xDCTCP – Extended DCTCP

Checkpoint 2 Progress Report

Objective

Our target for Checkpoint 2 was completion of implementation and initial performance measurements.

Course Correction post checkpoint 1

By checkpoint 1, we were successful in implementing DCTCP[1] using 2 VMs on our own system. But this approach wasn't scalable because of limitations of our system and changes to DCTCP would have required recompilation of kernel on each of the VMs individually. So, after discussion with the TA and Dr. Porter, we decided to shift to NS2 for xDCTCP implementation and simulation. We asked the authors of DCTCP for the original NS2 code and started modifying it as per our requirement.

Topology and Parameters

Our topology consists of 8 senders (labelled 0-7), 1 switch and 1 receiver as shown in the figure 1.

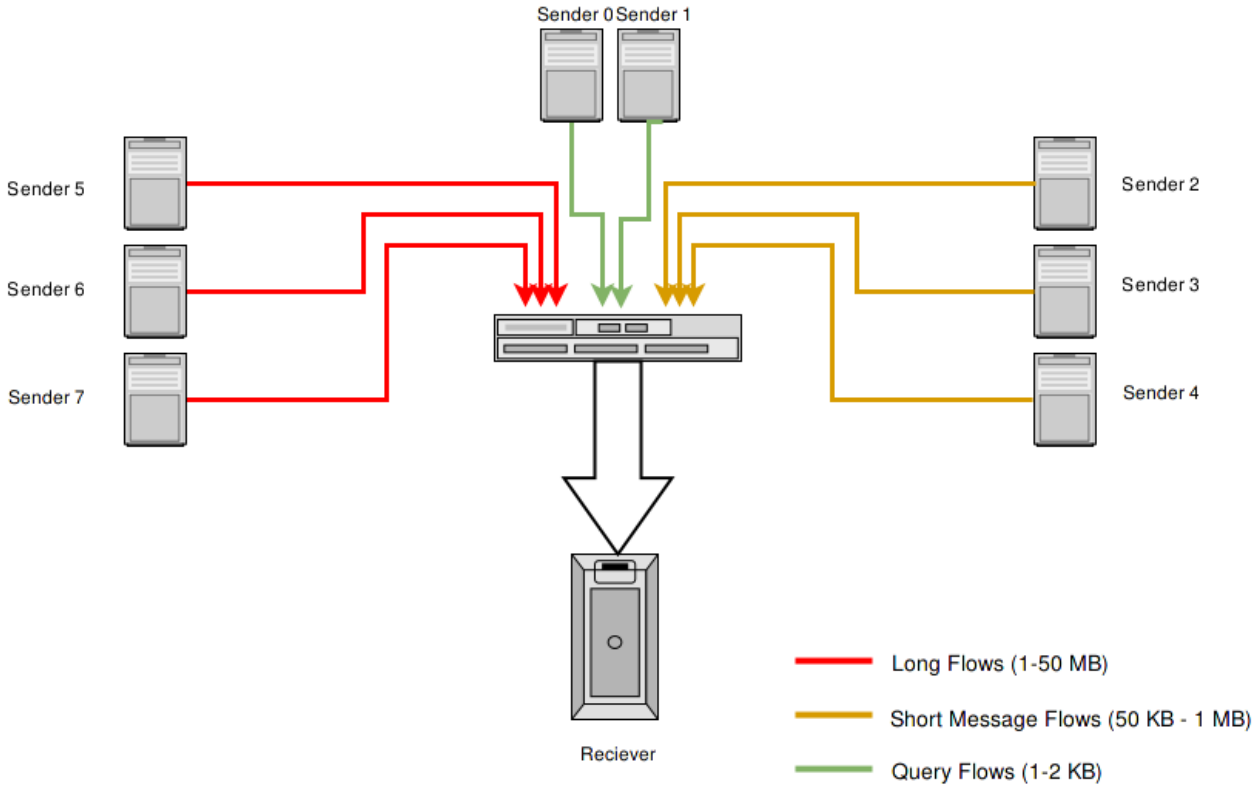

Figure 1: Topology

xDCTCP related parameters:

K = 65 (K is the threshold queue length for the switch to start marking ECN field in packets)
B = 250 (B is the max queue length post which switch starts dropping packets)
RTT = 100 µs (RTT in case of no queue delay)
Simulation time = 1 second
Packet Size = 1460 Bytes
Line Rate = 10 Gbps
Above parameters are same as the original DCTCP simulation.

