Approaches of Wireless TCP Enhancement and A New Proposal Based on Congestion Coherence *

Chunlei Liu
Lucent Technologies
6100 E Broad St, Columbus, OH 43213
liu.223@osu.edu

Raj Jain
Department of Computer and Information Science
Ohio State University, Columbus, OH 43210
jain@cis.ohio-state.edu

Abstract

TCP is known to have poor performance over unreliable wireless links where packet losses due to transmission errors are misinterpreted as indications of network congestion. TCP enhancements proposed in the literature differ in their signaling and data recovery mechanisms, applicable network configurations, traffic scenarios and locations where required changes are made. In this paper we categorize existing enhancements into several approaches. Motivated by these criteria, we propose a new enhancement that requires only local changes, but applies to a broad range of network and traffic configurations. Comparison with existing algorithms show this new enhancement achieves excellent performance.

1 Introduction

Transmission Control Protocol (TCP), the most widely used reliable transport protocol, was designed mainly for wired networks where transmission errors are rare and the majority of packet losses are caused by congestion. An important assumption of TCP algorithm is that packet losses and the resulting timeout at the source are indications of congestion and the source should reduce its traffic rate on timeout [1].

When applied to wireless networks where transmission errors are frequent, TCP is found to have poor performance if proper enhancement is not used. This is because the assumption behind TCP congestion control algorithm that the majority of packet losses are caused by congestion is no longer true. When a wireless loss[1] is treated as a congestion loss, the effective TCP transmission rate drops to half. If transmission errors happen frequently, the effective transmission rate of the wireless link becomes almost zero even though the network is not congested.

The rapid development of mobile and wireless networks is a driving force for wireless TCP enhancements. In the past few years, numerous enhancements have been proposed. Theses enhancements differ in their signaling and data recovery mechanisms, the applicable network and traffic configurations, and locations where needed changes are made. These approaches have big impacts on the feasibility, generality, effort and performance of the enhancements. In this paper we summarize the approaches used by major enhancement proposals in the literature, and use them as a set of criteria to evaluate different enhancements. Motivated by the set of criteria, a new enhancement is proposed and its performance is compared with existing methods.

2 Approaches of Wireless TCP Enhancement

Enhancement proposals in the literature differ widely in their mechanism and algorithm. This section is an attempt to categorize them into several approaches.

2.1 End-to-end v.s. Split

TCP is an end-to-end protocol — a packet is acknowledged only after it is received by its final destination. Enhancements that preserved this semantic are called **end-to-end**. Some enhancements split the entire path into a wired connection and a wireless connection and run TCP independently on both connections. When the transmission of a packet is complete in one connection, it is acknowledged to the source and relayed to the next connection. Such enhancements do not preserve TCP's end-to-end semantic and are called **split** enhancements.

I-TCP [2] is an early protocol of split enhancements. Its major drawback is that acknowledgments received by the sender do not mean that the packets have been received

*This research was supported in part by National Science Foundation grants 9980637 and 9809018.

[1] We call packet losses caused by wireless transmission errors "wireless losses" and those by network congestion "congestion losses".

Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03)
0-7695-1874-5/03 $17.00 © 2002 IEEE

by the intended destination. When the mobile host moves to another cell or when the base station crashes, some acknowledged packets may never be received. Since the end-to-end semantic is violated, the transmission of data is not reliable. The second drawback of split enhancements is buffering at the base station. If the two connections run TCP at different paces, a large number of packets may pile up at the base station and cause overflow.

2.2 Local v.s. Global

Another important criterion to evaluate enhancements is where the required changes are made. An enhancement is considered **local** if it requires changes only in network components that are under the control of a wireless service provider, such as base stations and mobile hosts. If an enhancement requires changes outside the control of a wireless service provider, it is regarded as **global**. When a wireless service provider offers an Internet service, he can modify the code in base stations and mobile hosts to improve TCP performance, but requiring changes in all web sites that his subscribers visit is virtually impossible.

Examples of global enhancements are **Partial Acknowledgment** [3] and **Control Connection** [4]. The first algorithm uses two types of acknowledgements to distinguish losses in the wired network and losses on the wireless link. The fixed sender needs to handle the two types of acknowledgements differently. The second algorithm creates a control connection that has the same path but terminates at the base station. Packets are sent periodically on the control connection to measure the congestion status of the wired network, which is then used to determine the cause of packet losses on the real connection. Both algorithms required changes in the wired networks.

Theoretically, global enhancements can be deployed incrementally — individual fixed hosts sites can update their software independently to improve the performance for wireless connections. However, in reality fixed hosts that mainly serve wired connections are less likely to take the overhead of wireless enhancement on all connections just for the benefit of a few wireless connections. We believe that mobile hosts or base stations are the right place for the enhancement, and the enhanced protocol shall be able to talk with existing TCP versions in the wired network.

