43

Incast in Data Center Networks

Mayur Vaidya

MSc. CS, School of Computer Science, MIT World Peace University Pune.

vaidyamr2001@gmail.com

Rohan Bilgundi

MSc. CS, School of Computer Science, MIT WPU World Peace University Pune.

rohan.bilgundi333@gmail.com

Paresh Pandit

MSc. CS, School of Computer Science, MIT World Peace University Pune.

pareshpandit5803@gmail.com

Dr. Mahendra Suryavanshi

School Of Computer Science,

MIT World Peace University Pune

mahendra.suryavanshi@mitwpu.edu.in

Corresponding Author: Mayur Vaidya,

MSc. CS, School of Computer Science, MIT-WPU,

vaidyamr2001@gmail.com

Abstract-

Several data center applications follow many-to-one communication pattern at the access layer of data center network (DCN), in which numerous workers (servers) transfers data towards same aggregator (client) through a common ToR switch at the same time. Due to this, goodput collapses and it is termed as incast problem in DCN. There are two types of incast problems. Transmission Control Protocol (TCP) incast problem and Multipath TCP (MPTCP) incast problem. In this paper, existing solutions to TCP incast and MPTCP incast are reviewed comprehensively.

Keywords: TCP, MPTCP, data center network, synchronous read, goodput, Incast.

I. INTRODUCTION

Data center is a pool of computing and storage resources clustered together using communication networks to host Internet-based applications (e.g., search engines, video data hosting, social networking, large-scale computing) and data storage [9][6]. Applications hosted by data center are either data intensive or communication intensive. The thousands of servers may be harnessed to fulfill a simple web search request or database query [6][5]. Building DCN using commodity TCP/IP and Ethernet networks is attractive because of the low cost, ease-of-use and desire to share the bandwidth over multiple compute resources [2]. DCNs are constructed by following switch-based, server-based or hybrid architecture. The FatTree switch-based architecture is most largely used to build large scale and high performing DCNs [10].

In TCP/IP protocol suite TCP is the most popular transport protocol and considered as the backbone of the internet. It provides reliable, byte-stream, connection-oriented services and operates over heterogeneous network topologies. TCP offers flow control and congestion control. Modern implementation of TCP uses slow start, fast-retransmit, fast-recovery and congestion avoidance algorithms. TCP used in DCNs to provide reliable communication between various clients and servers located at DCN on time [3]. Multipath TCP (MPTCP) [11] protocol can be used efficiently to achieve improved throughput, better fairness and robustness over multi-homed network architectures [12]. MPTCP is proposed to replace regular TCP in DCNs. More specifically, it is an extension to TCP which allows multi-homed servers in data center to use multiple paths simultaneously.

DCNs use ToR switches with small-sized buffers, low propagation delay (with hundreds of microseconds of RTT) and high-bandwidth (1 Gb/s and onwards) links [13]. Several data center applications follow many-to-one communication pattern at the access layer of DCN, in which numerous workers transfers data towards same aggregator through a common ToR switch at the same time. This simultaneous busty transmission may overload buffer of a ToR switch connected to aggregator and causes timeout events after frequent packet drops. Due to this, goodput collapses and it is termed as incast problem in DCN [1]. There are two types of incast problems i.e., Transmission Control Protocol (TCP) incast problem and Multipath TCP

(MPTCP) incast problem. In this paper, authors have comprehensively studied TCP incast and MPTCP incast problem in DCN. Section 2 analyses TCP incast and MPTCP incast in DCN. Existing solutions to TCP incast and MPTCP incast are reviewed in section 3. Section 4 provides analysis on existing solutions to TCP incast and MPTCP incast. Paper is concluded in section 5.

II. TCP INCAST AND MPTCP INCAST ISSUES IN DCN

Synchronized read/write operations are commonly performed in DCNs for multiple servers to one client. HDFS, Lustre, Panasas, pNFS, Cassandra, MapReduce are the network file systems used in DCNs to facilitate synchronized read/write operations [8]. Synchronized request workload requires that many-to-one communication pattern to take place between multiple servers and single client in reliable and on time manner.

