Datacentre TCP Protocol of Centralized Window Control

Chunlin Wang\textsuperscript{1,a}, Jianyong Sun\textsuperscript{1,c}, Wanjin Xu\textsuperscript{1,d}, Xiaolin Chen\textsuperscript{1,b}

\textsuperscript{1}School of Information Science & Technology, Chuxiong Normal University, P.R.China
Corresponding author: Xiaolin Chen
\textsuperscript{a}wcl@cxtc.edu.cn, \textsuperscript{b}chenxl@cxtc.edu.cn, \textsuperscript{c}sunny@cxtc.edu.cn, \textsuperscript{d}xwj_cx@cxtc.edu.cn

ABSTRACT There is an important class of applications in data centre called online data-intensive (OLDI) application, such as search engine, electronic shopping, and advertising, which has low latency, high throughput, and high burstiness. For better user experience, the query response delay in the data centre is controlled under 300 milliseconds. DCTCP can better meet the requirements of OLDI, but all the switches in the data centre are required to support RED. For data centre that does not fully support RED switch, TCP protocol can only be used. TCP Incast will appear in the application, which is far from the requirements of OLDI application. This paper proposes the TCP protocol of data centre based on window control. It is in existing switches and network devices, and centralized control makes the send window match with the switch queue cache. In the case of ensuring low latency, it keeps the switch queue in a small range of fluctuation, enabling the data centre to meet low latency, high throughput and high burstiness application requirements. Simulation experiments verify that WCTCP congestion control method can effectively solve the above problems. It is 5 times lower than TCPNewReno packet loss rate, 6 times higher than the network utilization rate, and the network performance is close to the DCTCP protocol.

1. INTRODUCTION
In recent years, the data centre has changed the application mode of computer from the traditional single-machine mode and network mode to the current computing mode with data centre as the core, and with the emergence of cloud computing service providers such as Alibaba, Microsoft and Google. In data center design, using low-cost switches and servers to build high-performance, manageable and maintainable data center is an ideal goal for all data center operators. On the other hand, with the continuous growth of Internet applications, Internet users expect shorter query response and higher throughput to meet different applications, such as network search, online e-commerce, advertising promotion and online on-demand. This type of application is called online data-intensive application (OLDI) [3], which controls the query response delay within the data centre to be less than 300 milliseconds. Online data-intensive application (OLDI) generates different short data stream and long data stream, and expects data centre network to meet the requirements of low latency, sudden fault tolerance, and high throughput of long data stream. After subtracting the typical Internet delay, the total network delay of OLDI within the data center is usually less than 230-300ms, which is also known as soft real-time constraint. If soft real-time constraint exceeds 300ms, it will affect user experience and operator revenue. OLDI soft real-time constraint (300ms) cannot meet the application requirements in traditional 1:1 client and server mode. Therefore, OLDI adopts a tree-based divide-and-conquer algorithm (the calculation mode of N: 1). Each query is decomposed into multiple sub-queries through the control centre, then distributed to different servers...
to complete the query, and the results of the query are fed back to the control centre. For OLDI application like this, soft real-time constraint is the key metric. To avoid missing soft real-time constraint, the control centre issues incomplete response without waiting for slow query that misses deadline. Soft real-time constraint.timeout can negatively impact OLDI application, reducing user experience and affecting the competitiveness of data centre and service providers. Therefore, reducing the missing soft real-time constraint is one of the focuses of data centre research.

2. CURRENT RESEARCH

2.1 Problem Is Introduced
In OLDI application, the client decomposes the query task into multiple sub-tasks assigned to hundreds of servers. When the server finishes the query and returns the query results to the client via the switch, the queue of the switch will be occupied in an instant until the buffer overflow occurs and the packet is lost due to the server sending the packet to the switch at the same time (N: 1). TCP has two ways to handle data retransmission after packet loss: one is to trigger timeout retransmission when the timer goes off. The second is that the sender receives three repeated ACK acknowledgment messages, triggering fast retransmission. Traditional TCP retransmission timer (RTO) is generally no less than 200ms, while data center network environment round-trip delay (RTT) is generally on the order of microseconds. During this period, the server's sending window continuously sends data until the sending window runs out. The network throughput drops dramatically when the switch is idle from the sending window running out to the start of retransmission. This phenomenon is called TCP Incast[6].

The preconditions for the occurrence of TCP Incast include: 1. the network has high bandwidth and low latency, and the cache of the switch is small. 2. There is synchronous many-to-one traffic in the network. 3. The data traffic on each TCP connection is small. Literature [4-5] all believe that packet loss leads to timeout retransmission, and that the timeout time of TCP retransmission of RTO does not match the RTT value, which leads to TCP Incast problem. Currently, three methods are commonly used in research to solve TCP Incast: 1. RTO is set according to the environment of the data centre, with reduced retransmission timer, so that it can be retransmitted earlier. 2. Larger switch cache, will increase the operating cost of the data centre. 3. Better congestion control enables better matching between the sending window and the switch cache to reduce packet loss and improve network utilization. In academia and industry, the TCP protocol congestion control of data centre is mainly studied. This paper also focuses on congestion control.

