Congestion Location Detection: Methodology, Algorithm, and Performance

Shao Liu, Mung Chiang, Mathias Jourdain, and Jin Li

***Abstract*—We address the following question in this study: Can a network application detect not only the occurrence, but also the location of congestion? Answering this question will not only help the diagnostic of network failure and monitor server's QoS, but also help developers to engineer transport protocols with more desirable congestion avoidance behavior. The paper answers this question through new analytic results on the two underlying technical difficulties: 1) synchronization effects of loss and delay in TCP, and 2) distributed hypothesis testing using only local loss and delay data. We present a practical Congestion Location Detection (CLD) algorithm that effectively allows an end host to distributively detect whether congestion happens in the local access link or in more remote links. We validate the effectiveness of CLD algorithm with extensive experiments.**

I. Introduction

A. Motivations

Internet is a best effort network. A major cause of quality degradation is network congestion, which can be caused by lack of network resource or uneven distribution of traffic. By detecting the *occurrence* of congestion, which is well investigated in the literature and in practice, congestion control can be applied to regulate the traffic entering into the Internet, thus avoiding oversubscription of link capabilities. However, if we can further 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 end hosts in the same premise (e.g., a home, an office, or a server farm), or on a *remote* link shared by TCP sessions across multiple premises, we could provide additional valuable functionalities.

For example, a network diagnostic module may use the information to inform the user the location of congestion, thus providing the user with valuable diagnostic information. If the location of the congestion is local, the user may be able to perform certain actions, e.g., shut down a bandwidth heavy local application, to ease the congestion. If the location of the congestion is remote, the user may decide not to perform any action to ease the congestion.

As another example, a server that is capable of detecting the location of congestion can more effectively monitor its QoS to the end users and decide whether it will need to subscribe more bandwidth from ISP. If most of the congestion location is at remote link, the server's ISP bandwidth is not a bottleneck to its QoS. If, on the other hand, most of the congestion location is at local link, the server will instead consider subscribing to more bandwidth from its ISP.

Congestion Location Detection (CLD) can also be used to design transport protocols with desirable congestion avoidance behavior. There are usually two types of Internet applications: 1) high-priority applications that are QoS-sensitive, such as real-time media streaming, VoIP, instant messaging, web browsing, and 2) low-priority applications that are QoS insensitive, such as P2P file sharing, FTP file download, software updates, and data backup application. Ideally, the second category of applications can be served with a low-priority transport protocol by end-to-end congestion control (e.g., TCP-Nice [1], TCP-LP [2], BATS [3], BITS [4], 4CP [5]). In fact, Windows OSes have already used BATS and Linux OSes have already used TCP-LP for automatic software update. These low-priority flows give up network bandwidth when the network is congested, and benefit high-priority flows. An issue that hinders the wide deployment of the 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 the local access link, e.g., a DSL or cable modem link, the aggressive back-off of the low-priority applications during congestion benefits the high-priority applications of the same home. 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 altruistic 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 access link (shared only by all flows from end hosts on the same premise) or a remote link. The low priority TCP only needs to back off when the congested link is local.

B. Challenges

While congestion avoidance and Congestion Occurrence Detection (COD) is a popular topic in the literature, there is virtually no rigorous results on CLD. Detecting the location of congestion is a challenging problem due to the following reasons:

- It is intractable to solve the CLD problem using traditional estimation and detection techniques, e.g., [6]. If we think of it as a hypothesis testing problem, the possible locations of congestions form too many hypotheses, each of which has to be evaluated through complicated statistical analysis.

Shao Liu (shaoliu@princeton.edu) and Mung Chiang (chiangm@princeton.edu) are with the Department of Electrical Engineering, Princeton University; Mathias Jourdain (mjourd@microsoft.com) is with Microsoft Corporation; Jin Li (jinl@microsoft.com) is with Microsoft Research.

This work was in part supported by NSF grants CNS-0519880 and CNS-0720570.