There are applications need many-to-one communication patterns in DCNs some of the examples are:

a) Social Networking sites: User logs in to the social networking site. If user’s complete profile is stripped across multiple storage servers, request for fetching complete profile can be sent to number of storage servers within the data center. These servers send their part of requested data to client simultaneously [4, 7].

b) Web search applications (search engine): Client submits search query to web search application. There could be hundreds of thousands of storage servers that contain requested data for search query. All storage servers respond with their part of result to the client simultaneously [4, 7].

c) Data warehousing applications and applications used to maintain big organizations data, banks data, government data, hospitals data etc. are all follows many-to-one communication pattern where multiple servers send data to the single client [4, 7].

Under many-to-one communication pattern, multiple single-homed concurrent workers use single path TCP protocol to simultaneously transmit rack-local short flows data towards single-homed aggregator connected with common ToR switch. This creates huge congestion at bottleneck ToR switch connected to single-homed aggregator. This results into **TCP incast** at access layer of single-homed DCN [14].

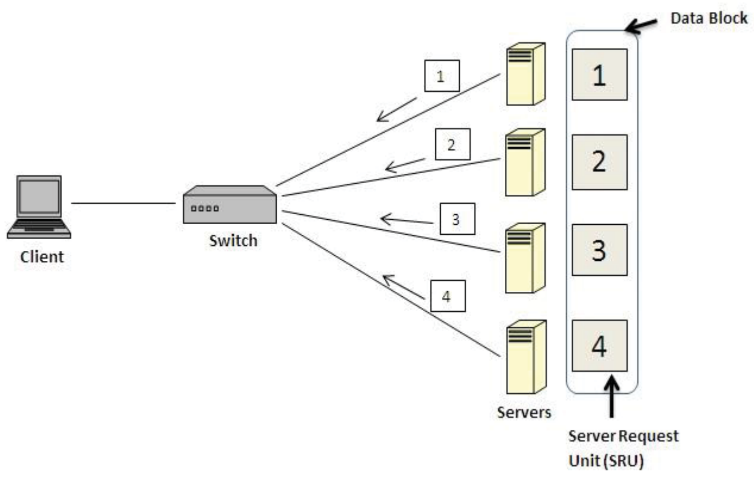

Figure-1: Many-to-one communication pattern in single-homed DCN causing TCP incast

Under many-to-one communication pattern, several multi-homed concurrent workers use MPTCP protocol to simultaneously transmit rack-local short flows data towards same multi-homed aggregator through their multiple subflows. This creates huge congestion at bottleneck ToR switches connected to multi-homed aggregator, causes severe packet loss and results into **MPTCP incast** at access layer of multi-homed DCN [15].

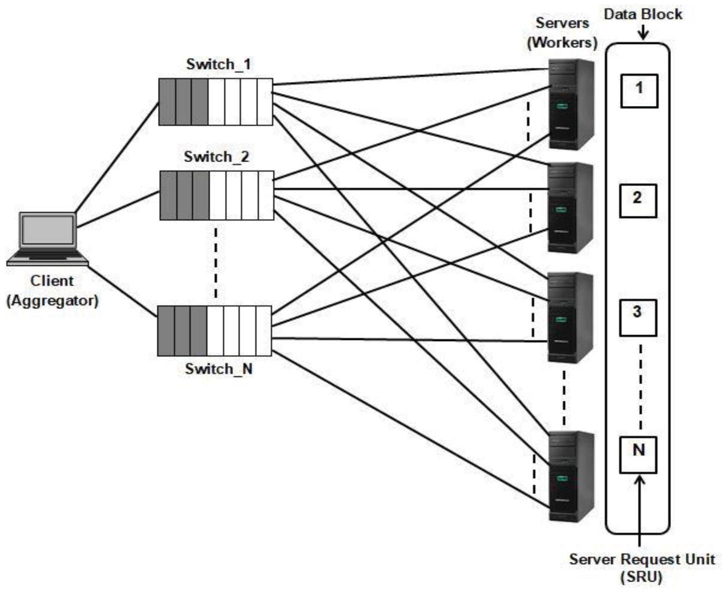

Figure-2: Many-to-one communication pattern in multi-homed DCN causing MPTCP incast

TCP incast and MPTCP incast problems have various consequences such as drop in goodput, increase in flow completion time and minimum fan-out in DCN. This eventually deteriorates data center user's experience. Several methods and protocols are available to mitigate TCP incast and MPTCP incast problems.

