# Utilizing spare network bandwidth to improve TCP performance

Manpreet Singh Saikat Guha Paul Francis
Cornell University
{manpreet, saikat, francis}@cs.cornell.edu

It is well-known that TCP Reno represents a performance bottleneck as the delay-bandwidth product increases. On high-bandwidth-delay links, the additive increase policy of one packet every RTT necessitates thousands of RTTs to reach full link utilization. This severely affects short TCP flows (most of the flows in the internet) as they cannot acquire bandwidth faster than "slow start" and waste precious RTTs ramping up even when bandwidth is available. XCP [2] addresses this problem by heavily modifying routers to explicitly tell the end-hosts about their fair share of network bandwidth. Fast-TCP [1] backs-off proactively without incurring any loss but cannot quickly ramp-up as the fair share increases.

We propose an approach where a TCP stream can quickly use the full available link bandwidth without requiring modifications to router software. Our approach rapidly fills the bandwidth-delay pipe while being just as cautious as TCP in avoiding causing losses to existing streams. This is accomplished by sending a large number of *low-priority* packets in addition to the conservative number of normal-priority TCP packets. Routers would be configured for strict priority queuing (already supported by routers) where low priority packets are routed only when no normal priority packets are in the queue, and normal priority packets are dropped only when there are no low priority packets in the queue. A diffserv-style packet marking scheme can be used to distinguish between normal-priority and low-priority packets.

Consequently, the low-priority part of the stream immediately utilizes unused network capacity while the more cautiously sent normal-priority packets engage in TCP congestion control. For example, a short 30-packet TCP flow would send 1 normal-priority packet (as per slow-start) and 29 low-priority packets. If enough bandwidth is available, the flow will terminate after only one RTT, instead of 5 RTTs as with regular TCP. For long lived flows, as more normal-priority packets are sent, fewer low-priority packets need to be sent to keep the pipe full. Feedback from the receiver governs the number of low and normal priority packets to be sent. When a normal-priority packet is lost (i.e. the network is truly congested), the normal-priority part scales back exponentially as mandated by TCP. Correspondingly, the low-priority part scales up to fill the potentially larger-than-necessary void created. Low-priority packets never increase congestion since they are preferentially dropped. In the worst case, all low-priority packets are dropped and the approach is identical to regular TCP. The overall performance, therefore, is strictly no-worse (and almost always better) than regular TCP; most strikingly so for short-lived TCP flows in high-bandwidth-delay networks.

While this idea of using "low-priority" packets to opportunistically utilize spare bandwidth is novel in itself, the key technical challenge lies in designing a new congestion control algorithm for the low-priority packets that maximize the overall goodput. Such an algorithm would utilize all the unused capacity, and automatically adapt itself to variations in the available bandwidth. It would have to be robust enough to do so despite high latencies for low-priority packets. Finally, it would enforce fairness among multiple low-priority streams competing for the spare bandwidth. We need to design a more aggressive scheme where efficiency and quick adaptivity (to changes in available bandwidth) take higher priority over fairness. Overall, this approach would speed up most flows in the internet and make full use of the over-provisioning in today's networks without causing more congestion. This opens up an interesting space to explore with non-trivial solutions. So, it might be worth considering various schemes like AIMD, AIAD, MIMD, MIAD, etc with different parameters.

## References

[1] Jin, C., Wei, D. X., and Low, S. H. Fast tcp: motivation, architecture, algorithms, performance. In *Proceedings of The 23rd Conference of the IEEE Communications Society (INFOCOM)* (Hong Kong, Mar. 2004).

[2] Katabi, D., Handley, M., and Rohr, C. E. Congestion control for high bandwidth-delay product networks. In *Proceedings of the Special Interest Group on Data Communications (SIGCOMM)* (Pittsburgh, PA, Aug. 2002), pp. 89–102.