2.2 The Research Progress
The DCTCP protocol [1] was proposed by Microsoft research institute at the SIGCOMM conference in 2010. The idea of DCTCP is to use explicit congestion notification (ECN). It first monitors the queue length in the switch. If the queue length exceeds the upper limit, the TCP receiver is explicitly

![Figure 1. Experimental topology](image)
notified through the ECN field. The receiving end reduces the sender's sending window through the window field, reduces the sending speed, and reduces the packet entering the switch queue. This equalizes the queue length of the switch to a certain length, reduces packet loss and increases network utilization. In 2011 SIGCOMM, Balajee Vamanan from Purdue university proposed the D2TCP congestion control protocol [2], which has better emergency response and lower latency. D2TCP adopts a new congestion avoidance algorithm, which uses ECN feedback and deadline to modulate the congestion window through gamma correction function, to speed up the delivery speed and priority of services when latency is close to soft real-time constraint, and to reduce soft real-time timeout.

2.3 Problems To Be Solved
The two previous approaches have two drawbacks: 1. All switches in the data centre are required to support ECN. 2. The TCP protocol needs to be modified, and the operating system should support the corresponding TCP protocol. These are hard to do in some smaller data centres. This paper focuses on small data centres that do not support ECN, conducts research around TCP protocol congestion control of data centres, and proposes solutions to the TCP Incast problem. This provides small data

3. The solution

3.1 Algorithm thought
This paper proposes a congestion control algorithm for centralized sending window control. The idea is: in OLDI application, if the sending window is too large, multiple senders will send data to the network continuously. This results in queue overflow, packet loss and idleness of the switch, which reduces network performance. On the other hand, a small sending window reduces network utilization and throughput. In this algorithm, the sum of the sending windows of all service groups is centrally controlled and constrained to match the cache size of the switch and keep the queue length of the switch within a certain range of fluctuation. It not only avoids overflow and packet loss in switch queue, but also ensures high network utilization and low latency. In the algorithm, the total window is initialized first, and the average value of the total window is the upper limit of each server sending window. If no packets are lost, adjust the value of the total window. If retransmission reduces the value of the total window, update the upper limit of the server sending window again. If the data sent by the server is smaller than that of other servers, the upper limit of the sending window of the server is reduced and the upper limit of the sending window of other servers is increased. The size of the total window in this algorithm is one of the researches focuses in this paper. The relationship between the total window, the size of the switch cache and the network utilization in this algorithm is described in figure 2.

3.2 Algorithm description
Start as: upper limit of the total window connected wnd_cap=min[switch_link_buf[1],switch_link_buf[2],…,switch_link_buf[n]]; switch_link_buf[n] is the number of switch cache in the connection, and the limit value of the connection window is window←0. Connection data N← number of servers
Update the upper limit of window\(_\_\_\_\) check_trans(N)\{
    
    If(connections_number < N)\{
        If (\(\text{wnd\_cap} / (\text{connections\_number} + 1)\) >= 2)
            connections_number \(\leftarrow\) connections_number + 1
            window\_\_\_\_ \(\leftarrow\) int(wnd\_cap / connections\_number)
            new_connection(connections\_number)
    
    For( i \(\leftarrow\) 0; i <= N; i++)
        
        If(prev\_npkts\_i >= npkts\_i)
            connections_number \(\leftarrow\) connections\_number - 1
            window\_\_\_\_ \(\leftarrow\) int(wnd\_cap / connections\_number)
        
        prev\_npkts\_i \(\leftarrow\) npkts\_i
    
    If(N > 0) check_trans(N)
}

Update the total size of the wnd\_cap sending window: \(g \leftarrow 0.8\), queue\_length: current switch queue length

\[
\left\{ \begin{array}{ll}
\text{wnd\_cap} = \text{wnd\_cap} - 1 & (\text{queue\_length} > g \times \text{swich\_link\_buf}) \\
\text{wnd\_cap} = \text{wnd\_cap} + 1 & (\text{queue\_length} < g \times \text{swich\_link\_buf})
\end{array} \right. 
\]

4. Evaluation
In the experiment, NS2 is used to conduct simulation experiments on the algorithm, and the experimental structure is shown in figure 1:8 servers, a switch and a client are used in the experiment. The algorithm is evaluated in terms of queue length, query delay, throughput rate and loss rate.

4.1 Queue Length Evaluation
In the queue length evaluation, TCPnewReno and DCTCP, two congestion control algorithms are respectively used for comparison. The switch cache swich\_link\_buf \(\leftarrow 16\) is set, and the amount of data to be sent is 1.6MB. The change of the switch queue length is observed, as shown in figure 3. After TCPnewReno is transmitted for a period of time, the phenomenon of Incast appears after packet loss, resulting in rapid decline of network utilization. WCTCP algorithm queue always fluctuates between [10,14], which has a smaller queue length than DCTCP algorithm, and the transmission delay is basically the same.