2.3 Transparent v.s. Snooping

The **Snoop protocol** [5] is the most well-known enhancement. It introduces a *snooping agent* at the base station to observe and cache TCP packets going out to the mobile host as well as acknowledgments coming back. By comparing the cached packets and acknowledgments, the agent is able to determine what packets are lost on the wireless link and schedule a local link layer retransmission. On the same time, duplicate acknowledgments corresponding to wireless losses are suppressed to avoid triggering an end-to-end retransmission at the source. Unlike the other proposals, the Snoop protocol can exactly find out the cause of packet losses and take actions to prevent the TCP sender from making unnecessary window reductions.

Explicit Loss Notification (ELN) [6] is a similar protocol for mobile senders. It uses a bit in the TCP header to communicate the cause of packet losses to the TCP sender, but no packets are cached.

Both Snoop and ELN can detect the exact cause of packet loss and prevent unnecessary congestion control actions, but the problem is that they need to read TCP header of the packets at the base station so they cannot be applied to encrypted traffic that is readable only at the final destination, such as IPSEC traffic.

If an enhancement needs to read header information in the IP payload at an intermediate node, we call it a **snooping** enhancement. Otherwise, we call it a **transparent** enhancement. Generally, snooping enhancements cannot be applied to encrypted traffic.

2.4 Two-Way v.s. One-Way

An enhancement can be **one-way** or **two-way**, depending on whether the enhancement can be applied to one-way or two-way traffic. Many enhancements were designed for traffic from the wired network to the mobile host under the assumption that downward traffic from the wired network is the major traffic activity. The Partial Acknowledgment and Control Connection protocols are both one-ways enhancements.

In reality, traffic in both directions is subject to transmission errors. In order to fully mitigate the impact of transmission errors, the enhancement should deal with traffic on both directions.

2.5 Intermediate-Link v.s. Last-Hop

Enhancements that work only for wireless links as the last hop of TCP connections are called **last-hop** enhancements. On the other hand, enhancements that work for intermediate wireless links, such as satellite links and those in the ad-hoc networks are called **intermediate-link** enhancements. Here, intermediate links include last-hop wireless links. Since intermediate wireless links also have the performance degradation problem, intermediate-link enhancements are preferred.

Partial Acknowledgment, Control Connection, Snoop and ELN protocols are all last-hop enhancements. Pure link layer protocols based the retransmission of failed packets on

wireless links are intermediate-links enhancements. They can be used regardless of the location of the wireless link.

Another intermediate-link enhancement is the **Delayed Duplicate Acknowledgments** (DDA) [13] algorithm. With the retransmissions in mind, DDA delays the third duplicate ACK, assuming that the missing packet is a wireless loss and is being retransmitted. In case that the missing packet does not arrive after a period, the receiver releases the deferred duplicate ACKs to trigger an end-to-end retransmission. This algorithm does not assume the location of the wireless link, so it is an intermediate-link enhancement.

2.6 Signaling v.s. Hiding

An enhancement is called **hiding** if it attempts to hide wireless losses from upper layers so that TCP code needs no changes. The opposite approach is called **signaling**, which detects and reports the cause of packet losses to TCP layer so that proper recovery actions can be taken.

Hiding is the philosophy of most pure local link layer retransmission enhancements. They assume that Forward Error Correction (FEC) and Automatic Retransmission Request (ARQ) can build a reliable link layer so that upper layers will not know the lossy characteristic of the wireless link. In reality, link layer retransmissions alter the characteristics of the network and may confuse the TCP layer. The retransmission mechanisms in the two layers may response to the same loss events and cause undesirable interaction. Although some studies show that reliable link layer through retransmissions can achieve good TCP performance [7, 8, 9, 10, 11, 12], they also point out that the retransmission schemes were designed for the characteristics of specific TCP connections and transmission error conditions. When the error condition changes or when applied to TCP connections of different characteristics, an undesirable interaction and performance degradation may happen.

Signaling enhancements, on the other hand, report the cause of packet losses to TCP layer so that proper actions can be taken to avoid the undesirable interaction. In this way, signaling enhancements can be applied to different connections and transmission error conditions. Their disadvantage is the need to change the existing TCP code.

2.7 Summary

As analyzed above, end-to-end, local, two-way, intermediate-link, transparent and signaling are desirable characteristics of wireless TCP enhancements. Table 1 is a summary of the major enhancement proposals in the literature. It is found that none of the major enhancements has all the desired characteristics.

3 Congestion Coherence: A New Proposal

Our proposal is to implement local retransmissions on the wireless link and use a scheme based on the coherence of congestion marking to detect the cause of packet losses. The details are described below.

3.1 Assumptions

An important assumption of our proposal is that Explicit Congestion Notification (ECN) is implemented in the wired network. ECN was developed from the binary congestion feedback scheme used in [14] and introduced into IETF by Floyd and Ramakrishnan [15]. It uses two bits in the IP header and two bits in the TCP header for ECN capability negotiation and feedback delivery. When its queue length exceeds a threshold, a router marks the packet as *congestion experienced*. At the destination, the Congestion Experience bit is copied to the *ECN-echo* bit in the TCP acknowledgment and sent back to the sender with the ACK. Upon receiving the ECN-echo, the sender reduces its congestion window to alleviate the congestion.

As a congestion control mechanism, ECN has proved to be more effective than using packet losses to signal the congestion status of the network. It avoids unnecessary packet drops and retransmissions. In RFC 2309 [16], it was recommended to be widely deployed as a router mechanism on the Internet, and was specified in RFC 2481 [17] in 1999. We expect it will be widely supported soon.

