IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received February 27, 2019, accepted February 28, 2019, date of current version March 25, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2902860

Fine-Grained Congestion Control for Multipath TCP in Data Center Networks

JIN YE[1], LUTING FENG[1], ZIQI XIE[1], JIAWEI HUANG[2], AND XIAOHUAN LI[3,4]

[1]Guangxi Key Laboratory of Multimedia Communications and Network Technology, School of Computer, Electronics and Information, Guangxi University, Nanning 530004, China
[2]School of Information Science and Engineering, Central South University, Changsha 410083, China
[3]School of Electronic and Information Engineering, Beihang University, Beijing 100083, China
[4]School of Information and Communication, Guilin University of Electronic Technology, Guilin 541004, China

Corresponding author: Xiaohuan Li (xhuan_lee@126.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61762030, Grant 61872387, and Grant 61572530, and in part by the China Education and Research Network Innovation Project under Grant NGII20160113, Grant NGII20170107, and Grant NGII20180305.

ABSTRACT In data center networks, MultiPath TCP (MPTCP) obtains both higher network utilization and fairer allocation of capacity by exploring multiple paths simultaneously. However, MPTCP experiences more queue oscillation in switch buffer with the increasing of the number of subflows. When using explicit congestion notification as the congestion indication to track the queue length in switch buffer, it is difficult for MPTCP to capture the accurate congestion state, resulting in wrong behavior in congestion control. Therefore, we propose an enhanced MPTCP protocol, namely, advanced MPTCP (AMP), which adjusts the time granularity of the congestion detection and control under a different number of subflows. The test results show that compared with Linked Increases Algorithm and eXplicit MultiPath, AMP achieves lower latency for small flows and higher throughput for large flows.

INDEX TERMS Data center networks, explicit congestion notification, MultiPath TCP.

I. INTRODUCTION

As the platform of cloud computing and the next generation networks, Data Center Networks (DCNs) plays an important role in providing diverse network services [1]–[4]. A significant number of online service providers like Amazon, Google, and Microsoft construct their data centers all around the world. In data center, the network traffic is classified into two types according to the flow size. The first one is large flow (a flow that consists of a large number of packets), which are usually generated by the virtual machine migration and massive data synchronization. The large flows are bandwidth-greedy, caring about the high link utilization. The second one is small flow usually created by request-based applications, such as Web search and MapReduce [5]. The small flows are short (i.e.,50KB to 1MB in size) and time-sensitive with soft real-time constraints, such as deadline requirements. If the transmission completion time exceeds the deadline, the user quality of experience (QoE) will be seriously degraded, resulting in decline of business revenue [6]–[10].

Unfortunately, the consistent trend in data center designs is to build high-performance computing and storage infrastructure using low-cost commodity components. For saving cost, these switches normally have small-size SRAM packet buffers. When the switch buffers are shared by both large and small flows, it is very common that the small switch buffer is occupied by a few number of large flow, leading to large queuing delay and TCP incast problem [11]–[14]. In order to satisfy the latency requirement of small flows, the low buffer occupation should be achieved. However, the small buffer queue length will decrease the link utilization, with the result that large flows cannot obtain high throughput [15].

To satisfy both latency and throughput requirements for different applications, MultiPath TCP (MPTCP) utilizes the multiple available paths that single-path TCP cannot use [16]–[18]. In MPTCP, the subflows take different paths to transfer packets simultaneously, effectively and seamlessly using available bandwidth [19]–[21].

However, the default congestion control algorithm of MPTCP is Linked Increases Algorithm (LIA), which is designed for traditional Internet [22]. In data center networks, LIA consumes switch buffer space, thus increasing

The associate editor coordinating the review of this manuscript and approving it for publication was Min Chen.

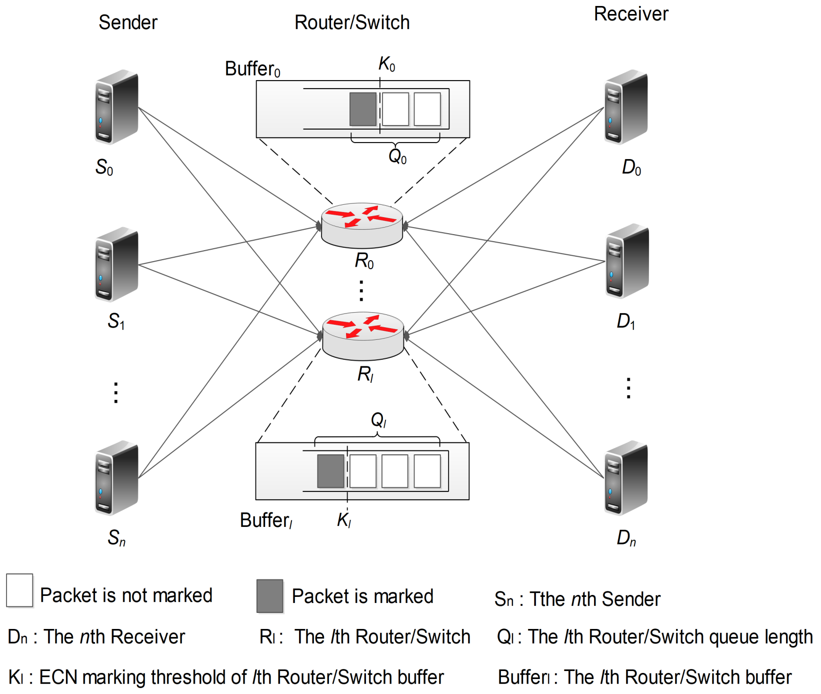


FIGURE 1. ECN-based MPTCP.

the round-trip time (RTT). In addition, when congestion happens, LIA blindly reduces the congestion window by half, easily resulting in low link utilization [23], [24]. To solve this problem, explicit MultiPath (XMP) [25] uses Explicit Congestion Notification (ECN) [26] to make quick response when congestion happens [27], [28]. Specifically, Buffer Occupancy Suppression (BOS) algorithm is used to maintain small buffer occupation and reduce queuing delay for small flows. To improve throughput for large flows, the traffic shifting (TraSh) algorithms is also used for shifting traffic from more congested links to less congested ones.

