International Journal of Soft Computing 3 (2): 93-98, 2008
ISSN: 1816-9503

Multi Agent System for Improving TCP/IP Performance over Wireless Networks

[1]B. Sasikumar and [2] V. Vasudevan
[1]Department of CSE, R.M.D Engineering College, Chennai, India
[2]Department of Information Technology, Arulmigu Kalasalingam College of Engineering, India

Abstract: As 3rd Generation (3G) wireless networks withhigh data rate get widely deployed, optimizing TCP performance over these networks would have a broad and significant impact on data application performance. One of the biggest challenges in optimizing TCP performance over the 3G wireless networks is adapting to the significant delay and rate variations over the wireless channel. In this study, we make two main contributions. First, we present a Window Regulator algorithm that uses the receiver window field in the acknowledgment packets to convey the instantaneous wireless channel conditions to the TCP source and an ack buffer to absorb the channel variations, thereby maximizing long-lived TCP performance. It improves the performance of TCP Sack by up to 100% over a simple drop-tail algorithm for small buffer sizes at the congested router. Second, we present a wireless channel and TCP-aware scheduling and buffer sharing algorithm that reduces the latency of short TCP flows by up to 90% while still exploiting user diversity, thus allowing the wireless channel to be utilized efficiently. TCP protocols implemented in multiagent system.

Key words: Agents, multiagent, radionetwork controller, wirelessnetwork, TCP/IP stack, window regulator

INTRODUCTION

Agent functionality: The history agent and the manuals agent are both, to a large extent, information agents. As such, they can receive requests for information and reply with the required information. Both of them receive the same type of requests-a single shot query. The content of the query includes one or more part-numbers and one or more fault descriptors for each fault in the request. The history agent parses each historical form to which it has access, usually limited to local archives, searching for complete or partial matches (We leave some flexibility to allow a search using ontology to increase hits on less obvious cases). The forms, which match the designated problem, are inserted into the content of the reply message. In contrast, the manuals agent does not need to perform parsing. It only has to search, using indexes, through the manuals. Matches are inserted into the content of the reply message Multi-agent organization. The description of information flow and processing provides only a partial view of the system developed. At least as important is the way in which the agents are organized to provide the required processing.

Third Generation (3G) wide-area wireless networks, based on the CDMA technology (TIA/EIA/cdma, 2000), are increasingly being deployed throughout the world. While voice and, to some extent, short messaging service have been the predominant applications on the low bandwidth wireless networks to date, the support for high speed data in 3G networks with bandwidths up to 2.4 Mbps should enable the widespread growth of wireless data applications. Since, the vast majority of data applications use the TCP/IP protocols, optimizing TCP performance over these networks would have a broad and significant impact on the user-perceived data application performance. TCP performance over wireless networks have been studied over the last several years. Early research, Bakre and Badrinath (1995) and Balakrishnan *et al.* (1995), showed that wireless link losses have dramatic adverse impact on TCP performance due to the difficulty in distinguishing congestion losses from wireless link losses. These results have been one of the main motivations behind the use of extensive local retransmission mechanisms in 3G wireless networks.

While these local retransmission mechanisms solve the impact of wireless link losses on TCP performance, they also result in unavoidable variations in packet transmission delay (due to local retransmissions) as observed by TCP.

In addition to these delay variations, 3G wireless links use channel-state based scheduling mechanisms (Bhagwat *et al.*, 1996), that result in significant rate variations. The basic idea behind channel state scheduling is to exploit user diversity. As wireless channel quality of different users vary due to fading, the total cell throughput can be optimized if scheduling