- In practice, we cannot send many probing packets. Sending a constant stream of probing packets causes too much overhead. If we send probing packets after the occurrence of congestion, it will lead to congestion collapse and inaccurate location detection due to delay.
- 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 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 [7]. There has been no systematic characterization on the number of flows seeing loss when the shared link is congested.
- Packet delay cannot give sufficient information on CLD, either. Packet delay measurements are often very noisy [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 issues above, we solve a simplified CLD problem statement: can an end host use only local loss and delay information to detect if congestion happens at a local access link or at a remote link? Even this binary detection problem is challenging. Indeed, we can draw an analogy with the much more extensively studied COD problem. In TCP, (1) events such as 3 duplicated ACK packets imply packet loss, which (2) in turn implies the occurrence of congestion (somewhere in the network). Neither implication relationship is always true, although the resulting TCP design has been working well enough. In our attempt to solve the more difficult CLD problem, (1) we use the synchronization of loss and delay behaviors across multiple TCP sessions in the area controlled by the same local gateway to imply synchronization of congestion across the sessions, which (2) in turn implies that congestion happens close to the end host. Again, neither implication relationships is always true. In fact, the "synchronization 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. Contributions and Organization

There are two contributions in terms of methodologies: 1) this is a comprehensive study of the synchronization behavior of packet loss and delay among multiple TCP sessions, and 2) we have developed 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 of a CLD algorithm 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 distributively detect whether congestion happens at the local access link or at the more remote link.

To highlight the proposed algorithm early on, we will first introduce the CLD algorithm in Section II, before showing how it was developed in the rest of the paper. We then quantify the synchronization of flow loss events in Section III. Next, we investigate the synchronization of delay increase, describe the distributed hypothesis testing, justify the CLD parameter configuration, and analyze the detection accuracy in IV. We finally provide extensive *ns-2* simulation results to test the performance of our algorithm in Section V, and conclude this paper in Section VI. Due to space limitation, all proofs of analytical results can be found in the technical report [10].

II. The Congestion Location Detection Algorithm

The key ideas of CLD are as follows. Let there be multiple TCP flows behind a certain local link, e.g., a home with DSL or cable modem connection. Whenever a 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" synchronized congestion (as defined later this section), then the local link is the congested link. This is because if the congested link is remote, it is less likely that many flows from the same host 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 is only a small number of flows seeing congestion, we perform CLD based on queueing delay patterns. If the local link is congested, typically most flows will experience high delays at 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. Since the delay measurements tend to be polluted with noise, we remove outlier samples when computing 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.

A. CLD Algorithm

We now first describe the CLD algorithm, then take detailed discussions afterwards. CLD periodically queries loss and delay information of all TCP flows in the home, and group upload and download flows in separate buckets, which gives congestion location detection for uplink and downlink independently. For either downlink or uplink, suppose one home has altogether N TCP flows, indexed by $i = 1, 2, \cdots, N$. At each query, let L_i be the total loss event number up till now, and q_i be the average queueing delay over the latest query period, for flow i. If none of the flows sees an increase in L_i compared with the previous query, then there is no congestion in the network during the last period. Otherwise, we use the following three modules 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, respectively; and the parameters, counters and state variables are summarized in Table I and Table II, respectively. The reasons to choose the default parameters are explained in Section III and IV.

1) Quick Detection (QD) Module: For each flow i, we say it "sees" congestion if either it experiences packet losses, or queuing delay q_i is larger than a threshold. As shown in Table II-A, we maintain four state variables, $\hat{\mu}_{LC}, \hat{\mu}_{RC}, \hat{\sigma}_{LC}$ and $\hat{\sigma}_{RC}$, which are estimations of the mean and standard deviation of delays among N flows at one congestion event, for local and remote congestions. The estimation mechanism is

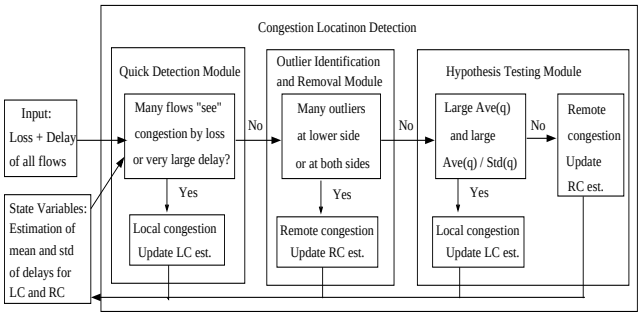

Fig. 1. **Flow chart of the CLD algorithm**.

