A Cross-Layer Packet Loss Identification Scheme to Improve TCP Veno Performance

Sachin Shetty sshetty@tnstate.edu
Department of Electrical and Computer Engineering
Tennessee State University
3500 John A. Merritt Blvd
Nashville, TN 37209

Ying Tang tang@rowan.edu
Department of Electrical and Computer Engineering
Rowan University
201 Mullica Hill Rd
Glassboro, NJ 08028

William Collani wcollani@gmail.com
Department of Electrical and Computer Engineering
Rowan University
201 Mullica Hill Rd
Glassboro, NJ 08028

Abstract

In wired-cum-wireless networks, one of the main design challenges for TCP is to accurately distinguish congestion losses from random losses caused due to channel noise and interference. TCP Veno has been widely deployed in wireless networks to address this challenge. TCP Veno has been demonstrated to show better performance than TCP Reno in wired-cum-wireless environments. However, TCP Veno does not take into consideration the effects of channel noise and interference on packet losses, which impacts the accuracy of TCP Veno's packet loss identification and thereby causes underutilization of bandwidth in lightly-loaded networks. In this paper, we propose TCP Venoplus which contains two refinements to improve the performance of TCP Veno in lightly-loaded networks. The first refinement incorporates a new variable, congestion loss window, into TCP Veno. This new variable will aid TCP Veno in proper utilization of bandwidth in presence of lightly-loaded traffic. The second refinement involves procuring power level of received packet from the MAC layer to aid in better detection of random losses. The simulation results demonstrate that, the two refinements in TCP Venoplus can significantly improve Veno's throughput with accurate packet loss identification and without compromising fairness.

Keywords – Transmission Control Protocol (TCP), System-on-Chip (SoC), congestion control, packet loss differentiation

1. INTRODUCTION

In recent times, communication networks have evolved greatly. Packet switching technologies have successfully merged the traditional voice networks and data networks into a converged and integrated multimedia network. This converged integrated network has been extended to incorporate varied flavors of wireless technologies. Transmission control protocol (TCP) has become the dominant communication protocol suite in almost all the Internet application for reliable data transfer services [1]. The great success of TCP is due to its end-to-end congestion control scheme, which is designed for fairness and friendliness with respect to coexistent TCP flows.

With the proliferation of mobile computing in recent years, more and more devices are communicating through wireless links. These links are often characterized by high random bit error rates (BER) due to channel fading or noise, and intermittent connectivity due to handoffs. Therefore, network protocols, originally designed for wired networks, have to be adapted for such lossy environment. Nowadays, about 90% of the Internet traffics are carried by TCP. TCP needs to depart from its original wired network oriented design and evolve to meet the challenges introduced by the wireless portion of the network. Transmission Control Protocol (TCP) has become the dominant communication suite in today's networking applications. TCP was designed ideally for wired network with low BER, where congestion is the primary source of packet losses. Although TCP was initially designed and optimized for wired networks, the growing popularity of wireless data applications has lead wireless networks such as WLAN to extend TCP to wireless communications as well. The initial objective of TCP was to efficiently use the available bandwidth in the network and to avoid overloading the network (and the resulting packet losses) by appropriately throttling the senders' transmission rates. Network congestion is deemed to be the underlying reason for packet losses [3]. Consequently, TCP performance is often unsatisfactory when used in wireless networks and requires various improvement techniques [2], [8], [9], [10], [12]. When applied to the wireless environment in which packet losses are often induced by noise, link errors, and reasons other than network congestion (random errors), TCP congestion control and avoidance mechanism [6] becomes incapable of dealing with the mixed packet losses. In fact, when a random loss occurs, TCP actually backs down the sending rate to reduce, what it perceives as, congestion in the network [4], resulting in significant performance degradation. There are three key factors responsible for the unsatisfactory performance of TCP in wireless networks. The first factor is that the radio link quality in wireless networks can fluctuate greatly in time due to channel fading and user mobility, leading to a high packet loss which is non-congestion loss[4][5][6]. The second factor is variability of transmission time and delay [7]. Finally, the third factor is high delay variability caused by radio link quality in wireless networks. A form of high delay variability, referred to as delay spike, is a sudden, drastic increase in delay for a particular packet or a few consecutive packets, relative to the delay for the preceding and following packets.

