
TCP in Wireless Environments: Problems and Solutions



YE TIAN, KAI XU, AND NIRWAN ANSARI

Abstract

The Internet provides a platform for rapid and timely information exchange among a disparate array of clients and servers. TCP and IP are separately designed and closely tied protocols that define the rules of communication between end hosts, and are the most commonly used protocol suite for data transfer in the Internet. The combination of TCP/IP dominates today's communication in various networks from the wired backbone to the heterogeneous network due to its remarkable simplicity and reliability. TCP has become the de facto standard used in most applications ranging from interactive sessions such as Telnet and HTTP, to bulk data transfer like FTP. TCP was originally designed primarily for wired networks. In a wired network, random bit error rate, a characteristic usually more pronounced in the wireless network, is negligible, and congestion is the main cause of packet loss. The emerging wireless applications, especially high-speed multimedia services and the advent of wireless IP communications carried by the Internet, call for calibration and sophisticated enhancement or modifications of this protocol suite for improved performance. Based on the assumption that packet losses are signals of network congestion, the additive increase multiplicative decrease congestion control of the standard TCP protocol reaches the steady state, which reflects the protocol's efficiency in terms of throughput and link utilization. However, this assumption does not hold when the end-to-end path also includes wireless links. Factors such as high BER, unstable channel characteristics, and user mobility may all contribute to packet losses. Many studies have shown that the unmodified standard TCP performs poorly in a wireless environment due to its inability to distinguish packet losses caused by network congestion from those attributed to transmission errors. In this article, following a brief introduction to TCP, we analyze the problems TCP exhibits in the wireless IP communication environment, and illustrate viable solutions by detailed examples.

Introduction

Created by the U.S. Department of Defense, the ARPANET, which made use of packet switching technology, was a milestone and the precursor for the modern Internet. Both TCP and IP were inspired and originally drafted in RFC 793 and RFC 791, respectively, in September 1981. They reside in the different layers of the open system interconnection (OSI) architecture, and thus have distinct but related functionalities in data communications. IP is a connectionless best-effort-based variable length packet delivery network layer protocol that does not guarantee the reliable, timely, and in-order delivery of packets between end hosts. It focuses on the routing mechanism that guides the packets from a host to one or multiple designated hosts based on the addressing scheme. As a best effort type of protocol, IP has fulfilled its task fairly well. TCP is a layer four transport protocol that uses the basic IP services to provide applications with an end-to-end and connection-oriented packet transport mechanism that ensures the reliable and ordered delivery of data.

TCP was primarily designed for wired networks. In wired networks random bit error rate (BER) is negligible, and congestion is the main cause of packet loss. Each TCP packet is associated with a sequence number, and only successfully received in-order packets are acknowledged to the sender by the receiver, by sending corresponding packets (acknowledgments, ACKs) with sequence numbers of the next expected packets. On the other hand, packet loss or reception of out-of-order packets indicates failures. To eradicate such failures, TCP implements flow control and congestion control algorithms based on the sliding window and additive increase multiplicative decrease (AIMD) [1] algorithms.

TCP Reno [2] is one of the most widely adopted TCP schemes. It has four transmission phases: slow start, congestion avoidance, fast recovery, and fast retransmit. TCP maintains two variables, the congestion window size ($cwnd$), which is initially set to be 1 maximum segment size (MSS), and the slow start threshold ($ssthresh$). At the beginning of a TCP connection, the sender enters the slow start phase, in which $cwnd$ is increased by 1 MSS for every ACK received; thus, the TCP sender's $cwnd$ grows exponentially in round-trip times (RTTs). When $cwnd$ reaches $ssthresh$, the TCP sender enters the congestion avoidance phase. Reno employs a sliding-window-based flow control mechanism allowing the sender to advance the transmission window linearly by one segment upon reception of an ACK, which indicates the last in-order packet received successfully by the receiver. When packet loss occurs at a congested link due to buffer overflow at the intermediate router, either the sender receives duplicate ACKs (DUPACKs), or the sender's retransmission timeout (RTO) timer expires. These events activate TCP's fast retransmit and recovery, by which the sender reduces the size of its congestion window ($cwnd$) to half and linearly increases $cwnd$ as in congestion avoidance, resulting in a lower transmission rate to relieve the link congestion.