In this work, we argue that existing MPTCP protocols ignore the problem that MPTCP will experience more queue oscillation in switch buffer with the increasing of number of highly concurrent subflows. Through empirical studies, we reveal that it is difficult for MPTCP to capture the accurate congestion state in the presence of severe congestion. Therefore, we propose an advanced MultiPath TCP (namely AMP), which adjusts the time granularity of congestion detection and control. The evaluation results show that AMP reduces the transmission delay of small flows and improves the throughput of large flows.

The remainder of the paper is organized as follows. In section II, we discuss shortcomings of MPTCP protocols by experiments. In section III, we summarize related work briefly. In section IV, we propose the AMP algorithm. Section V is dedicated to simulation test and performance analysis. Finally, we make conclusion of this paper in section VI.

II. MOTIVATION

In MPTCP, the single connection is striped across multiple network paths. Once MPTCP subflows have been established, the sender stripes data across the subflows. Each subflow has its own sequence space and maintains its congestion window so that it can adapt to congestion state along the path. If the MPTCP protocol uses the ECN as the congestion notification (i.e., XMP), each subflow will perform TCP-like additive increase and multiplicative decrease on ECN message. Specifically, as shown in Fig. 1, each sender has its subflows going through the core switch. If the queue length Q of the switch buffer exceeds a given threshold K, the ECN messages will be sent back to the senders by the destination nodes. Once receiving the ECN message, the sender will adjust its sending rate. This operation actively moves traffic from more congested paths to less congested ones, achieving the network load-balance.

Today's larger data centers have tens of thousands of hosts, which inevitably brings about high flow concurrency. Taken the 44 ports ToR switch as an example, the median number of concurrent flows is 36. In the multi-layer partition/aggregate pattern, the 99.99th percentile is even over 1,600 [29]. Obviously, MPTCP makes the flow concurrency become much higher as it uses multiple subflows to transfer data.

The high flow concurrency of MPTCP leads to insufficiency of available buffer space. To investigate the impact of MPTCP high flow concurrency in switch buffer, we conduct an experiment on a data center network testbed with its topology shown in Fig. 1. The number of subflows for each sender is increased from 4 to 8 during 3rd RTT to 8th RTT.

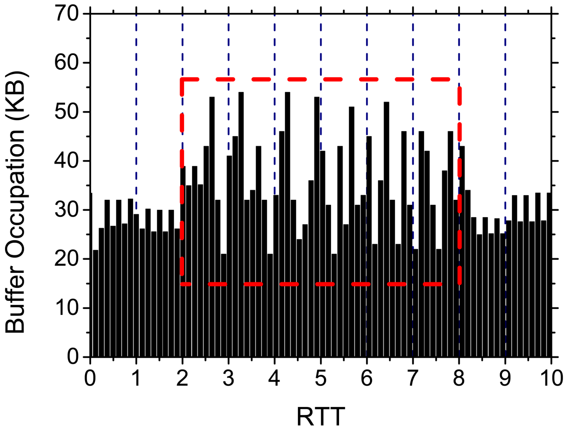


FIGURE 2. Buffer length under 4 concurrent subflows.

After 8th RTT, the number of subflows is recovered to 4. Fig. 2 describes how the queue length changed in the experiment. The changes of the queue length oscillations' frequency and the amplitude are due to the increase and decrease of the subflows numbers.

As shown in Fig. 2, the oscillations frequency of queue length is accelerated and the amplitude becomes larger. Using the traditional ECN scheme, it is hard to capture and analysis the actual congestion state. It is because adjusting the congestion window is just triggered through the ratio of ECN-marked, not considering how the ECN-marked being changed. Hence, a fine-grained detection scheme should be found, by which the adjustment of congestion window should be triggered.

In addition, we compare the queue length oscillations of MPTCP and a single path TCP. Fig. 3 shows the cumulative probability distribution of queue length oscillations. Compared with MPTCP, the queue length oscillation of the single path TCP is smaller. For MPTCP, the magnitude of oscillation becomes larger as the number of subflows increases from 4 to 8.

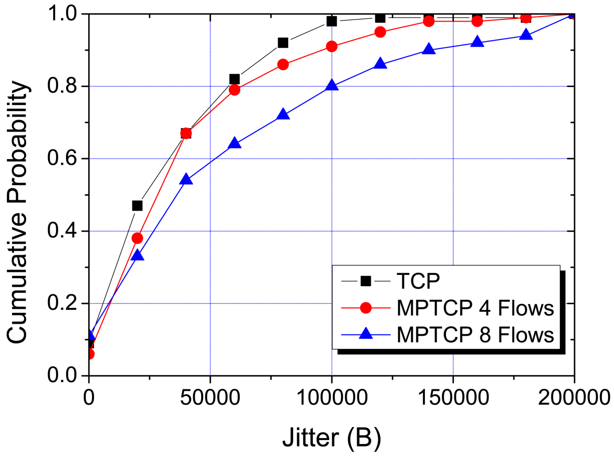


FIGURE 3. Cumulative probability distribution of queue length oscillations.