3.2 Local Link Layer Retransmission

In our proposed enhancement, local retransmissions are performed in the link layer. All packets transmitted on the wireless link are locally acknowledged before being deleted from the sender's buffer. Packets negatively acknowledged or not acknowledged after a short timer times out will be retransmitted.

Retransmissions of failed packets have higher priority than new packets. This is important to reduce the delay of the retransmitted packets, and minimize the chance of triggering end-to-end retransmissions from the source. One way of implementing the higher priority is to use the "insert from front" strategy. When a packet is detected to be lost, the link layer inserts the failed packet into the front of the transmission queue and transmits it when the medium is available.

The maximum number of retransmissions for a failed packet is configurable. The link layer can either retransmit persistently or stops after the maximum number of retransmissions is reached.

Because of the possibility of successive failures, link layer in-sequence delivery is not supported. Therefore, out-

Table 1. Approaches of existing enhancements

	End-to-end	Local	Two-way	Int.-link	Transparent	Signaling
Retransmission Protocols	√	√	√	√	√	
I-TCP		√	√	√		√
Partial Ack	√					√
Control Connection	√				√	√
Snoop	√	√	√			√
ELN	√	√	√			√
Delayed Dupacks	√	√		√	√	√

Note: Snooping and ELN were first proposed as one-way solutions, but they can be combined to provide a two-way solution.

of-order packets at the destination can imply a congestion loss or a wireless loss.

3.3 Detecting Cause of Packet Losses

In order to distinguish between congestion and transmission errors, the wireless end needs a mechanism to detect the cause of packet losses. For the introduction of the idea, we now assume the mobile host is a TCP receiver.

Out-of-order packets are the indications of packet losses. Both congestion losses and wireless losses cause out-of-order packets and create a holes in the packet sequence number space, but their consequences are different. A hole caused by a wireless loss will be filled when its retransmission arrives, but a hole caused by a congestion loss will not be filled without an end-to-end retransmission from the source. If the receiver knows the hole is a wireless loss, it should wait for the retransmission. Timeout at the source is a way to trigger the end-to-end retransmission, but timeout is usually associated with a prolonged period of idling. A better way to trigger the end-to-end retransmission is through the fast-retransmit. If the receiver knows the hole is a congestion loss, it should send the duplicate acknowledgments right away to trigger the fast retransmit.

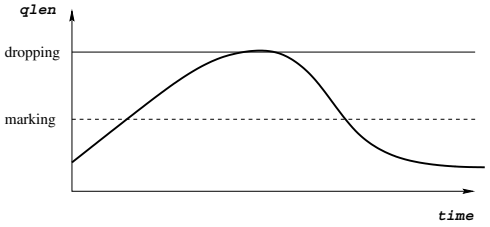

Figure 1. Smooth queue length change

The scheme to determine the cause of packet losses is based on the observation that congestion neither happens

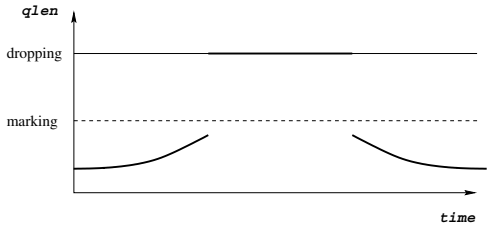

Figure 2. Abrupt queue length change

nor disappears suddenly. Before congestion becomes so severe that a packet has to be dropped, some packets must be marked as "Congestion Experienced" by ECN. Similarly, after a packet is dropped, congestion does not disappear immediately. The queue size falls gradually and some packets are marked. Figure 1 depicts a likely queue length change scenario at the congested router. Between the time that no packet is marked and the time that a packet is dropped, some packets must be marked. An abrupt change depicted in Figure 2 is very unlikely.

As a result, congestion losses are normally preceded and followed by marked packets, see Figure 3. We call this phenomenon **congestion coherence** of ECN marking.

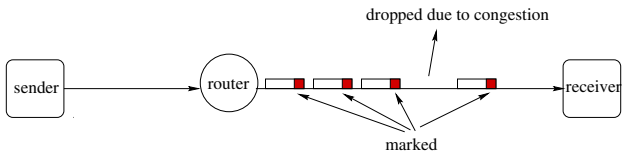

Figure 3. Congestion coherence

The neighborhood of a lost packet is defined by the *coherence context*. There are different ways to define the coherence context, but we find that defining the coherence context of packet n as packets $\{n-1, n+1, n+2\}$ yields

effective results. A packet loss will be considered as a congestion loss if any packet in its coherence context is marked by ECN. In this case, the mobile host responds with duplicate acknowledgments to trigger an end-to-end retransmission and window reduction at the source as specified in RFC 2481 [17].

In contrast to the congestion loss situation, occurrence of transmission errors is normally independent of congestion. If any packet in the coherence context is marked by ECN, congestion control actions are needed to reduce the TCP transmission rate. The chance of having a congestion loss without a marked packet in the coherence context is very small. So when the coherence context contains no marked packet, the mobile host holds the duplicate acknowledgments until the retransmitted packet is received.

