Shared Bottleneck Detection Based on Congestion Interval Variance Measurement

Wenjia Wei, Yansen Wang, Kaiping Xue, *Senior Member, IEEE*, David S. L. Wei, *Senior Member, IEEE*, Jiangping Han, and Peilin Hong

***Abstract*— The key technical issue of bottleneck fairness-based multipath congestion control is to detect whether two flows share a common bottleneck or not. Previous approaches perform poorly when the delay of the two paths from the shared bottleneck to the common receiver differs significantly (named the path lag problem). In this letter, we propose and implement a shared bottleneck detection scheme based on congestion interval variance measurement (SBDV). Our scheme uses only one-way delay measurement within each flow to detect whether two flows share a bottleneck or not. If the variance of time interval between the two flows experiencing congestion is smaller than a threshold, which is determined by the duration of congestion, the two flows can be considered to share a common bottleneck. Otherwise, they are considered to have different bottlenecks. Our simulation shows that the accuracy of SBDV is high in most of the experiments, even when the bottlenecks are partially overlapped, and our scheme is robust to the troublesome Path lag problem, as compared to other state-of-the-art techniques.**

***Index Terms*— Multipath TCP, one-way delay, shared bottleneck detection.**

I. Introduction

AN INCREASING number of network devices are now equipped with multiple communication interfaces which enable multiple paths for data transfer between a source and a destination. Since the concurrent transmission across multiple available paths can offer higher throughput and increase the resilience of the connectivity, multihoming-capable transport protocols, like MPTCP [1], have incentivized much research and standardization work in recent years.

A *multipath flow* is usually composed of more than one flow, where each flow corresponds to a path between the source and the destination. According to the bottleneck fairness criterion of multipath congestion control [2], if subflows of a multipath flow traverse a common bottleneck, they should be no more aggressive than a single TCP flow through the bottleneck. Otherwise, if they have different bottlenecks, they should seek to independently maximize the throughput along their paths.

A key technical issue underlying multipath congestion control which satisfies the bottleneck fairness is to detect whether two flows share a common bottleneck or not. However, the existing schemes cannot conduct accurate detection effectively. Rubenstein *et al.* [3] proposed two active techniques, one is based on One-Way Delay (OWD) measurement, and the other is based on packet losses. These two techniques both generate Poisson probe packets to measure the auto-correlation of packets within flows and the cross-correlation of the adjacent packets between flows. Then, by comparing the cross-correlation with the auto-correlation, whether the two flows share a bottleneck or not can be decided. However, deciding which packets from the two flows are adjacent at the bottleneck is not a simple task. The method in [4], based on the work proposed in [3], applies wavelet denoising on the original delay sequences to make the detection more robust. Katabi *et al.* [5] used the entropy of packet inter-arrival time to group flows into different clusters, where each cluster corresponds to a bottleneck. However, their technique only gives acceptable results in the case of low cross traffic and does not scale well with heavy cross traffic.

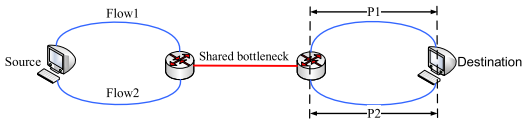


Fig. 1. A typical shared bottleneck scenario.

Recently, Hassayoun *et al.* [2] proposed a simple heuristic for detecting the shared bottleneck in MPTCP, named DWC, where two flows whose time interval between experiencing their congestion event (loss or significant increase in RTT) is smaller than $cwnd/2$ (about $RTT/2$) are considered to share a common bottleneck. However, DWC can't make accurate detection in scenarios where the delay of the two paths from the shared bottleneck to the common receiver differs more than $cwnd/2$. Another work, MPTCP-SBD [6], uses estimates of three summary statistics (i.e., variance, skewness, key frequency) of OWD measurement to detect the shared bottleneck, where flows whose summary statistic values are close can be considered to share the same bottleneck. This method assumes that different bottlenecks experience different loads, which does not always hold true in real network scenarios.