TCP's congestion control scheme is based on the fundamental premise that packet loss is an indicator for network congestion. Therefore, TCP senders react to packet loss by reducing its sending throughput. However, this premise is inaccurate as increasing number of wireless links are integrated into the Internet. Packet losses in wireless links are overwhelmingly caused due to bit errors, instead of buffer overflow at routers [2], [14], [15], [16]. This cause of packet loss due to bit errors has been largely ignored by TCP without any pertinent refinement. The result of such failure is that TCP will "blindly" scale back its sending rate upon packet losses with identical penalty, regardless of the reason of loss occurrence. So, TCP will suffer unnecessary performance degradation.

In recent years, different schemes have been proposed to aid TCP in distinguishing congestion losses from bit errors. We present the details of these schemes in the next Section. To provide an end-to-end congestion control a sender-side variation of TCP Reno, TCP Veno [2] was developed. Veno purely uses the information available at the sender to make decisions regarding the types of losses. The detection of packet losses is obtained from estimated congestion

information. A threshold value β is used to make the decision. If the estimated congestion level in the network exceeds the value of β, the packet loss is detected as a congestion loss; else it is classified as a loss due to random errors. In case of a random loss, the congestion window is reduced by a factor of 1/5, instead of halving the congestion window [2]. Therefore, if the estimated congestion information is accurate, the packet losses will be accurately detected at all times.

But TCP Veno has been demonstrated to not perform effectively in certain network scenarios. In [13], it has been shown that due to some uncertainties in network the actual network state could be associated with an irrelevant packet loss; or a packet loss is linked with the network state that is mistakenly estimated. These factors imply that the loss due to transient congestion is also a factor which can influence the prediction of packet losses. In [14], the authors claimed that the transient congestion period occurs so fast that a corresponding packet loss looks just like an episode of wireless loss. In presence of bursty traffic, it has been shown that Veno's packet loss identification scheme is not very accurate [15]. More recently, it has been demonstrated that Veno suffers from server bandwidth under utilization when operating under lightly loaded wireless networks [16].

In recent times, with the rapid progress of deep submicron technology, System-on-Chip (SoC) has revolutionized the design of Integrated Circuits, allowing all components of a computer or other electronic system into a single integrated circuit (i.e., chip). One of the applications in networking is offloading TCP/IP processing into an embedded device called TCP Offload Engine (TOE). TOE provides an emerging solution to release communication systems from burdened conventional TCP/IP stack, and significantly improves the network utilization [11]. Modern servers with multi-processor and multi-core architectures are now equipped with TOE to boost throughput and reduce CPU overload. The advent of SoC technology further opens a new venue for cross-layer design in wireless networking since the traditional layered architecture served well for wired networks is not suitable for wireless networks. More specifically, the integration of all layers of networking into a single chip presents the following two advantages: (1) the impact of breaking end-to-end network semantics is negligible; and (2) the implementation of cross-layer communication becomes easier.

Motivated by these observations, this paper addresses the aforementioned two problems together to improve TCP performance in both lightly and heavily loaded wireless networks. In this paper, we propose TCP Venoplus which incorporates two refinements in the TCP Veno scheme. The two refinements improve the congestion state measurement in Veno. By taking advantages of SoC technologies, Venoplus uses received signal strength information (RSSI) to compute BER and duplicate acknowledgements and timeouts to estimate congestion loss. The refinements in Venoplus are on the same line as the enhancedVeno scheme proposed in [16]. Enhanced Veno describes a scheme to distinguish congestion losses from random losses for lightly-loaded wireless networks. But enhancedVeno assumes that the nature of random errors in the link is known *a priori*. This is not practical in most wireless communication links due to the randomness associated with noise and interference sources. Venoplus differs from enhancedVeno by incorporating cross-layer functionality to compute random losses. The cross-layer property allows the TCP layer to learn RSSI for every received packet from the MAC layer.

