Thoughts on Real-Time Congestion Control

Murari Sridharan

There is an abundance of congestion control research out there which informs how we should approach the problem for interactive real-time communication. As one example, with Compound TCP (CTCP) we showed that it is possible to effectively combine multiple congestion signals into an effective rate control algorithm. By effectively combining delay and loss information we were able to design an algorithm that can fully utilize the pipe, yet yield to others when there is contention. It does so by monitoring changes to base delay. This clearly works well in terms of throughput and fairness; it however suffers from one problem which is fundamental to how TCP works. TCP is designed to cause a buffer overflow to sense available bandwidth because it inherently relies on the loss signal and self-induced loss is critical to fairness among multiple flows. Majority of the TCP variants out there are loss-only congestion control and therein lays the bulk of the problem. The entire body of congestion research has one primary goal, how to be fair to Standard TCP. Standard TCP returns the favor by filling up the buffers and increasing latency for everybody. Recent research has shown that there is a ton of buffering on the Internet starting with the Internet Gateway Device in people's homes.

Satisfying the needs of throughput sensitive flows with that of latency sensitive flows has eluded TCP research for a while. With Datacenter TCP (DCTCP) we showed a way to leverage Explicit Congestion Notification (ECN) in the network to provide *explicit* feedback to the end-hosts using ECN marks. Note delay signal provides *implicit* multi-bit feedback about the bottleneck queue. Instead of reacting to the presence of congestion, DCTCP reacts to the *extent* of congestion. Most importantly we found that DCTCP delivers same or better throughput using *90% less buffer space*.

ECN is *key* to improving performance on the Internet. For years we were in a "chicken and egg" situation due to lack of end-hosts and switch/network support. Most modern OS' today support ECN. One of the key reasons why ECN was not turned on in operator networks is because of the complexity involved in configuring AQM parameters. However we believe a simple marking scheme using a single parameter called the marking threshold (K) is enough. The AQM algorithm then is mark the packet if the buffer occupancy is greater than K. The end-hosts *derive multi-bit feedback from the information present in the single-bit sequence of marks*. The estimator derived from the ECN feedback at the sender estimates the probability that the queue size is greater than K.

There are some key takeaways from all this.

- Multiple views of the network i.e., multiple signal sources like loss, delay, bandwidth estimation etc. is not only useful but is *essential* for the congestion control algorithm to be effective
- Need for multi-bit feedback. There are several ways to derive this however I believe that explicit ECN is the best option. There are several advantages including reducing buffering and costs in switches, home routers, backend routers.
- Over the long term, we need to fix the dominant variant of TCP on the Internet so that elephant and mice flows co-exist *simultaneously.*

Even though the research stated above is around TCP, the controllers described above apply equally to any kind of transport framing, the only requirement is that they use IP protocol and allow switches to mark the packets when they sense congestion. While the support for ECN gets added they can instead rely on delay signals and explicit bandwidth estimation. It is interesting in the author's opinion that controllers such as the ones used in DCTCP and CTCP can be modified to rate control algorithms for real-time communication. By exposing the ECN marks from IP packets *directly* to the application layer provides a much tighter loop and therefore feedback to the real-time transport. Overall the use of ECN and a low marking threshold allows for sufficient headroom to prevent burst losses, but more important in this context, keeps the overall latency low for all traffic.

Modifying the above algorithms effectively for real-time traffic needs to incorporate the following.

- Incompatibility of some of TCP's congestion control algorithms on cellular networks which use opportunistic wireless schedulers needs to be addressed.
- Effective cross-layer feedback will go a long way in wireless networks which employ rate adaptation in the presence of interference.
- Differentiating interference loss from a congestion loss can borrow ideas from schemes such as FRTO (RFC 4138).
- Using the information from the IP packets it should be possible to construct and maintain a rate estimate with sufficient granularity to provide both an instantaneous rate and an aggregate rate estimate.

In summary

- This problem cannot be solved in isolation when the dominant versions of TCP on the Internet have a tendency to fill up buffers
- Multi-bit feedback from the network is very essential to get the rate adaptations right. ECN therefore needs to be incorporated into the algorithm. Using ECN marks per packet to build a powerful send side controller is definitely a good starting point.
- Use multiple congestion signals int the algorithm as they complement each other very nicely
- Special attention needs to be paid to cellular networks and interactions with opportunistic wireless schedulers must be addressed.

References

1. Kun Tan, Jingmin Song, Qian Zhang, and Murari Sridharan, A Compound TCP Approach for High-speed and Long Distance Networks, in *INFOCOM 06*, IEEE, 2006
2. Kun Tan, Jingmin Song, and Murari Sridharan, CTCP-TUBE: Improving TCP-Friendliness Over Low-Buffered Network Links, in *PFLDNet 08*, Association for Computing Machinery, Inc., 2008
3. Data Center TCP (DCTCP) , M. Alizadeh, A. Greenberg, D.A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. SIGCOMM 2010.
4. The Future of TCP: Train Wreck or Evolution. http://yuba.stanford.edu/trainwreck/agenda.html
5. Systems and methods for improving TCP-friendliness. United States Patent 7,729,249
6. Receive window auto-tuning. United States Patent 8,150,995

7. Communication Transport Optimized for Datacenter Environment. USPTO Application Number 20110211449, filed Feb. 26, 2010.