Path lag is a problem when a the delay of the two paths from the shared bottleneck to the common receiver (P1 and P2 in Fig. 1) is unequal, and is a troublesome problem in shared bottleneck detection. At the receiver, path lag makes the OWD statistics of the two flows look uncorrelated (as shown in Fig. 2). Existing approaches often show lower accuracy when the difference in delay of the two paths becomes larger. In this letter, we propose a Shared Bottleneck Detection scheme based on congestion interval Variance measurement (SBDV), which operates at the common receiver

Manuscript received September 8, 2018; accepted September 20, 2018. Date of publication October 1, 2018; date of current version December 10, 2018. This work is supported in part by National Key Research and Development Plan of China under Grant 2017YFB0801702, the National Natural Science Foundation of China under Grant 61671420, Youth Innovation Promotion Association of the Chinese Academy of Sciences under Grant 2016394 and the Fundamental Research Funds for the Central Universities. The associate editor coordinating the review of this paper and approving it for publication was V. Eramo. *(Corresponding author: Kaiping Xue.)*

W. Wei, Y. Wang, K. Xue, J. Han, and P. Hong are with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, China (e-mail: kpxue@ustc.edu.cn).

D. S. L. Wei is with the Computer and Information Science Department, Fordham University, New York, NY 10458 USA.

Digital Object Identifier 10.1109/LCOMM.2018.2872977

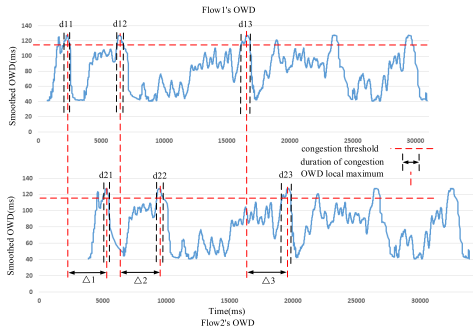


Fig. 2. Smoothed OWD statistics of the two flows at the receiver.

and is robust to the path lag problem. Congestion at the shared bottleneck can lead to significant increase in OWD measurement of the two flows. Considering the path lag, the significant increase in OWD of the two flows could not occur simultaneously at the receiver. But the time interval between the significant increase of the two flows should be approximately the same for each congestion event at the shared bottleneck. In this letter, we construct a threshold with the following property: If the two flows share a bottleneck, the variance of this time interval between the two flows is smaller than the threshold, which is determined by the duration of the congestion. If the two flows have different bottlenecks, the variance of this time interval is very likely to be larger than the threshold. Extensive experiments via NS-3 simulator show that the accuracy of SBDV is high, as compared with other state-of-the-art techniques.

II. Technique Description

In this section, we present our proposed scheme, namely SBDV. Informally, a bottleneck is the congested link that drops or excessively delays packets due to overflowing of queue. We assume that flows share no more than one bottleneck. For the convenience of presentation, we use the scenario with two flows to illuminate the details of our scheme.

A. Basic Scheme

Considering the shared bottleneck scenario in Fig. 1, let t_0 be the time when the shared bottleneck is congested. When packets of the two flows traverse the shared bottleneck at t_0, they would experience significant queuing delay. Making use of the sending timestamp carried in packets, the receiver could recognize the congestion event at the shared bottleneck by the time when flow's OWD measurement reaches local maximum. Let t_1 be the time when flow1's OWD reaches its local maximum, t_2 be the time when flow2's OWD reaches its local maximum. $t_1 - t_0$ corresponds to the delay of path P1 while $t_2 - t_0$ corresponds to the delay of path P2 such that $\Delta = t_2 - t_1 = (t_1 - t_0) - (t_2 - t_0)$ corresponds to the difference in delay of the two paths P1 and P2. It is worth noting that Δ is also the time interval between the receiver's recognized congestion events (by local maximums of OWD) of the two flows. Therefore, Δ can be measured by using only OWD measurement at the receiver.

Since the delay of non-congested links (P1 and P2) is relatively stable when compared with bottleneck links, the value of Δ can be approximately regarded as a constant in a specific network topology and its variance $Vari(\Delta)$ is theoretically close to 0. The value of $Vari(\Delta)$ reflects the fluctuation intensity of queuing delay on the non-congested links, *i.e.*, P1 and P2. In this letter, we construct a threshold value with the following property: If the two flows are sharing a bottleneck, $Vari(\Delta)$ is smaller than the threshold value. Otherwise, they are very likely having different bottlenecks.