The same idea can be applied to the wireless sender case. When the wireless sender receives duplicate acknowledgments, it checks whether the coherence context contains an ECN-Echo. If yes, then the duplicate acknowledgments are most likely caused by a congestion loss, so the sender invokes the congestion control. Otherwise, the duplicate acknowledgments are most likely caused by a wireless loss. The sender ignores duplicate acknowledgments until the local retransmission succeeds.

3.4 Algorithm

In our proposed approach, the modifications to the existing TCP algorithm are made in the wireless end. Based on the technique discussed above, this approach is named *Congestion Coherence* (CC). Figures 4 and 5 show the modified receiving and sending algorithms.

- The TCP sink follows existing algorithm for sending new acknowledgments, first and second duplicate acknowledgments.
- When the third duplicate acknowledgment is to be sent, TCP sink checks whether the coherence context is marked. If yes, the acknowledgment is sent right away. Otherwise, it is deferred for w ms, and a timer is started.
- If the expected packet arrives during the w ms, a new acknowledgment is generated and the timer is cleared.
- If the timer expires, all deferred duplicate acknowledgments are released.

Figure 4. CC receiving algorithm

- The TCP sender follows existing algorithm for sending packets and updating the congestion window upon receiving new acknowledgments, and first and second duplicate acknowledgments.
- When the third duplicate acknowledgment arrives, the sender checks whether any acknowledgment in the coherence context is an ECN-Echo. If yes, the packet corresponding to the duplicate acknowledgments is sent right away and the congestion window is reduced to half if a reduction has not been done in the previous RTT. Otherwise, the sender ignores the duplicate acknowledgement and a timer of w ms is started.
- If a new acknowledgment arrives during the w ms, the timer is cleared and new packets are sent as if the duplicate acknowledgments did not happen.
- If the timer expires, the packet corresponding to the duplicate acknowledgments is sent and the congestion window is reduced to half if a reduction has not been done in the previous RTT.

Figure 5. CC sending algorithm

It should be noticed that the modifications to the receiving and sending algorithms are made on the same end. The Congestion Coherence algorithm at the wireless end hides the lossy characteristic from the other end so no change is needed in the fixed end, intermediate routers and the base station. If the wireless link is in the middle, such as a satellite link or in an ad-hoc wireless network, the modifications can be made on either end.

3.5 Proposal Summary

The proposed enhancement is a transport layer signalling enhancement with link layer retransmissions. By utilizing the congestion coherence of ECN marking, it provides a light-weight TCP enhancement on wireless links. It has all the desirable characteristics discussed in Section 2.

Even though this enhancement needs ECN support in all routers in the wired network, we still consider it as a local enhancement. This is because ECN is a protocol to improve wired networks even though no enhancement for wireless links is needed.

The proposed enhancement also applies to two-way traffic, intermediate wireless links and encrypted traffic.

4 Performance Comparison

In order to evaluate the performance of the proposed Congestion Coherence enhancement, we performed a set of simulations with the ns simulator [18], and the results are compared with several known enhancements.

4.1 Simulation Model

The simulations are performed on the simplified network model shown in Figure 6, where s_1, s_2 are the sources and d_1, d_2 are the destinations. The link between intermediate routers r_1 and r_2 is the bottleneck link. The link between r_2 and d_1 is a wireless link. The numbers beside each link represent its rate and delay.

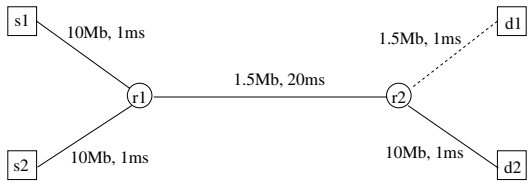

Figure 6. Simulation model

The experiment traffic is an FTP session from s_1 to d_1 using TCP Reno as the transport protocol. A UDP flow from s_2 to d_2 generated by an exponential on-off model is used as the background traffic. The mean burst period and the mean silence period are 100 ms, and the burst data rate is 500 kbps. Both TCP and UDP packet sizes are 1000 bytes, and TCP acknowledgments are 40 bytes long.

Packets transmitted on the wireless link are subject to random transmission errors. The raw packet error rate varies from 0.001 to 0.1. Considering the packet size of 1000 bytes, the bit error rate is roughly 10^{-7} to 10^{-5}. For transmission systems that use FEC, this bit error rate should be the residual error rate of FEC.

Link layer retransmissions are implemented on the wireless link. Packets sent but not acknowledged at the link layer in 40 ms are resent. Retransmitted packets have high priority than new packets, but they are also subject to transmission errors at the same rate. The waiting time w at the receiver is set at 81 ms so that packets delivered within two retransmissions are accepted.

The marking policy of the Random Early Detection (RED) algorithm [19] has a big impact on the congestion coherence. We found that actual queue length and a deterministic marking region provide better coherence than average queue length and random marking. This can be done by configuring the queue weight as 1 and choosing a th_{max} smaller than the buffer size.