Communication networks have evolved greatly in the past decades. Packet switching technologies have merged the traditional voice and data networks together into converged and integrated multimedia networks. The horizon of the converged and integrated network is extending further to incorporate wired, wireless, and satellite technologies. The all-IP wired and wireless hybrid network is becoming a reality. TCP/IP has become the dominant communication protocol suite in today's multimedia applications. A large amount of Internet traffic is carried by TCP. Hence, TCP/IP needs to depart from its original wired network oriented design and evolve to meet the challenges introduced by the wireless portion of the network.

Challenges in the Heterogeneous Network

With the advances of wireless technologies and ever increasing user demands, the IP protocols suite has to extend its capability to encompass the wireless aspect. In reality, future all-IP networks will most likely be heterogeneous, implying that the communication path from one end to another will consist of both wired and wireless links. However, the TCP we have enjoyed and relied on in wired networks exhibits weaknesses in

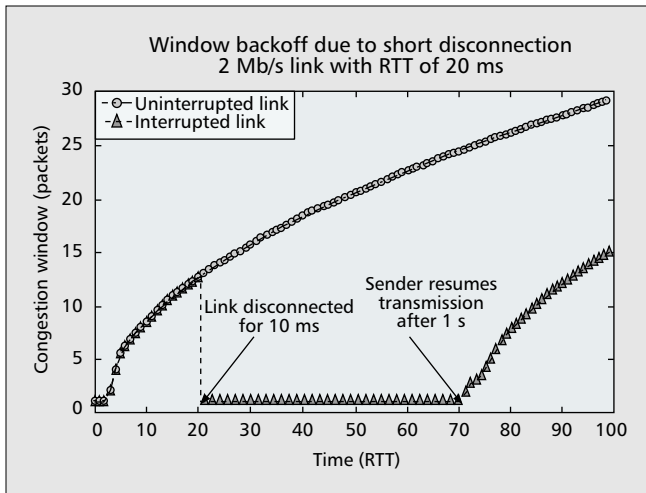


FIGURE 1. The effect of a short disconnection in TCP transmission.

such hybrid environments. The problems stem from the unique characteristics of wireless links as compared to wired links, and the current TCP's design assumption of the packet loss model. The problems manifest in various applications as degradation of throughput, inefficiency in network resource utilization, and excessive interruption of data transmissions [3, 4].

Unlike the fiber optical backbone and copper wired access networks, wireless links use the open air as the transmission medium and are subject to many uncontrollable quality-affecting factors such as weather conditions, urban obstacles, multipath interferences, large moving objects, and mobility of wireless end devices. As a result, wireless links exhibit much higher BERs than wired links. Also, limitations of radio coverage and user mobility require frequent handoffs, thus resulting in temporal disconnections and reconnections between the communicating end hosts during a communication session. Note that a short disconnection event can actually stall the TCP transmission for a much longer period. This effect is demonstrated in Fig. 1, where the evolution of the congestion window reflects TCP transmission throughput, and the time axis is in units of RTTs. This result is taken from a single TCP connection between two hosts, of which the link capacity is 2 Mb/s and the RTT is 20 ms. During the disconnection, both data packets and ACKs are dropped, and each retransmission attempt leads to an unsuccessful retransmission. These consecutive failed attempts of retransmission will cause the TCP sender to exponentially back off its retransmission timer by a typical granularity of 500 ms. The total stall time of the retransmission, as shown in Fig. 1, is added up to about 50 RTTs, or 1 s.

A standard TCP like Reno cannot handle the high BER and frequent disconnections effectively. Since all packet losses are inferred to be the result of network congestion in standard TCP, random packet loss caused by the high BER of the wireless link would mistakenly trigger the TCP sender to reduce its sending rate unnecessarily. The fast retransmit and fast recovery algorithms introduced by TCP Reno can recover from sporadic random packet losses fairly quickly if such losses only occur once within an RTT. However, noises and other factors in the wireless environment usually cause random bit errors to occur in short bursts, thus leading to a higher probability of multiple random packet losses within one RTT. Again, multiple unsuccessful retransmissions within one RTT would cause the retransmission timer to exponentially back off. Therefore, for instance, two random packet losses within an RTT would cause the TCP sender to stall its transmission for a period of about 1 s. This phenomenon is illustrated in Fig. 2. In comparison, two packet drops far apart in time do not cause a retransmission stall. The inability of the standard TCP protocol to