Due to the fact that congestion events often last for a short while, the local maximum of OWD can appear at any point in the duration of a congestion event. The duration time of a congestion event, observed by receiver, is also affected by the queuing delay on the non-congested links from the shared bottleneck to the receiver. Let d_{1i} be the i-th congestion event duration time of flow1, and d_{2i} be the i-th congestion event duration time of flow2, if the two flows share a bottleneck, $\forall i$, we have $\left|\Delta_i - \overline{\Delta}\right| \leqslant |d_{1i} + d_{2i}|$ ($\overline{\Delta}$ is approximated as not changing due to the change of specific Δ_i). Therefore $Vari(\Delta) = \sum_{i=1}^{N} \frac{\left(\Delta_i - \overline{\Delta}\right)^2}{N} \leqslant \sum_{i=1}^{N} \frac{(d_{1i}+d_{2i})^2}{N}$. On the other hand, if the two flows have different bottlenecks, congestion events of the two flows have no correlation at all. As a result, Δ could be an arbitrary value. It is very likely that $Vari(\Delta) > \sum_{i=1}^{N} \frac{(d_{1i}+d_{2i})^2}{N}$. In our simulation, we will show that $\sum_{i=1}^{N} \frac{(d_{1i}+d_{2i})^2}{N}$ is an appropriate threshold for determination.

B. Implementation

We use the timestamp option to acquire OWD statistics of the two flows and no clock synchronization between the sender and the receiver is required. After every detection cycle T, our technique runs with OWD statistics collected over the last T to give a result: shared or non-shared. A flow enters a congestion event when its smoothed OWD, OWD_{S}, is greater than the congestion threshold, OWD_{cth}. When its OWD_{S} becomes smaller than OWD_{cth}, the flow exits this congestion event. A flow is considered to be non-congested when its OWD_{S} is smaller than the non-congestion threshold, OWD_{nth}, which is smaller than OWD_{cth}. $OWD_{\mathrm{S}}(i) = (1 - \alpha) \cdot OWD_{\mathrm{S}}(i-1) + \alpha \cdot OWD_{\mathrm{M}}(i)$, where OWD_{M} is the latest measured OWD and $\alpha = 1/8$. Here, α is an empirical value, which refers to the smoothed value used for the average RTT measurement. OWD_{nth} is computed as follows: $OWD_{\mathrm{nth}} = OWD_{\mathrm{avg}} + \beta \cdot OWD_{\mathrm{std}}$, where OWD_{avg} is the average value of OWD_{S}, and OWD_{std} is the standard deviation of OWD_{S} in T. We set $\beta = 0.5$ in our simulation, and it has little effect on the experimental results. OWD_{nth} divides OWD_{S} statistics into congestion episode and non-congestion episode alternately. Non-congestion episode is the part of OWD_{S} values which are smaller than OWD_{nth}. OWD_{cth} is updated when $OWD_{\mathrm{S}}(i) > OWD_{\mathrm{nth}}$ and $OWD_{\mathrm{S}}(i+1) < OWD_{\mathrm{nth}}$ using $OWD_{\mathrm{cth}} = OWD_{\mathrm{min}} + \gamma \cdot (OWD_{\mathrm{max}} - OWD_{\mathrm{min}})$, where OWD_{min} is the minimum of OWD_{S} in the next non-congestion episode, and OWD_{max} is the maximum of OWD_{S} in the next congestion episode. $0 < \gamma < 1$, and we set $\gamma = 0.9$ as it is found the most effective parameter value in simulation experiments.

Each congestion episode whose maximum of OWD_{S} (OWD_{max}) is greater than OWD_{cth} is considered to correspond to a congestion event. The time of OWD_{max} is used as the occurrence time of the corresponding congestion event. The time interval during which OWD_{S} is greater than OWD_{cth} with respect to OWD_{max} is used as the duration time of the congestion event. Therefore, we can get the number of congestion events and the corresponding occurrence time, and duration time in a detection cycle of the two flows.