Parameter	κ_{QD}	κ_{low}	κ_{both}	β	g
Default Value	0.8	0.2	0.3	0.9	2/3/5
Used in module	QD	OIR	OIR	QD	OIR
Discussed in section	IV-C/D	IV-B	IV-B	IV-D	IV-B

TABLE I
SUMMARY OF CLD PARAMETER SETTINGS.

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, 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. In 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 (21) or Figure 2. If both conditions satisfy, then the queueing delays of all inliers are sufficiently large and close, which we detect local congestion, and 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	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 $N \le 5$, $g = 3$ if $5 < N \le 10$, 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.

B. Further Discussions

The query of loss and delay can be done by kernel level modifications, or can be obtained through the existing TCP-E-STATS-MIB (Extended Statistics Management Information Base) module [11]. The query is straight forward for upload bucket, as the home users are senders of these flows. For download bucket, the home users are receivers of these flows, and we can use the mechanism in [12], which uses out of order packet arrival to detect loss and estimates window size to estimate delay. For each bucket, the loss and delay are only acquired through local information, so CLD is a one-sided

algorithm that does not need sender and receiver cooperation.

If there are multiple machines in one home, and they use Ethernet or WiFi to share one access link, all machines may broadcast their loss and delay counters periodically, and one delegate machine may run CLD algorithm and broadcasts the detection results to all other machines. If not all TCP flow information can be collected from all machines, running CLD based on a subset of flows still works, as long as we collect enough number of flows: from our simulations, the performance of CLD is sufficiently good if the number of collected TCP flows is larger than 4.

One query period lasts 0.1 second by default (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. Obviously, the more frequent CLD queries, the more accurate and the quicker 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 will be 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.

III. Synchronization of Packet Loss

We now summarize the development of CLD algorithm, study the parameter configuration, and analyze the accuracy of detection. We focus on packet loss in this section, and will study delay in Section IV. 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 distribution of H?*

Suppose during one congestion, altogether M packets are dropped at the link. To derive the distribution of H, we must know the distribution of M first. Therefore, the synchronization problem is divided to the following two subproblems: *1) What is the distribution of M? and 2) what is the distribution of H after we know the distribution 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, then totally there are $\sum_{i=1}^{N} W_i$ packets pushed into the network pipe, and we know that the network pipe can hold at most $CD + B$ packets. A congestion event starts when the queue becomes full and the next arrival packet has to be dropped[1], and ends when a flow sees a loss event, backs off its TCP window size, and the total arrival rate at the link drops to below its capacity. We denote by T_a the length of the congestion event duration, and by W_i^- and W_i^+ the window sizes of flow i right before and after the congestion event, respectively. Taking randomness of packet arrival process into consideration, we have $E[\sum_{i=1}^{N} W_i^-] = CT + B$. If T_a does not exceed one RTT, then there are altogether $\sum_{i=1}^{N} W_i^+$ packets pushed into the network pipe during the congestion event, and we have $M = \sum_{i=1}^{N} W_i^+ - (CT + B)$. So $E[M] = \sum_{i=1}^{N} E[W_i^+ - W_i^-]$. Since W_i is quantized, and it increases by 1 per RTT, we have $E[W_i^+ - W_i^- | T_a] = T_a/T$. Therefore, we have the following proposition:

Proposition III.1. *If a link shared by N homogeneous-RTT flows 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 why 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. This advantage of Dropfront has been qualitatively stated in [14], 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. 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. 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.2. *For heterogeneous RTT case, we have $E[M] = N$ for Droptail queue, and for Dropfront queue,*

$$\eta_{min} N \le E[M] \le \eta_{max} N\,, \tag{1}$$

[1]This may not be true for Active Queue Management (AQM), like Random Early Detection (RED). However, AQM and RED are in general not enabled at Internet routers, especially in home routers [13].

where

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

Furthermore, $Std(M)$ is also proportional to N.

From Proposition III.1 and III.2, both $E[M]$ and $Std(M)$ are always proportional to N, and we can write

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

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.