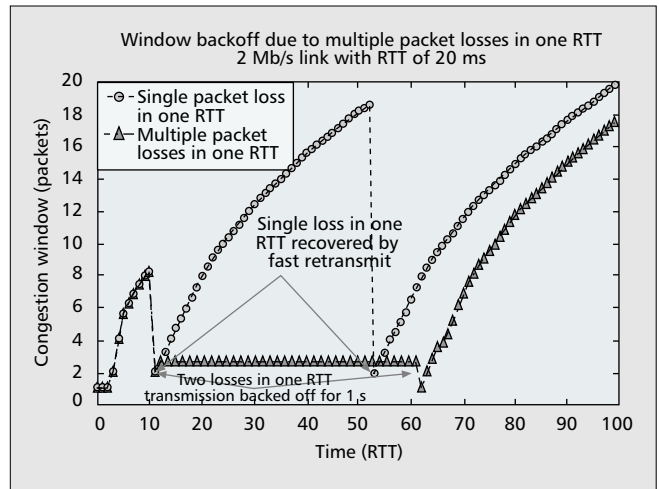


FIGURE 2. The effect of multiple losses in one RTT in TCP transmission.

distinguish between congestive and random packet loss contributes to a drastic decrease of TCP's throughput.

Link asymmetry in the network with respect to TCP performance refers to the situation where the forward and reverse paths of a transmission have different capacities [5]. In the 3G cellular networks, multimedia-based data transmission is carried through the wireless channel, and so are the conventional voice data. In practice, the traffic on the downlink from the base station to the mobile device tends to consume more bandwidth than the traffic on the uplink. Also, the transmission power of a base station is much higher than that of a mobile device. Third-generation (3G) wireless networks consider these facts in their design and can possibly control the link asymmetry (e.g., via pricing policy). Link asymmetry results in rate and delay variation. In a data path, the reverse channel may suffer from packet loss even when the condition on the forward channel is moderate. Link asymmetry is less problematic in a wired network than in a wireless network because the probability of transmission error in wired networks is much smaller than in wireless networks; the assumption that all packet losses are results of link congestion is "reasonably held" in wired networks. Because of the relatively smaller size of the ACK packets, most analytical and simulation-based studies of TCP dynamics ignore the fact that ACKs could also experience packet losses induced by both congestion as well as transmission errors. Most TCP schemes assume that packet loss occurs only in the forward channel; the TCP self-clocking mechanism based on ACK feedbacks could therefore be affected and even dominated by poor conditions on the reverse channel. A sender could mistakenly interpret a bad reverse channel condition as congestion on the forward channel and unnecessarily reduce the sending rate, thus resulting in TCP performance degradation.

One problem further exacerbated by link asymmetry is the ACK compression effect. TCP congestion control is self-clocking. In essence, arrivals of ACKs at the sender trigger the sending of new packets and advancing of the congestion window. However, the queuing in the reverse path of a TCP flow can cause the almost instantaneous arrival of successive ACKs at the sender end; this is often referred to as ACK compression. ACK compression can break TCP's self-clocking and cause long bursts of packet transmission in the forward direction, and thus possible congestive packet losses. Suppose several ACKs have been lost during two successfully received ACKs; the sender will send out the number of packets indicated by the newly received ACK back to back. This burdens the forward path with an instant load and exacerbates the forward path condition. This can be devastating to some TCP schemes, which control the sending rate based on interarrival gaps of the returning ACKs.

Usually such schemes assume that short intervals between successive ACKs at the sender imply large network capacity, since it is assumed that the ACK stream on the reverse path preserves the interpacket dispersion on the forward path, an assumption that only holds for ideal reverse paths. The TCP self-clocking mechanism would therefore be affected; the sender would make incorrect judgments of the forward path.

Other than the high BER problem, different types of wireless networks also pose their own challenges. Handoff is an inevitable process in mobile wireless communications due to user mobility as well as limited radio frequency coverage of wireless devices, either base stations or terminals. In a wireless ad hoc network, because of the infrastructurelessness and high node mobility of the network, frequent route changes and network partitions also present a great challenge to TCP performance. In a satellite network, long propagation delay makes long-range transmission inefficient.

Approaches to Improve Wireless TCP Performance

Traditional TCP schemes may suffer from severe performance degradation in a mixed wired and wireless environment. Modifications to standard TCP to remedy its deficiency in wireless communications have been an active research area. Many schemes have been proposed.

