xi_{min}} - \gamma^2 \tfrac{e^{-\eta}}{2}) \,. \end{aligned} \quad (39)$$

The lower bound of $E[H]$ in (39) depends on ξ_{min} and γ. For γ, we know that $\gamma \leq 1$ in general. For $\xi_{min} = \min_i x_i/\bar{x}$, where $\bar{x} := C/N$, it also depends on the RTT distribution. For homogeneous RTT case, $E[x_i] = \bar{x} = C/N, \forall i$, and due to the AIMD nature of TCP, at steady state, $\xi_{min} \geq 1/2$. For heterogeneous RTT case,

$$E[x_i/C] = \frac{\frac{1}{T_i}}{\sum_{j=1}^{N} \frac{1}{T_j}} \,,$$

and at steady state,

$$\xi_{min} \geq \frac{N}{2 \sum_{j=1}^{N} \frac{T_{max}}{T_j}} \,.$$

Plugging into (39), we have

$$E[H] \geq N(1 - e^{-\eta/(2\alpha)} - \frac{e^{-\eta}}{2}) \,, \quad (40)$$

where $\alpha := T_{max}(\sum_{j=1}^{N} 1/T_j)/N$. We now check the lower bound of $E[H]$ for a special case of homogeneous RTT and Droptail, where $\eta = 1$ and $\alpha = 1$. Under that case,

$$E[H] \geq N(1 - e^{-1/2} - \frac{e^{-1}}{2}) \approx 0.21N \,.$$

We next consider the standard deviation of H. From Taylor expansions for the moments,

$$E[g(M)] = g(E[M]) + \frac{g''(E[M])}{2} Var(M) \,.$$

In the above equaiton, $g(E[M])$ is proportional to N from (7), $Var(M)$ is proportional to N^2 from (3). It is easy to verify that $g''(E[M]) << 1$. So for asymptotical analysis, we can write

$$E[g(M)] = \epsilon N^2 + O(N) \,,$$

where $\epsilon := g''(E[M])/2 << 1$. We know that

$$\begin{aligned} Var(H) &= Var(E[H|M]) + E[Var(H|M)] \\ &= Var(f(M)) + E[g(M)] \,. \end{aligned} \quad (41)$$

Plugging (5) into (41), and use the Taylor expansions for the second moment, we have

$$\begin{aligned} & Var(f(M)) \approx (f'(E[M]))^2 Var(M) \\ = \; & (\sum_{i=1}^{N}(-)\log(1-\lambda_i)(1-\lambda_i)^{E[M]})^2 Var(M) \\ \approx \; & (\sum_{i=1}^{N}\lambda_i(1-\lambda_i)^{\eta N})^2 Var(M) \end{aligned}$$

Clearly, $Var(f(M))$ dominates $E[g(M)]$ for large N values, and we have

$$Var(H) \gtrapprox Var(f(M)) \approx (\sum_{i=1}^{N}\lambda_i(1-\lambda_i)^{\eta N})^2 Var(M)\,,$$

which means

$$Std(H) \gtrapprox (\sum_{i=1}^{N}\lambda_i(1-\lambda_i)^{\eta N}) Std(M)\,.$$

We now check the ratio between $Std(H)$ and $Std(M)$. If $\lambda_i \equiv 1/N, \forall i$,

$$\frac{Std(H)}{Std(M)} \gtrapprox (1-\frac{1}{N})^{\eta N} \approx e^{-\eta}\,.$$

For heterogeneous λ_i case, since $x(1-x)^m$ has maximum value at $x^* = 1/(1+m)$ and decreases as x moves far from x^* in both directions, we know that

$$\lambda_i(1-\lambda_i)^m \geq \min(\lambda_{min}(1-\lambda_{min})^m, \lambda_{max}(1-\lambda_{max})^m)\,,$$

and

$$\lambda_i(1-\lambda_i)^{\eta N} \geq \min(\frac{\xi_{min}}{N}e^{-\eta\xi_{min}}, \frac{\xi_{max}}{N}e^{-\eta\xi_{max}})\,.$$

Henceforth,

$$\frac{Std(H)}{Std(M)} \gtrapprox \min(\xi_{min}e^{-\eta\xi_{min}}, \xi_{max}e^{-\eta\xi_{max}})$$

Since xe^{-x} has the maximum value at $x=1$, we know that the ratio between $Std(H)$ and $Std(M)$ is the largest if $\lambda_i \equiv 1/N, \forall i$, and it becomes smaller if x_i's are more diversified. This indicates that the ratio is smaller for heterogeneous RTT case, which is consistence with the result that $Std(M)$ is much larger in heterogeneous RTT case than in homogeneous RTT case.

Numerically, from the AIMD behavior, at steady state, we can assume that $\xi_{min} \geq 1/(2\alpha_{max})$ and $\xi_{max} \leq 2/\alpha_{min}$, where

$$\alpha_{max} := T_{max}(\sum_{j=1}^{N}1/T_j)/N \text{ and } \alpha_{min} := T_{min}(\sum_{j=1}^{N}1/T_j)/N\,, \tag{42}$$

and therefore,

$$\frac{Std(H)}{Std(M)} \gtrapprox \min(\frac{1}{2\alpha_{max}}e^{-\frac{\eta}{2\alpha_{max}}}, \frac{2}{\alpha_{min}}e^{-\frac{2\eta}{\alpha_{min}}})\,.$$

For the special case of homogeneous RTT with Droptail, $\alpha_{min} = \alpha_{max} = 1$ and $\eta = 1$, then

$$\frac{Std(H)}{Std(M)} \gtrapprox \min(\frac{e^{\frac{-1}{2}}}{2}, 2e^{-2}) = 0.271$$

□

F. Proof of Proposition IV.1

.