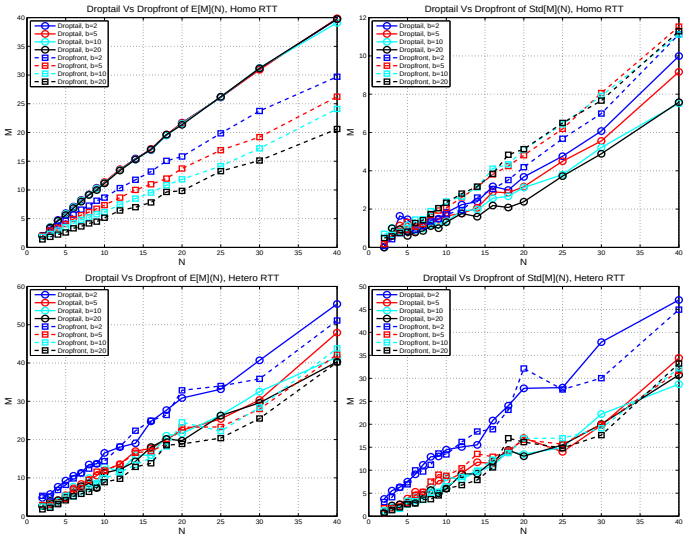

Fig. 3. $E[M]$ **and** $Std(M)$ **as functions of** N, B**.** Top row: homogeneous RTT. Bottom row: heterogeneous RTT. Left column: $Ave(M)$. Right column $Std(M)$. For all simulations, we vary N from 2 to 40, set $C = 2N$ Mbps, set $B = 2bN$ packets and vary b from 2 to 20, set $D_1 = 60$ Mbps, and set $D_N = 140$ ms for heterogeneous RTT case.

B. Number of Flows Seeing Loss (H)

We now study the mean and variance of H by first considering the conditional distribution of H given M, then extending to unconditional distribution of H.

1) Conditional Mean and Variance 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 $f(m) := E[H|M=m]$ 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 $\sum_{i=1}^N A_i \geq 1$, A_i's are not independent, and thus $g(m)$ depends on $Cov(A_i, A_j), \forall i \neq j$. 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

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

Proposition III.3. *For the conditional expectation, we have*

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

and

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

Furthermore, for conditional variance, we have

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

2) Unconditional Mean and Variance of H: For unconditional distribution, we have the following equations:

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

From Jensen's inequality, $E[f(M)] \leq f(E[M])$, 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 [15]. Then, we have the following approximations:

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

Plugging the results of Proposition III.1, Proposition III.2 and Proposition III.3 to (8) and (9), we have the following result:

Proposition III.4. *The unconditional expectation of H has the following bounds:*

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

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:

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

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

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

Furthermore, for the standard deviation of H,

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

Proposition	III.1	III.2	III.3	III.4	IV.1	IV.2
Section of Discussions	III-A1	III-A2	III-B1	III-B2	IV-C	IV-D
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)	(20)	

TABLE III
SUMMARY OF 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 (NUMBER OF DROPPED PACKETS) AND H (NUMBER OF FLOWS SEEING CONGESTION).

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:

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

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 $N \to \infty$.

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, and for both Droptail and Dropfront queue. For our simulations, we choose 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 summarize typical examples of M (total number of dropped packets) and H (number of flows seeing loss) with regard to N (total number of flows) in Figure 3 and 4, respectively.

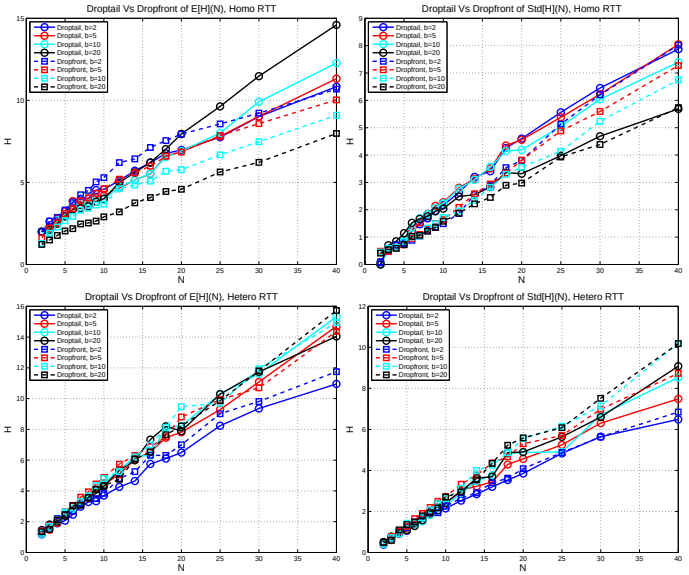

Fig. 4. $E[H]$ **and** $Std(H)$ **as functions of** N **and** B. Top row: homogeneous RTT users. Bottom row: heterogeneous RTT users. Left column: $E[M]$. Right column: $Std(M)$. Simulation setups are the same as those in Figure 3

