information

MDPI

Article

Multiple Congestion Points and Congestion Reaction Mechanisms for Improving DCTCP Performance in Data Center Networks

Prasanthi Sreekumari

Department of Computer Science, Grambling State University, Grambling, LA 71245, USA; sreekumarip@gram.edu

Received: 23 February 2018; Accepted: 6 June 2018; Published: 8 June 2018

Abstract: For addressing problems such as long delays, latency fluctuations, and frequent timeouts in conventional Transmission Control Protocol (TCP) in a data center environment, Data Center TCP (DCTCP) has been proposed as a TCP replacement to satisfy the requirements of data center networks. It is gaining more popularity in academic as well as industry areas due to its performance in terms of high throughput and low latency, and is widely deployed in data centers. However, according to the recent research about the performance of DCTCP, authors have found that most times the sender's congestion window reduces to one segment, which results in timeouts. In addition, the authors observed that the nonlinear marking mechanism of DCTCP causes severe queue oscillation, which results in low throughput. To address the above issues of DCTCP, we propose multiple congestion points using double threshold and congestion reaction using window adjustment (DT-CWA) mechanisms for improving the performance of DCTCP by reducing the number of timeouts. The results of a series of simulations in a typical data center network topology using Qualnet network simulator, the most widely used network simulator, demonstrate that the proposed window-based solution can significantly reduce the timeouts and noticeably improves the throughput compared to DCTCP under various network conditions.

Keywords: TCP; data centers; timeouts; double threshold

1. Introduction

Modern data centers host a variety of services and computation-intensive distributed applications [1]. Nowadays, universities and large IT enterprises such as Microsoft, IBM, Google, and Amazon build their own data centers because of their lower operating costs, improved data protection, and cheap networking equipment [2–4]. Figure 1 shows the conventional Data Center Network (DCN) architecture for data centers adopted from Cisco [2,5]. The typical data center consists of core, aggregation, and access layers. Among these, the core layer provides the high-speed packet switching for all incoming and outgoing flows of the data center. In addition, it provides connectivity to multiple aggregation modules and serves as the gateway.

From the clients in the Internet, data requests to multiple rack of servers (A) are routed through border (BR) and access (AR) routers in Layer 3 to the Layer 2 domain based on the destination Virtual IP address—that is, the IP address to which the request is sent and which is configured onto the two load balancers in Layer 2. The aggregation switches (S) in the top of Layer 2 forward data requests to the top-of-rack switches (TRS) which provide connectivity to servers mounted on every rack. The two load balancers (LBs) in the S contain the list of private and internal addresses (DIP) of physical servers in the racks. This list of DIPs defines the pool of servers that can handle requests to that VIP, and the LB spreads requests across the DIPs in the pool.

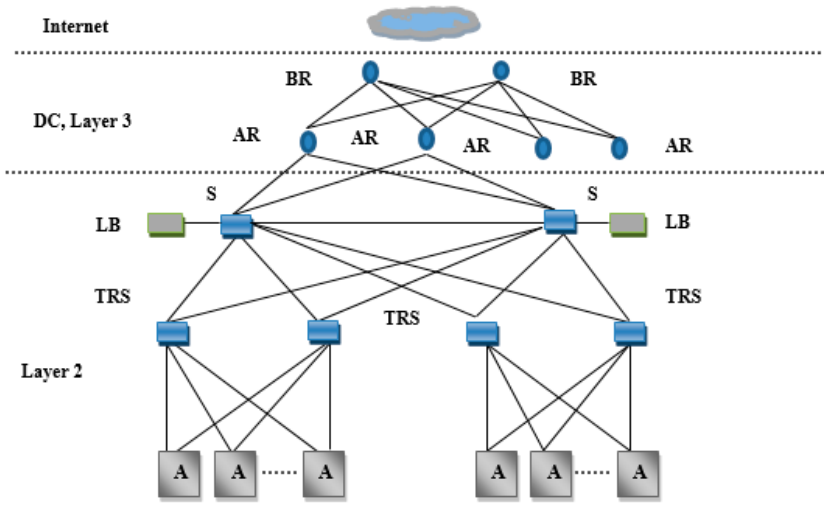

Figure 1. Conventional data center (DC) architecture.