In short, the experimental study indicates that, when MPTCP is deployed, high subflows concurrency brings about fast and large queue oscillations in switch buffer, making it hard for traditional ECN scheme to track the congestion state when using fixed time granularity in congestion detection. This is the fundamental drawback in current MPTCP protocol. It motivates us to explore more elaborate congestion detection and control.

III. RELATED WORKS

In order to make the tradeoff between latency-sensitive and throughput-sensitive flows, researchers have proposed lots of solutions.

Alizadeh *et al.* [29] propose the DCTCP protocol, which combines the explicit congestion notification and rate adjustment of source to keep the queue length under a certain range. Though DCTCP effectively mitigates the buffer congestion, it is very sensitive to parameter setting and easily leads to unfair bandwidth allocation.

RepFlow replicates multiple copies for mice flows and transmits these flows simultaneously. It is very likely for each copy flow to be transferred on different paths. Due to the mutual independent congestion state of different paths, the probability of all copies experience congestion is very small [30]. Therefore, RepFlow reduces the transmission delay for small flows, but it will increase redundant data which impacts the elephant flows throughput in reverse.

MPTCP divides one flow into multiple subflows to transmit on parallel paths simultaneously, and it implements an united congestion control to adjust rates according to the congestion state dynamically. When some paths become congested, MPTCP will transmit data on the other available paths, achieving the load balance at transport layer.

MMPTCP scatter packet under a single congestion window when switch queue length is under a certain threshold, and switch to standard MPTCP while reaching a switch threshold [31]. In this way, the short flow completion time is reduced.

The researchers also proposed multipath transmission solutions on other layers. For instance, ECMP selects the transmission path by a hash mapping in terms of five-tuple [32]. However, due to the existence of hash collision, it is likely that several elephant flows are mapped to the same path, resulting in the large queuing delay for mice flows. DeTail combines load balance and priority flow control schemes at network layer and link layer, respectively [33]. It transfers data on multiple paths to satisfy different application requirements. DeTail monitors the queue length of each egress port on switch and records the ports whose queue length is under a certain threshold. When a packet reaches to the switch, it is forwarded to a less congested path. In this way, the network traffic is evenly distributed among multiple available paths. CONGA perceives global congestion information and splits TCP flows into flowlets, which are transmitted on different paths based on the congestion feedback from remote switches [34]. Therefore, CONGA reduces the flow completion time and improves the network throughput.

In contrast with the above protocols, our solution AMP tackles the problem that multipath transmission will experience more jitter in switch buffer under the highly concurrent subflows. By adjusting the time granularity of congestion detection and control, AMP reduces the transmission delay of small flows and improves the throughput of large flows, where existing solutions become less effective.

IV. PROTOCOL DESIGN

Since the traffic workload of data center is highly dynamic, a fixed period for sampling and controlling congestion cannot obtain the maximum efficiency under all scenarios. This section presents AMP, which adjusts the time granularity of congestion detection and control under different number of subflows.

The protocol design involves several key challenges that are addressed in this section. First, we need to obtain the time granularity of congestion detection and control, taking the queue length oscillation into consideration. Second, we need a congestion control method to deal with rapid changes of network dynamics.

A. CONGESTION DETECTION

AMP utilizes ECN message to adjust the time granularity of congestion detection and control. In ECN scheme, when the queue length of switch buffer is larger than the threshold K, the packet arriving at the switch and its corresponding ACK packet are marked with ECN bit. In our design, the sender of AMP records the ECN information of each subflow. For the ith subflow with the congestion window cw_i, its jth ACK packet $p_{i,j}$ in the current congestion window is marked with ECN bit by the destination node. Thus, the matrix M is used to record the ECN informations of whole subflows.

$$M = \begin{pmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,j} & \cdots & m_{1,s} \\ m_{2,1} & m_{2,2} & \cdots & m_{2,j} & \cdots & m_{2,s} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ m_{i,1} & m_{i,2} & \cdots & m_{i,j} & \cdots & m_{i,s} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ m_{n,1} & m_{n,2} & \cdots & m_{n,j} & \cdots & m_{n,s} \end{pmatrix} \tag{1}$$

where n is the number of subflows, s is the largest congestion window size among all subflows, and $m_{i,j}$ is as:

$$m_{i,j} = \begin{cases} 0, & packet\ p_{i,j}\ is\ not\ marked; \\ 1, & packet\ p_{i,j}\ is\ marked; \\ \varnothing, & cw_i < j. \end{cases} \tag{2}$$

When the number of subflows increases, there will be larger queue length oscillations. We need fine-grained reflection of the queue length oscillation. Thus, We utilize the variety of ECN messages between the current and previous RTT to reflect queue length oscillations, namely, the variety ratio β_i of ith subflow is defined as

$$\beta_i = \frac{\sum_{j=1}^{cw_i}(m_{i,j} \oplus m'_{i,j})}{cw_i} \tag{3}$$

where $m'_{i,j}$ is the ECN information of the ith subflow's jth packet in the previous RTT.

The variety ratio β_i reflects network state changes in adjacent RTTs.

B. ADJUSTING THE TIME GRANULARITY

In tradition MPTCP protocols, the sender increases or reduces its congestion window according to the congestion state in the whole round trip time as shown in Section II. However, using a whole RTT as sampling period is too large for highly concurrent subflows, resulting in inaccurate congestion estimation. Thus, our proposed AMP protocol adjusts the time granularity of congestion detection and control to achieve more accurate congestion control. In addition, in our proposed AMP protocol, when detecting the congestion state, the sender gives the received ACK packets different weights, which are determined by the subflow's variety ratio β and the packet position j in the congestion window.