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. Other simulations in [10] show that these properties hold for a wide range of scenarios (combination of N, C, B, D_1, D_N, and queueing policy) and the distribution of M and H do not depend on the volume of background traffic.

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 have been a few modeling and measurement studies [8], [16], [17] on TCP RTTs, and the following properties of TCP RTTs are generally observed from measurement studies: 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 sessions with abnormally large RTT samples. Therefore, in this paper, we model the measured queueing delay to be sum of the real queueing delay and a noise term. As the instantaneous 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 average window do not change and the queueing delay sequence is Martingale, and assume that the moving window contains sufficiently large number of samples. From the Martingale Central Limit Theorem, we can model the current delays to be Gaussian random variables: $q_i = n_i + s_i$, 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: $q_i = n_i + s_i + \omega_i$, where ω_i is a random variable whose expectation is much larger than μ_i and μ_n. We assume that during each query period, only a small proportion of flows yields extreme values.

We now study how to use $\{q_i, \forall i\}$ to detection congestion location. Assume q_i's follow 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 containing the extra term ω_i. For local congestion, 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 further divide delay based CLD into two modules: Outlier Identification and Removal (OIR) Module and Hypothesis Testing (HT) Module.

B. Outlier Identification and Removal

There has been a lot of algorithms for outlier identification and removal, like Chauvenet's criterion [18], Peirce's criterion [19], Hampel's identifier [20], [21], etc. Major metrics for outlier identifer are *masking* (miss) and *swamping* (false positive) probabilities [21]. We choose 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 first briefly introduce Hampel's Identifer. Suppose we have N samples: $\mathbf{X}_N = (X_1, \cdots, X_N)$. Among them, $N-k$ samples are regular (inliers) with common distribution $\mathcal{N}(\mu, \sigma)$, and k samples are irregular (outliers). Neither (μ, σ) nor k is known. Let $med(\mathbf{X}_N)$ be the median of $\mathbf{X}_N$, and $mad(\mathbf{X}_N)$ 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 re-normalization factor to make the MAD value Fisher consistent [20]. 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

$$|X_i - med(\mathbf{X}_N)| \geq g(N, \alpha_N) mad(\mathbf{X}_N)\,, \tag{16}$$

where $g(N, \alpha_N)$ is a critical value that is chosen 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 takes 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 [20], [21].

2) Our Modification: The standard Hampel's identifier takes 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 take 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:

$$\begin{cases} P\left((\sum_{i=1}^N \mathbb{1}_{|q_i - med| > g_1(N, \kappa_{both}, \alpha) mad}) > \kappa_{both} N\right) < \alpha\,, \\ P\left((\sum_{i=1}^N \mathbb{1}_{q_i < med - g_2(N, \kappa_{low}, \alpha) mad}) > \kappa_{low} N\right) < \alpha\,. \end{cases} \tag{17}$$

We call the two probabilities in (17) "strong two-side swamping probability" and "strong lower-side swamping probability", respectively.

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 5. From the plot, we see that, if we set $g = 5$ for $N \leq 5$, $g = 3$ for $5 < N \leq 10$, 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 5. 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 Distribution