Corresponding Author: B. Sasikuma, Department of CSE, R.M.D Engineering College, Chennai, India

priority is given to the user with higher channel quality. For example, Qualcomm's proportional fair (Bender *et al*., 2000) exploits this idea while providing for long-term fairness among different users. Thus, while these scheduling mechanisms maximize overall linklayer throughput, they also can result in significant variations in instantaneous individual user throughput as observed by TCP. Only recently, researchers have begun investigating the impact of these wireless link rate and delay variations on TCP, (Chan and Ramjee, 2002; Ludwig and Katz, 2000). These variations cause TCP performance degradation due to the difficulty in estimation of the appropriate throughput (i.e., congestion window size and round trip time) of the endto- end path at the TCP source. When the source over-estimates the available through-put, it causes multiple and frequent packet drops at the congested router buffer, resulting in poor TCP throughput. In, we (Chan and Ramjee, 2002) propose a network-based solution called the Ack Regulator to address this problem. Ack Regulator determines the available buffer space at the congested router and the expected number of data packets arriving at the router and manages the release of acks to the TCP source so as to prevent an undesired overflow of the buffer.

While Ack Regulator was shown to increase the throughput of long-lived TCP flows, it has a few drawbacks due to the limitation of not being able to modify headers of TCP packets. First, Ack Regulator needs to estimate the number of data packets in transit from the source - errors in this estimation, for example due to variations on the wired network, could lead to multiple-packet drops resulting in lowered throughput.

Second, since it intentionally causes single packet drops to force TCP source to go into congestion avoidance phase, it inherently cannot achieve maximum goodput. Finally, it also does not address the performance of short-lived flows such as HTTP. In this study, we make two important contributions. First, we design a network-based solution called the Window Regulator that maximizes TCP performance for any given buffer size at the congested router. Second, we present a scheduling and buffer sharing algorithm that reduces the latency for short flows while exploiting user diversity, thus allowing the wireless channel to be utilized efficiently. The proposed Window Regulator algorithm uses the receiver window field in the acknowledgment packets to convey the instantaneous wireless channel conditions to the TCP source and an ack buffer to absorb the channel variations, thereby maximizing long-lived TCP performance. Window Regulator ensures that the source TCP operates in the window-limited region resulting in a congestion loss-free operation. While the receiver window field of the ack packets have been used for ensuring fairness and regulating flows in wired network (Karandikar *et al*., 2000), we show that these schemes do not perform as well over wireless links with variation. We show that the Window Regulator results in highest goodput and maximum TCP performance for even small values of the buffer size, reasonably large wiredlatencies and small amount of packet losses. For example, it improves the performance of TCP Sack by up to 100% over a simple drop-tail policy for small buffer sizes at the congestedrouter. The use of a small buffer for long flows is important when we consider the impact of having both short and long flows sharing the buffer since the buffer needs to have spaceto be able to absorb the burst of packets from the short flows. While minimizing short flow latency is important for the user perceived performance of applications like HTTP, any short flow differentiation scheme has to take into account the wireless channel condition in order to take advantage of user diversity. We show that a scheduling algorithm that provides differentiation but does not fully exploit user diversity canhave the adverse effect of increasing short flow latencies and decreasing long flow throughput at the same time. We present a wireless channel and TCP-aware buffer sharing and scheduling algorithm that decreases the latency of short TCP flows by up to 90% while still exploiting user diversity, thus allowing the ireless channel to be utilized efficiently.

Receive windows in the next generation TCP/IP stack: To solve the problem of correctly determining the value of the maximum receive window size for a connection based on the current conditions of the network, the Next Generation TCP/IP stack supports Receive Window Auto-Tuning. Receive Window Auto-Tuning continually determines the optimal receive window size by measuring the bandwidth-delay product and the application retrieve rate and adjusts the maximum receive window size based on changing network conditions (Bender *et al*., 2000; TIA/EIA/cdma, 2000). Receive Window Auto-Tuning enables TCP window scaling by default, allowing up to a 16 MB window size. As the data flows over the connection, the Next Generation TCP/IP stack monitors the connection, measures the current bandwidth-delay product for the connection and the application receive rate and adjusts the receive window size to optimize throughput. The Next Generation TCP/IP stack no longer uses the TCPWindowSize registry values.

With better throughput between TCP peers, the utilization of network bandwidth increases during data transfer. If all the applications are optimized to receive TCP data, then the overall utilization of the network can increase substantially, making the use of Quality of Service (QoS) more important on networks that are operating at or near capacity.