III. REVIEW OF EXISTING SOLUTIONS TO TCP INCAST AND MPTCP INCAST

3.1 Existing solutions to TCP incast:

Solutions to TCP incast problem in single-homed DCN are available at application layer, transport layer and data-link layer.

3.1.1) Application layer solutions to TCP incast

Optimal Staggering Data Transfers (OSDT) technique utilizes maximum link capacity without causing any packet loss. OSDT achieves this by limiting worker's transmission rate and number of concurrent workers. OSDT uses application parameters like size of SRU, size of data block, number of concurrent workers and so on. It also considers network parameters like link bandwidth, switch buffer size, RTT, advertised window size. This application and network information is used to build an optimization model [16].

Delayed Server Response at Application Layer (DSRAL) is sequential and delay-based SRU transmission technique. Aggregator uses packet scheduling time, average RTT and SRU size values to calculate time taken by each worker for completing SRU packets transmission. Aggregator enforces workers to perform sequential SRUs transmission by communicating distinct SRU transmission time through request packets [17].

Adaptive Request Schedule (ARS)is a cross-layer design. ARS acquires congestion state information from transport layer by tracking out-of-order data packets received at aggregator TCP. Based on this congestion state information, ARS adjusts number of concurrent flows dynamically by scheduling SRU requests in batches [18].

A serialized SRU packet transmission technique (SERIAL) states that, SRU packets will be transmitted by only one worker at a time. In this application layer scheme, aggregator send request to workers one after another. Until aggregator receives all SRU packets from currently transmitting worker, aggregator does not issue request to next worker for SRU transmission. In this manner, serialized requests transmitted by aggregator makes all workers to transmit their SRUs in sequential order [19].

Enhanced DSRAL (EDSRAL) technique ensures that before currently transmitting worker finishes its SRU transmission, next worker in the list begins its SRU transmission. Hence instead of allowing workers to transmit their SRUs sequentially one after another, EDSRAL allows two consecutive workers in the list to overlap their SRU transmissions. In EDSRAL technique, the Flow Overlapping Factor (FOF) is determined to overlap two consecutive SRU transmissions made by two successive workers in the list. In such a way, EDSRAL helps to utilize data center links at maximum level and produce higher goodput [20].

3.1.2) Transport Layer Solutions to TCP incast

Reducing TCP minimum RTO value states that TCP timeouts are unavoidable, but time spent on waiting for a timeout can be reduced. TCP implementations use an RTO_{min} value of 200ms. This value is generally greater than round-trip times. By reducing the TCP's RTO value, TCP Incast problem can be avoided [8].

Proactive ACK Control (PAC) allows aggregator to receive data packet, create ACK packet, store it at the end of ACKqueue and update incoming traffic volume value. Aggregator determines whether releasing ACK packet will make threshold value less than incoming traffic volume. Until switch buffer has enough space, aggregator does not release ACK. Incoming traffic value is decremented and incremented based on receiving data packet and transmitting ACK packet respectively. ACK packets are scheduled using Multi-level Feedback Queue mechanism [21].

In Adaptive Pacing (AP), worker transmits packets by adding time interval between them. Furthermore, worker dynamically adjusts the time intervals while sending outgoing packets. This is done based on count of simultaneous flows. In AP, to determine starting time for each of n simultaneously transmitting flows, available time interval (i.e. between 0 to t_0) is split into n number of identical time slots. Each worker transmits packets at specific time slot assigned to it. AP requires modification to worker's TCP [22].

Timely Retransmitted ACKs (T-RACKs) update flow-table entries to keep current state of all flows by intercepting TCP header of each arriving and outgoing TCP packet. This helps aggregator to perform fast retransmit and recovery by transmitting adequate duplicate ACKs (dupAcks) after detecting packet loss. Before RTO_{min} timer expires, workers retransmit lost segment. T-RACKs transmit spoofed dupAcks towards worker, if aggregator not receives adequate packets (due to short flows) for generating dupAcks [23].