To reflect the steady state measurement, the simulation time should be long enough to minimize the effect of the initial transient state. In our experiment, we tried various simulation time and found the results of 500 seconds show the essential features without noticeable differences from longer simulations, so all aggregate measurements are collected from 500-second simulations.

Our proposal is compared with base TCP with retransmissions, DDA, Snoop and Congestion Coherence. When the mobile host is a sender, Snoop is replaced by ELN. To show that ECN without congestion coherence does not work, we also compared plain ECN. We did not compare with I-TCP, Partial Acknowledgment and Control Connection because they are not end-to-end or not local.

We experimented with various network configurations, including wireless link as the last hop (mobile receiver), as the first hop (mobile sender) and as an intermediate link. Congestion Coherence works for all three configurations and has a similar performance. The performance comparisons presented below are for the mobile receiver configuration.

4.2 Performance Results

Our first group of results, shown in Figure 7, is the TCP congestion window and queue length of each proposal. They are collected from 40-second simulation traces. The packet error rate in the simulation is 0.1, corresponding to a bit error rate of roughly 10^{-5}. A calculation shows that the delay and bandwidth the wireless connection can support a window size of about 10 packets, but as shown in the figure, the window size of base TCP with retransmissions and ECN is reduced frequently. Their corresponding queue size graph shows the queue at the bottleneck link is almost always empty. Therefore, their link efficiency is very low. Snoop and DDA solve the problem of unnecessary window reductions caused by transmission errors. The window size is significantly increased and the bottleneck link is better utilized. Nevertheless, the spikes in the bottom of Snoop and DDA cwnd figure indicate these two methods suffer severe degradation from timeouts. Congestion Coherence is a thorough solution. Unnecessary window reductions and timeouts are avoided. The queue size figure shows that Congestion Coherence has high link efficiency.

The major metric to evaluate the enhancement proposals is *goodput*, which is defined as the number of packets successfully received and acknowledged by the mobile host, excluding the retransmitted packets. The goodput of the five proposals under different packet error rates is drawn in Figure 8. Base TCP performs reasonably well when the packet error rate is very small, but as the packet error rate increases, its performance degrades quickly. The performance curve confirms that TCP needs enhancement on wireless links.