TCP for Different Wireless Applications

From the application point of view, there are various wireless environments, and wireless TCP can be specifically designed to accommodate their needs. The most common wireless networks include satellite networks, ad hoc networks, and general wireless platforms such as wireless LANs and cellular systems. The design of wireless TCP considers the characteristics of a particular type of wireless network and its needs; for instance, a satellite network has long propagation delay, and an ad hoc network is infrastructureless. Nevertheless, an obstacle that all wireless networks have to face is high BER. In heterogeneous networks, to explicitly differentiate the cause of packet loss becomes the primary goal of TCP design. Such efforts aim to find an explicit way to inform the sender of the cause of packet dropping, be it congestion or random errors. Therefore, the sender is able to make appropriate decisions on how to adjust the congestion window. The standard Reno scheme halves its window size when experiencing a packet loss regardless of the cause. If the loss is due to network congestion, this behavior alleviates network congestion. However, it would degrade the performance for random loss.

Satellite Networks — TCP schemes for satellite networks are designed based on the observation that it takes the sender of a TCP connection a fairly long time to reach a high sending rate during the traditional slow start phase. While many TCP applications such as HTTP are based on the transfer of small files, with the long propagation delay of the satellite link it can happen that the entire transfer occurs within the slow start phase and the connection is not able to fully utilize the available network bandwidth. TCP-Peach [6] employs two new mechanisms, sudden start and rapid recovery, in combination with the traditional TCP's congestion avoidance and fast retransmit algorithms, to cope with TCP performance degradation over satellite communication links characterized by large propagation delay and high link error rate. Sudden start introduces the use of dummy packets that are copies of the last data packet the sender has sent. The sender sends multiple dummy packets between two consecutive data packets; the reception of ACKs for the dummy packets indicates the availability of the unused network resource, and the sender is triggered to increase its sending window quickly. Dummy packets are labeled with low priority; therefore, a router would drop the dummy packets first

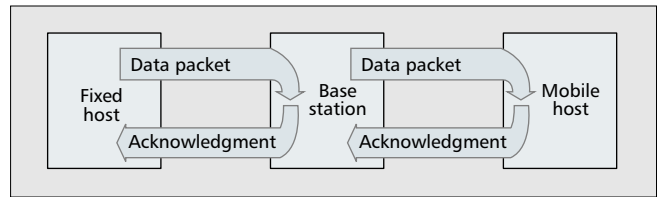


FIGURE 3. Split connection of wireless TCP.

in the event of congestion. Simulations show that sudden start can reach the available bandwidth within two RTTs, whereas traditional slow start in TCP-Reno takes about seven RTTs to reach the same bit rate. Rapid recovery replaces the classical fast recovery mechanism in an effort to improve throughput in the presence of high link error rate. In the rapid recovery phase, instead of trying to distinguish congestive loss from error loss, the sender uses the lower-priority dummy packets to interleave the data packets and inflates the sending window size upon receptions of ACKs for the dummy packets.

Ad Hoc Networks — In an ad hoc network, high error rate and frequent route changes and network partitions are typical. Both characteristics result in packet loss besides network congestion, and hence should be treated differently. ATCP [7] is designed as an end-to-end solution to improve TCP throughput for such an environment. It is implemented as a thin layer inserted between the standard TCP and IP layers. It relies on explicit congestion notification (ECN) to detect congestion and distinguish congestion loss from error loss, and the ICMP Destination Unreachable message to detect a change of route or temporary partition in an ad hoc network. According to these feedbacks from the network, the ATCP layer puts the TCP sender into either the persist, congestion control, or retransmit state accordingly. When the packet loss is due to a high BER, ATCP retransmits the lost packet so that TCP does not perform congestion control; if the ICMP message indicates a route change or network partition, ATCP puts the TCP sender into the persist state waiting for the route to reconnect. For packet reordering, ATCP reorders the packets so that TCP would not generate DUPACKs. When ECN indicates real congestion, TCP enters the normal congestion control stage. By inserting the ATCP layer between the TCP and IP layers, the scheme does not modify the TCP code; in addition, since the ATCP layer does not generate or regenerate ACK packets by itself, the end-to-end TCP semantic is maintained.