In **Many to one TCP (M21TCP),** switch continually calculates number of concurrently transmitting workers by tracking presently active discrete flows traversing through its bottleneck interface. Validity of switch determined worker count is for one RTT period only. Switch put calculated worker count in additional 32-bit TCP field of a data packet and forwards data packet towards aggregator. Aggregator receives data packet, retrieves concurrently transmitting workers count and through ACK packet forwards it towards particular worker. Worker receives ACK packet from aggregator, retrieves workers count and then determine its congestion window size. Congestion window size is determined by worker with the help of workers count, maximum header size and switch buffer size [24].

Stochastic Adjustment TCP (SA-TCP) performs congestion control using additive increase and multiplicative decrease. It generates random values for variables α and β, instead of keeping constant values. Arrival of highly busty traffic at bottleneck switch due to concurrent transmissions of multiple workers can be avoided by having random values of α and β. SA-TCP permits fair sharing of aggregate bandwidth among multiple workers [25].

3.1.3) Data Link Layer Solutions to TCP incast

Provisioning Larger Switch Buffers: Timeouts are the primary cause of incast, and the root cause of timeouts is packet losses. Use of larger switch buffers can reduce the significant packet losses at client side. Increasing the switch buffer size at client side doubles the number of servers that can transmit before the system experiences incast. Unfortunately switches with larger buffers costs more. Hence this cannot be the optimal solution for TCP incast problem [1].

Flow Control Method: Switch supporting Ethernet Flow Control (EFC) transmit a pause frame towards packet transmission interface, once it is overloaded by packets. Device attached to packet transmission interface pause the transmission of packets for particular time period upon receiving pause frame. The EFC mechanism is not effective in a network containing workers and aggregator connected through more than one switch. EFC face an issue of head-of-line blocking [1].

Congestion Control Method: Quantized Congestion Notification (QCN) is IEEE 802.1 QAU based standard and commonly used link layer congestion control method. Congestion Point (CP) i.e., switches, Reaction Point (RP) i.e. worker are 2 different locations where QCN is implemented. CP exists between worker and aggregator which tracks queue length growth rate by counting number of received packets. CP transmits negative feedback packet to worker, if it detects congestion. RP reduces transmission rate if it receives negative feedback packet from CP [26].

3.2 Existing solutions to MPTCP incast

MPTCP-L aims to solve MPTCP incast by avoiding packet loss and reducing latency of rack local short flows. This is done by replicating each packet of short flow on two subflows and ensuring that aggregator will receive transmitted packet on at least one subflow [27].

Adaptive Multipath Transmission Control Protocol (AMTCP) aimed to mitigate MPTCP incast by minimizing scheduling and resource overhead for short flows. Furthermore, AMTCP has goal of improving throughput of large flows. According to application workloads, number of subflows of a multipath worker is dynamically adjusted by AMTCP [28].

Fast multi-path loss recovery (FUSO) is a recovery-based multipath transport protocol that immediately retransmit lost packet by exploiting multipath diversity in DCNs. In FUSO, subflow having spare congestion window slots and less congestion is used by multipath transport worker to immediately transmit recovery packet whenever worker detects packet loss on another congested subflow [29].

Balanced Multipath TCP (BMPTCP) efficiently mitigate MPTCP incast in multi-homed DCNs. BMPTCP computes and controls subflow congestion window sizes by considering ToR switch buffer size, total header size of data packets and count of total flows traversing through bottleneck interface of ToR switch. It maintains identical congestion window size for all concurrent subflows to avoid timeout events due to full window loss at ToR switch [30].

eXplicit Multipath (XMP) protocol balances latency with throughput. XMP use Buffer Occupancy Suppression (BOS) algorithm to control bottleneck ToR switch buffer occupancy by employing ECN mechanism. Furthermore, Traffic Shifting (TraSh) algorithm is used to shift traffic between multiple paths to equally balance traffic and relieve congestion. XMP does not utilize whole capacity of link buffer [31].

Efficient Scheduling scheme with Explicit congestion notification (ESE) scheme schedules DCTCP and XMP mixed flows. Compared to DCTCP short flows, buffer of ToR switch is occupied more aggressively by XMP large flows, hence for XMP and DCTCP flows, dual service queues with ECN is maintained. Urgent flows are delivered with highest priority by providing extra queue. According to the congestion situation, ESE scheme dynamically adjusts number of subflows of workers [32].