Since data center traffic consists of flows of different sizes, we tried to simulate the same using following traffic composition:
Query Flows: 0, 1 -> Send 10KB every 1 ms
Short Flows: 2, 3, 4 -> Send 500Kb every 10 ms
Long Flows: 5, 6, 7 -> Continuous flow from start to end

Implementation Approach

$$cwnd = cwnd \times \left(1 - \frac{\propto}{n}\right) \quad \text{.....(1)}$$

$$\text{where } \propto = (1 - g) \times \propto + g \times F \quad \text{.....(2)}$$

(n=2, F = Fraction of packets marked in last window, g = Weight given to new samples against the past ones)

As per initial proposal, we planned on studying the performance of DCTCP on varying g and n. DCTCP claims to reduce delay for short flows while keeping the throughput for long flows reasonably high. But it uses the same factor for reducing the congestion window for both short and long flows (n = 2 for both short and long flows in equation (1)). We decided to differentiate the treatment of short and long flows using separate factors as per equation (3) and (4) respectively.

$$cwnd = cwnd \times (1 - dctcp\_ns\_ \times \propto) \quad \text{.....(3)}$$

$$cwnd = cwnd \times (1 - dctcp\_nl\_ \times \propto) \quad \text{.....(4)}$$

Calculation of ∝ remains the same as equation (2).

D2TCP[2] differentiates the window reduction by varying the congestion window as per the deadline, but that requires the aggregator to inform the worker of the deadline. Our approach is simpler and doesn't require any changes at the aggregator and comparatively few changes at the workers.

The reasoning behind our approach was that for reduction of delay for short flows, it makes sense to reduce their windows by a smaller value while reducing the windows for long flows by a greater value in case of congestion. But we need to make sure that this doesn't affect the throughput of long flows significantly. These were considerations while selecting the optimum triplet (g, dctcp_ns_, dctcp_nl_).

As per the DCTCP authors, $g < \frac{1.386}{\sqrt{2\,(C \times RTT + K)}}$ where C = link capacity in packets/second, RTT = the round trip time and K = threshold queue length for the switch to start marking ECN field in packets. For our setup, we get g < 0.112.

We ran our simulation for all possible triplets (g, dctcp_ns_, dctcp_nl_) to find the optimum triplet where each of these varied as below:

g -> [0.05, 0.055, 0.06, 0.0625, 0.065, 0.07, 0.075, 0.08, 0.085, 0.09, 0.095, 0.1, 0.105, 0.11]

dctcp_ns_ -> [0.1, 0.2, 0.3, 0.4, 0.5]

dctcp_nl_ -> [0.5, 0.6, 0.7, 0.8, 0.9, 1.0]

Results

The optimum point which corresponds to minimum average delay for which average throughput doesn't take significant hit is obtained at *(g = 0.095, dctcp_ns_ = 0.4, dctcp_nl_ = 1)* at which average throughput for long flows = *2689.97 Mbps*, average delay for short flows = *69.03 µs*. For the original DCTCP parameters *(g = 0.0625, dctcp_ns_ = 0.5, dctcp_nl_ = 0.5)*, average throughput for long flows = *2690.26 Mbps*, average delay for short flows = *73.80 µs*. Thus, we managed to reduce the average delay for short flows by *6.46%* while affecting the throughput by just *0.01%* as compared to DCTCP.