Data center traffic is classified as the traffic between two data center networks and the traffic within a single data center network. The analysis of data center traffic characteristics is important to designing efficient networking mechanisms for data centers. As the authors mentioned in [6,7], unique characteristics such as large number of short flows, short congestion periods, and significant traffic variability make datacenter traffic remarkably different compared with any other network traffic. The flows in data centers are either large, latency critical, or short. These flows typically operate using the conventional TCP due to its reliability [8]. However, recent research has shown that the TCP does not work well in the unique data center environment and is unable to provide high throughput [9]. One of the main reasons for the TCP throughput collapse in a DCN is TCP Incast congestion. Incast congestion is a catastrophic loss in throughput that occurs when the number of senders communicating with a single receiver by sending data increases beyond the ability of an Ethernet switch to buffer packets. It leads to severe packet loss and, consequently, frequent TCP timeouts, thereby reducing the performance of TCP. Hence, for better communication with the nodes, there is a need to propose a good solution to address the issues of conventional TCP in a data center environment.

Recently, a few solutions [10–18] have been proposed to increase TCP performance in a DCN. Among the existing solutions, Data Center TCP (DCTCP) has gained more popularity in academic as well as industry areas due to its better performance in terms of throughput and latency. DCTCP [10] uses a very small buffer space compared with other existing solutions. However, in recent research [19–25] about the performance of DCTCP, the authors found that most times the sender's congestion window reduces to one segment, which results in timeouts. In addition, the authors observed that the nonlinear marking mechanism of DCTCP causes severe queue oscillation which results in low throughput. To address these issues of DCTCP, we propose multiple congestion points using double threshold and congestion reaction using window adjustment (DT-CWA) mechanisms for improving the performance of DCTCP by reducing the number of timeouts. The results of a series of simulations in a typical data center network topology using Qualnet network simulator, the most widely used network simulator, demonstrate that the proposed window-based solution can significantly reduce the timeouts and noticeably improve the throughput compare with DCTCP under various network conditions.

The remainder of the paper is organized as follows: In Section 2, we describe the details of proposed algorithm, DT-CWA. In Section 3, we present our experimental methodology and explain our results. Finally, Section 4 concludes our work.

2. DT-CWA

DCTCP has greatly improved the throughput of conventional TCP in data center networks. However, the congestion window estimation of DCTCP is in the choice of α initialization value [25]. If we set α to 0, the sender may suffer from frequent packet losses and retransmission timeouts. On the other hand, if we set α to 1, the sender can minimize the queuing delay but the amount of packets to be transferred is much smaller. As a result, the throughput of DCTCP will be reduced. In addition, performance studies show that most times the sender's congestion window reduces to one segment, which results in timeouts. Furthermore, studies have shown that the nonlinear marking mechanism of DCTCP causes severe queue oscillation, which results in low throughput [20]. To address these issues of DCTCP, DT-CWA modifies the DCTCP sender and switch sides by adding two schemes: one is a congestion reaction for controlling the size of the congestion window at the sender side, and the other is multiple congestion points using double threshold at the switch for solving the problem of queue oscillation, as shown in Figure 2. First, we present multiple congestion points using a double threshold mechanism at the switch side and then we explain the congestion reaction mechanism at the sender side.