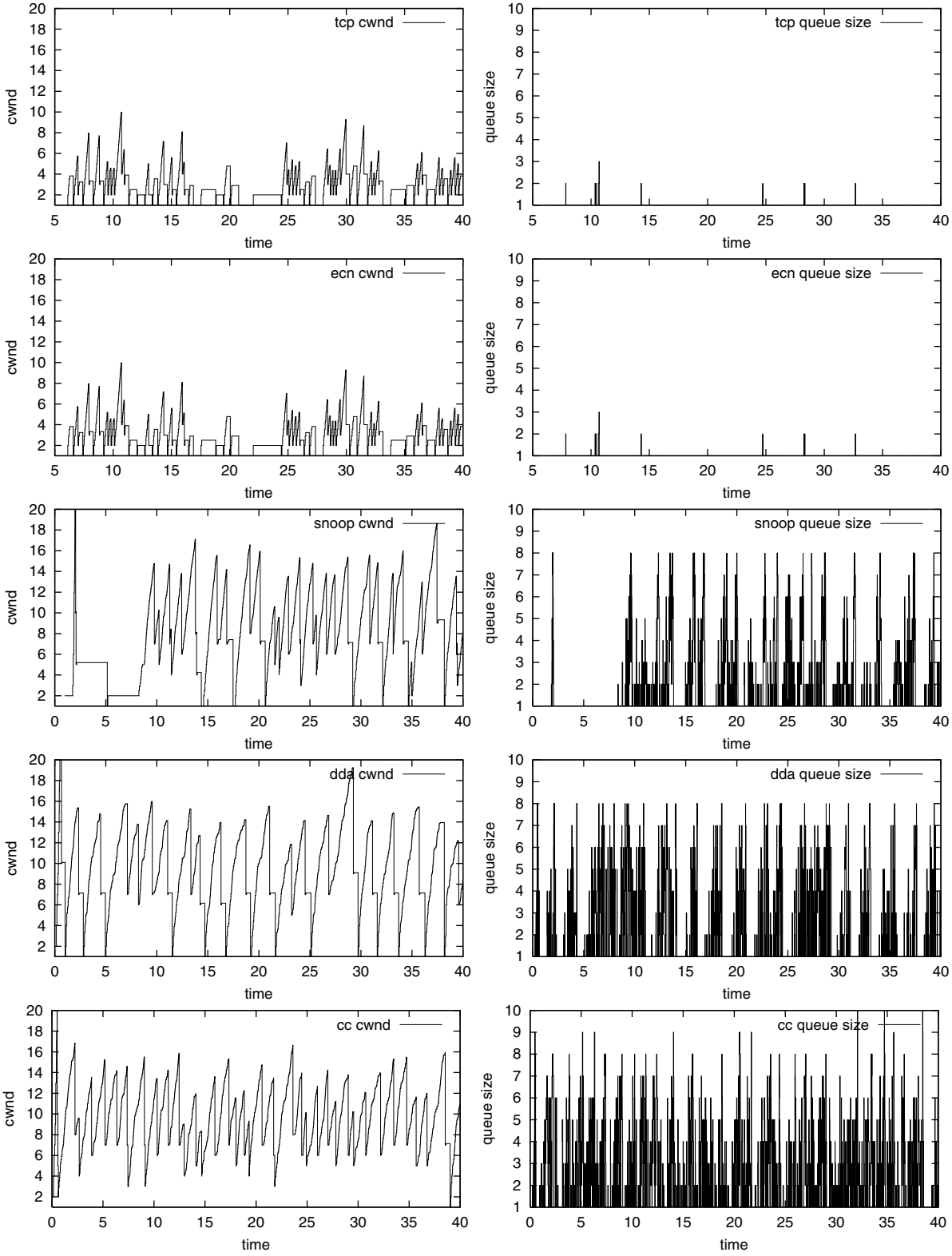

Figure 7. Congestion window and queue length for base TCP, ECN, Snoop, DDA and CC

Plain ECN performs better than base TCP with retransmissions when the error rate is very small, but its performance degrades quickly as the error rate increases. DDA does not degrade much with the error rate, but its performance under small error rates is low. Congestion Coherence proves to be better than all other methods for all ranges of error rates.

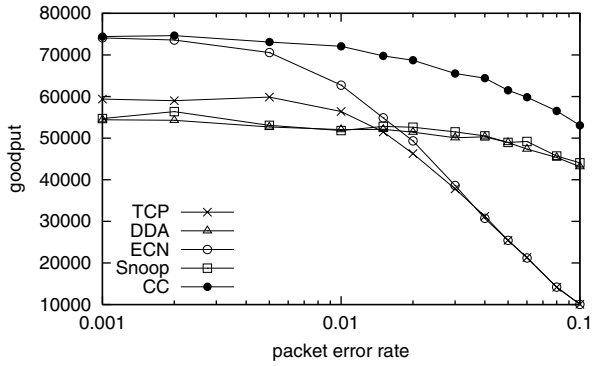

Figure 8. Goodput for the five methods

In addition to the goodput, we also analyzed the simulation trace and collected other data that helped us understand why one enhancement works better than another.

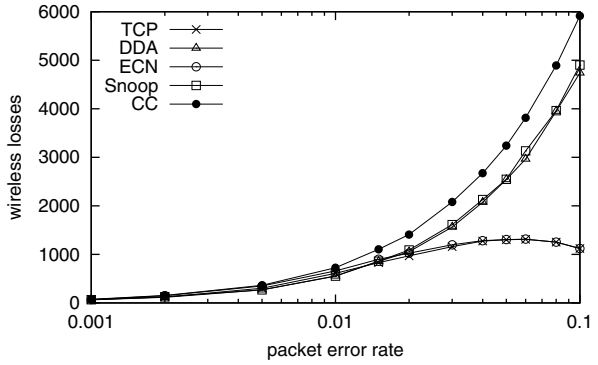

Figure 9. Wireless losses

Figures 9 and 10 show the number of wireless and congestion losses. The number of wireless losses equals the total number of packets transmitted on the wireless link times the packet error rate. The numbers of congestion losses of base TCP with retransmissions, Snoop and DDA are significantly more than other methods because they use packet losses as a congestion control mechanism. As the packet error rate increases, wireless losses reduce the congestion window so frequently that the window seldom grows to the level that a packet needs to be dropped. This explains the smaller number of congestion losses of base TCP with retransmissions in the right half of Figure 10. In contrast,

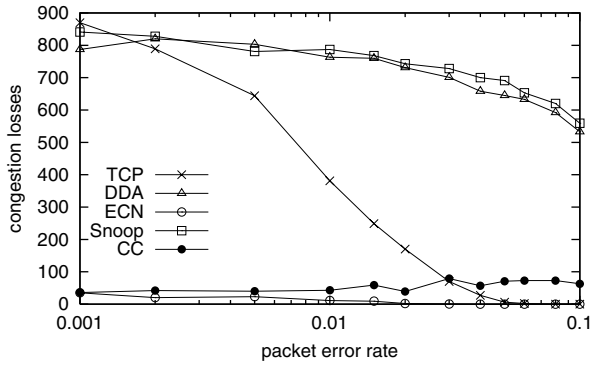

Figure 10. Congestion losses

methods using ECN do not suffer from congestion losses on a regular basis. Congestion losses happen only when bursts of background traffic generate so many packets that the buffer of bottleneck link cannot absorb. As analyzed in the beginning of Section 3, having fewer congestion losses helps to reduce end-to-end retransmissions and the chance of timeout.