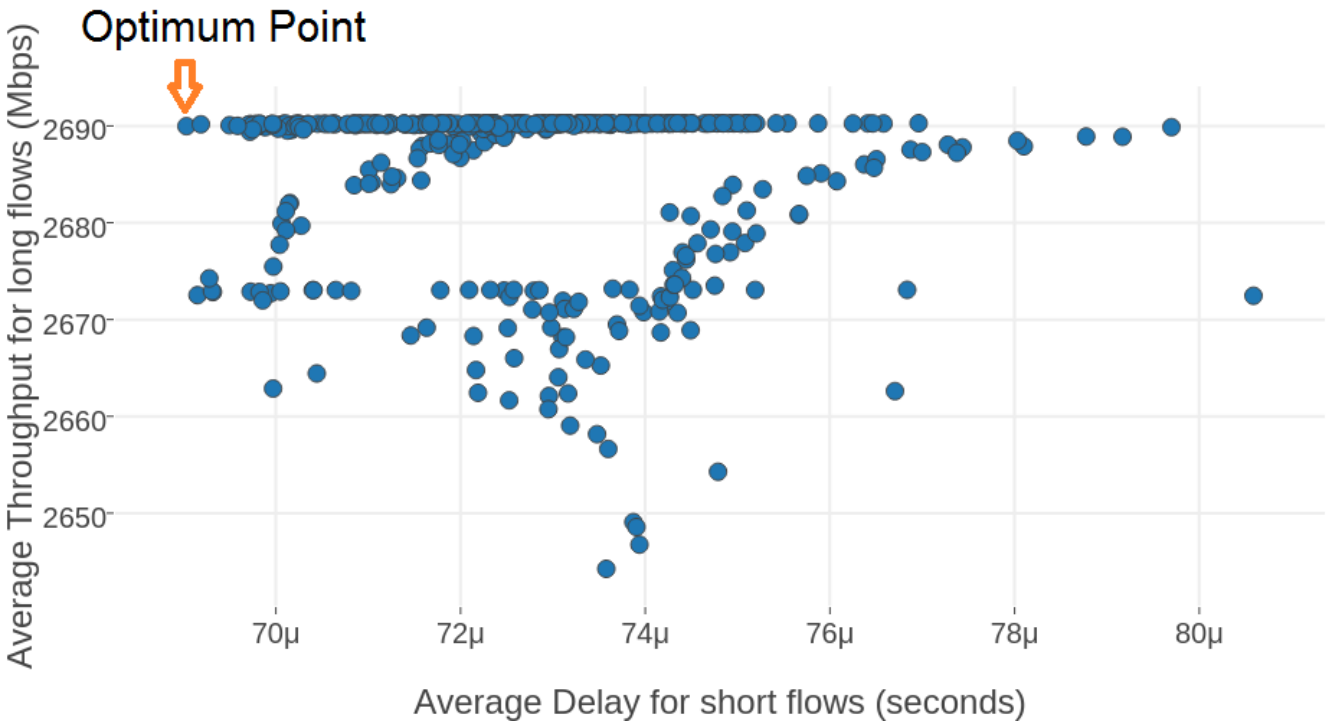

Figure 2: Avg Throughput for long flows vs Avg Delay for short flows for all possible triplets (g, dctcp_ns_, dctcp_nl_)

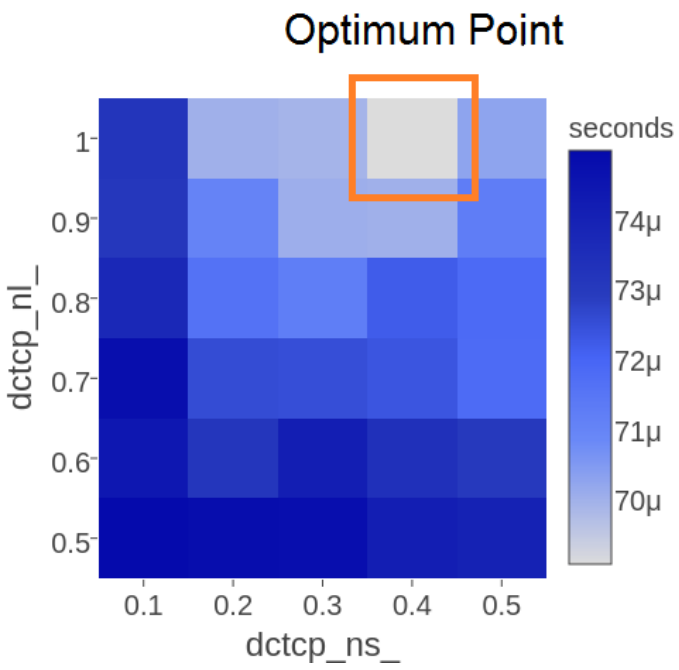

Figure 3: dctcp_ns_ vs dctcp_nl_ vs average delay for short flows for optimum g = 0.095

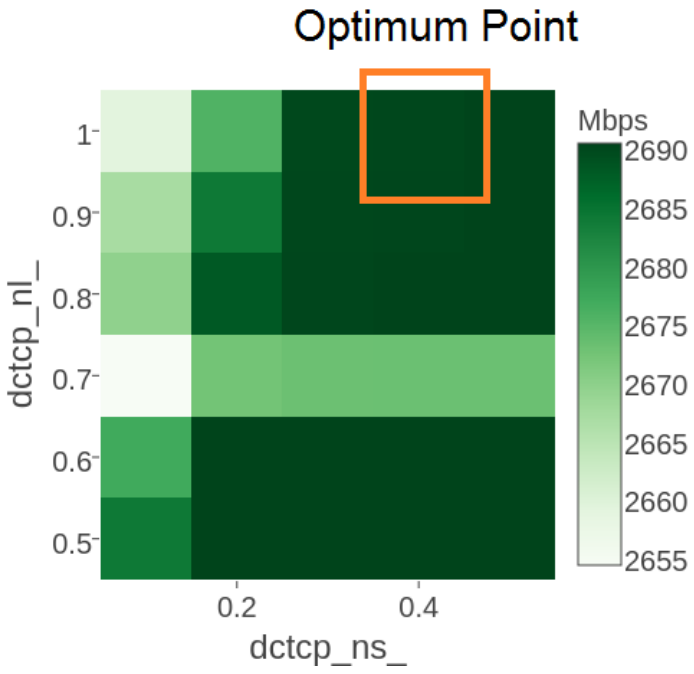

Figure 4: dctcp_ns_ vs dctcp_nl_ vs average throughput for long flows for optimum g = 0.095

Future Work

We need to improve the traffic composition to match data center traffic and also obtain corresponding data related to congestion window variation and queue length variation.

References

[1] Alizadeh, Mohammad, et al. "Data center tcp (dctcp)." ACM SIGCOMM computer communication review 41.4 (2011): 63-74.

[2] Vamanan, Balajee, Jahangir Hasan, and T. N. Vijaykumar. "Deadline-aware datacenter tcp (d2tcp)." ACM SIGCOMM Computer Communication Review 42.4 (2012): 115-126.