Ideally, if the two flows share a bottleneck, the respective numbers of congestion events experienced in the same bottleneck is supposed to be identical in a detection cycle. However, due to the queuing delay fluctuation on the non-congested links and the congestion events experienced on other links, the numbers of their respective congestion events, denoted as $M1$ and $M2$, may be unequal even if the shared bottleneck does exist. In this situation, we use **Algorithm 1** to interpolate the array with a smaller size in order to let the two arrays have the same size after interpolation. Then we compute $Vari(\Delta)$ and $\sum_{i=1}^{N} \frac{(d_{1i}+d_{2i})^2}{N}$, and the total computational complexity is $O(C_{M2}^{M2-M1} \cdot M2)$. To be noted, in a measurement cycle, the number of congestion events is limited, so the overhead of the proposed algorithm is acceptable.

Algorithm 1 Compute $Vari(\Delta)$ and QuadraticMean

Input:
The array of congestion events occurrence time in the two flows, *ToC1[ ]*, *ToC2[ ]*;
The array of congestion events duration time in the two flows, *DoC1[ ]*, *DoC2[ ]*;
The size of *ToC1[ ]*, *ToC2[ ]*, $M1$, $M2$;
`// We assume` $M1 < M2$
Output: $Vari(\Delta)$, *QuMean* //QuadraticMean
Initialize: $Vari(\Delta) = 0$, $QuMean = 0$
Find an array *interp[M2]* whose elements are positive integers such that $\sum_{i=1}^{M2}(ToC2[i] - ToC1[interp[i]] - aver)^2$ is the smallest, where the average value $aver = \sum_{i=1}^{M2}(ToC2[i] - ToC1[interp[i]])/M2$, $0 \le interp[i+1] - interp[i] \le 1$, $interp[1]$=1 and $interp[M2]$=$M1$.
for i =1 **to** $M2$ **do**
  $Vari(\Delta) += (ToC2[i] - ToC1[interp[i]] - aver)^2$
  $QuMean += (DoC2[i] - DoC1[interp[i]])^2$
end
$Vari(\Delta) = Vari(\Delta)$ / $M2$
$QuMean = QuMean$ / $M2$
return $Vari(\Delta)$, *QuMean*

Also to be noted that if the number of flows is more than two, our scheme can be extended easily. For each flow in a set, if any two of them are detected to share a bottleneck, we can say that all flows in this set share the same bottleneck. Otherwise, they don't share any bottleneck. Furthermore, In multi-flow scenario, we can get different sets, where each one is corresponding to a shared bottleneck flow set.

III. Performance Evaluation

It is difficult to verify our technique on a real network because we could not find out whether the two flows really share a bottleneck or not to confirm the correctness of our scheme. Therefore we evaluate our scheme's accuracy and adaptivity via NS-3 simulator and compare it with DWC [2] and MPTCP-SBD [6].

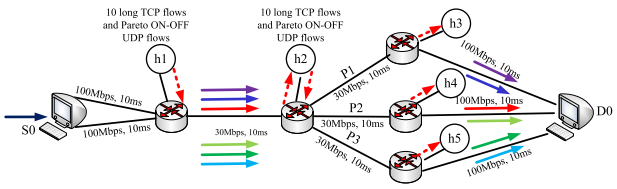


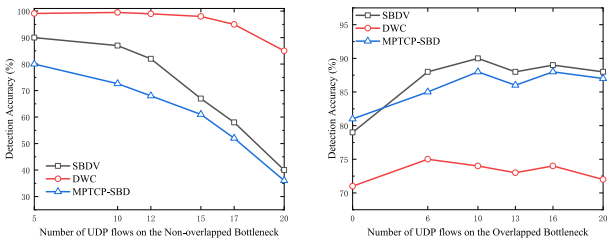
Fig. 3. Simulation topology.

A. Simulation Setup

Fig. 3 shows the simulation topology where we can construct a Shared Bottleneck (SB) scenario and a Non-Shared Bottleneck (NSB) scenario. The bottlenecks' bandwidth is set to 30Mbps and the link delay is set to 10ms. Bandwidth of other links is set to 100Mbps and their link delay is set to 10ms. All links are loss free. The Drop-Tail buffer size at bottlenecks is set to 600 packets. Packet size is 536 bytes. An MPTCP device S0 generates 6 flows to D0 through three different paths in which each path is loaded with two flows. To reflect the real network condition, the background traffic consists of a mixture of long-lived TCP flows and bursty UDP traffic. In each bottleneck, we set 10 TCP flows and 5-25 Pareto ON-OFF UDP flows, with shape parameter 1.5 and mean 400ms "ON" and "OFF" time, to compete with the MPTCP flows. The sending rate of UDP flows is uniformly distributed between 500 and 700 kbps. Detection Cycle T is set to 20s. Furthermore, we also give the performance analysis with different values of T.

