TCP Congestion control in Heterogeneous Network

Md. Ehtesamul Haque* and Md. Humayun Kabir
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology, Dhaka 1000, Bangladesh

August 20, 2007

Abstract

TCP (Transmission control protocol) is the most widely used transport layer protocol in the current networks. Congestion control is one of the main feature of TCP. TCP congestion control technique is conservative in the sense that it assumes any packet loss as a sign of congestion. This approach works good in wired network but fails where wireless links are present in the network as packet losses are mainly caused by bit errors rather than by congestion in wireless links. In this paper, we document the characteristics of wireless links that create problems in TCP congestion control. We also describe a taxonomy of approaches that are used to solve these problems. Moreover we discuss three versions of TCP, TCP Peach, TCP Westwood and TCP Hybla, which performs better in heterogeneous environment.

1 Introduction

Since the introduction of computer network, network was mainly wired where every system is connected to network by some wire. However with the advance in technology wireless networks are common now-a-days and wireless links are everywhere in todays network. Due to the flexibility supported by wireless technology, users are increasing drastically in this area. Although wireless networks are getting popular, wired network is not of less use as bandwidth and reliability is very low in wireless environment.Thus network engineers are faced with challenges to provide an unified support for both wired and wireless. This need introduces the concept of Heterogeneous network. *Heterogeneous network* is a network where the connected computers have different operating systems, protocols or access technologies. In this paper, we focus on heterogeneity due to different access technologies namely wired and wireless links.

Reliability and flow control are very much craved properties by network users over the year. These craving make TCP as a popular transport layer protocol as TCP provides reliable service without overloading the network. Popularity of TCP lead the network engineers to apply TCP in wireless technology as well. But packet losses due to Bit Errors, long Round Trip Time (RTT) and

*#100605034P

variation in RTT confuses TCP as congestion and TCP performance may be far less than optimal in wireless network. Making TCP better in the presence on wireless links in the network is one of the hottest research are over the last few years. In this paper, we present some approaches to solve this problems and three specific variations of TCP which can perform better in heterogenous environment.

In section 2, we describe TCP basics and previous works on TCP. We also discuss the characteristics of wireless links that create problems for TCP. In section 3, Different approaches followed to improve TCP are discussed. Section 4 discusses three recent variations of TCP and Section 5 concludes with future direction and conclusion.

2 Background

2.1 Transmission Control Protocol

Transmission Control Protocol (TCP) is the most popular transport layer protocol. It is a byte-oriented reliable protocol. The simple best effort service provided by IP can be transformed to be a reliable, ordered service by adding TCP on top of IP. TCP is also a conservative protocol in that it always avoids overloading network. Early from the introduction, TCP is equipped with congestion control mechanism. A *congestion window* (cwnd) is maintained which denotes current estimate of the available capacity of the network. At any moment, sender is allowed to send no more than cwnd number of segments without any acknowledgement from the receiver. Initially, cwnd is set to one maximum segment size (MSS) [3]. During slow start phase cwnd is increased exponentially at each ack received from receiver. Slow start ends when a segment loss is detected or cwnd reaches some threshold named *ssthreshold*. Once cwnd reaches ssthreshold, sender enters in *congestion avoidance* state where congestion window is increased linearly. TCP assumes each packet loss as a sign of congestion and sets ssthreshold to half of the cwnd and sender enters again into slow start phase. *Fast retransmit* is a technique used for recovery when packet loss is detected before timeout [5]. Fast recovery was added by TCP Reno to improve recovery further after fast retransmission. New Reno made a slight improvement by modifying normal fast recovery to handle multiple losses in one window [3].

2.2 Wireless Link Characteristics

Wireless links exist in different technology. Different types of wireless link create different problem. Wireless links in general have high latency, high error rate which can affect TCP congestion control mechanism as TCP considers all losses are caused by congestion [4]. Thus high latency, high transmission error and high delay-bandwidth product of wireless links necessitates some extra before applying TCP in wireless successfully.

3 Approaches followed to improve TCP

Improving TCP over wireless links is an active research area since last few years. Many different approaches have been proposed for this purpose. A classification of these schemes identifies two categories: Link-layer solutions and Transport-layer solutions [4]. However more recently many cross layer and layer independent end-to-end solutions are proposed. So, a broad classification is proposed named, Hiding non-congestion losses from sender and sender aware solutions [5]. Here we discuss some of basic and popular approaches followed to improve TCP.