Cellular Network — For cellular networks, where the base station interconnects a fast fixed network and a slow mobile network, modifications of TCP algorithms focus on cellular characteristics such as handoff and the commonly shared problem of all wireless networks, high BER. Freeze-TCP [8] is one end-to-end solution to improve TCP performance in the mobile environment. It imposes no restrictions on routers and only requires code modifications at the mobile unit or receiver side. It addresses the throughput degradation caused by frequent disconnections (and reconnections) due to mobile handoff or temporary blockage of radio signals by obstacles. The assumption is that the mobile unit has knowledge of the radio signal strength, and therefore can predict the impending disconnections. The Freeze-TCP receiver on the mobile unit proactively sets the advertised window size to zero in the ACK packets in the presence of impending disconnection. Because the TCP sender selects the window size to be the minimum of its vision of the window size and the receiver's advertised window size, this zero window size ACK packet would force the sender into persist mode, where it ceases sending more packets while keeping its sending window unchanged. To prevent the sender from exponentially backing off, when it detects the reconnection the receiver sends several positive ACK packets to the sender

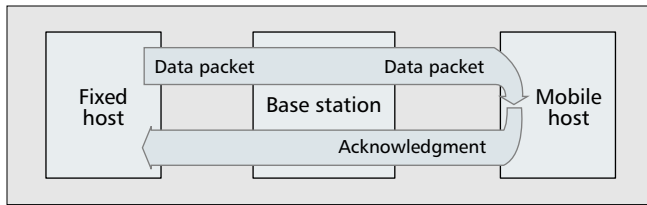


FIGURE 4. End-to-end connection of wireless TCP.

acknowledging the last received packet before disconnection so that the transfer can resume quickly at the rate before the disconnection occurs. To implement the scheme proposed in [8], cross-layer information must be exchanged, and the TCP layer protocol must be exposed to some details of the roaming and handoff algorithms implemented by network interface card (NIC) vendors on the interface devices.

Implementation of Wireless TCP

From the implementation point of view, wireless TCP algorithms can be designed either in split mode or end-to-end. Due to the significant difference in characteristics between wireless and wired links, the split mode divides the TCP connection into wireless and wired portions, and ACKs are generated for both portions separately. By doing so, the performance on the wired portion is least affected by the relatively unreliable wireless portion. On the other hand, the end-to-end mode treats the route from the sender to the receiver as an end-to-end path, and the sender is acknowledged directly by the receiver. This maintains the end-to-end semantics of the original TCP design.

Split Mode — The wired portion of the heterogeneous network is more reliable than the wireless portion in terms of link capacity and error rate; however, transmission would be bottlenecked at a slow and lossy wireless link. Split mode attempts to shield the wireless portion from the fixed network by separating the flow control at the intermediate router (or a base station for a cellular network), so that the wireless behavior has the least impact on the fixed network. The intermediate router behaves as a terminal in both the fixed and wireless portions. Both end hosts communicate with the intermediate router independently without knowledge of the other end. The intermediate router is reinforced with functionalities to coordinate the transaction between the two network portions. This is illustrated in Fig. 3. In Indirect TCP (I-TCP) [9], the mobile support router (MSR) connects the mobile host (MH) to the fixed host (FH), and establishes two separate TCP connections with the FH and MH, respectively. The MSR communicates with the FH on behalf of the MH. The congestion window is maintained separately for the wireless and the fixed connections. When the MH switches cells, a new MSR takes over the communication with the FH seamlessly. Therefore, the FH is shielded from the unreliable feature of wireless connections.

End-to-End Approach — In split mode, an intermediate router has to reveal the information in the TCP packet and process related data before it reaches the destination, thus violating the end-to-end semantics of the original TCP. In the end-to-end approach, only the end hosts participate in flow control. The receiver provides feedback reflecting the network condition, and the sender makes decisions for congestion control. This is illustrated in Fig. 4.

In the end-to-end approach, the ability to accurately probe for the available bandwidth is the key to better performance, which is still a great challenge. The available bandwidth of a flow is the minimum unused link capacity of the flow's fair share along the path. The end-to-end approach can have its congestion control mechanism realized in two ways, reactive

and proactive. By reactive congestion control, the sender rectifies the congestion window when the network situation becomes marginal or has crossed a threshold. By proactive congestion control, feedbacks from the network guide the sender to reallocate network resources in order to prevent congestion.

Reactive Congestion Control — The standard Reno scheme employs reactive flow control. The congestion window is adjusted based on the collective feedback of ACKs and DUPACKs generated at the receiver. TCP probes for the available bandwidth by continuously increasing the congestion window gradually until the network reaches the congestion state. In this sense, congestion is inevitable. TCP will then fall back to a much slower transmission rate, which may be unnecessary for wireless random loss. Many TCP schemes have been proposed as an enhancement to the standard Reno scheme in this reactive manner.