Two new variables, congestion loss window and random loss rate are added to TCP Veno. The congestion loss window variable aids TCP Veno in improving bandwidth utilization without compromising fairness. The random loss rate variable incorporates the information procured from BER to signify the number of packet losses due to random errors. The computation of the random loss rate involves the implementation of cross-layer functionality. As stated earlier, the implementation complexity in terms of cross-layer communication becomes negligible with recent SoC advancements. Extensive simulations demonstrate that the throughput and accurate detection of congestion and random losses are much improved than TCP Veno. We also observed that Venoplus yields better fairness in presence of competing TCP connections.

The remainder of this paper is organized as follows: Section 2 presents the related work. In Section 3 we analyze the issue of bandwidth underutilization in TCP Veno and present the details for TCP Venoplus. The simulation and performance evaluation is described in Section 4. In Section 5, we present conclusions and future work.

2. RELATED WORK

In recent years, different schemes have been proposed to aid TCP in distinguishing congestion losses from random errors. More specifically, the TCP schemes address the following two problems (1) how to distinguish a packet loss due to congestion and the corruption of a packet due to random losses; and (2) how to refine the congestion-window adjustment parameters according to network conditions. Many research efforts have been directed towards the modification of TCP protocol to address these two problems. All these research efforts fall under two main categories. The two categories are: Wireless TCP Refinements and Routers with RED and ECN.

2.1 Wireless TCP Refinements

Existing TCP schemes suffer from severe performance degradation in hybrid wired-cum-wireless networks [18], [19]. There have been numerous research efforts to alleviate this problem. These research efforts can be categorized into the split mode approach, link layer approach, and end-to-end TCP modifications. In the split mode approach [20]–[22], the traffic in a wireless network is protected from the wired network by separating the TCP connections of the wired and wireless networks. The base station is responsible for the recovery of the packet losses caused by wireless links. This approach requires large buffers at base stations which violates the end-to-end TCP semantics. The link layer approach [23]–[26] rectifies wireless link errors by employing a suite of techniques such as forward error correction (FEC), automatic repeat request (ARQ), and explicit loss notification (ELN). However, these techniques require protocols at different layers to interact and coordinate closely, which increase the complexity of protocol implementation. In the end-to-end approach, TCP senders and receivers are responsible for the flow control. TCP-Peach [27] is particularly designed for the satellite communication environment. TCP-Peach replaces slow start and fast recovery phases with sudden start and rapid recovery, respectively. In sudden start and rapid recovery, the sender probes the available network bandwidth in only one RTT with the help of low-priority dummy packets. An important assumption made by TCP-Peach is that the routers must support priority queuing. In TCP-Peach Plus [28], the actual data packets with lower priority replace the low-priority dummy packets as the probing packets to further improve the throughput. TCP-Westwood [29] is a rate based end-to-end approach, in which the sender estimates the available network bandwidth dynamically by measuring and averaging the rate of returning ACKs. TCP-Westwood claims improved performance over TCP-Reno and -Sack, while achieving fairness and friendliness. The end-to-end approach maintains the network layer structure and requires minimum modification at end hosts and router.

2.2 RED and ECN enabled routers

In this approach, routers play a significant role in controlling TCP transmission rates by implementing active queue management (AQM). Random early detection (RED), proposed in [30], is an AQM scheme implemented in routers that informs the sender of incipient congestion by probabilistically dropping packets before the buffer overflows. The packet drop triggers the receiver to send DUPACKs to the sender. The sender is therefore able to adjust its window size in response to the packet drop by means of congestion control. The early packet drop prevents the router to enter the fully congested state. By doing so, the average queue length at the router can be kept small, hence reducing the queueing delay and improving the TCP throughput. Explicit congestion notification (ECN) [31] is an extension to RED. Instead of randomly early dropping packets, an ECN router marks packets to alert the sender of incipient congestion. ECN is an explicit signaling mechanism designed to convey network congestion information from routers to end stations; however, since the signaling only uses one bit for such congestion information, the

information conveyed is not quantitative. For TCP flow control, ECN works by configuring the intermediate router to mark packets with congestion experienced (CE) bit in the IP header when the router's average queue occupancy exceeds a threshold, so that the TCP receiver can echo this information back to the sender via ACK by setting the explicit congestion echo (ECE) bit in the TCP header. TCP Jersey [13] integrates ECN with a modified version of TCP Westwood. Although its throughput performance is improved, the additional requirement on network routers to provide information about the onset of congestion limits their practical applications.

