Improving TCP Performance over Networks with Wireless Links†

Shin-Yi Tsai and *Chunhung Richard Lin*

Department of Computer Science and Information Engineering, National Chung Cheng University
Email: ivantsai@tw.ibm.com, chlin@acm.org

Abstract - TCP is now a widely used Internet protocol in the most of networks. All systems which want to communicate with other computers in the Internet must follow the rules of TCP. It is a reliable transport layer protocol, responsible for the end-to-end performance in the network, and it performs well in traditional networks made up of links with low bit-error rates. But when networks contain high bit-error rate links such as wireless links, TCP will degrade end-to-end performance caused by packet error in wireless link. At this time, TCP can not distinguish weather the packet loss is caused by congestion or the data error in wireless links. TCP is impractical in networks that contain wired and wireless links, and especially wireless links are part of backbone networks. we will present a novel scheme in base stations to achieve good performance without modifying the existing TCP protocol. This new scheme is quite simple in which we only add some local acknowledgments and a checksum module to guarantee the correction of packets in wireless links. The simulation results show that our scheme has better performance than the pure TCP. Especially, in high bit-error rate, our scheme provides very good performance.

1. INTRODUCTION

Wireless networks grew rapidly in the last years due to advancement of communication technologies. Wireless media has many different properties from the existing wired network, such as high bit-error rate, limited bandwidth, high latency movement of data, etc. TCP assumes that network links rarely lose packets or cause error because of high channel quality [3]. All the loss of packets in network is assumed to be only caused by congestion. If TCP sender find packets lost, it will reduce its send-packet rate (sliding window) and initiate congestion avoidance or control mechanisms to relieve congestion. Such operations in TCP sender make network traffic from heavy load to light load [11]. It seems fine when there exists a wireless link in the path from source to destination and the channel quality of this wireless link is always good. In such a situation, TCP can still yield high performance in this TCP connection. But this wireless link is not always to have good channel quality. It is quite dependent on the environment situation. When bit-error rate of wireless link is high, the sender can not distinguish that the loss of packets is caused by congestion or the other reasons, such as bad channel quality of wireless link. If the loss of packets is always due to packet error in wireless link, the window of sender will never reach its optimal value, and also make TCP sender do unnecessary congestion control or avoidance mechanisms, so as to waste bandwidth utilization of each TCP connection.

† This work was in part supported by the National Science Council, Taiwan, R.O.C., under Contract NSC-86-2213-E-194-024-T "QoS Support for Wireless, Mobile, Multimedia Networks".

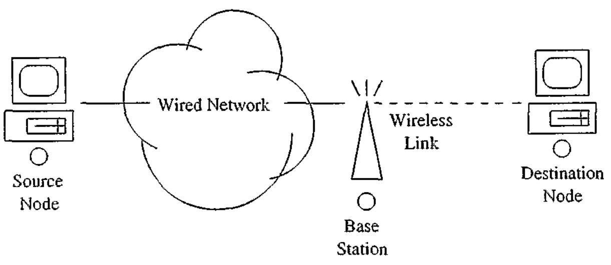

Figure 1: Networks topology of cellular wireless networks

We know there are several schemes have been proposed to solve such kind of problems to yield good performance when wireless links have high bit-error rates [3, 5, 8, 16-18]. Some papers also make comparisons of these schemes [4]. These schemes use a variety of mechanisms such as local retransmission (snoop), split-TCP method (I-TCP) [2, 3]. It is notable that all of these schemes only modify some functions in base stations or mobile hosts instead of the existing TCP in traditional wired (fixed) networks. This makes the wireless network easier to be integrated with the wired network. However, these schemes all consider the same network model: the wireless link is either on the first or on the last hop of a TCP connection. That is, the only wireless link connects a base station and a mobile host as shown in Figure 1. Thus, only one connection of a source-destination pair passes through the wireless link. However, the network model we discuss in this paper is that the wireless links (maybe more than one) are parts of the backbone network. That is, a wireless link is not on the first or on the last hop of a TCP connection and this wireless link may contain several TCP connection pairs. In Figure 2, this wireless link is the connection between two base stations instead of between a base station and a mobile host. Under this network environment, packet error in wireless links may cause more serious problems than in traditional networks. Those schemes we described above may not work as fine as in the original model they presented.