The fast recovery algorithm of Reno takes care of a single packet drop within one window. After one lost packet is recovered, Reno terminates the fast recovery mechanism. Due to the nature of wireless networks, correlated errors may induce multiple packet drops. Therefore, the Reno scheme would be forced to invoke multiple fast recovery procedures back and forth, slowing down recovery of the lost packet. New Reno [10] modifies the fast recovery mechanism of Reno to cope with multiple losses from a single window; this is one of the characteristics of wireless networks, where a fading channel may cause contiguous packet loss. In New Reno the fast recovery mechanism does not terminate until multiple losses, indicated by the reception of partial ACKs, from one window are all recovered. The limitation of New Reno is, however, that it cannot distinguish the cause of the packet loss; thus, a more effective fast recovery algorithm cannot be implemented.

TCP SACK [11] is a selective ACK (SACK) option for TCP, targeting the same problem Reno tries to tackle. While the feedback of Reno and New Reno is based on cumulative ACKs, SACK employs a selective repeat retransmission policy. It indicates a block of data that has been successfully received and queued at the receiver when packet loss occurs instead of sending a partial ACK as in New Reno. Hence, the sender has better knowledge about the exact number of packets that have been lost rather than limited knowledge about the loss at the left edge of the window only, as in the standard ACK scheme. SACK requires modifications at the sender and receiver sides. SACK blocks are encoded in the TCP option field, which inherently limits the number of SACK blocks one ACK can carry. Moreover, as in Reno, SACK reactively responds to packet losses, and therefore has limited ability for congestion avoidance.

Proactive Congestion Control — In proactive congestion control, the sender attempts to adjust the congestion window proactively to an optimal rate according to the information collected via feedback, which can be translated to an indication of the network condition. By doing so, the sender reacts intelligently to the network condition or the cause of the packet loss, due to either congestion or random errors, and therefore prevents the network from entering an undesired state (e.g., congestion or unnecessary decrease of the congestion window). Different strategies can be employed in the design to provide the sender with the explicit network condition.

TCP Vegas [12] estimates the backlogged packets in the buffer of the bottleneck link. Vegas selects the minimal RTT as a reference to derive the optimal throughput the network can accommodate. It also records the actual sending rate at the sender during transmission to derive the actual throughput. The difference between the optimal throughput and the actual sending rate can be used to derive the amount of backlogged data in the network. It then sets two thresholds corresponding to two network stages, one of which represents too little backlogged data, the other too much. For the former stage, Vegas

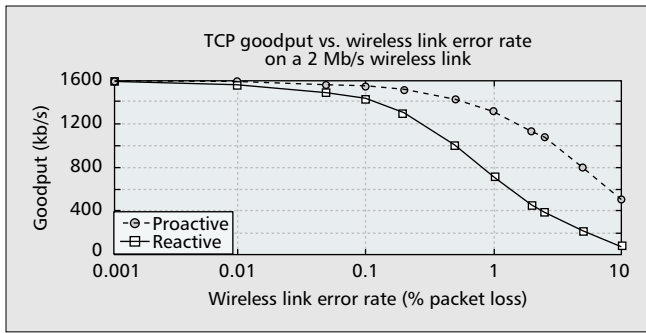


FIGURE 5. Performance of a typical proactive scheme (TCP-Jersey) and typical reactive scheme (TCP-Reno) in the wireless environment.

increases the congestion window linearly; for the latter, Vegas decreases the congestion window linearly. In doing so, Vegas tries to stabilize the network congestion state around the optimal point by proactively adjusting the congestion window without a dramatic change in the congestion window.

TCP Vegas [13] adopts the same methodology as Vegas to estimate the backlogged packets in the network. It further suggests a way to differentiate the cause of packet loss. If the number of backlogged packets is below a threshold, the loss is considered to be random. Otherwise, the loss is said to be congestive. If the loss is congestive, Vegas adopts the standard Reno scheme. For loss due to a random error, it increases the congestion window in a conservative manner (i.e., sending one packet for every other ACK received).

TCP Westwood [14] 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. Under the assumption of an ideal error-free and congestions-free reverse path, the inter-ACK gap reflects the available network resource. Westwood deploys an available bandwidth measurement module at the sender side based on the interval of returning ACKs. It calculates the explicit available bandwidth and uses it to guide the sending rate. When TCP-Westwood determines that the link is congested after having received three DUPACKs, it sets the slow start threshold to reflect its estimated bandwidth-delay product. 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 in some cases also the routers.