3. TCP VenoPlus

In this paper, we propose TCP VenoPlus which incorporates a congestion loss window to keep track of the level of congestion in the bottleneck links and leverages the availability of signal strength information for the received packet at the MAC layer to accurately detect the presence of random losses due to noise and/or interference on the physical links. VenoPlus adjusts the congestion window in a smart fashion by utilizing the information obtained from the congestion loss window and the signal strength of the received packet. The congestion loss window is calculated as follows: Considering a sequence of packet losses during the lifetime of a TCP connection. Let W_i be the packet loss monitoring window. The monitoring window represents the number of packets lost to be observed in the current window. The window is a configurable parameter which represents the number of received packets for observation of packet losses. Let T_i be the time stamp for window W_i. The congestion losses are computed during every window cycle. The number of congestion losses occurring during the current window W_i is called the congestion loss (C_i). The congestion loss window $conw_i$ can be calculated as follows:

$$conw_i = C_i / (T_i - T_{i-1}) \qquad (2)$$

Next, we will show how TCP Venoplus utilizes the received signal strength information (*RSSI*) for a packet to calculate random losses due to noise and interference. Each radio receiver in a wireless node is equipped with power sensitivity, which allows the receiver to detect and decode signals with strength larger than this sensitivity. There are two threshold values when receiving radio signals: receive threshold (*RXThresh*) and carrier sense threshold (*CSThresh*). If the power of the received signal is higher than *RXThresh*, it is regarded as a valid packet and passed to the TCP layer. During the current monitoring window cycle, Venoplus obtains the *RSSI* for every received packet from the MAC layer. This requires a cross-layer implementation to facilitate transfer of *RSSI* values between the MAC layer and TCP layer. In recent years, cross-layer approaches have gained significant attention, due to their ability to allow critical information of the wireless medium in the MAC and physical layers to be shared with higher layers, in order to provide efficient methods of allocating network resources and applications over the Internet [19].
The packets whose RSSI is less than the *RXThresh* or *CSThresh* are dropped by the MAC layer. But Venoplus procures the number of dropped packets (R_i) from the MAC layer and computes the random loss rate ($ranl_i$). R_i is calculated at the MAC layer by keeping track of the received signal strength (RSS) of each packet. If the RSS value falls below a certain threshold the packet is dropped.

$$ranl_i = R_i / (T_i - T_{i-1}) \qquad (3)$$

We use a low-pass filter to calculate $conw_i$ and $ranl_i$. For the low pass filter we use the exponential weighted moving average (EWMA) [18].

if (congestion loss) **then**
  $conw_{inst}$=α conw + (1-α)conw
else if (random loss) **then**
  $ranl_{inst}$=α ranl + (1-α)ranl
end if

Figure 1 illustrates the process of computation of congestion loss window and random loss rates. The packet train is categorized into different windows during a given period of time. The values for *conw* and *ranl* are calculated during every window. Having calculated the values for *conw* and *ranl*, we now present the packet loss identification scheme:

if ($conw_i > conw_{i-1}$ and $ranl_i > ranl_{i-1}$) **then**
cwnd=cwnd x 1/2; { Packet losses due to congestion and random errors}
if ($conw_i < conw_{i-1}$ and $ranl_i > ranl_{i-1}$) **then**
cwnd=cwnd x 4/5; { Packet losses due to random errors}
if ($conw_i > conw_{i-1}$ and $ranl_i < ranl_{i-1}$) **then**
cwnd=cwnd x 1/2; { Packet losses due to congestion}
if ($conw_i < conw_{i-1}$ and $ranl_i < ranl_{i-1}$) **then**
cwnd=cwnd ; { No packet losses}