Adaptive Multipath (AMP) protocol is designed to handle last hop unfairness and MPTCP incast problems. AMP converts multipath flow into single-path flow immediately after detecting last hop unfairness or MPTCP incast problem. AMP again reverses single-path connection to multipath connection once last hop unfairness and MPTCP incast problems disappear and perform data transmission through multiple paths [33].

IV ANALYSIS AND DISCUSSION

In this section, we have performed comparative analysis of existing solutions to the TCP incast and MPTCP incast. Solutions to the TCP incast are compared in the below Table-1. Table-2 contains MPTCP incast solutions and mechanisms followed by those protocols.

Table-1: Comparative analysis between TCP incast solutions

Technique	Implementation At	Additional shim layer	Server modification	Switch modification
OSDT	Application Layer	No	No	No
DSRAL	Application Layer	No	No	No
ARS	Application Layer	No	No	No

SERIAL	Application Layer	No	No	No
EDSRAL	Application Layer	No	No	No
Reducing RTO$_{min}$	Transport Layer	No	No	Yes
PAC	Transport Layer	No	Yes	No
AP	Transport Layer	No	No	Yes
T-RACKs	Transport Layer	Yes	No	No
M21TCP	Transport Layer	No	Yes	Yes
SA-TCP	Transport Layer	No	No	Yes
Large Switch Buffer	Link Layer	No	Yes	No
EFC	Link Layer	No	Yes	Yes
QCN	Link Layer	No	Yes	Yes

As given in Table-1 OSDT, DSRAL, ARS, SERIAL, EDSRAL are some of the application layer techniques that not require additional shim layer implementation, server modification and switch modification. Hence application layer techniques to mitigate TCP incast are considered as an efficient technique compared to transport layer and switch layer techniques. But as compared to transport layer solutions, application layer solutions do not support large number of servers under many-to-one communication pattern. Transport layer solutions demand modification at server, at ToR switch or at both the devices. T-RACKs which is transport layer solution does not require server and switch side modification but an additional shim layer need to be implemented above transport layer to allow T-RACKs to function. Lastly, link layer techniques require modification at server as well as at switch devices.

Compared to transport layer techniques, link layer techniques do not support large number of concurrent servers under many-to-one communication pattern.

Table-2: MPTCP incast solutions and their corresponding mechanisms

Protocol	**Mechanism**
MPTCP-L	Replicates packets on multiple subflows
AMTCP	Based on congestion situation dynamically adjusts number of subflows of a worker
FUSO	Retransmit recovery packets over another non-congested subflow
BMPTCP	Controls workers subflow congestion window size based on bottleneck switch buffer handling capacity
XMP	Employs ECN-based mechanism to control growth of subflow congestion window
ESE	Employs ECN-based mechanism to control growth of subflow congestion window
AMP	Based on congestion situation dynamically adjusts number of subflows of a worker

As per Table-2, MPTCP-L protocol replicates packets on multiple subflows. AMP and AMTCP protocols dynamically adjust number of subflows of a worker. FUSO protocols retransmit recovery packets over another non-congested subflow. BMPTCP controls congestion based on switch buffer size. XMP and ESE protocols employs ECN-based mechanism to control growth of subflow congestion window. It is important to consider bottleneck ToR switch buffer size while determining subflow congestion window size hence controlling congestion based on ToR switch buffer size is an effective method to mitigate MPTCP incast in multi-homed DCN.

V CONCLUSION

Many-to-one traffic pattern is common in DCNs, where simultaneous data transfer from multiple servers to a single client overloads the clients switch buffer. This results into one or more timeouts, retransmissions, throughput collapse and is called as incast problem in DCN. TCP incast and MPTCP incast are two types of incast problems. TCP incast is mitigated at application, transport and link layer. Application layer solutions are easy and efficient but fail to support large number of concurrent servers. More research is required to provide rate-based transport layer solutions which has potential to support large number of concurrent servers. Researchers have proposed variety of mechanisms to avoid MPTCP incast in multi-homed DCN.ECN-based mechanisms efficiently mitigate MPTCP incast problem by supporting large number of concurrent servers. Additional research is required to improve window-based solutions for mitigating MPTCP incast in multi-homed DCN.