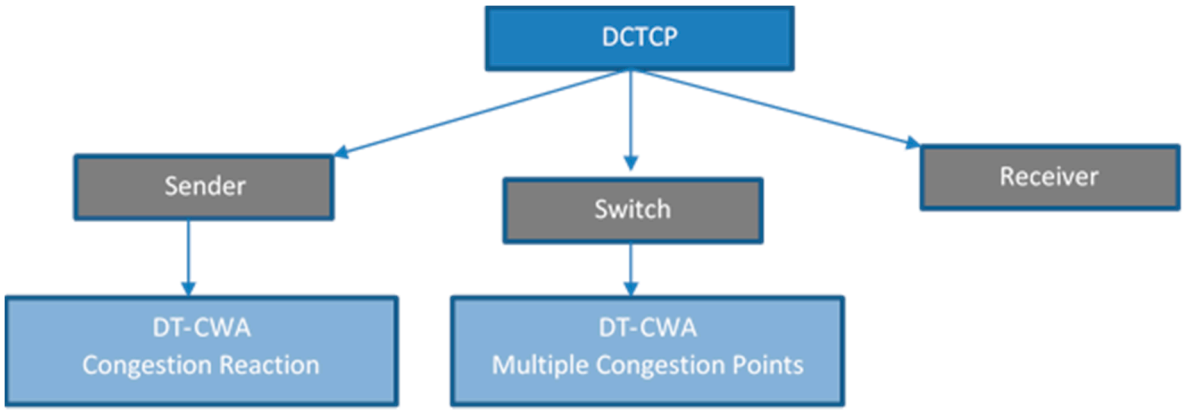

Figure 2. Double threshold and congestion reaction using window adjustment (DT-CWA) modifications over DCTCP.

2.1. Multiple Congestion Points

For avoiding the queue oscillation at the switch, we improved the single threshold mechanism of DCTCP by adding multiple congestion points at the switch side using a double threshold mechanism which we adopted from [20].

Double Threshold Mechanism: DCTCP uses an active queue management policy with an explicit congestion notification (ECN) in which the switch detects the congestion and sets the congestion encountered (CE) codepoint in the IP header. When the number of packets that are queued exceeds the single marking threshold 'K', the switch marks the incoming packets with the CE codepoint and informs the sender about network congestion. In data center networks, the queue length increases and decreases rapidly due to the concurrent arrival of bursts of flows, and the ECN-enabled switch marks packets until the queue length drops back to the threshold. As a result, the notified senders reduce their window size for a while based on the value of α in DCTCP. In addition, some packets may drop due to severe oscillation in queue length. As a result, the sender needs to wait until the timeout happens. As the author mentioned in [20], one of the main reasons for queue length oscillation is the nonlinear marking process of single threshold.

As we explained in Section 1, in data center networks, TCP Incast occurs when the flow of data increases from many senders to a single receiver. Using a single threshold 'K', DCTCP cannot mitigate the problem of TCP Incast when severe packet loss occurs and the frequency of timeouts thereby

increases. For mitigating the TCP Incast problem due to frequent packet losses and timeouts, we adapted the double threshold mechanism from [20] and incorporated it into DT-CWA.

Instead of using single threshold 'K', we split 'K' into 'K1' and 'K2'. The lower threshold 'K1' is for starting ECN packet marking, i.e., the starting point of congestion, and the upper threshold 'K2' is for ending the marking of packets, i.e., the ending point of congestion, as shown in Figure 3. The switch sends the message of network congestion to senders by marking packets to decrease their congestion window size from when the queue length increases beyond 'K1' until when the queue length decreases under 'K2'. Using double threshold values, we can control the queue length and improve the performance of DCTCP by marking the packets earlier than DCTCP would, particularly from when the queue length increases to the first threshold 'K1' and by continuously marking the packets until the queue length falls back to 'K2'.

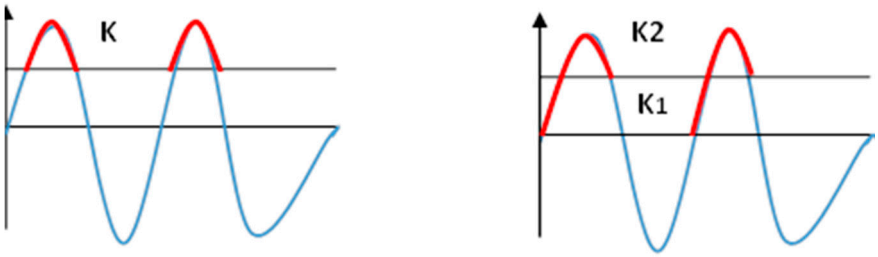

Figure 3. (**a**) Single threshold and (**b**) double threshold.