The packet identification scheme illustrates four different scenarios which are summarized as follows: 1) If $conw_i > conw_{i-1}$ and $ranl_i > ranl_{i-1}$, the network suffers from losses due to congestion and random errors in the link simultaneously. This situation requires slowing down the transmission of packets which is reflected by the reduction of the congestion window by half. 2) If $conw_i < conw_{i-1}$ and $ranl_i > ranl_{i-1}$, the packet losses can be attributed to random errors in the link only. The congestion window is cut down by 1/5. 3) If $conw_i > conw_{i-1}$ and $ranl_i < ranl_{i-1}$, the packet losses can be attributed to congestion losses in the network. The congestion window is now cut down by 1/2. 4) Finally, if $conw_i < conw_{i-1}$ and $ranl_i < ranl_{i-1}$, the congestion in the network and the random errors in the link are not getting worse, which means that the congestion window remains unchanged.

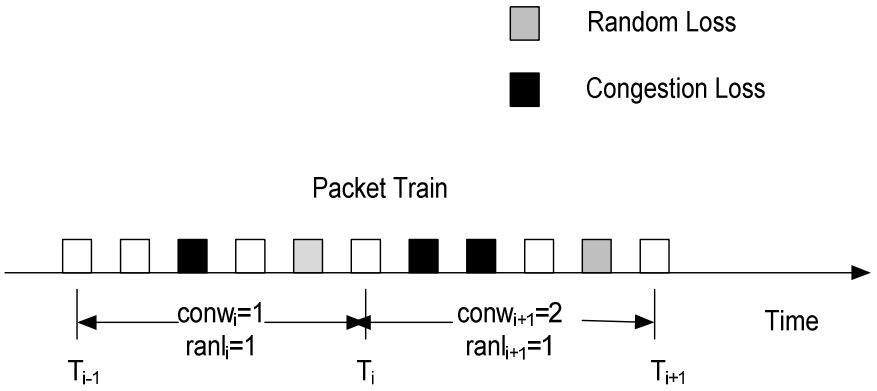

Figure 1: Computation of congestion loss window and random loss rate.

4. SIMULATION & EVALUATION

In this section, we will evaluate the following performance metrics of VenoPlus as compared to Veno: throughput, and fairness. The comparison will be performed with varied number of nodes and random loss rates.

4.1 Simulation Model

We use network simulator ns-2 to conduct the simulations. Figure 2 illustrates the heterogeneous wired-cum-wireless topology. The wired nodes represent the source of traffic and the wireless nodes represent the sinks. The wired links between R2 and the traffic sources are set with 10 Mbps bandwidth, 0.1 ms roundtrip time and buffer size of 50 packets. The wireless links between R1 and the traffic sink are set with 10Mbps bandwidth, 0.1 ms round-trip time, and buffer size of

50 packets. The link between R1 and R2 represents the wired bottleneck link. The bandwidth of the bottleneck link is set to be 10 Mbps and the propagation delay is 80 ms. The random loss rate in the wireless links changes from 0.0001 to 0.1 in packet unit. The shadowing model for physical links is appropriately configured to generate the desired random loss rates.

4.2 Throughput

To compare the throughput between Venoplus and Veno, multiple TCP connections ranging from 1 to 64 are setup to create a congestive bottleneck link. The throughput comparison is computed by a normalized throughput ratio. The normalized throughput is given by

$$TH_{norm} = \frac{TH_{venoplus}}{TH_{veno}}$$

where $TH_{venoplus}$ and TH_{veno} are the average throughput of flows for Venoplus and Veno respectively. The normalized throughput will signify the improvement in throughput obtained by Venoplus as compared to Veno.

Figure 3 illustrates the normalized throughput for different number of TCP connections and bit error rates. The bit error rates range from 10^{-4} to 10^{-1}. As shown in Figure 3, TCP Venoplus is capable of better throughput improvements than Veno when the number of TCP connections is small. The maximum improvement is close to 80 % when the bit error rate is 10^{-3}. But as the number of TCP connections increase, which is an indicator of heavy network traffic, the throughput offered by Venoplus is closer to Veno. This proves our hypothesis that Venoplus is better than Veno when the network load is light and similar to Veno when the network load is heavy.

