Dynamics and Congestion Control of Alternative TCP Variants on Asymmetric Lines

Peter Orosz, Janos Sztrik, Chesoong Kim

***Abstract*—A growing number of network computers are connected to the Internet through asymmetric connections such as ADSL, CableNet, satellite link and other novel broadband solutions. However, the asymmetric nature of such connections may have significant impact on the performance of TCP transmission as upstream bandwidth can be easily overloaded. In this situation TCP's self-clocking mechanism may be misled whereas TCP ACK packets are delayed on the congested upstream link. It will result in degradation of TCP throughput. The purpose of the present letter is to show the effect of asymmetric physical lines on the dynamics of novel TCP variants based on our empirical measurement results. Therefore, we addressed the evaluation of alternative congestion control mechanisms.**

***Keywords*—TCP, congestion control, ACK, asymmetric bandwidth, traffic dynamics.**

I. INTRODUCTION

ADSL is a data communication technology that provides faster data transmission over copper phone lines than a conventional voice-band modem can provide. The available bandwidth is greater in one direction (downlink) than the other (uplink). There are technical reasons of the implementation of asymmetric bandwidth on long distance connections. More crosstalk is likely to be experienced between the local loops that are connected and aggregated to the DSLAM (Digital Subscriber Line Access Multiplexer) with high density interfaces. Therefore, the uplink signal is the weakest at the noisiest part of the local loop. Another limiting factor is the available frequency range for upstream traffic that is much narrower than that of the downstream which enables lower uplink bandwidth.

We can easily point out how lower uplink bandwidth limits the performance of TCP in some common traffic situation. As we know TCP's congestion control aims to effectively utilize network bandwidth[1]. Congestion control feature of TCP is actually a self clocking mechanism where reception of ACK (acknowledgement) triggers the transmission of the next packet. Spending some time in buffer queues during their transit through the network path ACKs arrivals may be delayed. As a consequence arrival time of ACKs can get closer to each other (ACK burst). This behavior actually misleads self-clocking mechanism on sender side to send more data than the network link can handle. It will cause congestion and loss of efficiency[2].

II. INVESTIGATED TCP VARIANTS

1. TCP BIC

BIC solves the performance problem of TCP in fast long distance networks. Accordingly, the main feature of BIC-TCP is its unique window growth function[3]. TCP severely underutilizes the available bandwidth because of its slow response to available bandwidth. BIC solves this problem by modifying TCP window increase function. It can scale its window increase function to be more aggressive without diminishing the fairness properties of TCP too much. This binary search technique allows the window increase to be logarithmic.

2. TCP Reno

Reno is the implementation of Van Jacobson research and was the default congestion control scheme until recently.

3. TCP Vegas

Vegas algorithm is based on Reno and tries to track the sending rate through looking at variances to the RTT (Round Trip Time) along with other enhancements.

4. HighSpeed TCP

HS-TCP introduces modified TCP response function for systems with higher congestion windows. Because HighSpeed TCP's modified response function would only take effect with higher congestion windows, HighSpeed TCP does not modify TCP behavior in environments with heavy congestion, and therefore does not introduce any new dangers of congestion collapse[4].

III. TEST-BED

- End-point with 512/128 Kbit/sec ADSL connection towards the ISP's DSLAM.
- Server box with 100Mbit/sec full-duplex LAN connection. 1Gbit/sec backbone bandwidth (throughout the network path).
- Characteristics of the measurement data: FTP transmission of large binary files (both directions)
- OS environment: Linux based system – Fedora Core 7, kernel v2.6.22-7
- FTP server: Pure-FTPd v1.0.21-12
- Traffic analysis tools: Wireshark v0.99.5, tcpdump

Linux kernel (version 2.6.13+) supports pluggable TCP congestion control therefore our tests were performed based on this kernel feature. Normally, TCP memory buffers are set by the kernel at boot stage. According to the network link bandwidth and latency, we calculated the BDP value of the applied physical link so to determine optimal buffer sizes. Based upon these values we could tune TCP buffer parameters within the kernel *(tcp_mem, tcp_rmem, tcp_wmem, tcp_app_win, tcp_sack, tcp_wmax, tcp_rmax)* if required.

Result of BDP calculation for local link and full path:

The latency *(D_L - RTT)* of the ADSL link (from modem to DSLAM) was 20ms while the latency of the full path *(D_P)* was ~30ms. Maximum physical bandwidth *(B)* was 512Kbit/sec on downlink and 128Kbit/sec on uplink direction.