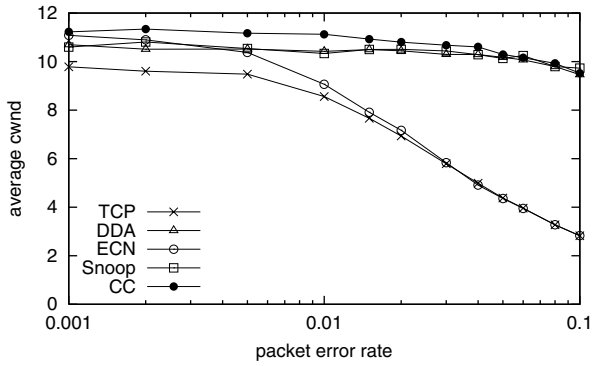

Figure 11. Average congestion window size

Figure 11 shows the average congestion window size. As the packet error rate increases, wireless losses cause unnecessary window reductions in TCP and plain ECN, but the window size of Snoop, DDA and Congestion Coherence is not affected much by transmission errors. The slight drop in the right upper corner is caused by transmission errors in the retransmit packets. This figure confirms that Snoop, DDA and Congestion Coherence solve the problem of unnecessary window reductions.

Figure 12 shows the number of timeouts that occurred during the simulation period. When the packet error rate is small, TCP, Snoop and DDA have the largest number of timeouts because they use packet losses for congestion control. Their buffer occupancy at the bottleneck link can grow

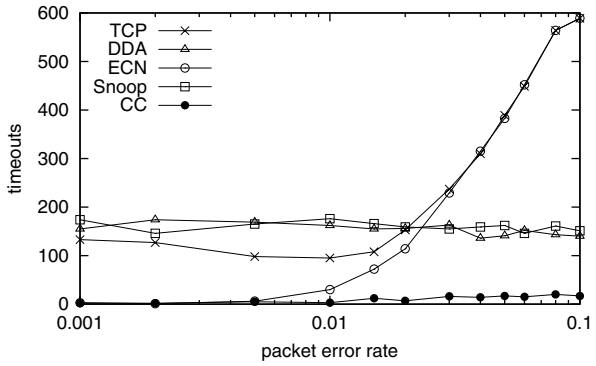

Figure 12. Number of timeouts

so high that bursts in background traffic can cause continual losses. Since two or more losses in a window causes a timeout. This translates to a large number of timeouts. ECN and Congestion Coherence have very few timeouts because most of their congestion losses are avoided. Background traffic causes occasional losses, but seldom become multiple losses in one window. As the packet error rate increases, the number of timeouts in TCP and plain ECN increases dramatically because a larger number of wireless losses increase the chance of multiple losses in one window. When the error rate is below 0.014, TCP has more timeouts than plain ECN. As the congestion window of TCP is reduced frequently by wireless losses (Figure 11) and congestion losses become fewer (Figure 10), TCP behaves almost identical to to ECN. The timeouts of Snoop and DDA are mainly caused by congestion losses. They remains constant for all packet error rates. Congestion Coherence has the smallest of all proposals. This figure is the evidence showing that only our proposal avoids the degradation caused by timeouts.

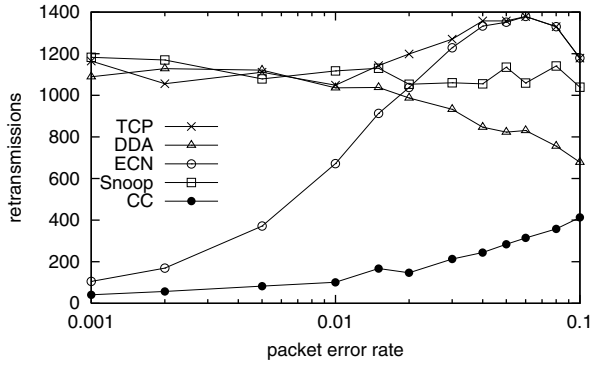

Figure 13. End-to-end retransmissions

Figure 13 shows the number of end-to-end retransmissions. This number depends on the number of congestion losses, wireless losses and timeouts, as well as the enhancement method used. In fact, congestion losses in all methods are retransmitted. Wireless losses in TCP and plain ECN are retransmitted. When timeout happens, one full window of packets are retransmitted. Snoop and DDA avoid the majority of end-to-end retransmissions of wireless losses, but they still have a large number of retransmissions because of congestion losses and timeouts. Plain ECN reduces congestion losses, but cannot recover from wireless losses. All its wireless losses are retransmitted. Congestion Coherence avoids the majority of congestion losses, and waits for the local retransmission for wireless losses, so it has the smallest number of retransmissions.

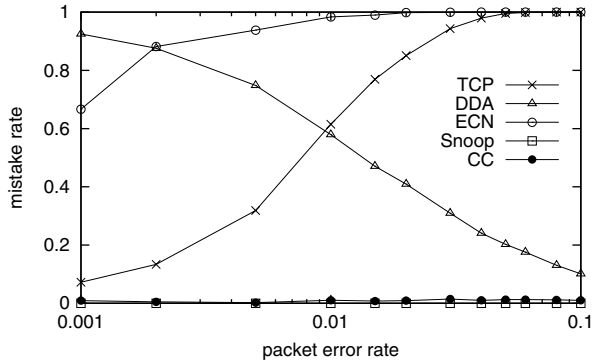

Figure 14. Mistake rate

Finally, the mistake rate in determining the cause of packet losses is shown in Figure 14. Both base TCP with retransmissions and plain ECN assume all losses are caused by congestion, so their mistake rate is the percentage of wireless losses in all losses. Plain ECN makes almost the same number of mistakes as base TCP with retransmissions, but it has a much higher mistake rate because of its small number of congestion losses. DDA assumes all packet losses are due to transmission errors, so its mistake rate decreases when the packet error rate increases. Snoop knows the exact cause of all packet losses by monitoring packets arriving at the base station, so its mistake rate is zero. Congestion Coherence takes advantage of congestion coherence and makes the right guess in most cases, but it makes mistake when very bursty traffic causes sudden packet losses without having neighboring packets marked. In our simulations, Congestion Coherence's mistake rate ranges from 0.06% to 1.2%. This rate is very small compared to other methods (except Snoop), and has a minimal impact on performance.