TCP-Jersey [15] is another proactive scheme that adapts the sending rate proactively according to the network condition. It consists of two key components, the available bandwidth estimation (ABE) algorithm and the congestion warning (CW) router configuration. ABE is a TCP sender side addition that continuously estimates the bandwidth available to the connection and guides the sender to adjust its transmission rate when the network becomes congested. CW is a configuration of network routers such that routers alert end stations by marking all packets when there is a sign of incipient congestion. The marking of packets by CW configured routers helps the sender of the TCP connection to effectively differentiate packet losses caused by network congestion from those caused by wireless link errors. Based on the congestion indication implied by CW and the estimation from ABE, TCP-Jersey calculates the optimum congestion window size at the sender. ECN needs to be supported at the routers in order to implement CW.

Proactive schemes, in general, exhibit the ability to handle random loss more effectively. Figure 5 illustrates the goodput achieved by a typical proactive scheme (TCP-Jersey) vs. a typical reactive scheme (TCP-Reno). The experimental network is a reliable fixed link connected to a lossy wireless link. The link error rate on the wireless link is varied to show how proactive schemes perform in the wireless environment in comparison to reactive schemes. The basic trend is that the performance of both schemes degrades with increasing wireless link error rate. However, TCP-Jersey, which represents the proactive approach, outperforms the standard TCP-Reno (reactive) substantially in a higher-error-rate environment. For instance, at 2 percent error rate, 280 percent more goodput is achieved by TCP-Jersey than by TCP-Reno. This is mainly due to the fact that proactive schemes are able to better distinguish random loss from congestive loss than reactive schemes. Hence, there are fewer unnecessary decreases of the congestion window in transmission. More simulation results and analysis among wireless TCP schemes can be found in [15].

Summary

This article focuses on illustrating the typical wireless TCP techniques with examples. The characteristics of wireless networks vary as access technologies differ; therefore, a universal solution for all types of wireless networks is unlikely to be developed. Each wireless TCP solution tends to tackle specific problems. While it is necessary to analyze each solution's properties and suitable application scenarios, it is difficult to make an overall conclusive evaluation of one solution vs. another. A summary of the wireless TCP schemes listed in this article and their targeted applications is presented in

| | Schemes | Need intermediary | TCP semantics | Support for mobility | Modification requirement | Targeted application |
|-------------------------------|--------------|-------------------|---------------|----------------------|--------------------------|----------------------|
| Application-specific approach | I-TCP | Yes | Split | High | Base station | Cellular |
| | TCP-Peach | Yes | End-to-end | High | Router and end stations | Satellite |
| | ATCP | No | End-to-end | High | TCP stack | Ad hoc |
| | Freeze-TCP | No | End-to-end | High | Base station | Cellular |
| Reactive approach | TCP-New Reno | No | End-to-end | Low | Sender side | Heterogeneous |
| | TCP-SACK | No | End-to-end | Low | Sender side | Heterogeneous |
| Proactive approach | TCP-Vegas | No | End-to-end | Low | Sender side | Heterogeneous |
| | TCP-Veno | No | End-to-end | Low | Sender side | Heterogeneous |
| | TCP-Westwood | No | End-to-end | High | Sender side | Heterogeneous |
| | TCP-Jersey | No | End-to-end | High | Router and sender side | Heterogeneous |

TABLE 1. Comparison of various TCP modification schemes for wireless communications.

Table 1. Nevertheless, proactive TCP schemes have attracted much attention recently. They address the general issue of the heterogeneous network (i.e., frequent random errors in the wireless network). More important, such schemes are able to manage and utilize the available bandwidth in the network more efficiently.