The vast majority of related work on TCP performance over wireless networks have concentrated on reducing the impact of TCP mis-reacting to wireless losses as congestion losses (Bakre and Badrinath, 2005) that result in poor throughput. As mentioned earlier, link layer retransmission in 3G wireless links have effectively reduced the loss rate of wireless links to well under 1%, thereby minimizing the impact of loss on TCP performance. Recently, there have been several studies that examine the impact of wireless link variations on TCP performance. Chan and Ramjee (2002) Large delay variations resulting in delay spikes can cause spurious timeouts in TCP where the TCP source incorrectly assumes that a packet is lost while the packet only delayed; this unnecessarily forces TCP into slow start, adversely impacting TCP performance (Ludwig and Kat, 2000). In, the authors propose enhancements to the TCP timer calculations to better track the round trip time of the connection, thereby avoiding spurious timeouts. The authors in Inamura *et al.* (2002) present several recommendations for TCP hosts such as enabling the timestamp option and the use of large windows, for improving performance over wireless networks. As discussed in the introduction, in Chan and Ramjee (2002), the authors present the Ack Regulator solution for avoiding multiple packet drops at the congested router for long-lived flows. The use of receiver-window field in the ack packets to throttle TCP source is not new. In Karandikar *et al.* (2000) the authors implement a receiver window-based technique with ack pacing at the bottleneck router to reduce burstiness and ensure fairness among different flows. In Kalampoukas *et al.* (2002), the authors use the receiver window field to better manage TCP throughput over a connection that spans both IP and ATM networks. However, since these solutions do not explicitly cater to significant rate and delay variations, they do not perform as well over wireless networks.

Differentiation for short flows over long flows in wired networks have been studied. The basic idea is to identify short flows heuristically through the use of a simple threshold for bytes transmitted and another threshold for idle period and then give priority to short flows. The authors in Chen and Heidemann (2001) use RED with different weights for short and long flows to provide differentiation. However, tuning RED for wireless links that exhibit significant variation will be hard. Furthermore, wireless networks already employ peruser buffering in order to implement reliable link layers with local retransmissions; thus utility of an algorithm like RED is reduced since we already have per-user state information available. Differentiation for short flows in wireless networks have also been studied. In Shao and Madhow (2002) Foreground-Background (FB) scheduling is used to schedule flows within a user and Proportional Fair (PF) scheduling is used to schedule packets across users. No new algorithm is proposed for inter-user scheduling in place of PF. In the goal is to minimize the average stretch (ratio of actual job completion time over minimum job completion time) over all jobs. One drawback is that it requires advance knowledge of all job sizes which may not be available.

ARCHITECTURE

A simplified view of the 3rd generation wireless access network architecture is shown in Fig. 1. The base stations are managed by a Radio Network Controller (RNC). The RNC performs handoffs and terminates the Radio Link Protocol (RLP), that is responsible for improving the reliability of the wireless link through link-layer retransmissions. The Packet Data Service Node (PDSN) terminates PPP, performs the function of a Mobile IP Foreign Agent and interfaces to the public Internet. In this architecture, the RNC receives IP packets encapsulated in PPP from the PDSN. These IP packets are fragmented into smaller radio frames using the RLP protocol and transmitted to the base station. The base station then schedules the transmission of the packet over the air. In the case of a wireless frame loss, the RLP protocol performs retransmission of the radio frames. In this architecture, the RNC maintains a per-user packet buffer and drop packets during congestion when the per-user buffer is full. For the algorithms discussed in this study, we assume that the RNC can be extended to distinguish between different TCP flows based on the IP addresses and port numbers inside the packet headers. However, we still use the same per-user buffer limit and ensure that all the flows belonging to a single user share the per-user buffer judiciously. The RNC must carefully choose how much buffer is to be allocated to a single user. Placing a strict upper limit on the maximum buffer allocated to a single user is necessary because of several