REFERENCES

1) Amar Phanishayee, Elie Krevat, Vijay Vasudevan, David G. Andersen, Gregory R. Ganger, Garth A. Gibson, Srinivasan Seshan, “Measurement and Analysis of TCP Throughput Collapse in Cluster-based Storage Systems”, 6th USENIX Conference on File and Storage Technologies (FAST ’08) Feb. 26-29, 2008.

2) Elie Krevat, Vijay Vasudevan, Amar Phanishayee, David G. Anderson, Gregory R. Ganger, Ganger A. Gibso, Srinivasan Seshan, “On Application-level Approaches to Avoiding TCP Throughput Collapse in Cluster-based Storage Systems”, proceedings of the 2nd international Petascale Data Storage Workshop (PDSW ’17) November 11, 2007.

3) Irfan Riaz Shohab, Muhammad Younas, Ramzan Talib, Umer Sarwar, “Application Layer’s Approaches for TCP Incast problem in data center Networks”, International Journal of Computer Science and Mobile Computing, Vol.3 Issue 4, April 2014, pg 459-474.

4) Jiao Zhang, Fengyuan Ren, Li Tang, Chuang Lin, “Modeling and Solving TCP Incast Problem in Data Center Networks”, IEEE Transactions on Parallel and Distributed Systems, VOL. 26, No. 2, FEB 2015.

5) Juha Salo, “Data Center Network Architectures”, http://www.cse.tkk.fi/en/-publications/B/10/papers/Salo_final.pdf
6) Kashif Bilal, Samee U. Khan, Joanna Kolodziej, Limin Zhang, Khizar Hayat, Sajjad A Madani, Nasro Min- Allah, Lizhe Wang, Dan Chen, “A Comparative Study of Data Center Network Architectures”, Proceedings 26th European Conference on Modelling and Simulation.
7) Vijay Vasudevan, Amar Phanishayee, Hiral Shah, Elie Krevat, David G. Andersen, Gregory R. Ganger, Garth A. Gibson, Brian Mueller, “Safe and Effective Fine-grained TCP Retransmissions for Datacenter Communication”, SIGCOMM ’09, August 17-21, 2009.
8) Vijay Vasudevan, Hiral Shah, Amar Phanishayee, Elie Krevat, David Anderson, Greg Ganger, Garth Gibson, “Solving TCP Incast in Cluster Storage Systems”, http://www.pdl.cmu.edu/PDL-FTP/Storage/vasudevan_fast09wip.pdf
9) Yang Liu, Jogesh K. Muppala, MalathiVeeraraghavan, “A Survey of Data Center Network Architectures”, http://www.ece.virginia.edu/mv/pubs/recent-samples/Data-Center-Survey.pdf.
10) Suryavanshi, M. M. "Comparative analysis of switch-based data center network architectures." J MultidiscipEng Sci Technol (JMEST) 4, no. 9 (2017): 2458-9403.
11) Ford, Alan, Costin Raiciu, Mark Handley, and Olivier Bonaventure. TCP extensions for multipath operation with multiple addresses. No. rfc6824. 2013.
12) Raiciu, Costin, Sebastien Barre, Christopher Pluntke, Adam Greenhalgh, Damon Wischik, and Mark Handley. "Improving datacenter performance and robustness with multipath TCP." ACM SIGCOMM Computer Communication Review 41, no. 4 (2011): 266-277.
13) Yu, Ye, and Chen Qian. "Space shuffle: A scalable, flexible, and high-bandwidth data center network." In 2014 IEEE 22nd International Conference on Network Protocols, pp. 13-24. IEEE, 2014.
14) Chen, Yanpei, Rean Griffith, Junda Liu, Randy H. Katz, and Anthony D. Joseph. "Understanding TCP incast throughput collapse in datacenter networks." In Proceedings of the 1st ACM workshop on Research on enterprise networking, pp. 73-82. 2009.

15) Li, Ming, Andrey Lukyanenko, SasuTarkoma, and Antti Ylä-Jääski. "MPTCP incast in data center networks." China Communications 11, no. 4 (2014): 25-37.

16) Zhang, Shuli, Yan Zhang, Yifang Qin, Yanni Han, Zhijun Zhao, and Song Ci. "OSDT: A scalable application-level scheduling scheme for TCP Incast problem." In 2015 IEEE International Conference on Communications (ICC), pp. 325-331. IEEE, 2015.