$B \times D_L = 64kB \times 0.02sec = 1{,}28kBytes$ (local DSL link)

$B \times D_P = 64kB \times 0.03sec = 1{,}92kBytes$ (full path between the client and the server)

The following general rule calculates the optimal TCP receiver buffer size derived from BDP:

Receiver buffer >= RTT x BW

As default maximum buffer sizes *(>16kB)* are far larger than the calculated BDP value, we left all TCP parameters on their default values. However, on both sides selective ACK (SACK) TCP extension was turned on.

IV. TEST METHOD

Our test had three phases where we generated TCP-based binary FTP traffic between the server and the client. We reached 100% downlink load with one flow: phase 1. In uplink direction only ACKs belonging to that TCP flow passed through. At phase 2 we initiated an independent TCP flow to the opposite direction from the client towards the server machine. Accordingly, uplink load easily reached 100%. At phase 3 another independent TCP flows were started on the client in order to stress the uplink therefore increase effective RTT.

V. MEASUREMENT RESULTS

Nevertheless, TCP throughput is squarely determined by the arrival time of ACK packets. They are transmitted through the uplink with latency due to the congested link. In Phase 2, throughput of downstream TCP flow falls down to near 70% of the available physical bandwidth, depending on the applied congestion control. By initiating another TCP flow from client to server – resulting greater congestion on the uplink – further dramatic degradation was experienced on downstream side: phase 3. Effective throughput now falls down to an average of 40% or less. Under these circumstances physical downlink bandwidth cannot be effectively utilized by TCP. Accordingly, low bandwidth capability of the uplink has a direct effect on downstream TCP performance. As novel TCP variants use specifically designed congestion control algorithms it seems to be a good idea to evaluate them under these circumstances. Even if not all of these algorithms are designed to operate optimally on asymmetric connections. However the measurement results can show unknown effects of these algorithms.

The investigated congestion control mechanisms performed similarly well at the first phase while they showed different results within the last two ones. Considering the asymmetric nature of ADSL, active network users (who generate considerable upstream traffic such as video conferencing, VoIP, sending large Emails, uploading files etc.) may face with TCP performance problem on downlink whilst it is actually not congested.