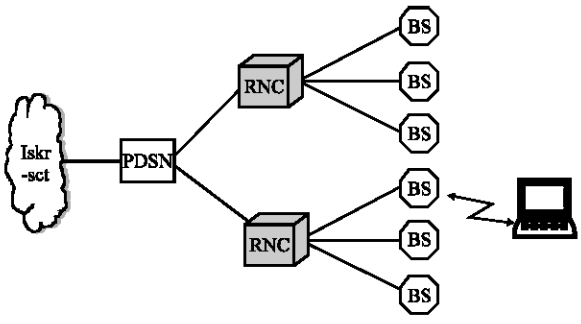

Fig. 1: Wireless network architecture

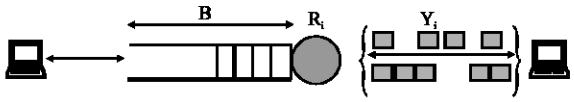

Fig. 2: Simple model of the wireless network

reasons: During handoffs, the per-user buffers have to be either quickly moved from one RNC/base station to another or flushed; large buffers can result in long handoff latencies and/or wasted bandwidth on the access network, Scalability and cost considerations also place a limit on the buffer size as the RNC must scale to the order of hundred thousand users or more, Stale data (for example, user clicking "reload" on a browser or terminating a FTP flow) will still be sent over the wireless link; large buffers imply larger amount of stale data, wasting limited wireless bandwidth.

Window regulator: Consider a simple model of a single flow in a wireless network shown in Fig. 2. The base stations and the RNC are collapsed into a single bottleneck link with the per-flow buffer size at the bottleneck node set to B. For simplicity of analysis, let us assume that there are no losses or re-ordering on the wired or the wireless links and let the latency on the wired network be insignificant compared to wireless delays. We will relax these assumptions later when we evaluate the performance of Window Regulator, in order to maximize TCP throughput, we need to ensure that there is always at least one packet available in the bottleneck node to be transmitted (no underflow) and there is no packet loss due to buffer overflow (this leads to retransmission, which is wasted bandwidth and reduction in congestion window size or timeout that could then lead to underflow).

In order to understand the performance of the Window Regulator, we focus on the arrival events on the data queue. Consider the arrival of the ith packet. Let Yi represent the number of packets in the "wireless pipe" when packet I arrives. Yi is a function of the varying rates and delays on the forward and reverse directions on the wireless link. Finally, let Ni($\leq$ B) be the number of packets in the bottleneck link when packet i arrives. Since we assume that wired network latency is insignificant, all data packets are buffered in the bottleneck link and all acknowledgments are acknowledged immediately with no outstanding packets in the wired network. The sum of data packets in the buffer and in the "wireless pipe" equals the TCP window size,

$$Wi.\ Wi = Ni + Yi + 1 \quad (1)$$

Equation 1 is always true since wired network latency is assumed to be insignificant. Here we study the impact of relaxing this assumption on the throughput of the different algorithms. Now, for TCP to operate without buffer underflow, the window size, Wi, must obey

$$\forall I\ Yi + 1 < Wi \text{ (no underflow)} \quad (2)$$

Equation 2 states that the window size must be greater than the instantaneous number of packets in the wireless pipe and there must be at least one packet in the buffer for there to be no underflow. In general, this is a necessary but not sufficient condition. However, since we assume that Eq. 1 is true, Ni$\geq$1 when there is no underflow and thus Eq. 2 is also a sufficient condition for no underflow. For there to be no overflow (packet drop), from Equation 1, B$\geq$Wi-Yi. In other words,

$$\forall I\ Yi + B \geq Wi \text{ (no overflow)} \quad (3)$$

Similarly, Eq. 3 is necessary and sufficient to prevent overflow. The difficulty in choosing an appropriate TCP window size is in adapting to the variations in Yi while ensuring that Eq. 2 and 3 are not violated. Next, we consider three Window Regulator algorithms that set the receiver window size, Wr, in the ack packets in order to manage the window size at the TCP source. For the purpose of this discussion, we assume that each packet is acked. However, the algorithms can be generalized to handle acks that represent more than one packet as well.