Due to the TCP ACK-clocking, the packets are spread at the bottleneck rate, resulting in well-spaced traffic during the round-trip time. Therefore, it is reasonable to give the higher weight to the packets arriving in more recent time. Meanwhile, when the subflow's variety ratio β is large, the packets arriving later is also given higher weight to obtain the current congestion state in accurate and timely manner.

Here, we calculate the weight $w_{i,j}$ of the jth packet in ith subflow as

$$w_{i,j} = (\frac{j}{cw_i})^{\beta_i} \tag{4}$$

If the packet number j and the variety ratio β_i are large, the jth packet's weight $w_{i,j}$ is also large. If there is no change in the ECN messages between consecutive RTTs (i.e., β_i=0), packets get the same weight. On the contrary, if β_i is large, the packet with larger packet number has a higher weight to track the more recent congestion state. And, if the j is large, the packets arriving later is given higher weight.

Hence, the weight $w_{i,j}$ of the jth packet in ith subflow is determined by the variety ratio β_i of the subflow and the packet position in the congestion window. That means, different packet position in the congestion window reflects different variety of congestion state, achieving to reflect more accuratly current congestion state.

Finally, for the ith subflow s_i, the congestion ratio α_i is defined as

$$\alpha_i = \frac{\sum_{j=1}^{cw_{i,j}} w_{i,j} \times m_{i,j}}{cw_i} \tag{5}$$

The ith subflow's congestion ratio α_i reflects the weight ratio of ACK packets with ECN-marked in the current whole congestion window. Thus, α_i reflects the network congestion state within the current transmission time. If α_i is large, it means that the network congestion is more obvious.

By setting the different weights to the packets in the congestion window, AMP elaborately utilizes the ECN information in consideration of the variety of congestion state

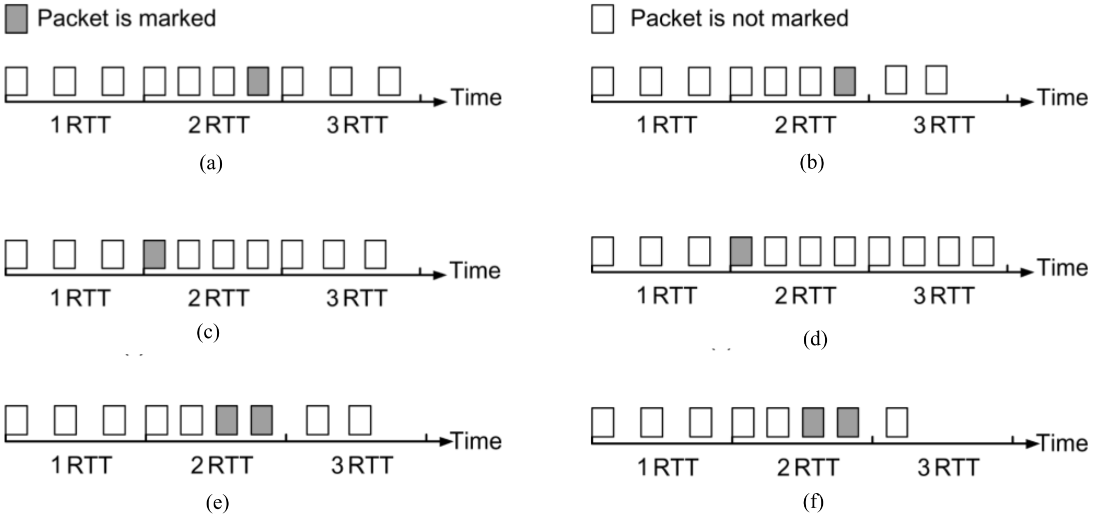


FIGURE 4. Congestion control design of AMP and XMP. (a) Scenario 1 with XMP. (b) Scenario 1 with AMP. (c) Scenario 2 with XMP. (d) Scenario 2 with AMP. (a) Scenario 3 with XMP. (a) Scenario 3 with AMP.

and the packet position in congestion window, achieving the fine-grained congestion detection.

C. CONGESTION CONTROL

There are three principles in designing the congestion control algorithm of AMP: (1) With the help of ECN message, the switch buffer occupation is maintained at a low level. As the result, the transmission delay for small flows becomes small. (2) By splitting its traffic into multiple subflows, AMP avoids from transmitting much traffic on congested path, improving the throughput of large flows. (3) By adjusting the time granularity of congestion detection, AMP copes with the network dynamics in a more accurate way.

The congestion control design of AMP and XMP are compared in Fig. 4. And the AMP's detailed design of congestion control is as described as the three scenarios shown in Fig. 4. Here it should be noted that, for simplicity, XMP reduces the congestion window by the ratio of ACK packets with ECN-marked in the congestion window.

As shown in Fig. 4, in the first RTT, both AMP and XMP send 3 packets in their congestion windows. Since the queue length does not exceed the marking threshold, none packet is marked with ECN bit. Then the sender increases the congestion window to 4 in the 2nd RTT. However, as the number of in-flight packets increased, some packets are marked with ECN bit due to the increasing of queue length in switch buffer. Thus, we give the three different scenarios to show the difference between AMP and XMP.

In scenario 1, the last packet in the congestion window is marked in 2nd RTT. In Fig. 4(a), XMP decreases its congestion window to 3 according to the ratio of marked packets in the 3rd RTT. However, as shown in Fig. 4(b), since the last packet has a large weight in estimating congestion state, AMP sender reduces the congestion window to 2 in the 3rd RTT. The results show that AMP deals with the most recent marked packet in a more active manner compared with XMP.