In summary, the simulation results show that Congestion Coherence avoids the majority of congestion losses and is able to distinguish wireless loss from congestion losses.

It is the only enhancement that avoids the three degradations of TCP performance over wireless links — end-to-end retransmissions, unnecessary window reductions and timeouts. Therefore, the performance of TCP is improved to a level that current enhancements cannot achieve.

5 Conclusion

In this paper, we summarize the approaches used by existing wireless TCP enhancement proposals and analyze their impacts on the feasibility, generality and performance. Our proposed new enhancement makes use of the congestion coherence between consecutive packets to determine the cause of packet losses. It requires only local changes and applies to a wide range of network configurations and traffic scenarios. By using local link layer retransmissions and signaling the cause of packet losses, this method eliminates the majority of end-to-end retransmissions, unnecessary window reductions and timeouts caused by transmission errors, and achieves a high level of performance.

References

[1] R. Jain, A timeout-based congestion control scheme for window flow-controlled networks, *IEEE Journal on Selected Areas in Communications*, Vol. SAC-4, No. 7, pp. 1162-1167, October 1986.

[2] A. Bakre, B. R. Badrinath, I-TCP: indirect TCP for mobile hosts, *Proceedings - International Conference on Distributed Computing Systems*, Vancouver, Canada, pp. 136–146, 1995.

[3] S. Biaz, N. Vaidya et al, TCP over wireless networks using multiple acknowledgements, *Texas A&M University, Technical Report 97-001*, www.cs.tamu.edu/faculty/vaidya/papers/mobile-computing/97-001.ps, January 1997.

[4] S. Banerjee and J. Goteti, Extending TCP for wireless networks, University of Maryland, College Park, www.cs.umd.edu/users/suman/docs/711s97/711s97.html.

[5] H. Balakrishnan, S. Seshan, and R. H. Katz; Improving reliable transport and handoff performance in cellular wireless networks; *Wireless Networks*, 1, 4, pp. 469 – 481, February 1995.

[6] H. Balakrishnan and R. H. Katz; Explicit loss notification and wireless web performance, in *Proceedings of IEEE Globecom 1998*, Sydney, Australia, November, 1998.

[7] G. Xylomenos, Multi Service Link Layers: An approach to enhancing internet performance over wireless links, *PhD dissertation at University of California, San Diego*, 1999.

[8] P. Bhagwat, P. Bhattacharya, A. Krishna, S. K. Tripathi, Using channel state dependent packet scheduling to improve TCP throughput over wireless LANs, *ACM/Baltzer Wireless Networks Journal*, Vol. 3, No. 1, January 1997.

[9] D. A. Eckhardt, P. Steenkiste, Improving wireless LAN performance via adaptive local error control, *Proceedings of IEEE ICNP 98*, 1998.

[10] R. Ludwig, A. Konrad, A. D. Joseph, Optimizing the end-to-end performance of reliable flows over wireless links, pp. 113-119, *Proceedings of ACM/ IEEE MOBICOM 99*.

[11] M. Meyer, TCP Performance over GPRS, *Proceedings of IEEE WCNC 99*.

[12] R. Ludwig, R. H. Katz, The Eifel algorithm: making TCP robust against spurious retransmissions, *ACM Computer Communication Review*, Vol. 30, No. 1, January 2000.

[13] N. H. Vaidya, M. Mehta, C. Perkins, G. Montenegro, Delayed duplicate acknowledgements: a TCP-unaware approach to improve performance of TCP over wireless, *Technical Report 99-003*, Computer Science Dept., Texas A&M University, February 1999.

[14] K. K. Ramakrishnan and R. Jain, A binary feedback scheme for congestion avoidance in computer networks, *ACM Transactions on Computer Systems*, Vol. 8, No. 2, pp. 158-181, May 1990.

[15] S. Floyd, TCP and explicit congestion notification, *ACM Computer Communication Review*, V. 24 N. 5, p. 10-23, October 1994.

[16] B. Braden et al, Recommendations on queue management and congestion avoidance in the Internet, *RFC 2309*, April 1998.

[17] K. Ramakrishnan and S. Floyd, A proposal to add explicit congestion notification (ECN) to IP, *RFC 2481*, January 1999.

[18] UCB/LBNL/VINT Network Simulator - ns (version 2), http://www-mash.CS.Berkeley.EDU/ns/.

[19] S. Floyd and V. Jacobson, Random early detection gateways for congestion avoidance, *IEEE/ACM Transactions on Networking*, Vol. 1, No. 4, pp. 397-413, August 1993.