Proof: From GLR [6], define general likelihood ratio

$$\rho := \frac{\max_{\theta=(\mu,\sigma)\in\Gamma_1} P_\theta(\mathbf{Q})}{\max_{\theta=(\mu,\sigma)\in\Gamma_0} P_\theta(\mathbf{Q})}\,,$$

where $\mathbf{Q} := \{q_1, \cdots, q_N\}$ is the vector of all q_i's. Then, we detect H_1 if ρ is larger than a threshold, and detect H_0 otherwise.

We now check $max_\theta P_\theta(\mathbf{Q})$:

$$P_\theta(\mathbf{Q}) = \Pi_i \frac{1}{\sqrt{2\pi}\sigma} exp(-\frac{q_i-\mu)^2}{2\sigma^2})\,,$$

and

$$\begin{aligned} & \log(P_\theta(\mathbf{Q})) = \textstyle\sum_{i=1}^{N}(-\frac{(q_i-\mu)^2}{2\sigma^2} - \log\sigma - \log\sqrt{2\pi}) \\ = \; & -\textstyle\sum_{i=1}^{N}(\frac{(q_i-\mu_q+\mu_q-\mu)^2}{2\sigma^2} + \log\sigma + \log\sqrt{2\pi}) \\ = \; & -\textstyle\sum_{i=1}^{N}(\frac{(q_i-\mu_q)^2+(\mu_q-\mu)^2}{2\sigma^2} + \log\sigma + \log\sqrt{2\pi}) \\ = \; & -((N-1)\frac{\sigma_q^2}{2\sigma^2} + N\frac{(\mu_q-\mu)^2}{2\sigma^2} + Nlog\sigma + N\log\sqrt{2\pi})\,, \end{aligned}$$

where μ_q and σ_q are the sample mean and standard deviation of $\mathbf{Q}$. It is easy to verify that $P_\theta(\mathbf{Q})$ is maximized if

$$\mu = \mu^* = \mu_q \text{ and } \sigma = \sigma^* = \sqrt{\frac{N-1}{N}}\sigma_q$$

If we assign equal cost to false detection and miss, the threshold in the GLR test is 1, then we have the following detection rule:

$$\begin{cases} (\mu^*,\sigma^*) \in \Gamma_1 \Leftrightarrow \mu_q > \alpha_1 \text{ and } \frac{\mu_q}{\sigma_q} > \sqrt{\frac{N-1}{N}}\alpha_2 \Rightarrow H_1\,, \\ (\mu^*,\sigma^*) \in \Gamma_0 \Leftrightarrow \mu_q \leq \alpha_1 \text{ or } \frac{\mu_q}{sigma_q} \leq \sqrt{\frac{N-1}{N}}\alpha_2 \Rightarrow H_0\,. \end{cases}$$

□

G. Proof of Proposition V.1

We first consider false detection probability. There are altogether two possibilities for false detections: 1) in Quick Detection Module and 2) in hypothesis testing module. Suppose the false detection probabilities for the two modules are denoted by P_w^f and P_{ht}^f, then the total false detection probability $P^f < P_w^f + P_{ht}^f$. We already know P_w^f, and we now check P_{ht}^f.

Assume after removing outliers, we have N_i samples left, and they all are $\mathbf{N}(\mu_n, \sigma_n)$ distributed. After finding the average, μ_q is $\mathbf{N}(\mu_n, \sigma_n/N_i)$ distributed. As number of outliers is less than $N/2$, we have $N_i > N/2$, and therefore, the standard deviation of μ_q is at most $\sigma^* := 2\sigma_n/N$. Suppose the estimations on $\bar{\mu}_{LC}$ and $\bar{\mu}_{RC}$ are accurate. Then, we have

$$P_{ht}^f < 1 - \Phi(\frac{\frac{\mu_s-\mu_n}{2}}{\frac{2\sigma_n}{N}}) = 1 - \Phi(\frac{N(\mu_s-\mu_n)}{4\sigma_n}$$

Assume that $\mu_s \geq 2\mu_n \geq \sigma_n$, then

$$P_{ht}^f < 1 - \Phi(\frac{N}{4})\,.$$

Therefore, if $N > 8$, $P_{ht}^f < 2.5\%$ and if $N > 12$, $P_{ht}^f < 0.15\%$. Furthermore, for any value of μ_s, μ_n, and σ_n, as long

as $\mu_s > \mu_n$, which is typically true, we have $P_{ht}^f \to 0$ as $N \to \infty$.

We then consider the miss probabilities, which is the sum of the miss probabilities at the outlier removal module and hypothesis testing module, and denote these probabilities by $P^m = P_o^m + P_{ht}^m$. From our choice of g, κ_{both} and κ_{low}, $P_o^m < 0.05$ for all N, and $P_o^m < 0.01$ if $N > 20$ (see right plot of Figure 8).

For P_{mt}^m, it has the similar formula as P_{mt}^f except that we replace σ_n by $\sigma_1 = \sqrt{\sigma_n^2 + \sigma_s^2}$. Similarly, we have $P_{mt}^m \to 0$ as $N \to \infty$. □