In scenario 2, the first packet in congestion window is marked in 2nd RTT, which the marked packet appears in a different position from that of scenario 1. In Fig. 4(c), XMP changes the congestion window to 3 even it does not receive the congestion notification recently. In our opinion, it is reasonable not to reduce the congestion window in the next RTT under this situation because it seemed that the congestion has disappeared in a short time. As shown in Fig. 4(d), AMP sender dose not reduce the congestion window due to the small weight of the first packet. The results show that AMP can detect more accurate congestion conditions according the different weight of packet compared with XMP.

In scenario 3, the network appears persistent congestion state during packets transmission process because there are two consistent marked packets in 2nd RTT. In Fig. 4(e), XMP decreases the congestion window by half according to the ratio of marked packets. However, as shown in Fig. 4(f), AMP sender decreases the congestion window to 1 according to ECN notifications. The reason is that AMP decreases more sending rate under higher oscillations of queue length so that it can leave more buffer room to accommodate the dynamic traffic.

Based on the above analysis, AMP's congestion control algorithm is shown as following.

For subflow s_i, if no ACK packets are marked in its current RTT, it is considered that the path is in good condition. Therefore, in this case, ith subflow will enlarge its congestion window as

$$cw_i = cw_i + \frac{1-\alpha_i}{cw_i} \tag{6}$$

According to congestion state of the respective transmission path, AMP adjusts the increasing rate of congestion window of each subflow. If the congestion ratio α_i of ith subflow is large, the sending rate will become slow, avoiding the congested path to become the transmission bottleneck.

When ith subflow detects packet loss, AMP reduces the congestion window according to the corresponding congestion ratio α_i. The congestion window w_i is updated as

$$cw_i = (1 - \alpha_i) \times cw_i \tag{7}$$

Based on the above, the pseudo-code of AMP algorithm is shown in Algorithm 1.

Algorithm 1 The Pseudo-code of AMP

```
At Sender:
// Perform per-subflow-operations
On subflow $i$:
At receiving the ACK packet $j$, record the state of ECN bits with matrix $M$ as:
// Perform per-ACK-operations
if $ECE == 0$ then
    $M[j] = 0$;
else
    if $ECE == 1$ then
        $M[j] = 1$;
    else
        $M[j] = null$;
    end if
end if
Calculate the variety ratio $\beta$ of subflow $i$ as:
for $j = 1$ to $sendcwnd$ do
    $\beta = \beta + M[j] \oplus M[j-1]$;
end for
$\beta = \beta / sendcwnd$;
Calculate the weight $w$ of packet $j$ on subflow $i$ as:
$w[j] = (j/sendcwnd)^{\beta}$;
Calculate the congestion ratio $\alpha$ of subflow $i$ as:
for $j = 1$ to $sendcwnd$ do
    $\alpha = \alpha + w[j] * M[j]$;
end for
$\alpha = \alpha / sendcwnd$;
For each ACK, increase $sendcwnd$ as:
$sendcwnd = sendcwnd + (1 - \alpha)/sendcwnd$;
For each lost, decrease $sendcwnd$ as:
$sendcwnd = (1 - \alpha) * sendcwnd$;
```

V. PERFORMANCE EVALUATION

In this section, we use NS3.21 simulation to evaluate the performance of AMP. The test topology is Fat-Tree [29], [31], [35], which consists of 80 switches with 8 ports and 128 servers. The routing algorithm is Equal Cost MultiPath (ECMP) [32]. The link bandwidth is 1Gbps and the RTT without queuing delay is 100us between any two servers. In the following performance tests, we randomly select some servers as senders and receivers to simulate Many-to-One traffic pattern or One-to-Many traffic pattern.

A. PERFORMANCE COMPARISON OF CONGESTION CONTROL ALGORITHM

To explain how AMP avoids the performance impairments described in Section II, we randomly select 8 servers as the senders and one server as the receiver to establish the simulation environment with Many-to-One traffic pattern, in which 10 subflows are synchronously sent.

We calculated switch buffer occupancy of three different congestion control algorithms, which is AMP, LIA and XMP.

In order to verify that AMP can effectively control the buffer occupation, we change the switch buffer size and ECN marking threshold K. The buffer occupation of bottleneck link is shown in Fig. 5 and Fig. 6.

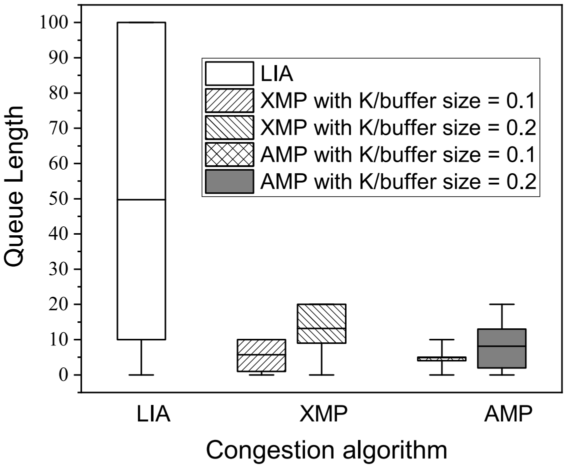


FIGURE 5. Buffer size occupation with buffer size =100.

From Fig. 5, we observe that: (1) The buffer size of LIA fluctuates from 0 to 100, while XMP and AMP can effectively control the buffer size changing from 0 to K, according to use ECN to detect congestion. (2) Since AMP is more finely than XMP, AMP can more sensitively adjust the change of congestion and control buffer size occupation to be lower, further reducing the queue delay time of network traffic.

The similar results of large buffer size and ECN marking threshold K are also shown in Fig. 6. It means that under different switch buffer size, AMP can also effectively control switch buffer occupation, while being less affected by switch buffer size.