Split TCP is one the most primitive solution where TCP connections are split between wired and wireless parts and congestion control scheme is applied differently. A proxy sits between these two parts and handles transmission in different parts differently. This approach however breaks the end-to-end semantics of TCP as ack received by sender does not mean that data has reached to receiver [5]. *Snoop TCP* [4, 5] is another proposal where an intermediate agent tracks TCP data and acks but not terminate TCP connection. Snoop agent suppresses duplicate acks sent by destination and performs local retransmission on reception of duplicate ack or local timeout. Local timer is set less than that of the sender. Handoff in mobile network causes much of the packet loss in cellular network. A predictive heuristic method is described where TCP can exploit mobility hints and identify losses due to handoff [4]. For these losses methods other than congestion control scheme can be followed.

Decoupling of congestion detection from loss detection also tried in literature [5]. In these schemes, network can recognize congestion and notify sender. Any loss when congestion is not reported, is considered as non-congestion loss. Some other methods try to implement some sender modification to reduce shrinking of congestion window when non-congestion loss happens. After the basic TCP Tahoe is introduced many schemes like TCP Reno, TCP New Reno, TCP Vegas etc are proposed. All these schemes maintain TCP end-to-end semantics and approaches differentiating congestion and non-congestion losses.

4 Variations in TCP

Various effort in improving TCP performance in wireless network resulted in different versions of TCP. Each TCP version has been developed with backward compatibility so that all of them can coexists without any problem. We discuss three of these variations, TCP Peach, TCP Westwood, TCP Hybla.

4.1 TCP Peach

TCP Peach congestion control scheme stresses on avoiding slow start used by previous TCPs and thus increasing initial utilization. The concept used in this approach introduces dummy packets which are probed to determine the network condition [1]. Dummy packets are small and low-priority as a result they do not affect original data delivery. TCP Peach introduces two new algorithm,

sudden start and rapid recovery. It also uses congestion avoidance and fast retransmit as used by TCP Tahoe.

Sudden Start algorithm replaces Slow Start algorithm. Let, $rwnd$ be the receivers advertised window and RTT be the round trip time estimated by sender during TCP setup. The sender transmits 1 data segment and $(rwnd - 1)$ dummy packets within first RTT time. If network has enough bandwidth then within $2 \times RTT$ time all packets are acked and cwnd jumps to $\min\{rwnd, \phi\}$, where ϕ is the maximum achievable rate with current bandwidth [1]. Thus TCP peach avoids initial under-utilization. *Rapid Recovery* is a technique used in TCP Peach to restore congestion window to previous value if any on-congestion loss happens. As like other TCP, TCP Peach assumes all losses due to congestion and halves the congestion window, cwnd. However in Rapid Recovery, after halving cwnd it transmits $2 \times cwnd$ dummy packets within the next RTT along with sending normal data packets. If all dummy packets are acked then cwnd is restored to the value that it had before packet loss.

4.2 TCP Westwood

TCP Westwood follows the principle of TCP Peach to assess the network condition and set the initial congestion window based on this assessment. Unlike TCP Peach, TCP Westwood does not send any dummy packets. Rather it relies on the ack reception rate to estimate the current bandwidth available in network [2]. Here duplicate acks are also counted in bandwidth estimation. It is considered that duplicate ack denotes an average size segment has reached to receiver. A non-duplicate ack conveys the amount of information has reached. Thus exact amount of information reached before non-duplicate ack can be used to estimate bandwidth.

Let, kth ack is received at time t_k which notifies about reception of receiving d_k amount of data. Then we can measure sample bandwidth as $b_k = d_k/\Delta_k$, where $\Delta_k = t_k - t_{k-1}$ and t_{k-1} is the time when (k-1)th ack was received. To avoid abrupt change in the estimation, the estimated bandwidth is calculated as follows

$$\hat{b}_k = \alpha_k \hat{b}_{k-1} + (1 - \alpha_k)\frac{b_k + b_{k-1}}{2}.$$

Here, α_k is a constant that depend on Δ_k to counter the effect of non-deterministic interarrival times [2]. Next, this bandwidth estimation is used to set initial value of ssthreshold. Congestion window is set to ssthreshold after three duplicate acks and 1 after a retransmission timeout.