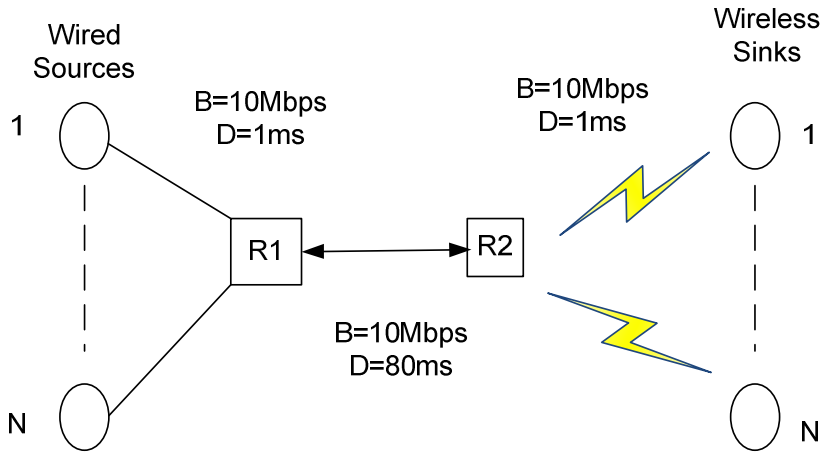

Fig. 2: Simulation Topology

4.3 Fairness

The goal of TCP Fairness is that if N TCP connections share same bottleneck link, each connection should get 1/N of link capacity. In our simulations, we set the value of N to be 64 and verify if each connection receives 1/64 of the link capacity. The fairness ratio is computed using the Jain Fairness index *f* which is defined as:

$$f = \frac{\left(\sum_{i=1}^{n} th_i\right)^2}{n\sum_{i=1}^{n} th_i^2}$$

where, n is the number of TCP connections, th_i is the throughput for the ith connection. The goal of TCP fairness is then to ensure that the value of f is closer to 1. Figure 4 illustrates the fairness ratio for Venoplus. The fairness ratio is close to 1 from light traffic to heavy traffic and for low bit error rate to high bit error rate. This proves that Venoplus is capable of providing better throughput without compromising fairness.

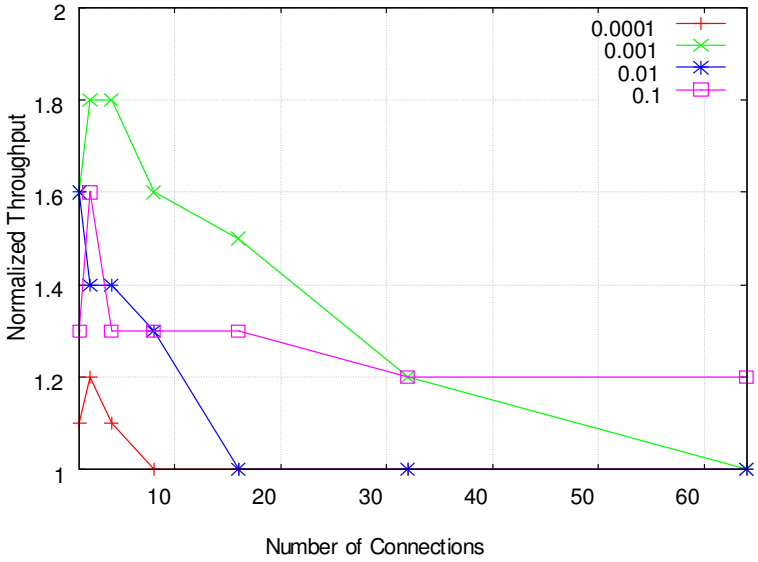

Figure 3: Throughput improvement of Venoplus in presence of different bit error rates and TCP connections.