B. PERFORMANCE COMPARISON OF TCP AND MPTCP PROTOCOL

We compare the performances of DCTCP, MMPTCP and MPTCP with LIA, XMP and AMP algorithms in the web browse application. A server is randomly selected as the aggregator to send requests to 8 servers simultaneously for retrieving objects on the Web page. When receiving the requests, the 8 servers immediately transmit back response objects. Only when the response objects are completely rendered by all servers, the browser application is completed. The size of objects obeys the uniform distribution from 10KB

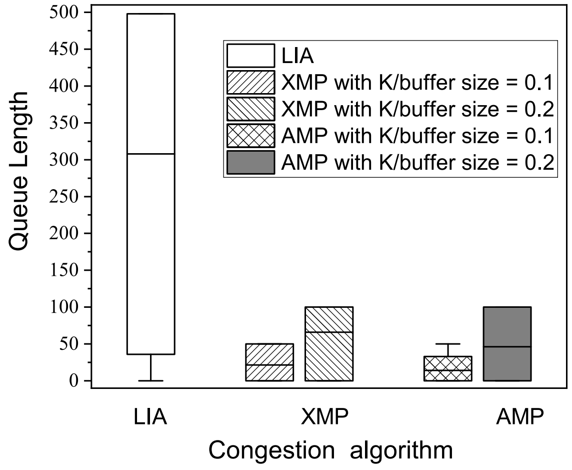


FIGURE 6. Buffer size occupation with buffer size =500.

to 128MB. Here, we evaluate the flow completion time and network throughput.

1) FLOW COMPLETION TIME

In data centers, many applications (i.e., web browser and web search) prefer the shorter flow completion time (FCT), which is the time between the first packet's departure and the last packet's arrival for a given flow. We measure the average flow completion time of different protocols as shown in Fig. 7, and the number of subflows is 4.

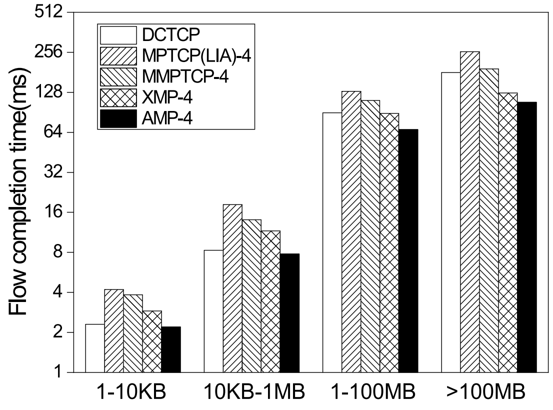


FIGURE 7. Average flow completion time.

Fig. 7 shows that, compared with other four algorithms, AMP has the least flow completion time for both small and large flows. When the flow size is between 10KB and 1MB, AMP achieves the reduction of FCT by 8.9%, 26.2%, 19.5% and 16.3% compared with DCTCP, MMPTCP, LIA-4, and XMP-4, respectively. When the flow size is larger than 100MB, the reduction of FCT become 21.2%, 38.3%, 24.9% and 12.8% in contrast with DCTCP, MMPTCP, LIA-4 and XMP-4, respectively. This result indicates that more AMP flows complete their transmission during the same period due to the fine-grained congestion control.

2) NETWORK THROUGHPUT

Throughput is the most concerned performance for large flow in the data center network, so we count the average throughput of DCTCP, LIA, XMP and AMP for the transmission of 100M files. Specially, in multipath transmission scenario, we change the number of subflows from 2 to 8. The evaluation result is as shown in Fig. 8.

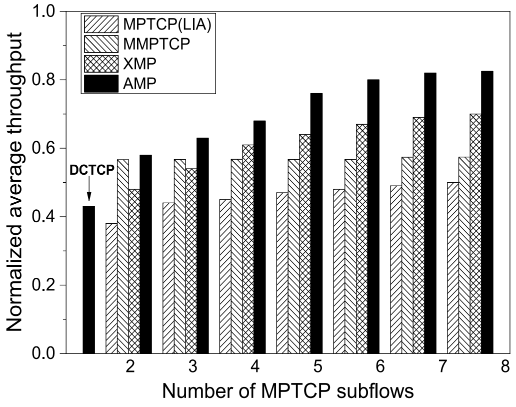


FIGURE 8. Normalized average throughput.

The following conclusions can be drawn from Fig. 8. (1) With increasing of the number of subflows, all MPTCP protocols get the higher network throughput. However, when the number of subflows exceeds a certain value, the network throughput increases slowly. (2) Under the same number of subflows, AMP achieves the highest network throughput. Even for only two subflows, AMP increases the network throughput by 1.4% compared to MMPTCP, though MMPTCP dynamically adjusts dupthresh to reduce the number of congestion window reductions. Compared with DCTCP, LIA and XMP, AMP achieves 10%-35% higher throughput. (3) When achieving the similar throughput, AMP needs less subflows, effectively saving the system resources.

VI. CONCLUSION

High concurrent flows in data centers is common. It means that MPTCP will experience more queue oscillation in switch buffer with the increasing of number of concurrent subflows. In order to accurately detect congestion state, this paper proposes an enhanced MPTCP protocol AMP, which adjusts the time granularity of congestion detection and control. The evaluation results show that AMP achieves lower latency for small flows, and higher throughput for large flows. In addition, MPTCP with AMP needs fewer subflows when achieving the same throughput with comparison of the existing MPTCP protocol. To verify more detail performance and detail overhead of our improved MPTCP, we will deploy AMP in real networks and have more test in different situations in future.

