Reviving Delay-based TCP for Data Centers

Changhyun Lee
KAIST
Daejeon, Korea
chlee@an.kaist.ac.kr

Keon Jang
Microsoft Research
Cambridge, UK
keonjang@microsoft.com

Sue Moon
KAIST
Daejeon, Korea
sbmoon@kaist.edu

ABSTRACT

With the rapid growth of data centers, minimizing the queueing delay at network switches has been one of the key challenges. In this work, we analyze the shortcomings of the current TCP algorithm when used in data center networks, and we propose to use latency-based congestion detection and rate-based transfer to achieve ultra-low queueing delay in data centers.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols

General Terms

Performance

Keywords

Data centers, Latency, TCP

1. INTRODUCTION

The recent growth of data centers has brought new challenges to network research. Among the challenges, minimizing network delay within a data center is one of the key concerns for service operators. Previous work [1] has well pointed out that user experience is badly affected in data center applications when even a single flow suffers from a large latency.

In today's data centers, the largest part of network delay comes from the queueing delay at switches since the propagation delay within a data center can be almost negligible; ideally, 100 meters of network cabling between two nodes adds only 0.5μs of propagation delay, while a single 1,500 byte packet queued at a 10Gbps switch already costs 1.2μs. Recently, much effort has been put to reduce the queueing delay at switches. DCTCP [2] uses ECN marking to slow down flows before the queue becomes full. HULL [3] takes a further step and give up a little bandwidth to have even lower latency than DCTCP. Our work is also motivated to reduce the queueing delay in data centers, but we focus on the end-to-end congestion control not on queue management at switches. The final goal is to keep the queue occupancy low and fully utilize the link bandwidth at the same time.

In this work, we demonstrate why the current TCP cannot keep latency low when operating in data centers even with AQM policies and lay out directions for overcoming the hurdles. First, we present the problems in loss-based congestion detection and window-based congestion control. Second, we propose to use a delay-based congestion detection and rate-based transfer, and show why it could be a feasible solution for data center environment with simulation results. Finally, we present future directions and challenges of our approach.

SIGCOMM'12, August 13–17, 2012, Helsinki, Finland.
ACM 978-1-4503-1419-0/12/08.

2. ROOT SOURCES OF LARGE LATENCY

Loss-based congestion detection

How to detect congestion in network links and notify the sender is an important part of the congestion control mechanism. The most commonly used TCP version in the Internet, TCP Reno, relies on the packet loss event; a sender does not stop until one of the queues in the network path gets full and drops packets. This mechanism is born to keep the queues occupied. Queue management schemes using ECN may alleviate the situation, but they still suffer from the same dilemma; you need to fill up the queue first in order to empty the queue. As queueing delay is the major source of network delay in data centers, full queues are the last thing that data centers would like to have.

Window-based congestion control

TCP Reno controls the rate of a flow by adjusting the window size, and packets in the same window are sent back-to-back. Therefore packets arriving at a switch have a high level of burstiness and easily fill the queue. As minimizing the queue length is the key issue in data center networks, window-based transmission is not a proper scheme to be used.

The burstiness may fade out as the size of a window decreases, but there still is a limitation in window-based transfer. When network congestion is severe, the size of a window can reduce to one in low latency networks. We run a simulation for this scenario and show in Figure 1. We use a dumbbell topology where data nodes are connected through a single 10Gbps switch, and end-to-end propagation delay is 10μs. Here we gradually increase the number of TCP flows from 0 to 50 until 2.1 second. The congestion window size of each flow is fixed to one during the simulation, so each flow repeatedly sends one data packet, waits for an ACK packet, and then sends another data packet; we call it pingpong transfer. This pingpong transfer is the slowest way of sending data in the current TCP, but we see that as soon as the number of flows exceeds 19, the queue is starting to grow and experience increasing delay. In this scenario, there is no way of doing throughput control on these flows other than stopping them for a retransmission timeout after a loss occurs. The wide-area Internet does not have to worry about this case because of large RTT, but it is a practical concern in networks of small propagation delay like data centers.

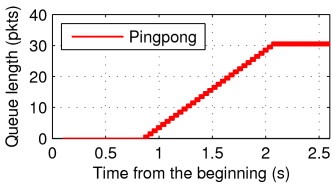

Figure 1: Minimum sending rate simulation of TCP

3. FUTURE DIRECTIONS

Based on the limitations of current TCP in data center networks, we present a preliminary design of our proposed congestion control mechanism. The primary goal is to achieve low queueing delay without adding any features to switches.