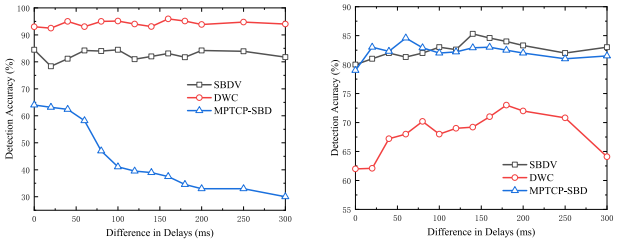
B. Detection Accuracy With Bottlenecks Partially Overlapped

We set the delay of links P1, P2 and P3 to 10ms to avoid the path lag problem. In the SB scenario, 10 TCP flows and 15 Pareto UDP flows are sent from h1 to h2. The packet loss rate at the shared bottleneck is about 0.5-1.5%. The numbers of UDP flows sent from h2 to h3, h4, and h5, respectively, are all increased from 0 to 20 to construct the scenario where the bottlenecks of the six flows are partially overlapped. When the numbers of UDP flows sent from h2 to h3, h4, and h5, respectively, are all 20, packet loss rate at links P1, P2, and P3 is about 0.1-0.8%. The shared bottleneck makes congestion events of the two flows correlated, while the non-shared bottlenecks makes congestion events of the two flows uncorrelated. The packet loss rate at the shared bottleneck is higher and the two flows are considered to share a bottleneck. Fig. 4(a) shows the accuracy of the three schemes with the number of background UDP flows on links P1, P2, and P3 varying from 0 to 20 in the SB scenario. The accuracy of all the three schemes is reduced with increasing number of UDP flows on the non-overlapped bottleneck, as the increasing number of background UDP flows causes the flows to behave less correlated. With a large observation window (about one RTT) and a low R_{th} (It is defined in DWC and represents the congestion threshold.), DWC often correlates the related and even the disrelated congestion events of the two flows to draw a shared bottleneck conclusion. Thus, DWC performs the best in the SB scenario, but it increases error detection. However, in the NSB scenario, DWC with a large observation window and a low R_{th} of DWC will have a negative effect and lead to poor accuracy in the NSB scenario. SBDV performs better than MPTCP-SBD with light background traffic load.



(a) SB scenario (b) NSB scenario

Fig. 4. Detection accuracy with bottlenecks partially overlapped in the SB and NSB scenario.



(a) SB scenario (b) NSB scenario

Fig. 5. Detection accuracy with difference in delay varying in the SB and NSB scenario.

With heavy background traffic load, the detection accuracy of both is not acceptable.

In the NSB scenario, 10 TCP flows and 25 Pareto UDP flows are sent from h2 to h3, h4, and h5, respectively. Packet loss rate at the links P1, P2, and P3 is about 0.3-0.8%. The number of UDP flows sent from h1 to h2 is increased from 0 to 20. When the number of UDP flows sent from h1 to h2 is 20, packet loss rate at the shared bottleneck is about 0.3-0.8%. Fig. 4(b) shows the accuracy of the three schemes with the number of background UDP flows on the shared bottleneck varying from 0 to 20 in the NSB scenario. As we discussed above, with a large observation window and a low R_{th}, DWC often correlates the disrelated congestion events of the two flows and falsely concludes that they share a bottleneck. Therefore, DWC performs the worst in the NSB scenario. SBDV and MPTCP-SBD both have the accuracy of 86% on average in this scenario.