2.2. Congestion Reaction

At the sender side: When the sender receives the acknowledgment (ACK) with a congestion notification, the sender checks the current network condition based on the outstanding packets and adjust the size of congestion window according to Equation (1):

$$CWstart = 1/\alpha(CWmax - CWmin) - CWcurrent \quad (1)$$

where α is the number of outstanding packets in the network, i.e., the difference between the total number of packets sent and the number of ACK received so far in the network; *CWcurrent* is the current size of the congestion window at the time of receiving the ACK packets with congestion notifications; and *CWmax* and *CWmin* are the maximum and minimum size, respectively, of the congestion window adjusted before receiving the congestion notification. When the sender receives the ACK of all outstanding packets, the sender adjusts the sending rate according to Equation (2):

$$CWend = CWcurrent + \beta(CWmax - CWmin) \quad (2)$$

where *CWstart* and *CWend* are the starting and ending points of congestion, respectively. The above modifications in DCTCP using the two components (congestion reaction at the sender side and multiple congestion points at the switch side) help to overcome the problem of oscillations in queue length as well as frequent retransmission timeouts, and thereby increase the performance in data center networks.

For setting the value of β, we did an experiment by calculating the throughput using 10 senders with different values for β. All senders send data to a single aggregator through different switches. As shown in Figure 4, from the result, we observed that when the value of β increases from 0.2 the throughput decreases due to frequent timeouts. As a result, we set β to be 0.2 in Equation (2).

With the above modifications to DCTCP, DT-CWA can improve performance by reducing the packet losses by controlling the size of congestion window as well as the packet marking mechanism using double threshold.

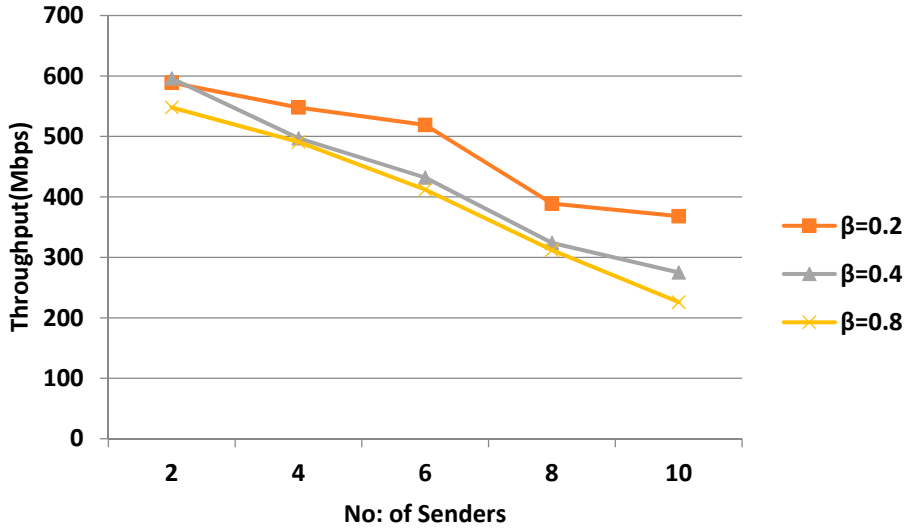

Figure 4. Comparison of β values.

3. Performance Evaluation

In this section, we present the performance of our improved DCTCP algorithm through comprehensive simulations using Qualnet network simulator [26]. We compare the performance of our algorithm with that of DCTCP as it is the most popular data center transport protocol. We implemented DCTCP in Qualnet]. Our main goal of this work is to increase the performance of DCTCP by controlling the size of the congestion window as well as the queue length in data center networks.