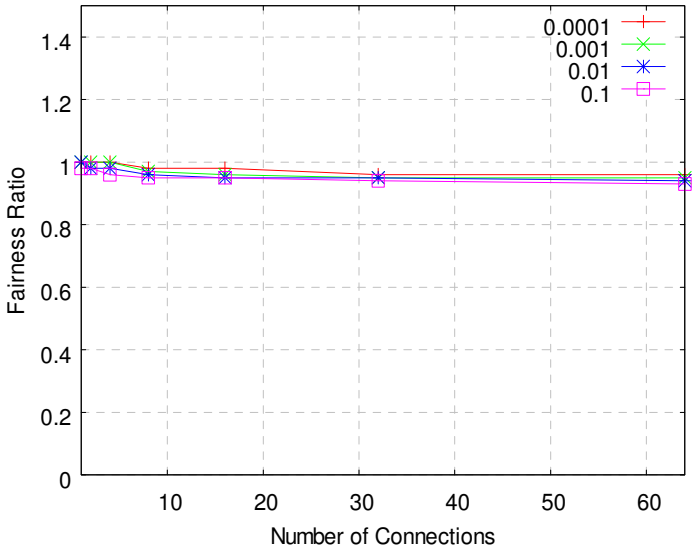

Figure 4: Fairness index of Venoplus

5. CONCLUSION & FUTURE WORK

TCP schemes have to adapt to the changing trends in modern networks. In this paper we address the problems with existing TCP schemes to accurately detect the reason for packet losses. We highlight the performance issues for TCP Veno in presence of bursty congestion and lightly loaded wireless networks. We propose TCP Venoplus which incorporates two refinements to TCP Veno to alleviate the performance degradation in lightly loaded wireless networks. Simulations results demonstrate that Venoplus can improve the accuracy of congestion loss identification in Veno in presence of light traffic. Venoplus provides significant improvement in throughput over Veno without sacrificing fairness.

For future work, we plan to extend the simulations to include bursty traffic and evaluate the performance of Venoplus in presence of transient congestion. We also plan to incorporate Venoplus in the TCP/IP suite of a Xilinx Virtex-II Pro FPGA and perform experimental evaluations.

6. ACKNOWLEDGEMENT

This work was supported in part by the National Science Foundation under award number DUE-0633512. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the National Science Foundation.

7. REFERENCES

1. W. Richard Stevens, “TCP/IP Illustrated,” Addison-Wesley 1994.
2. C. P. Fu and S. C. Liew. “TCP Veno: TCP Enhancement for Transmission Over wireless Access Networks,” IEEE Journal on Selected Areas in Communication, February 2003.
3. V. Jacobson, "Congestion avoidance and control," SIGCOMM 88, ACM, Aug. 1988
4. C. L. Zhang, C. P. Fu, M. T. Yap, C. H. Foh, K. K. Wong, C. T. Lau, and E. M. K. Lai, “Dynamics comparison of TCP Reno and Veno,” in Proc. IEEE Globecom 2004.
5. C. P. Fu, W. Lu, and B. S. Lee, “TCP Veno revisited,” in Proc. IEEE Globecom 2003.
6. Y. Tian, K. Xu, and N. Ansari. “TCP in Wireless Environments: Problems and Solutions,” IEEE Radio Communications Magazine, March 2005.
7. A. Gurtov, “Effect of Delays on TCP Performance,” Proc of IFIP Personal Wireless Commun., Aug. 2001.
8. Balakrishnan, H., Padmanabhan, V. N., Seshan, S., and Katz, R. H., “A Comparison of Mechanisms for Improving TCP Performance over Lossy Links,” *IEEE/ACM Transactions on Networking*, Dec 1997
9. Balan, R.K.; Lee, B.P.; Kumar, K.R.R.; Jacob, L.; Seah, W. K. G., Ananda, A.L., “TCP HACK: TCP header checksum option to improve performance over lossy links,” *Proceedings of Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, pp. 309-318, 2001.
10. Comer, Douglas E. (2006). *Internetworking with TCP/IP* (5E ed.). Prentice Hall: Upper Saddle River, NJ.
11. Chung, S. M., Li, C. Y., Lee, H. H., Li, L. H., Tsai, Y. C. and Chen, C. C., “Design and implementation of the high speed TCP/IP Offload Engine,” *Proceedings of IEEE International Symposium on Communications and Information Technologies,* Oct. 17-19, 2007, pp. 574-579.
12. Krishnan, R., Allman, R. M., Partridge, C. and Sterbenz, J. P. G., “Explicit Transport Error Notification for Error-Prone Wireless and Satellite Networks,” *BBN Technical Report* No. 8333, BBN Technologies, Feb. 2002.
13. Xu, K., Tian, Y. and Ansari, N., “TCP-Jersey for Wireless IP Communications,” *IEEE Journal on Selected Areas in Communications*, Vol. 22, No. 4, 2004, pp. 747-756.

