Sync-TCP: Using GPS Synchronized Clocks for Early Congestion Detection in TCP

Michele Clark Weigle, Kevin Jeffay, F. Donelson Smith
University of North Carolina at Chapel Hill
Department of Computer Science
Chapel Hill, NC 27599-3175 USA
http://www-dirt.cs.unc.edu

In TCP Reno, the most common implementation of TCP, packet loss is the sole indicator of network congestion. Even so, TCP hosts are not explicitly notified of lost packets, but must rely on timeouts and duplicate acknowledgements (ACKs) to indicate loss. We believe that changes in one-way transit times (OTTs) could also be used to indicate congestion. Queues in routers build up before they overflow, resulting in increased OTTs. If all senders could directly measure changes in OTT and back off when the OTT increases past a threshold, congestion could be alleviated. In this work, we introduce a variant of TCP, called Sync-TCP, which uses the Global Positioning System (GPS) to gather a connection's OTT data. We use Sync-TCP as a platform for investigating techniques for detecting and responding to changes in OTTs.

Given information about the OTT along a connection's path, we conjecture that the performance of TCP's congestion control mechanisms can be improved. By equipping end systems with cards that receive GPS synchronized clock signals, we can accurately (within 1-2 *ms*) determine the OTT between pairs of machines. By observing variation in OTT, end hosts could be aware of increases in queue occupancy at the bottleneck router. This would allow a sender to reduce its sending rate before queues in routers overflow, avoiding losses and costly retransmissions. Monitoring OTTs for indications of congestion also has the effect of allowing senders to quickly realize when periods of congestion have ended, allowing them to increase their sending rate more aggressively than TCP Reno does.

We added a TCP header option to be sent in every packet, both data and ACK, which contains timestamps from synchronized clocks used for computing the OTT in each direction. Separating the OTT for the data and ACK paths is important because these paths are often asymmetric. Our TCP option includes the packet's sending time, the sending time of the last packet received, and the OTT calculated for that packet. Given this information, upon receiving an ACK a sender can calculate three important metrics:

- OTT of the data packet — This is provided in the OTT field of the option.
- OTT of the ACK — This is the receive time minus the time the ACK was sent.
- The time the data packet was received — This is the time the data packet was sent plus the OTT.

Sync-TCP is a congestion control algorithm which uses OTTs and bottleneck bandwidth estimates to estimate the maximum, minimum, and current queue sizes at the bottleneck router. The algorithm operates like TCP Reno, except that it also halves the congestion window when the bottleneck queue is estimated to be half full. We use techniques similar to those in ECN [Flo94] to determine when to reduce the congestion window in response to an early congestion indication (at most once per round-trip time). To estimate queue sizes, we assume that the queue is empty at the point of the lowest observed OTT and that the queue is full at the point of the highest observed OTT. Sync-TCP runs only during congestion avoidance, so we can assume that previously there has been at least one packet loss (*i.e.*, the bottleneck queue was full and then overflowed). The difference between a packet's OTT and the highest observed OTT represents the queuing delay seen by the packet [Pax99]. If we know the bottleneck bandwidth, we can then determine the queuing capacity by dividing the maximum queuing delay by the bottleneck bandwidth.

We performed preliminary experiments to understand the dynamics of this new congestion control algorithm. We found that under Sync-TCP, connections sent far fewer retransmissions than TCP Reno connections, while maintaining a comparable throughput. Current experiments focus on investigating additional methods of estimating the bottleneck queue size using OTTs, estimating a connection's bottleneck and available bandwidth as in [AP99], and adjusting the congestion window in response to early congestion indications.

References

[AP99] Allman, Mark and Vern Paxson. "On Estimating End-to-End Network Path Properties." SIGCOMM 1999.

[Flo94] Floyd, Sally. "TCP and Explicit Congestion Notification." Computer Communications Review, 24(5):10-23, October 1994.

[Pax99] Paxson, Vern. "End-to-End Internet Packet Dynamics." IEEE/ACM Transactions on Networking 7(4), pp. 1-16. (Also in SIGCOMM 1997.)