17) Suryavanshi, Mahendra, Ajay Kumar, and Jyoti Yadav. "An application layer technique to overcome TCP incast in data center network using delayed server response." International Journal of Information Technology 13 (2021): 703-711.

18) Huang, Jiawei, Tian He, Yi Huang, and Jianxin Wang. "ARS: Cross-layer adaptive request scheduling to mitigate TCP incast in data center networks." In IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, pp. 1-9. IEEE, 2016.

19) Yang, Yukai, Hirotake Abe, Ken-ich Baba, and Shinji Shimojo. "A scalable approach to avoid incast problem from application layer." In 2013 IEEE 37th Annual Computer Software and Applications Conference Workshops, pp. 713-718. IEEE, 2013.

20) Suryavanshi, Mahendra, and Jyoti Yadav. "Mitigating TCP incast in data center networks using enhanced application layer technique." International Journal of Information Technology 14, no. 5 (2022): 2523-2531.

21) Bai, Wei, Kai Chen, Haitao Wu, Wuwei Lan, and Yangming Zhao. "PAC: Taming TCP incast congestion using proactive ACK control." In 2014 IEEE 22nd International Conference on Network Protocols, pp. 385-396. IEEE, 2014.

22) Zou, Shaojun, Jiawei Huang, Jianxin Wang, and Tian He. "Flow-aware adaptive pacing to mitigate TCP incast in data center networks." IEEE/ACM Transactions on Networking 29, no. 1 (2020): 134-147.

23) Abdelmoniem, Ahmed M., and BrahimBensaou. "Curbing timeouts for TCP-incast in data centers via a cross-layer faster recovery mechanism." In IEEE INFOCOM 2018-IEEE Conference on Computer Communications, pp. 675-683. IEEE, 2018.

24) Adesanmi, Akintomide, and LotfiMhamdi. "Controlling TCP Incast congestion in data centre networks." In 2015 IEEE International Conference on Communication Workshop (ICCW), pp. 1827-1832. IEEE, 2015.

25) Ren, Yongmao, Jun Li, Guodong Wang, Lingling Li, and Shanshan Shi. "SA-TCP: A novel approach to mitigate TCP Incast in data center networks." In 2015 International Conference on Computing and Network Communications (CoCoNet), pp. 420-426. IEEE, 2015.

26) Devkota, Prajjwal, and AL Narasimha Reddy. "Performance of quantized congestion notification in TCP incast scenarios of data centers." In 2010 IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, pp. 235-243. IEEE, 2010.

27) Wang, Wei, Liang Zhou, and Yi Sun. "Improving multipath TCP for latency sensitive flows in the cloud." In 2016 5th IEEE International Conference on Cloud Networking (Cloudnet), pp. 45-50. IEEE, 2016.

28) Li, Long, Nongda Hu, Ke Liu, Binzhang Fu, Mingyu Chen, and Lixin Zhang. "Amtcp: an adaptive multi-path transmission control protocol." In Proceedings of the 12th ACM international conference on computing frontiers, pp. 1-8. 2015.

29) Chen, Guo, Yuanwei Lu, Yuan Meng, Bojie Li, Kun Tan, Dan Pei, Peng Cheng et al. "FUSO: fast multi-path loss recovery for data center networks." IEEE/ACM Transactions on Networking 26, no. 3 (2018): 1376-1389.

30) Suryavanshi, Mahendra, Ajay Kumar, and Jyoti Yadav. "Balanced Multipath Transport Protocol for Mitigating MPTCP Incast in Data Center Networks." International Journal of Next-Generation Computing 12, no. 3 (2021).

31) Cao, Yu, Mingwei Xu, Xiaoming Fu, and Enhuan Dong. "Explicit multipath congestion control for data center networks." In Proceedings of the ninth ACM conference on Emerging networking experiments and technologies, pp. 73-84. 2013.

32) Zhang, Xinming, Sikun Liu, and Jia Xu. "An efficient scheduling scheme for XMP and DCTCP mixed flows in commodity data centers." IEEE Communications Letters 22, no. 9 (2018): 1770-1773.

33) Kheirkhah, Morteza, and Myungjin Lee. "AMP: An adaptive multipath TCP for data center networks." In 2019 IFIP networking conference (IFIP networking), pp. 1-9. IEEE, 2019