REFERENCES

[1] M. Chen, J. Yang, L. Hu, M. S. Hossain, and G. Muhammad, "Urban healthcare big data system based on crowdsourced and cloud-based air quality indicators," *IEEE Commun. Mag.*, vol. 56, no. 11, pp. 14–20, Nov. 2018. Accessed: Feb. 27, 2019. doi: 10.1109/MCOM.2018.1700571.

[2] M. Chen, Y. Hao, K. Lin, Z. Yuan, and L. Hu, "Label-less learning for traffic control in an edge network," *IEEE Netw.*, vol. 32, no. 6, pp. 8–14, Nov./Dec. 2018. Accessed: Feb. 27, 2019. doi: 10.1109/MNET.2018.1800110.

[3] K. Hwang and M. Chen, *Big–Data Analytics for Cloud, IoT and Cognitive Computing*. Hoboken, NJ, USA: Wiley, 2017.

[4] M. Chen, H. Jin, Y. Wen, and V. C.M. Leung, "Enabling technologies for future data center networking: A primer," *IEEE Netw.*, vol. 27, no. 4, pp. 8–15, Jul./Aug. 2013. Accessed: Feb. 27, 2019. doi: 10.1109/MNET.2013.6574659.

[5] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *Proc. IMC*, 2010, pp. 267–280.

[6] S. Liu, J. Huang, Y. Zhou, J. Wang, and T. He, "Task-aware TCP in data center networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 389–404, Feb. 2019. Accessed: Feb. 27, 2019. doi: 10.1109/TNET.2018.2890010.

[7] M. Chen, Y. Qian, Y. Hao, Y. Li, and J. Song, "Data-driven computing and caching in 5G networks: Architecture and delay analysis," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 70–75, Feb. 2018. Accessed: Feb. 27, 2019. doi: 10.1109./MWC.2018.1700216.

[8] T. Zhang, J. Wang, J. Huang, J. Chen, Y. Pan, and G. Min, "Tuning the aggressive TCP behavior for highly concurrent HTTP connections in intra-datacenter," *IEEE/ACM Trans. Netw.*, vol. 25, no. 6, pp. 3808–3822, Dec. 2017. Accessed: Jan. 22, 2019. doi: 10.1109/TNET.2017.2759300.

[9] J. Ye, J. Lin, and J. Huang, "Priority probability deceleration deadline-aware TCP," *J. Syst. Eng. Electron.*, vol. 26, no. 3, pp. 595–602, Jun. 2015. Accessed: Feb. 27, 2019. doi: 10.1109/JSEE.2015.00067.

[10] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowtron, "Better never than late: Meeting deadlines in datacenter networks," in *Proc. SIGCOMM* Toronto, ON, Canada, 2011, pp. 50–61.

[11] S. Memon, J. Huang, and H. Saajid, "Gentle slow start to alleviate TCP incast in data center networks," *Symmetry*, vol. 11, no. 2, p. 138, 2019. Accessed: Feb. 27, 2019. doi: 10.3390/sym11020138.

[12] J. Huang, Y. Huang, J. Wang, and T. He, "Adjusting packet size to mitigate TCP incast in data center networks with COTS switches," *IEEE Trans. Cloud Comput.*, to be published. Accessed: Mar. 1, 2018. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8305485. doi: 10.1109/TCC.2018.2810870.

[13] Y. Qin, W. Yang, Y. Ye, and Y. Shi, "Analysis for TCP in data center networks: Outcast and incast," *J. Netw. Comput. Appl.*, vol. 68, pp. 140–150, Jun. 2016. Accessed: Nov. 13, 2018. doi: 10.1016/j.jnca.2016.04.014.

[14] W. Chen, F. Ren, J. Xie, C. Lin, K. Yin, and F. Baker, "Comprehensive understanding of TCP incast problem," in *Proc. IEEE INFOCOM*, Hong Kong, Apr./May 2015, pp. 1688–1696.

[15] S. Memon *et al.*, "Novel multi-level dynamic traffic load-balancing protocol for data center," *Symmetry*, vol. 11, no. 2, p. 145, 2019. Accessed: Feb. 27, 2019. doi: 10.3390/sym11020145

[16] Q. Peng, A. Walid, J. Hwang, and S. H. Low, "Multipath TCP: Analysis, design, and implementation," *IEEE/ACM Trans. Netw.*, vol. 24, no. 1, pp. 596–609, Feb. 2016. Accessed: Nov. 13, 2018. doi: 10.1109/TNET.2014.2379698.

[17] K. Yedugundla *et al.*, "Is multi-path transport suitable for latency sensitive traffic?" *Comput. Netw.*, vol. 105, pp. 1–21, Aug. 2016. Accessed: Nov. 13, 2018. doi: 10.1016/j.comnet.2016.05.008.

[18] W. Wang, X. Wang, and D. Wang, "Energy efficient congestion control for multipath TCP in heterogeneous networks," *IEEE Access*, vol. 6, pp. 2889–2898, Dec. 2017. Accessed: Nov. 13, 2018. doi: 10.1109/ACCESS.2017.2785849.

[19] C. Lee, S. Song, H. Cho, G. Lim, and J.-M. Chung, "Optimal multipath TCP offloading over 5G NR and LTE networks," *IEEE Wireless Commun. Lett.*, vol. 8, no. 1, pp. 293–296, Feb. 2019. Accessed: Feb. 27, 2019. doi: 10.1109/LWC.2018.2870595.

[20] G. Carofiglio, M. Gallo, and L. Muscariello, "Optimal multipath congestion control and request forwarding in information-centric networks: Protocol design and experimentation," *Comput. Netw.*, vol. 110, pp. 104–117, Dec. 2016. Accessed: Feb. 27, 2019. doi: 10.1016/j.comnet.2016.09.012.