To achieve our goal, we evaluate the performance of our algorithm in a typical network topology [20] which consists of 10 end hosts and four switches as shown in Figure 5. Among them, Switch 1 is connected to the aggregator; Switch 2 is connected to Workers 1, 2, and 3; Switch 3 is connected to Workers 4, 5, and 6; and Switch 4 is connected to Workers 7, 8, and 9. In addition, Switches 2, 3, and 4 are connected to Switch 1.

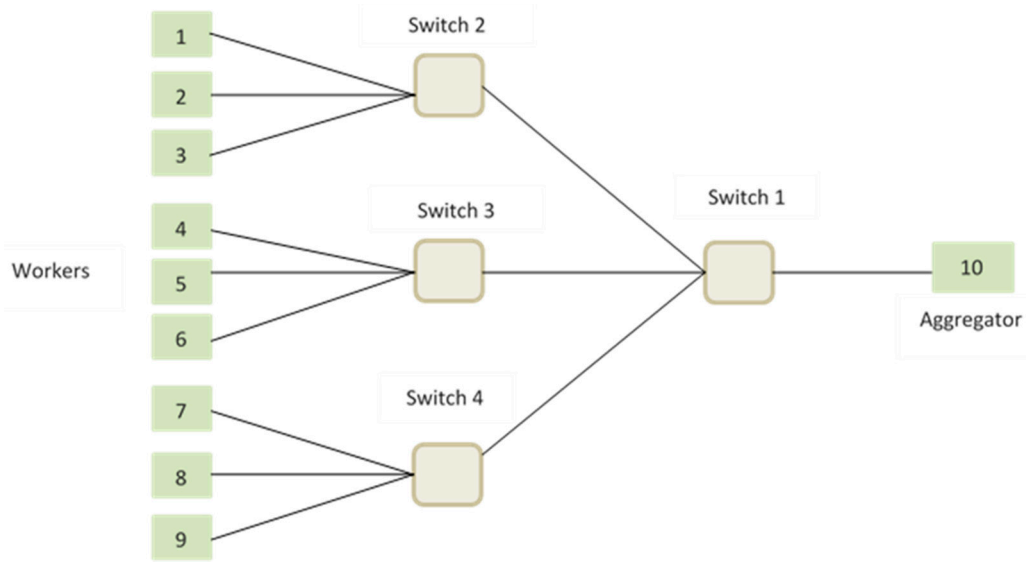

Figure 5. Network topology.

The link capacity was set to 1 Gbps and link delay was set to 25 µs, RTT 100 µs, and RTO min which is equal to 10 ms. The buffer size was set to 128 KB for Switch 1 and 512 KB for Switches 2, 3, and 4. The marking threshold value 'K' for DCTCP was set according to [10,21] for 1 Gbps link

capacity. The value of the weighted averaging factor 'g' for DCTCP was set to 0.0625. We set the lower threshold value 'K1' to be 35 and upper value 'K2' to be 30. An FTP-generic application was run on each source to send the packets as quickly as possible. Figures 6–8 present the results of our evaluation of the proposed algorithm, DT-CWA, in comparison with DCTCP in terms of throughput, average query completion time, and average short message completion time as they are the most important performance metrics.

Figure 6 shows the performance of DT-CWA compared with that of DCTCP in terms of throughput with different numbers of flows. From the result, we observe that the throughput of DT-CWA is higher than the throughput of DCTCP. After 20 flows, the throughput of DT-CWA increased significantly compared to that of DCTCP, but the throughput of DT-CWA reduced to less than 200 Mbps after 35 flows.