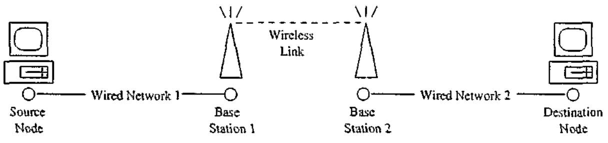

Figure 2: Networks topology discussed in this paper

In this paper we consider a network topology in which there are two separate LANs or two WANs. Both LANs (WANs) can not communicate with each other by using wired links. For example, such two LANs(WANs) may be separated by a inlet, chasm, etc. In order to make a connection between both LANs (WANs), we may use a wireless link (Figure 2). In order to make both LANs yield high performance when channel quality of the wireless link is bad, we design a simple method to improve the end-to-end performance without changing the existing TCP protocol in fixed hosts. We just add some modules between transport layer and data link layer, and some databases in both base stations. The sender need not initiate the congestion control or congestion avoidance mechanisms if the base stations can retransmit the lost or error packet in time in wireless link. Thus, the performance of this TCP connection between sender and receiver can be improved.

This paper is organized as follow. In section 2 we describe the network model and point out the problems when using TCP protocol to transmit packets. In section 3 we will describe our method in detail. Our simulation results by comparing with the normal TCP is shown in section 4. Section 5 draws some conclusions.

2. OUR PROTOCOL MODEL

We consider a network topology, as in Figure 2. There are two wired LANs or WANs, says WN1 and WN2. Between both wired networks, we use a wireless link to connect WN1 and WN2. We assume all source nodes in WN1 and all destination nodes in WN2. We can say this wireless link is part of the backbone network. Any path between a computer in WN1 and another computer in WN2 has to pass through the wireless link. Suppose that we use the pure TCP in this network environment, the most interesting thing we want to know is that if TCP performs well in such a network environment.

In addition to limited bandwidth, wireless media has a very special characteristic different from wired one, i.e. high bit-error rate. It always has periodic bit errors or burst errors in data transmission. So if we send packets from source node to destination node, we can not guarantee that all packets received by destination node are correct. TCP checksum will check the correctness of the packet. If destination node find that errors occur and can not be recovered, the packet will be dropped and the acknowledgment will not be sent.

Source node uses the acknowledgments it received to determine which packets have been received by destination node. The source has to retransmits the packet when timeout occurs. When timeout is detected, TCP sender (source) will initiate congestion control or avoidance mechanisms, update its round-trip time of TCP packets (so does measurement of timeout), decrease its window size and then initiate slow start method. In traditional wired network environment, the purpose of these serial operations in source node is to decrease the traffic load in the network. The loss of packets is only assumed to be caused by congestion. With wireless links, if the lost packet is not caused by congestion but by packet error in wireless link, all congestion control or avoidance mechanisms in source node are unnecessary. Unfortunately, TCP can not distinguish both situations. Therefore, not only the optimal throughput is never reached, but also the bandwidth of network is wasted.

For the above example, we know that there are some problems with TCP when a connection contains both wired and wireless links. We have to investigate solutions to solve this problem. Any acceptable solution for the wireless network environment should inter-operate with the existing infrastructure and should not make any modifications to the fixed networks [13]. That is, the solution for the integration of wireless media (network) with the existing traditional network must be transparent to all applications and protocols that are run in fixed network environment [15].

3. The Local Acknowledgment Method

3.1. Overview of Our Method

We add some modules between link layer and transport layer and two types of database, packet pool and packet buffer, in base station 1 (BS1) and base station 2 (BS2), respectively. In BS1, packet_monitor_modules and ack_monitor_module are added to monitor all packets in packet pool, and to deal with local acknowledgments from BS2. In BS2, we put packet_check_module to check packets from BS1 and send local acknowledgments to BS1. In addition, there is a packet pool in BS1 to store packets temporarily from Wired Network 1, and there is a packet buffer in BS2 to store out-of-sequence packets from wireless link. Figure 3 shows our solution.