Latency-based congestion detection

To overcome the shortcomings of loss-based congestion detection, many TCP variants have been proposed in the literature. One of the solutions designed for the Internet is XCP [4]. It adds a congestion header to a packet which is frequently updated by specialized XCP routers. While the queueing delay reduction of XCP is notable, using it for data centers requires modification of all switches and routers in the networks, which is not truly practical. Recently proposed techniques for data centers such as DCTCP [2] and HULL [3] also require some level of modification at switches.

As a better way of congestion detection in an end-to-end manner, we propose to use a latency-based algorithm from TCP Vegas. Although TCP Vegas is not the most popular protocol in the Internet, it has much potential in data centers. First, network topology and routing in data centers are much more static than in the wide-area Internet. Stability in routing ensures that latency change occurs only from network congestion while delay measurement in the Internet suffers from various type of noises. Second, the portion of propagation delay and transmission delay in total end-to-end delay is much smaller than that of queueing delay. In this environment, one additional packet queueing greatly increases the total network delay unlike the Internet. Provided that accurate delay measurement in microseconds is feasible and affordable, delay is a good source to determine the congestion level in data center networks.

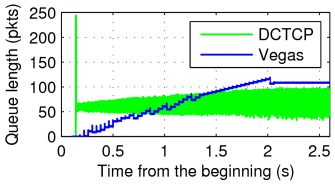

Figure 2: Queue length of TCP Vegas and DCTCP simulation

To explore the potentials of TCP Vegas in data center networks, we run a basic simulation and compare the result with DCTCP; we use the ns-2 code and sample simulation script of DCTCP available online. The same dumbbell topology is used as in the previous section, but the end-to-end propagation delay is changed to 50μs. The maximum queue size at the switch is 250 packets. During the simulation, we increase the number of flows from 0 to 50 until 2.1 second. In Figure 2, we find that latency-based congestion detection achieves comparable queue length reduction with DCTCP and exhibits much less queue length variation. The queue length of TCP Vegas, unlike DCTCP, is proportional to the number of flows in the simulation result, and how to maintain the queue small regardless of the number of flows will be the primary goal of our future work.

Rate-based transfer using inter-packet gap

We have shown that window-based transfer is not an efficient way for low latency networks. To achieve fine-grained throughput control, we propose to use rate-based transfer similar to TFRC protocol [5]. According to the difference between the base RTT and the current RTT, we decide the time gap between two consecutive packets to be sent. When a network path is congested, the time gap is increased to slow down a flow. In this approach, we can micro-manage the rate of a flow by adjusting the time gap a bit by bit. When a network is too congested, a flow can be sent even slower than the pingpong manner by setting the time gap larger than RTT. The micro-grained time gap can be gradually reduced or increased to the point where network bandwidth is fully utilized and queue occupancy remains low.

Challenges

There are several remaining challenges in our new algorithm design and implementation. First, measuring the base RTT for a flow can be easily affected by background flows. The ideal base RTT value will be the sum of propagation delay and transmission delay of a network path, but getting an accurate base RTT in presence of background traffic is not an easy problem. We currently resort to the minimum value of observed RTTs since the beginning of a flow, but we consider creating a high priority channel only for base RTT measurement that is separated from data transfer channel. Second, sending packets with the exact inter-packet gap may not be easy in real world implementation. To fully benefit from rate-based transfer, inter-packet gap in practice can be as low as sub-microseconds, and precisely scheduling packets at operating system level can be hard to be done. Therefore we plan to offload TCP packet scheduling to a specialized hardware that helps accurate timing control. We are currently revising our algorithm design to have better performance and planning to do more simulations and experiments with various network topologies and traffic pattern scenarios.

4. ACKNOWLEDGMENTS

This research was supported by the KCC (Korea Communications Commission), Korea, under the R&D program supervised by the KCA (Korea Communications Agency) (KCA-2011-08913-05002).

5. REFERENCES

[1] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowtron. Better Never than Late: Meeting Deadlines in Datacenter Networks. In *Proceedings of the ACM SIGCOMM conference*, 2011.

[2] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. Data Center TCP (DCTCP). In *Proceedings of the ACM SIGCOMM conference*, 2010.

[3] M. Alizadeh, A. Kabbani, T. Edsall, B. Prabhakar, A. Vahdat, and M. Yasuda. Less Is More: Trading a Little Bandwidth for Ultra-Low Latency in the Data Center. In *Proceedings of USENIX NSDI conference*, 2012.

[4] D. Katabi, M. Handley, and C. Rohrs. Congestion Control for High bandwidth-delay Product Networks. In *Proceedings of the ACM SIGCOMM conference*, 2002.

[5] S. Floyd, M. Handley, J. Padhye, and J. Widmer. TCP Friendly Rate Control (TFRC): Protocol Specification. RFC 5348 (Proposed Standard), Sept. 2008.