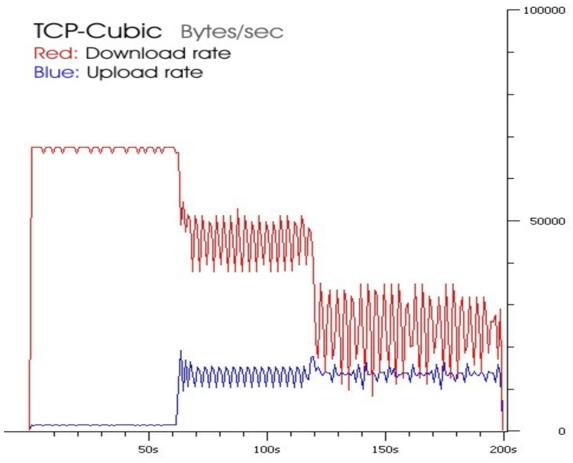

Fig. 1. TCP Cubic throughput

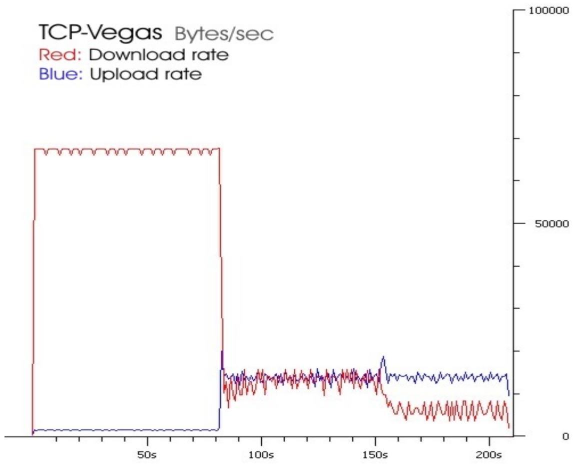

Fig. 2. TCP Vegas throughput

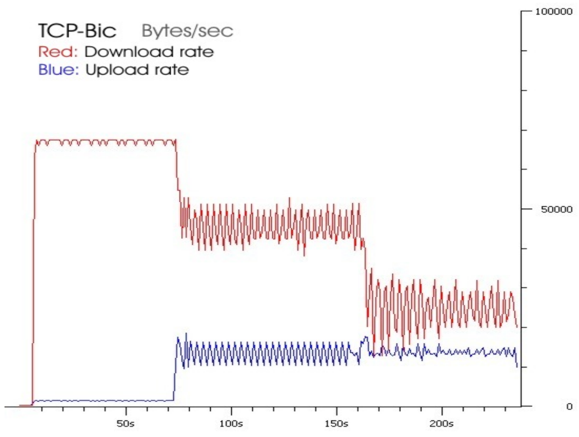

Fig. 2. TCP BIC throughput

For an extreme example, the throughput of TCP variant Vegas – that is actually an earlier TCP implementation – *(Figure 2.)* became far the lowest as long as any large TCP flow is present on the uplink. Its effective throughput fell down dramatically and became equal to or under the uplink throughput rate. Vegas monitors at the variance of RTT to tune the window size. In congested uplinks the ACK packets may have the arrival time fluctuating in a quite large interval that misleads Vegas' algorithm. As we know, conventional TCP Reno algorithm is known by its relatively slow response to the available bandwidth while novel high speed TCP variants have more aggressive window growth algorithms *(Figures 1., 3., 4.)*.

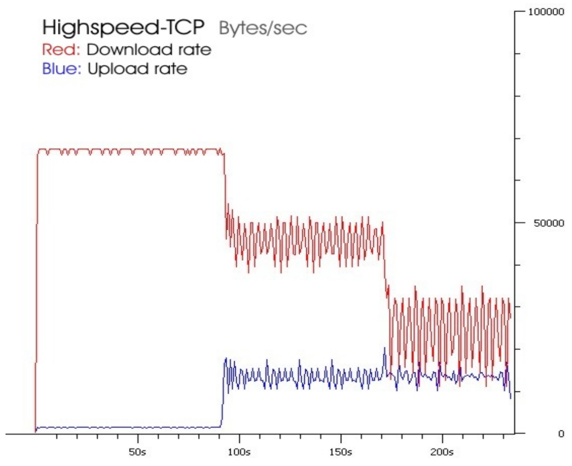

Fig. 4. HSTCP throughput

As soon as uplink becomes congested TCP throughput shows a large fluctuation (phase 3) due to the aggressive window growth function of the high speed TCP variants.
Besides the performance degradation, the originally flat transmission characteristics of downstream TCP flow became very irregular which can result in unreliable performance of real-time networking applications (such as video and voice conversations).

Naturally, uplink congestion may happen even in symmetric connections as well. However, its probability is much smaller. We demonstrated the direct effect of downlink/uplink bandwidth ratio on TCP performance. Accordingly, the higher the bandwidth ratio the higher is the probability of throughput degradation.

VI. Conclusion

Uplink of ADSL connection has a very low bandwidth that can be easily overloaded. Therefore it may seriously degrade downstream TCP performance. TCP self-clocking may be misled into window size degradation by uplink congestion. Moreover ACK compression on the uplink may cause downlink congestion. Under these circumstances, physical downlink bandwidth cannot be effectively utilized by TCP. Accordingly, we demonstrated that uplink load has a direct effect on downstream TCP performance. TCP Vegas' window growth function could not adapt to the asymmetric connection therefore overall downstream TCP performance is dramatically degraded in congested uplink situation. We found that all high speed variants (HTCP, Bic and Cubic) showed significant throughput fluctuation as soon as the uplink became congested due to their more aggressive window growth functions. Size of TCP memory buffers are set by the kernel at the system boot. According to network link bandwidth and latency, BDP should be calculated so to determine optimal TCP buffer sizes. Based upon these values TCP buffer parameters can be tuned for optimal performance. However, by using any novel TCP variant low uplink bandwidth remains a serious limiting factor of asymmetric connections in the investigated traffic situation.

References

[1] Allman, M. – Paxson, V. – Stevens, W.: *TCP congestion control, RFC 2581 (RFC2581),* http://www.faqs.org/rfcs/rfc2581.html
[2] Mogul, Jeffrey C.: *Observing TCP Dynamics in Real Networks*, ACM Press, 1992
[3] TCP Bic/Cubic reference pages: http://www.csc.ncsu.edu/faculty/rhee/export/bitcp/index_files/Page703.htm
[4] TCP Westwood reference pages: http://www.cs.ucla.edu/NRL/hpi/tcpw/