[21] M. Chen *et al.*, "M-plan: Multipath planning based transmissions for IoT multimedia sensing," *Proc. IEEE IWCMC*, Sep. 2016, pp. 339–344.

[22] R. Khalili, N. Gast, M. Popovic, and J.-Y. Le Boudec, "MPTCP is not Pareto-optimal: Performance issues and a possible solution," *IEEE/ACM Trans. Netw.*, vol. 21, no. 5, pp. 1651–1665, Oct. 2013. Accessed: Nov. 13, 2018. doi: 10.1109/TNET.2013.2274462.

[23] S. Habib, J. Qadir, A. Ali, D. Habib, M. Li, and A. Sathiaseelan, "The past, present, and future of transport-layer multipath," *J. Netw. Comput. Appl.*, vol. 75, pp. 236–258, Nov. 2016. Accessed: Nov. 13, 2018. doi: 10.1016/j.jnca.2016.09.005.

[24] P. Dong *et al.*, "Reducing transport latency for short flows with multipath TCP," *J. Netw. Comput. Appl.*, vol. 108, pp. 20–36, Apr. 2018. Accessed: Jan. 22, 2019. doi: 10.1016/j.jnca.2018.02.005.

[25] Y. Cao *et al.*, "Explicit multipath congestion control for data center networks," in *Proc. CoNEXT*, Santa Barbara, CA, USA, 2013, pp. 73–84.

[26] K. Ramakrishnan, S. Floyd, and D. Black, *The Addition of Explicit Congestion Notification (ECN) to IP*, document RFC 3168, 2001.

[27] J. Luo, J. Jin, and F. Shan, "Standardization of low-latency TCP with explicit congestion notification: A survey," *IEEE Internet Comput.*, vol. 21, no. 1, pp. 48–55, Jan./Feb. 2017. Accessed: Nov. 13, 2018. doi: 10.1109/MIC.2017.11.

[28] H. He, T. Li, L. Feng, and J. Ye, "Frame transmission efficiency-based cross-layer congestion notification scheme in wireless ad hoc networks," *Sensors*, vol. 17, no. 7, p. 1637, 2017. Accessed: Feb. 27, 2019. doi: 10.3390/s17071637.

[29] M. Alizadeh *et al.*, "Data center TCP (DCTCP)," in *Proc. SIGCOMM*, New Delhi, India, 2010, pp. 63–74.

[30] H. Xu and B. Li, "RepFlow: Minimizing flow completion times with replicated flows in data centers," in *Proc. IEEE INFOCOM*, Toronto, ON, Canada, Apr./May 2014, pp. 1581–1589.

[31] M. Kheirkhah, I. Wakeman, and G. Parisis, "MMPTCP: A multipath transport protocol for data centers," in *Proc. IEEE INFOCOM*, San Francisco, CA, USA, Apr. 2016, pp. 1–9.

[32] C. Hopps, *Analysis of An Equal-Cost Multi-Path Algorithm*, IETF, document RFC 2992, 2000.

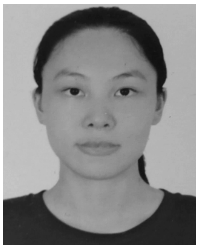
[33] D. Zats, T. Das, P. Mohan, D. Borthakur, and R. Katz, "DeTail: Reducing the flow completion time tail in datacenter networks," in *Proc. SIGCOMM Rev.*, vol. 42, no. 4, Oct. 2012, pp. 139–150. Accessed: Nov. 13, 2018. doi: 10.1145/2377677.2377711.

[34] M. Alizadeh *et al.*, "CONGA: Distributed congestion-aware load balancing for datacenters," in *Proc. SIGCOMM*, Chicago, IL, USA, 2014, pp. 503–514.

[35] M. Coudron and S. Secci, "An implementation of multipath TCP in ns3," *Comput. Netw.*, vol. 116, pp. 1–11, Apr. 2017. Accessed: Nov. 13, 2018. doi: 10.1016/j.comnet.2017.02.002.

JIN YE received the Ph.D. degree with the School of Science and Engineering, Central South University, in 2008. She is currently a Professor with the School of Computer, Electronics and Information, Guangxi University. Her main research interests include network protocol design and data center networks. She is also a member of the China Computer Federation.

LUTING FENG received the B.S. degree from Guangxi University, Nanning, China, in 2016, where she is currently pursuing the M.S. degree with the School of Computer, Electronics and Information. Her main research interests include MPTCP, SDN, and protocol optimization.



ZIQI XIE received the B.S. degree from the Chongqing University of Industrial Engineering, Chongqing, China, in 2016. She is currently pursuing the M.S. degree with the School of Computer, Electronics and Information, Guangxi University, Nanning. Her main research interests include datacenter networks, SDN, and protocol optimization.

JIAWEI HUANG received the Ph.D. degree from the School of Science and Engineering, Central South University, in 2008, where he is currently a Professor with the School of Science and Engineering. He is also the Deputy Director of the Information Security Department. His main research interests include protocol optimal, data center networks, and vehicle networks.

XIAOHUAN LI received the B.Eng. and M.Sc. degrees from the Guilin University of Electronic Technology, Guilin, China, in 2006 and 2009, respectively, and the Ph.D. degree from the South China University of Technology, Guangzhou, China, in 2015. He was a Visiting Scholar with the Université de Nantes, France, in 2014. He is currently an Associate Professor with the School of Information and Communication, Guilin University of Electronic Technology, and also a Research Fellow with Beihang University. His current research interests include wireless sensor networks, vehicular ad hoc networks, and cognitive radios.