C. Detection Accuracy With Difference in Delay Increasing

We increase the delay difference of links P1, P2, and P3 in Fig. 3 to evaluate the path lag problem of the three schemes. In the SB scenario, 10 TCP flows and 15 Pareto UDP flows are sent from h1 to h2, and 10 TCP flows and 12 Pareto UDP flows are sent from h2 to h3, h4, and h5, respectively. The changing of queuing delay at links P1, P2, and P3 makes it difficult to draw the right conclusion that the six flows share the same bottleneck. Fig. 5(a) shows the accuracy of the three schemes with difference in delay of links P1, P2, and P3, increasing from 0 to 300ms, in the SB scenario. We observe that SBDV outperforms MPTCP-SBD even when the difference is 0, which is the best result of MPTCP-SBD. With the increasing difference, the accuracy of SBDV is still high and stable, while MPTCP-SBD deteriorates rapidly, which is affected by path lag problem. With an overlarge observation window and a low R_{th}, DWC still performs the best even when the difference is 300ms, but it increases error detection as we have explained above.

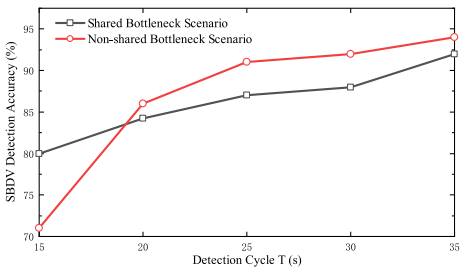


Fig. 6. SBDV detection accuracy under different detection cycle.

In the NSB scenario, 10 TCP flows and 25 Pareto UDP flows are sent from h2 to h3, h4, and h5, respectively. 10 TCP flows and 5 UDP flows are sent from h1 to h2. Fig. 5(b) shows the accuracy of the three schemes with difference in delay of links P1, P2 and P3, increasing from 0 to 300ms, in the NSB scenario. Since the six flows have three distinctive bottlenecks, the increase of difference in delay has no obvious effect on the accuracy of these three schemes. DWC performs the worst, while the detection accuracy of SBDV and MPTCP-SBD is nearly 86% in the NSB scenario.

D. SBDV Detection Accuracy Under Different Detection Cycle

In the previous simulation, we set the detection cycle T of SBDV to 20s. The value of T is a tradeoff between the accuracy and responsiveness of SBDV. A larger T leads to higher accuracy but less responsiveness, and it also takes more time for SBDV to react to the changes of network conditions. Fig. 6 shows SBDV's accuracy with T increasing from 15s to 35s. SBDV is more accurate in the SB and NSB scenario when T becomes larger. This is because SBDV can collect more OWD statistics to draw a conclusion.

IV. Conclusion

In this letter, we presented and realized a shared bottleneck detection scheme which uses only one-way delay measurement within each flow. Compared with previous approaches which perform poorly to the path lag problem, our proposed SBDV is more robust and effective. Extensive experiments via NS-3 simulator show that SBDV outperforms MPTCP-SBD in the SB scenario and DWC in the NSB scenario. Moreover, SBDV is more accurate with a larger detection cycle.

References

[1] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, *TCP Extensions for Multipath Operation With Multiple Addresses*, document IETF RFC 6824, 2018.

[2] S. Hassayoun, J. Iyengar, and D. Ros, "Dynamic window coupling for multipath congestion control," in *Proc. 19th IEEE Int. Conf. Netw. Protocols (ICNP)*, Oct. 2011, pp. 341–352.

[3] D. Rubenstein, J. Kurose, and D. Towsley, "Detecting shared congestion of flows via end-to-end measurement," *IEEE/ACM Trans. Netw.*, vol. 10, no. 3, pp. 381–395, Jun. 2002.

[4] M. S. Kim, T. Kim, Y.-J. Shin, S. S. Lam, and E. J. Powers, "A wavelet-based approach to detect shared congestion," *IEEE/ACM Trans. Netw.*, vol. 16, no. 4, pp. 763–776, Aug. 2008.

[5] D. Katabi, I. Bazzi, and X. Yang, "A passive approach for detecting shared bottlenecks," in *Proc. IEEE Int. Conf. Comput. Commun. Netw. (ICCCN)*, Oct. 2001, pp. 174–181.

[6] S. Ferlin, Ö. Alay, T. Dreibholz, D. A. Hayes, and M. Welzl, "Revisiting congestion control for multipath TCP with shared bottleneck detection," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Apr. 2016, pp. 1–9.