Figure 3. Queue Length
4.2 Delay
In the latency measurement experiment, 8KB, 24KB, ..., 157MB data blocks are sent respectively in 1Gb network, as shown in figure 4. In the experiment, since WCTCP and DCTCP algorithms differ very little in transmission delay ($10^{-4}$), they are highly overlapping in the figure. In data block transmission under 17MB, WCTCP can control the transmission time well below soft real-time constraint (300ms), meeting the requirements of OLDI application for short data stream with low latency.

4.3 Throughput
In the throughput measurement (as shown in figure 5), WCTCP can maintain a high network utilization rate in the network bandwidth of 0.1Gb to 5Gb. Throughput rate is decreasing after 5Gb and consistent with TCP. This is mainly because in the 5Gb and consistent with TCP. This is mainly because in the experimental test, the total switch cache of the algorithm is 100KB, and the service sending window is 18PKT.

![Figure 4. Delay](image)

In a network of more than 10Gb, the mismatch between the switch cache and the sending window is the cause of the decrease in network utilization. Solving this problem can increase the size of the switch cache and the sending window.

![Figure 5. Throughput](image)
4.4 Loss Rate

In the test of packet loss rate, 1.6MB packets are sent, and the packet loss of switch is recorded in the process of the transmission rate increasing from 0.1Gb to 50Gb. The packet loss rate of WCTCP algorithm is gradually decreasing. After 5Gb, the network packet loss rate is close to zero. Experimental results show that it has lower packet loss rate than TCP protocol.

5. CONCLUSION

Simulation experiments verify that WCTCP algorithm is in line with expectations, with low switch queue length, high throughput and low latency. In low-cost data center application, the WCTCP algorithm does not need to support RED switch. The dynamic allocation of the sending window can reduce the network congestion, which is 5 times lower than the packet loss rate of TCPNewReno, 6 times higher than the network utilization rate, and the network performance is close to the DCTCP protocol. In a network of more than 10Gb, more support of sending window and high-performance switch is needed, otherwise the performance drops dramatically.

ACKNOWLEDGMENT

This work is supported in part by Yunnan Provincial Department of Education Project Data Center and the Science and Technology Program of Yunnan Province, China. And the soft timeout transmission control protocol project number and the Grant Number are respectively 2018JS454 and 2017FH001-124

REFERENCES

[1] Alizadeh M, Greenberg A, Maltz D A, et al. Data center tcp (dctcp)[J]. ACM SIGCOMM computer communication review, 2011, 41(4): 63-74.
[2] Vamanan B, Hasan J, Vijaykumar T N. Deadline-aware datacenter tcp (d2tcp)[J]. ACM SIGCOMM Computer Communication Review, 2012, 42(4): 115-126.
[3] Meisner D, Sadler C M, Barroso L A, et al. Power management of online data-intensive services[C]//ACM SIGARCH Computer Architecture News. ACM, 2011, 39(3): 319-330.
[4] Vasudevan V, Phanishayee A, Shah H, et al. Safe and effective fine-grained TCP retransmissions for datacenter
[5] Chen Y, Griffith R, Liu J, et al. Understanding TCP incast throughput collapse in datacenter networks[C]//Proceedings of the 1st ACM workshop on Research on enterprise networking. ACM, 2009: 73-82.
[6] Nagle D, Serenyi D, Matthews A. The panasas activescale storage cluster: Delivering scalable high bandwidth storage[C]//Proceedings of the 2004 ACM/IEEE conference on Supercomputing. IEEE Computer Society, 2004: 53.
[7] Kwan B, Agarwal P, Ashvin L. Flexible buffer allocation entities for traffic aggregate containment: U.S. Patent 8,532,117[P]. 2013-9-10.

[8] Al-Fares M, Loukissas A, Vahdat A. A scalable, commodity data center network architecture[C]//ACM SIGCOMM Computer Communication Review. ACM, 2008, 38(4): 63-74.

[9] Floyd S. RED: Discussions of setting parameters[J]. http://www.aciri.org/floyd/REDparameters.txt, 1997.

[10] Qazi I A, Andrew L L H, Znati T. Congestion control using efficient explicit feedback[C]//INFOCOM 2009, IEEE. IEEE, 2009: 10-18.

[11] Vasudevan V, Phanishayee A, Shah H, et al. Safe and effective fine-grained TCP retransmissions for datacenter communication[C]//ACM SIGCOMM computer communication review. ACM, 2009, 39(4): 303-314.

[12] Xia Y, Subramanian L, Stoica I, et al. One more bit is enough[J]. ACM SIGCOMM Computer Communication Review, 2005, 35(4): 37-48.

[13] Wilson C, Ballani H, Karagiannis T, et al. Better never than late: Meeting deadlines in datacenter networks[J]. ACM SIGCOMM Computer Communication Review, 2011, 41(4): 50-61.

[14] Ellison N B, Steinfield C, Lampe C. The benefits of Facebook “friends:” Social capital and college students’ use of online social network sites[J]. Journal of Computer - Mediated Communication, 2007, 12(4): 1143-1168.