Window Regulator-Static (WRS): The first algorithm, called the Window Regulator-Static (WRS), is a common algorithm used in wired routers for guaranteeing bandwidth and ensuring fairness (for example, Packeteer) (Karandikar *et al*., 2000). The receiver window size, Wr, is set statically On Enque of each Ack set Wr = B transmit the Ack to the source Fig. 3. Window Regulator-Static to the buffer allocated for that flow. That is, Note that, this algorithm is very conservative as it easily satisfies Eq. 3 (no overflow) since Wi = B and $\forall$iYi = 0 as the number of packets in the wireless pipe cannot be negative. However, depending on the size of the buffer in relation to the variation of the wireless pipe, the queue can be idle, resulting in loss of throughput. In other words, the utilization of the queue, QWRS, of this algorithm is approximated k is the total number of packets arrived and 1{B > Yi + 1} is the delta function:

$$1\{B > Yi + 1\} = \_\ 1,\ B > Yi + 1$$
$$0,\ \text{otherwise}$$

The approximation is exact if the arrival process is Poisson (PASTA). For the case of bursty arrivals, which is the expected case here, Eq. 4 is an upper bound.

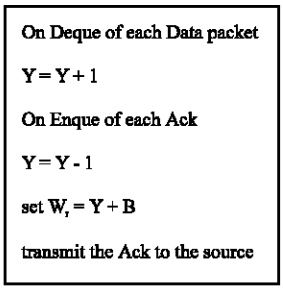

Fig. 3: Window regulator-dynamic

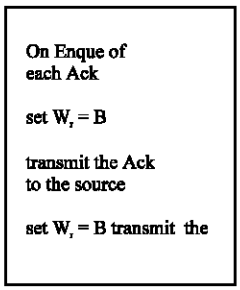

Fig. 4: Window regulator-static

Window Regulator-Dynamic (WRD): One simple way to extend the WRS algorithm is to track the changes in Yi and convey this in each ack packet flowing back to the sender. We call this the Window Regulator-Dynamic (WRD) algorithm and it operates as shown in Fig. 4, where Y is the current estimate of the size of the wireless pipe. On Deque of each Data packet Y = Y + 1.

On Enque of each Ack Y = Y - 1 set Wr = Y+B transmit the Ack to the source Fig. 4. Window Regulator-Dynamic Assuming that wired latency is insignificant, if a data packet i arrives due to this ack departure, then Wi = Wr and Yi = Y. Note that this algorithm might end up reducing the receiver window size Wi compare to Wi-1 as Y is a varying quantity.

However, we never reduce the receiver window size between consecutive acks by more than one packet (reception of an ack reduces the packets in the wireless pipe by one since we assume every packet is acknowledged). Thus, transmitting an ack with a reduced window size does not shrink the window but just freezes the window from advancing (no new packets will be transmitted in response to an ack with reduced window). In the implementation, if packets are not individually acknowledged, Y can be easily measured as the difference between the sequence number of the packet being transmitted and the sequence number of the ack being receivedl.

Care has to be taken to handle retransmissions and duplicates. This algorithm is also conservative as it satisfies Eq. 3 (no overflow) but it uses a higher window size compared to WRS. Since both these algorithms operate in the window limited region of TCP (where throughput is given by W/RTT, where W is the window size and RTT is the round trip time), the throughput of WRD is as good or better than the throughput of WRS since If W

RTT ≤ R then
W
RTT ≤
W+k
RTT+k/R∀k ≥ 0

Where, R is the average rate of the connection. Window size of WRS is B and window size of WRD is B+Y. Y is the difference in window size between WRD and WRS and is always non-negative (Ludwig *et al*., 1999). However, this algorithm can also result in underflow when the whole buffer is drained before the reception of an ack (causing a sudden large increase in the number of packets in the wireless pipe.