References

- [1] D. Chiu and R. Jain, "Analysis of the Increase/Decrease Algorithms for Congestion Avoidance In Computer Networks," *J. Comp. Net.*, vol. 17, no. 1, June 1989, pp. 1–14.
- [2] K. Fall and S. Floyd, "Simulation-Based Comparisons of Tahoe, Reno and SACK TCP," *Comp. Commun. Rev.*, vol. 26, no. 3, July 1996, pp. 5–21..
- [3] T. V. Lakshman and U. Madhow, "The Performance of TCP/IP for Networks with High Bandwidth-Delay Products and Random Loss," *IEEE/ACM Trans. Net.*, vol. 5, no. 3, June 1997, pp. 336–50.
- [4] F. Lefevre and G. Vivier, "Understanding TCP's Behavior over Wireless Links," *Proc. Commun. and Vehic. Tech.*, 2000 SCVT-200, 2000, pp. 123–30.
- [5] H. Balakrishnan, V. N. Padmanabhan, and R. H. Katz, "The Effects of Asymmetry on TCP Performance," *Proc. ACM/IEEE Mobicom*, Sept. 1997, pp. 77–89.
- [6] I. F. Akyildiz, G. Morabito, and S. Palazzo, "TCP-Peach: A New Congestion Control Scheme for Satellite IP Networks," *IEEE/ACM Trans. Net.*, vol. 9, no. 3, June 2001, pp. 307–21.
- [7] J. Liu and S. Singh, "ATCP: TCP for Mobile Ad Hoc Networks," *IEEE JSAC*, vol. 19, no. 7, pp. 1300–1315, 2001.
- [8] T. Goff *et al.*, "Freeze-TCP: A True End-to-end Enhancement Mechanism for Mobile Environments," *Proc. IEEE INFOCOM 2000*, 2000, pp. 1537–45.
- [9] A. Bakre and B. R. Badrinath, "I-TCP: Indirect TCP for Mobile Hosts," *Proc. ICDCS '95*, May 1995, pp. 136–43.
- [10] S. Floyd and T. Henderson, "The New Reno Modification to TCP's Fast Recovery Algorithm," RFC 2582, Apr. 1999.
- [11] M. Mathis *et al.*, "TCP Selective Acknowledgment Options," RFC 2018, Oct. 1996.
- [12] L. Brakmo and L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet," *IEEE JSAC*, vol. 13, no. 8, Oct. 1995, pp. 1465–80.
- [13] C. P. Fu and S. C. Liew, "TCP Veno: TCP Enhancement for Transmission over Wireless Access Networks," *IEEE JSAC*, vol. 21, no. 2, Feb. 2004, pp. 216–28.
- [14] C. Casetti *et al.*, "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links," *ACM Mobicom*, July 2001, pp. 287–97.
- [15] K. Xu, Y. Tian, and N. Ansari, "TCP-Jersey for Wireless IP Communications," *IEEE JSAC*, vol. 22, no. 4, May 2004, pp. 747–56.

Biographies

YE TIAN received a B.E. (Honors) degree in electrical and electronic engineering from the University of Canterbury, Christchurch, New Zealand in 1999, and an M.S.E.E. degree in electrical and computer engineering from New Jersey Institute of Technology (NJIT), Newark, in 2002. He is a Ph.D. candidate in electrical and computer engineering at NJIT. His current research interests include QoS routing, WLAN, and IPSec.

KAI XU received a B.S. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 1989, and an M.S. degree in computer imaging engineering from Chiba University, Japan, in 1993. He worked for two years in the Center for Advanced Information Processing (CAIP) of Rutgers University as a research assistant and doctorate candidate before he started his career in the industry. He is now expecting to receive a Ph.D. degree in computer engineering from NJIT. Before his current academic pursuits, he held senior level positions, including senior system architect, senior systems engineer, and senior R&D consultant in several telecommunications and data networking companies. His last position was Distinguished Member of Technical Staff in Lucent Technologies. His current research interests include wireless IP, TCP performance analysis, Internet congestion control, active queue management, wireless sensor networks, and wireless multimedia embedded systems.

NIRWAN ANSARI (Nirwan.Ansari@njit.edu) received B.S.E.E. (summa cum laude), M.S.E.E., and Ph.D. degrees from NJIT, University of Michigan, and Purdue University in 1982, 1983, and 1988, respectively. He joined the Department of Electrical and Computer Engineering, NJIT, as an assistant professor in 1988, and has been a full professor since 1997. He authored with E. S. H. Hou *Computational Intelligence for Optimization* (Kluwer, 1997, translated into Chinese in 2000), and edited with B. Yuhua *Neural Networks in Telecommunications* (Kluwer, 1994). He is a technical editor of *IEEE Communications Magazine*, *Computer Communications*, *ETRI Journal*, and *Journal of Computing and Information Technology*. His current research focuses on various aspects of broadband networks and multimedia communications. He organized (as General Chair) the First IEEE International Conference on Information Technology: Research and Education (ITRE 2003).