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 $\kappa_{both}N <$

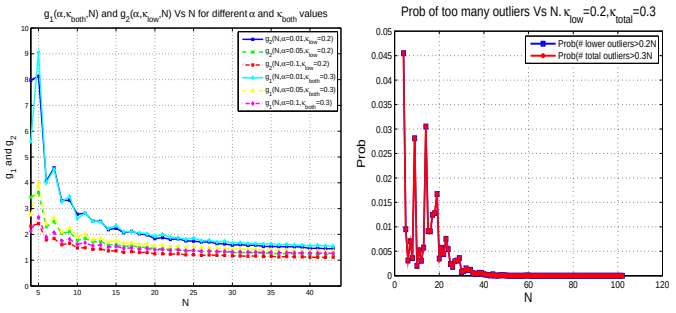

Fig. 5. $g(N, \kappa_{both}, \kappa_{low}, \alpha)$ **and** $P($**Number Outliers** $> \kappa N)$ **versus** N. Left: $g_1(N, \kappa_{both}, \alpha)$ and $g_2(N, \kappa_{low}, \alpha)$ versus N, for $\alpha = 0.01, 0.05$, or $\alpha = 0.1$, with $\kappa_{both} = 0.3$ or $\kappa_{low} = 0.2$. Right: The 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 binary hypotheses testing problem:

$$\begin{array}{ll} H_0: & q_i \sim \mathcal{N}(\mu_n, \sigma_n)\,, \forall i \in I\,, \\ H_1: & q_i \sim \mathcal{N}(\mu_{LC}, \sigma_{LC})\,, \forall i \in I\,, \end{array} \tag{18}$$

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

This 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:

$$\left\{ \begin{array}{lcl} \Gamma_1 & = & \{\mu > \alpha_1 \text{ and } \mu > \alpha_2\sigma\}\,, \\ \Gamma_0 & = & \{\mu < \alpha_1 \text{ or } \mu < \alpha_2\sigma\}\,, \end{array} \right. \tag{19}$$

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] and we have:

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:*

$$\left\{ \begin{array}{l} \mu_q > \alpha_1 \text{ and } \frac{\mu_q}{\sigma_q} > \sqrt{\frac{N-1}{N}}\alpha_2 \Rightarrow H_1\,, \\ \mu_q \leq \alpha_1 \text{ or } \frac{\mu_q}{\sigma_q} \leq \sqrt{\frac{N-1}{N}}\alpha_2 \Rightarrow H_0\,. \end{array} \right. \tag{20}$$

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

$$\left\{ \begin{array}{lcl} \alpha_1 & = & (\hat{\mu}_{LC} + \hat{\mu}_{RC})/2\,, \\ \alpha_2 & = & (\hat{\mu}_{LC}/\hat{\sigma}_{LC} + \hat{\mu}_{RC}/\hat{\sigma}_{RC})/2\,. \end{array} \right. \tag{21}$$

D. Summary: CLD Accuracy

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

Proposition IV.2. *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 IV.2 gives both the parameter configuration guideline and asymptotical result on detection accuracy. From our simulations, we see that the accuracy of CLD is insensitive to parameter configurations as long as they are set within a reasonable wide range.

V. NS-2 Simulations and Validation

We perform simulations for the following network scenario illustrated in Figure 6. As shown in the left plot, the network has three links L_1, L_2 and L_3. There are N_1 and N_2 persistent flows passing $L_1 + L_2$, and $L_1 + L_3$, respectively. There are further 3 groups of M on-off flows passing the single link L_1, L_2 and L_3, and are turned on one at a time at slot 2, 4, 6, respectively. The link capacities and the receiver window sizes are chosen 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 more noisy than those in ns-2, to test robustness of the CLD algorithm, we add noise (either Gaussian or uniformly distributed) into each RTT measurement. For our simulations, we choose 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. Due to space limitation, we summarize typical examples in Figure 7. The left figure gives the detection result for one simulation setup of $N_1 = N_2 = 9$ and $n = 8$ ms. In this figure, positive values represent local congestion detection and negative values represent remote detection, and CLD gives correct detection except two out of hundreds of congestion

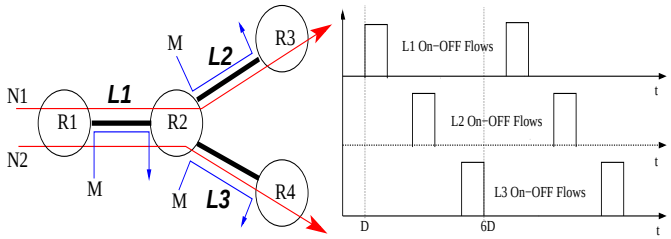

Fig. 6. **Simulations Scenario**. (Left) topology of simulations: three links, N_1 and N_2 two-hop flows, and M single-hop ON-OFF flows passing each individual link. (Right) Pattern of On-Off flows: second, fourth and sixth phases are on phases for L_1, L_2, and L_3 on-off flows, respectively.