CONCLUSION

In this study, we make two important contributions. First, we proposed a network-based solution called the Window Regulator that maximizes TCP performance in the presence of channel variations for any given buffer size at the congested router. We analyzed the performance of various versions of the Window Regulator schemes and make the following observations. Window Regulator-Static (WRS), a common algorithm used in wired routers, performs poorly. The Window Regulator with ack Buffer (WRB) scheme, which explicitly adapts to the wireless channel conditions and also performs ack regulation, improves the throughput by up to 100% over a drop-tail scheme. WRB also delivers robust performance gains even with reasonably large wired latencies and small amount of packet losses. Next, we presented a scheduling and buffer sharing algorithm that reduces the latency for short flows while exploiting user diversity, thus allowing the wireless channel to be utilized efficiently. We found that the proposed PF-RP/SFP scheme provides robust performance over different user channel conditions and improves latency up to 54% over PF/FIFO The open question remains as to what is the best way to balance all three parameters: diversity, fairness and preference for short flows. Answering this question would require a better understanding of the trade-off and comparison metric. We are exploring this issue as part of future work.

REFERENCES

Bakre, A. and B.R. Badrinath, 1995. Handoff and System Support for Indirect TCP/IP. In proceedings of Second Usenix Symposium on Mobile and Location-Independent Computing.

Balakrishnan, H., S. Seshan, E. Amir and R.H. Katz, 1995. Improving TCP/IP Performance over Wireless Networks. In Proceedings of ACM Mobicom.

Bender, P. *et al*., 2000. A Bandwidth Efficient High Speed Wireless Data Service for Nomadic Users. In IEEE Communications Magazine.

Bhagwat, P., P. Bhattacharya, A. Krishna and S. Tripathi, 1996. Enhancing Throughput over Wireless LANs Using Channel State Dependent Packet Scheduling. In Proc. IEEE. INFOCOM, pp: 1133-40.

Chan, M.C. and R. Ramjee, 2002. TCP/IP Performance over 3G wireless links with rate and delay variation. In Proceedings of ACM Mobicom.

Chen, X. and J. Heidemann, 2001. Preferential Treatment for Short Flows to Reduce Web Latency. USC/ISI Technical Report ISI-TR-548 to appear in Computer Networks.

Guo, L. and I. Matta, 2001. The war between mice and elephants. In Proceedings of ICNP.

Inamura, H. *et al*., 2002. TCP over 2.5G and 3G Wireless Networks. Draftietf-pilc-2.5g3g-07.

Joshi, N.S., S.R. Kadaba, S. Patel and G.S. Sundaram, 2000. Downlink Scheduling in CDMA Data Networks. In Proceedings of Mobicom.

Kalampoukas, L., A. Varma and K.K. Ramakrishnan, 2002. Explicit window adaptation: a meth d to enhance TCP performance. IEEE/ACM Transactions on Networking.

Karandikar, S., S. Kalyanaraman, P. Bagal, B. Packer, 2000. TCP rate control. ACM Computer Communication Review.

Khafizov, F. and M. Yavuz, 2000. TCP over CDMA2000 networks, Internet Draft, draft-khafizov-pilc-cdma2000-00.txt.

Ludwig, R. and R. H. Katz, 2000. The Eifel Algorithm: Making TCP Robust Against Spurious Retransmissions. In: ACM Computer Communications Review, Vol. 30, No. 1.

Ludwig, R., B. Rathonyi, A. Konrad, K. Oden and A. Joseph, 1999. Multilayer Tracing of TCP over a Reliable Wireless Link. In Proceedings of ACM SIGMETRICS.

Shao, Z. and U. Madhow, 2002. Scheduling Heavy-tailed Data Traffic over the Wireless Internet. In Proceedings of VTC.

Third Generation Partnership Project, 1999. RLC Protocol Specification (3G TS 25.322:).

TIA/EIA/cdma, 2000. Mobile Station-Base Station Compatibility Standard for Dual-Mode Wideband Spread Spectrum Cellular Systems. Washington: TIA.

TIA/EIA/IS-707-A-2.10, 2000. Data Service Options for Spread Spectrum Systems: Radio Link Protocol Type 3.