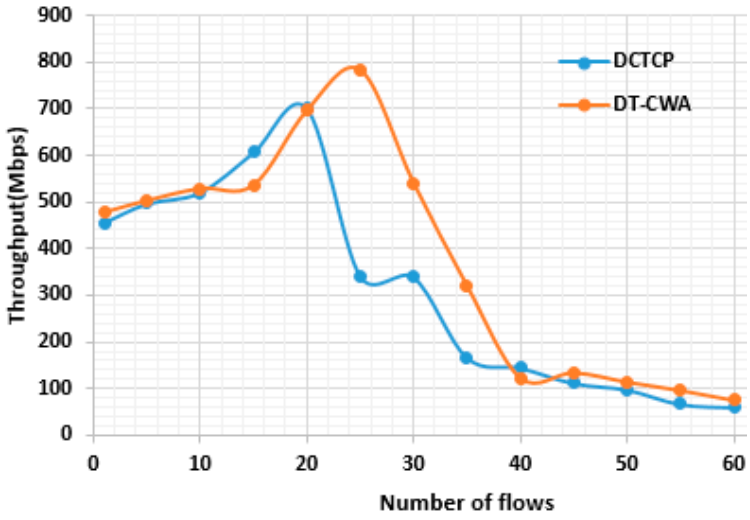

Figure 6. Comparison of throughput.

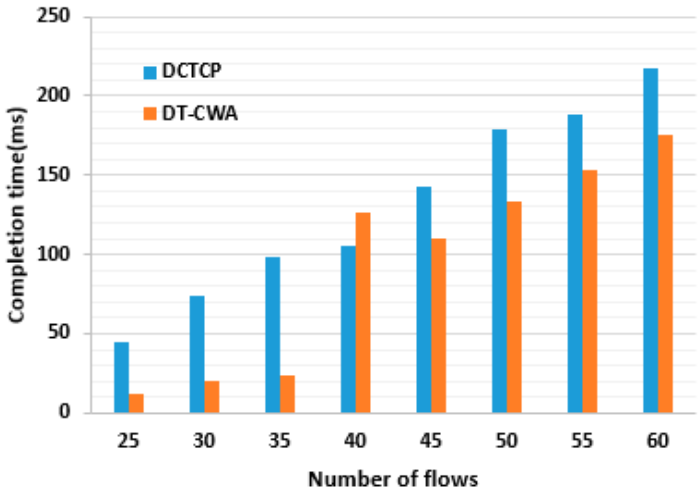

Figure 7. Comparison of average query completion time.

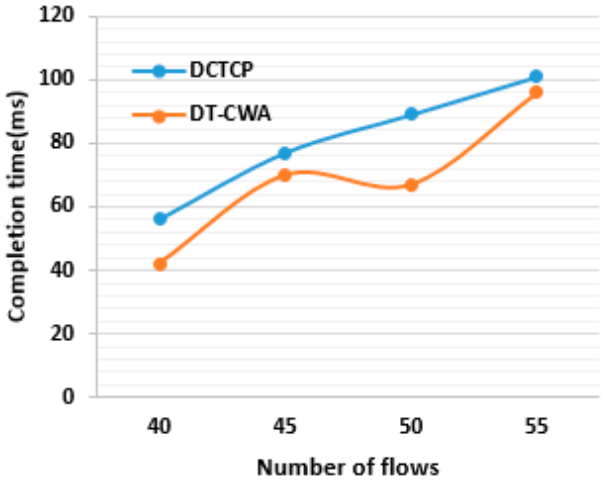

Figure 8. Comparison of average short message completion time.

However, the performance of DT-CWA is higher compared to that of DCTCP. For 30 flows, DT-CWA achieves 59% throughput increment and for 55 flows, DT-CWA achieves 43% throughput increment compared to DCTCP. One of the main reasons for this achievement is our multiple congestion points and rate adjustment mechanisms. Figure 7 presents the comparison of average query completion time with flows from 25 to 60. The result shows that the average query completion time of DT-CWA is less than that of DCTCP. Compared to DCTCP, the query completion time of DT-CWA is less than 50 ms until 35 flows. When the number of flows increases, the query completion time also increases. However, DT-CWA took less time than DCTCP, thereby increasing the performance. Figure 8 depicts the results of the average short message completion time of DT-CWA compared to that of DCTCP. Compared to the average query completion time, for short messages, DCTCP and DT-CWA took only less time. However, the performance of DT-CWA is better than that of DCTCP. DT-CWA suffers from timeouts due to port black-out [27] but not frequently like DCTCP during the communication with the end host.