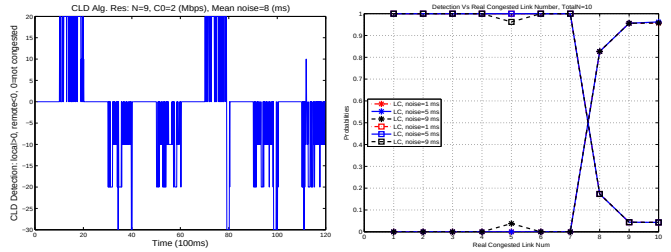

Fig. 7. **Examples of CLD Detection Result**. (Left) Detection of one simulation with $N_1 = N_2 = 9$ and noise level = 8 ms. The x-axis is time, and y-axis gives the detection result: 0 for no congestion, positive values for local congestion and negative values for remote congestion. Right: Detection Result Vs Number of Congested Links. 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 $N_c < 8$, the detection is very accurate.

events. The right figure gives a summary of detection as N_c varies while N is fixed. From this figure, as long as $N_c < \min(1-\kappa_{low}, \kappa_{QD})N = 0.8N$, the detection is accurate even with large noise: the false detection probability is less than 1% when $N_1 \neq N_2$, and is less than 5% if $N_1 = N_2$. The performance degradation when $N_1 = N_2$ is because the OIR Module is less likely to remove outliers for this special case, and thus the hypothesis testing assumption of identical inlier distributions no long holds. For the $N_1 = N_2 = N/2$ worst case, we vary N and n, and the CLD performance is demonstrated in Figure 8, which shows that as long as $N_1 = N_2 > 2$ and the noise level is less than the RTT level, the false local or false remote probability is always less than 5%. Furthermore, the false detection probabilities increase as noise level increases, and decrease as flow number increases.

Due to space constraint, more extensive simulation results are shown in [10], with different network topology, different combinations of N, C, B, T, noise distribution and magnitude, RTT heterogeneity and queueing policies, and different parameter configurations. They show that our CLD algorithm gives very accurate detection for a wide range of network topology and scenarios, and is insensitive to parameter configurations as long as they lie in a wide reasonable range.

VI. Conclusion and Future Work

We proposed an end application Congestion Location Detection (CLD) algorithm, which may be used for network diagnosis, server QoS monitoring, and low priority TCP. We described the CLD algorithm in details, showed that it can accurately detect whether the congested link is local or

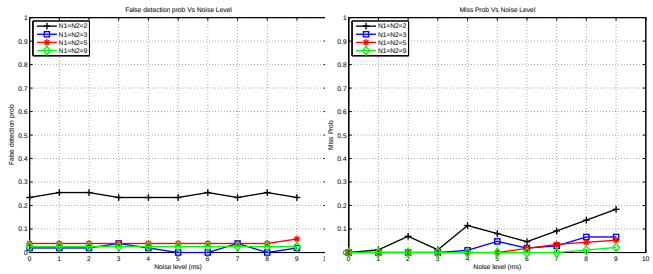

Fig. 8. **Detection Result Vs Noise Level**. The x-axis is the noise level, and the y-axis gives false detection probabilities. (Left) False local detection probability. (Right) False remote detection probability. As long as $N_1 = N_2 \geq 3$, the detection is accurate.

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.

References

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

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

[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. IMC*, 2003.

[9] J. Aikat, J. Kaur, F. Smith, and K. Jeffay, "Variability in tcp round-trip times," in *In Proc. IMC*, 2003.

[10] S. Liu, M. Chiang, M. Jourdain, and J. Li, "Congestion location detection: Methodology, methodology, performance, and applications," MSR Techinical Report, 2008. Available at http://www.princeton.edu/~shaoliu/cldtr.pdf.

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

[12] P. Key et al, "Round trip time estimation," available at http://www.freepatentsonline.com/EP1744495.html.

[13] M. Dischinger et al, "Characterizing residential broadband networks," in *Proc. IMC*, 2007.

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

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

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

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

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

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

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

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