4.3 TCP Hybla

TCP Hybla relies on the fact that congestion window in TCP relies much on Round Trip Time (RTT) of the network. So, TCP Hybla tries to avoid the dependence of congestion window from RTT. If we denote cwnd at time t by $W(t)$ and t_γ be time at ssthreshold value, γ is reached, then in TCP Tahoe,

$$W(t) = \begin{cases} 2^{(t/RTT)}, & \text{if } 0 \le t < t_\gamma \quad \text{SS} \\ \frac{t - t_\gamma}{RTT} + \gamma, & t \ge t_\gamma \quad \text{CA} \end{cases}$$

where $t_\gamma = RTT \log_2 \gamma$. Also amount of segments transmitted per second, $B(t) = W(t)/RTT$ [3].TCP Hybla tries to compensate this dependency by introducing a normalized round trip time, ρ, defined as, $\rho = RTT/RTT_o$, where RTT_o is the round trip time of the reference connection to which we want to equalize our TCP performance. Thus we have in TCP Hybla,

$$W^H(t) = \begin{cases} \rho 2^{(\rho t/RTT)}, & \text{if } 0 \leq t < t_{\gamma,o} \quad \text{SS} \\ \rho[\rho \frac{t-t_{\gamma,o}}{RTT} + \gamma], & t \geq t_{\gamma,o} \quad \text{CA} \end{cases}$$

where $t_{\gamma,o} = RTT_o \log_2 \gamma$. So, segment transmission rate in Hybla, $B^H(t) = W^H(t)/RTT$, assumes the following expression,

$$B^H(t) = \begin{cases} \frac{2^{t/RTT}}{RTT_o}, & \text{if } 0 \leq t < t_{\gamma,o} \quad \text{SS} \\ \frac{1}{RTT_o}[\frac{t-t_{\gamma,o}}{RTT_o} + \gamma], & t \geq t_{\gamma,o} \quad \text{CA} \end{cases}$$

which is independent of RTT and equal to segment transmission rate of a TCP connection with Round Trip Time, RTT_o [3]. TCP Hybla also enhances performance by Westwood like bandwidth estimation as discussed in 4.2 and New Reno like SACK support.

4.4 Comparative Discussion

A quick overview of all these three TCPs described above can unveil the similarities and dissimilarities among them. All these TCP versions tried to compensate the problems introduced by wireless links in reliable transmission. TCP Peach and TCP Westwood tried to estimate the bandwidth and to avoid overloading network. TCP Hybla, on the other hand, follows TCP Westwood in estimating the bandwidth.

Despite addressing the similarities in the problem approached by these TCPs, they differ in their principles about solving them. TCP Peach tries to estimate the bandwidth by sending dummy packets. Dummy packets are marked in changing the bits in TOS field in IPv4 header or flow label in IPv6 header. This needs slight change in receiver as well as in sender as receiver must not take dummy packets as data and mark the acks as for dummy packets. Dummy packets, although small adds some congestion to the network which is solved by keeping their priorities low. TCP Westwood, on the other hand, tries to estimate the bandwidth of the network by considering the ack reception rate. This approach does not add any packet to the network but it assumes acks has the same amount of bandwidth as data and Round Trip Time variation is not that much. These assumptions are impractical in the context of wireless network. TCP Hybla applies a novel approach making segment transmission rate independent of Round Trip Time. This is an analytical approach and a good way to improve TCP performance. TCP Hybla is however not much worthy proposal as it uses previous many TCPs good characteristics and added only one proposal to improve. Thus TCP Hybal though a good proposal is not that much complete.

5 Conclusion

TCP in heterogenous environment is very critical as it cannot differentiate between congestion loss and non-congestion losses. TCP in general considers all loses for congestion. Here we have discussed about three proposals where effort has been given not to decrease congestion window upon non-congestion loss. A correct approach to differentiate between losses is still desirable. We also discussed the characteristics of wireless links that are problematic not only for TCP but also in other aspect of networking. Work on to hide or avoid these problems may be a god research area. Moreover, TCP performance due link asymmetry is still wide open and is an challenge to the next generation researchers.

References

[1] Ian F. Akyildiz, Giacomo Morabito and Sergio Palazzo, *TCP-Peach: A New Congestion Control Scheme for Satellite IP Networks*, IEEE/ACM Transactions on Networking, vol. 9, no. 3, pp. 307-321, 2001.

[2] Claudio Casetti, Mario Gerla, Saverio Mascolo, M. Y. Sanadidi and Ren Wang, *TCP Westwood: End-to-End COngestion COntrol for Wired/Wireless Networks*, Wireless Networks Journal, vol. 8, pp. 467-479, 2002.

[3] Carlo Caini and Rosario Firrincieli, *TCP Hybla: A TCP Enhancement for Heterogeneous networks*, International Journal of Sattelite Communications and Networking, vol. 22, pp. 547-566, 2004.

[4] George Xylomenos,George C. Polyzos, Petri Mähönen and Mika Saaranen, *TCP Performance Issues over Wireless Links*, IEEE Communications Magazine, vol. 39, no. 4, pp. 52-58, 2001.

[5] Chadi Barakat, Eitan ALtman and Walid Dabbous, *On TCP Performance in a Heterogeneous Network: A Survey*,