The efficient utilization of the buffer leads to a reduction in the overflow of the queue length and thereby reduces the loss of packets from the network; thus, DT-CWA achieves better performance than DCTCP.

4. Conclusions

In this paper, we have developed a modified DCTCP protocol named DT-CWA for improving the performance of DCTCP in data center networks. In DT-CWA, we propose a new congestion reaction mechanism at the sender side and multiple congestion points at the switch for avoiding queue length oscillation and frequent retransmission timeouts, thereby utilizing the buffer space efficiently. We conducted extensive simulations using 10 end hosts and 4 switches in Qualnet to evaluate the performance and effectiveness of our algorithm, DT-CWA, compared with those of DCTCP in terms of throughput, average query completion time, and short message completion time. Our experimental results show that the proposed algorithm achieves better performance compared with DCTCP. In future work, we will design a protocol for mitigating TCP Incast as well as TCP Outcast issues for the improvement of data center networks.

Acknowledgments: The author would like to thank the anonymous reviewers for their valuable comments and suggestions that have contributed to improve this paper.

Conflicts of Interest: The author declare no conflicts of interest.

References

1. Zhang, Y.; Ansari, N. On Architecture Design, Congestion Notification, TCP Incast and Power Consumption in Data Centers. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 39–64. [CrossRef]
2. Cisco Data Center Infrastructure 2.5 Design Guide. Available online: http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Data_Center/DC_Infra2_5/DCI_SRND_2_5a_book.pdf (accessed on 7 June 2018).
3. Guo, J.; Liu, F.; Lui, J.; Jin, H. Fair Network Bandwidth Allocation in IaaS Datacenters via a Cooperative Game Approach. *IEEE/ACM Trans. Netw.* **2016**, *24*, 873–886. [CrossRef]
4. Guo, J.; Liu, F.; Wang, T.; Lui, J.C.S. Pricing Intra-Datacenter Networks with Over-Committed Bandwidth Guarantee. In Proceedings of the 2017 USENIX Conference on Usenix Annual Technical Conference (USENIX ATC '17), Santa Clara, CA, USA, 12–14 July 2017.
5. Data Center Architecture Overview. Available online: http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Data_Center/DC_Infra2_5/DCInfra_1.html (accessed on 7 June 2018).
6. Liu, F.; Guo, J.; Huang, X.; Lui, J.C.S. eBA: Efficient Bandwidth Guarantee under Traffic Variability in Datacenters. *IEEE/ACM Trans. Netw.* **2017**, *25*, 506–519. [CrossRef]
7. Tao, W.; Liu, F.; Xu, H. An Efficient Online Algorithm for Dynamic SDN Controller Assignment in Data Center Networks. *IEEE/ACM Trans. Netw.* **2017**, *25*, 2788–2801.
8. Chen, Y.; Griffith, R.; Liu, J.; Katz, R.H.; Joseph, A.D. Understanding TCP incast throughput collapse in datacenter networks. In Proceedings of the 1st ACM workshop on Research on Enterprise Networking (WREN '09), Barcelona, Spain, 16–21 August 2009; ACM: New York, NY, USA, 2009; pp. 73–82.
9. Kant, K. Data center evolution: A tutorial on state of the art, issues, and challenges. *Comput. Netw.* **2009**, *53*, 2939–2965. [CrossRef]
10. Alizadeh, M.; Greenberg, A.; Maltz, D.; Padhye, J.; Patel, P.; Prabhakar, B.; Sengupta, S.; Sridharan, M. Data center TCP (DCTCP). *ACM SIGCOMM Comput. Commun. Rev.* **2010**, *40*, 63–74. [CrossRef]