14. L. Brakmo, S. O'Malley, and L. Peterson. "TCP Vegas: new techniques for congestion detection and avoidance," SIGCOMM, ACM, 1994.
15. Z. Zou, C. Fu, B. Lee, "A refinement to improve TCP Veno performance under bursty congestion," in Proc. IEEE Globecom 2005.
16. B. Zhou, C. Fu, K. Zhang, C. Lau, C. Foh, "An enhancement of TCP Veno over light-load wireless networks," IEEE Communication Letters, 2006.
17. S. Shakkottai, T. S. Rappaport and P. C. Karlsson ,"Cross-layer Design for Wireless Networks,", IEEE Communications magazine, October, 2003.
18. T. V. Lakshman and U. Madhow, "The performance of TCP/IP for networks with high bandwidth-delay products and random loss," *IEEE/ACM Trans. Networking*, vol. 5, pp. 336–350, June 1997.
19. F. Lefevre and G. Vivier, "Understanding TCP's behavior over wireless links," in *Proc. Communications Vehicular Technology*, 2000, SCVT-2000, pp. 123–130.
20. V. Tsaoussidis and I. Matta, "Open issues on TCP for mobile computing," *J. Wireless Commun. Mobile Computi.*, vol. 2, no. 1, pp. 3–20, Feb. 2002.
21. A. Bakre and B. R. Badrinath, "I-TCP: Indirect TCP for mobile hosts," in *Proc. ICDCS 95*, May 1995, pp. 136–143.
22. K. Wang and S. K. Tripathi, "Mobile-end transport protocol: An alternative to TCP/IP over wireless links," in *IEEE INFOCOM*, vol. 3, Mar. 1998, pp. 1046–1053.
23. H. Balakrishnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *ACM/Baltzer Wireless Networks J.*, vol. 1, no. 4, pp. 469–481, Dec. 1995.
24. S. Keshav and S. Morgan, "SMART retransmission: Performance with overload and random losses," in Proc. IEEE INFOCOM'97, vol. 3, 1997, pp. 1131–1138.
25. K. Ratnam and I. Matta, "WTCP: An efficient mechanism for improving TCP performance over wireless links," in Proc. Int Symp. Computers Communications, 1998, pp. 74–78.
26. H. Balakrishnan and R. H. Katz, "Explicit loss notification and wireless web performance," in Proc. IEEE GLOBECOM Internet Mini-Conf., Sydney, Australia, Nov. 1998.
27. I. F. Akyildiz, G. Morabito, and S. Palazzo, "TCP-Peach: A new congestion control scheme for satellite IP networks," IEEE/ACM Trans. Networking, vol. 9, pp. 307–321, June 2001
28. I. F. Akyildiz, X. Zhang, and J. Fang, "TCP-Peach+: Enhancement of TCP-Peach for satellite IP networks," *IEEE Commun. Lett.*, vol. 6, pp. 303–305, July 2002.
29. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth estimation for enhanced transport over wireless links," *ACM Mobicom*, pp. 287–297, July 2001.
30. S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Networking*, vol. 1, pp. 397–413, Aug. 1993.
31. S. Floyd, "TCP and explicit congestion notification," *ACM Comput.Commun. Rev.*, vol. 24, pp. 10–23, Oct. 1994.