11. Hwang, J.; Yoo, J. FaST: Fine-grained and Scalable TCP for Cloud Data Center Networks. *KSII Trans. Internet Inf. Syst.* **2014**, *8*, 762–777. [CrossRef]
12. Wu, H.; Feng, Z.; Guo, C.; Zhang, Y. ICTCP: Incast Congestion Control for TCP in Data Center Networks. *IEEE/ACM Trans. Netw.* **2013**, *21*, 345–358.
13. Hwang, J.; Yoo, J.; Choi, N. IA-TCP: A Rate Based Incast-Avoidance Algorithm for TCP in Data Center Networks. In Proceedings of the 2012 IEEE International Conference on Communications (ICC), Ottawa, ON, Canada, 10–15 June 2012.
14. Zhang, J.; Wen, J.; Wang, J.; Zhao, W. TCP-FITDC: An adaptive approach to TCP incast avoidance for data center applications. In Proceedings of the 2013 International Conference on Computing, Networking and Communications (ICNC), San Diego, CA, USA, 28–31 January 2013; pp. 1048–1052.
15. Zhang, J.; Ren, F.; Yue, X.; Shu, R.; Lin, C. Sharing Bandwidth by Allocating Switch Buffer in Data Center Networks. *IEEE J. Sel. Areas Commun.* **2014**, *32*, 39–51. [CrossRef]
16. Zhang, J.; Ren, F.; Tang, L.; Lin, C. Modeling and solving TCP Incast problem in data center networks. *IEEE Trans. Parallel Distrib. Syst.* **2015**, *26*, 478–491. [CrossRef]
17. Wang, G.; Ren, Y.; Dou, K.; Li, J. IDTCP: An effective approach to mitigating the TCP Incast problem in data center networks. *Inf. Syst. Front.* **2014**, *16*, 35–44. [CrossRef]
18. Shukla, S.; Chan, S.; Tam, A.S.-W.; Gupta, A.; Xu, Y.; Chao, H.J. TCP PLATO: Packet labelling to alleviate time-out. *IEEE J. Sel. Areas Commun.* **2014**, *32*, 65–76. [CrossRef]
19. Sreekumari, P.; Jung, J.; Lee, M. An early congestion feedback and rate adjustment schemes for many-to-one communication in cloud-based data center networks. *Photon. Netw. Commun.* **2016**, *31*, 23–35. [CrossRef]
20. Chen, W.; Cheng, P.; Ren, F.; Shu, R.; Lin, C. Ease the Queue Oscillation: Analysis and Enhancement of DCTCP. Procedings of the 2013 IEEE 33rd International Conference on Distributed Computing Systems, Philadelphia, PA, USA, 8–11 July 2013.
21. Jiang, C.; Li, D.; Xu, M. LTTP: An LT-Code Based Transport Protocol for Many-to-One Communication in Data Centers. *IEEE J. Sel. Areas Commun.* **2014**, *32*, 52–64. [CrossRef]
22. Fang, S.; Foh, C.H.; Aung, K.M.M. Prompt congestion reaction scheme for data center network using multiple congestion points. In Proceedings of the 2012 IEEE International Conference on Communications (ICC), Ottawa, ON, Canada, 10–15 June 2012; pp. 2679–2683.

23. Zhang, J.; Ren, F.; Tang, L.; Lin, C. Taming TCP incast throughput collapse in data center networks. In Proceedings of the 2013 21st IEEE International Conference on Network Protocols (ICNP), Goettingen, Germany, 7–10 October 2013; pp. 1–10.
24. Wu, W.; Crawford, M. Potential performance bottleneck in Linux TC. *Int. J. Commun. Syst.* **2007**, *20*, 1263–1283. [CrossRef]
25. Improving Transmission Performance with One-Sided Datacenter TCP. Available online: https://eggert.org/students/kato-thesis.pdf (accessed on 7 June 2018).
26. QualNet Network Simulator Software. Available online: http://web.scalable-networks.com/content/qualnet (accessed on 7 June 2018).
27. Prakash, P.; Dixit, A.; Hu, Y.C.; Kompella, R. The TCP outcast problem: Exposing unfairness in data center networks. In Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation (NSDI'12), Lombard, IL, USA, 3–5 April 2013; USENIX Association: Berkeley, CA, USA, 2012; p. 30.