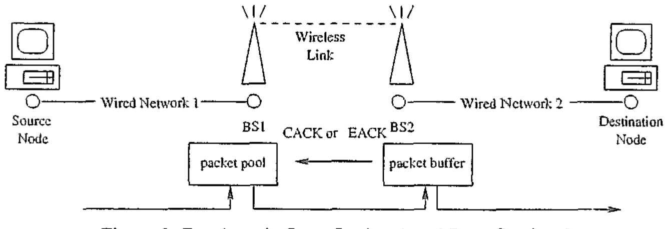

Figure 3: Database in Base Station 1 and Base Station 2

All packets which will pass through the wireless link are buffered in BS1. Here we define two types of local acknowledgments in wireless link, EACK and CACK, which are sent by BS2. All packets from BS1 to BS2 will be checked to make sure if they are correct. If the packet contains no error or contains error which can be recovered by BS2, then BS2 will forward this packet to wired network 2 (WN2) and send CACK to BS1. Otherwise, if the packet contains error and can not be recovered, BS2 will drop it and send EACK to BS1. When BS1 receives CACK from BS2, ack_monitor_module will remove the packet from the packet pool. When it gets EACK from BS2 or timeout occurs, it will retransmit the packet to BS2.

3.2. Modules and Database in Base Station 1

We define four hosts called source node, destination node, base station 1 (BS1), and base station 2 (BS2). There are two wired networks called WN1 and WN2, and a wireless link (see Figure 2). We first assume that all the packets are sent by the source node to the destination nodes. All packets have to pass through the wireless link. All fixed hosts use the original TCP. The new modules and databases are put in BS1 and BS2.

We add two modules and a packet pool in BS1. The two modules are packet_monitor_module and ack_monitor_module. The packet_monitor_module is in charge of dealing with the packets from WN1. The ack_monitor_module is responsible for the local acknowledgments from BS2. These two modules also manage the packet pool of BS1.

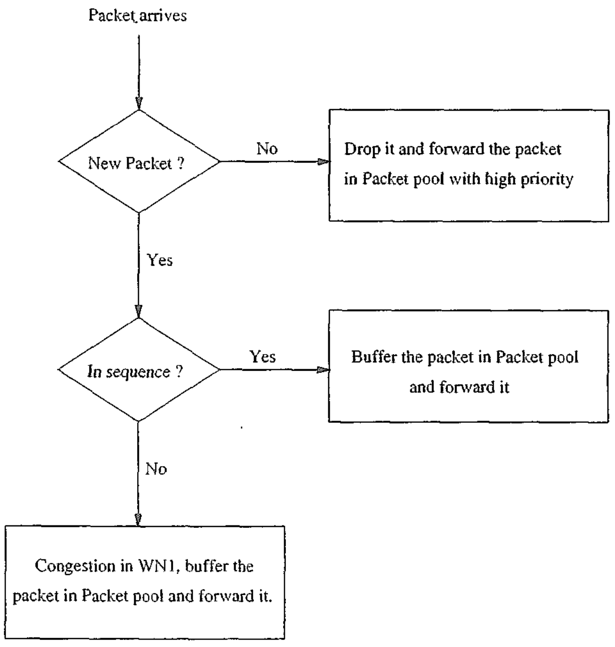
Packet arrives
New Packet ?
No
Drop it and forward the packet in Packet pool with high priority
Yes
In sequence ?
Yes
Buffer the packet in Packet pool and forward it
No
Congestion in WN1, buffer the packet in Packet pool and forward it.

Figure 4: The flow chart of packet_monitor_module in BS1

packet_monitor_module (Figure 4): This module deals with the packets from WN1. Besides forwarding packets to wireless link, the module also makes a copy in packet pool, and records the sequence number of the last packet it received by using an additional data structure in BS1. By the information of sequence number, BS1 can know which packet is lost in WN1 by congestion. Besides the normal sequence number of TCP, all packets they may be from different connections received by BS1 will be given an unique "local packet serial number" if they want to pass through wireless link to BS2. With this unique number for each packet, BS2 can detect the lost packet in wireless link by finding the out-of-order packet from wireless link.

There are several kinds of packets received by BS1 and packet_monitor_module deals with them in different ways:

(1) Get a new packet in normal sequence: This is a normal case. Congestion would not happen in WN1 and the network traffic in WN1 is stable. When BS1 gets such a packet, it will forward the packet to wireless link and copy the packet into packet pool. Packet pool is a special data structure in BS1. It is a temporary packet buffer for wireless retransmission while a packet has errors or is lost. Finally, BS1 will set the clock for the local round-trip time of the packet which is forwarded to wireless link. It can help BS1 to detect packet loss.

(2) Get a new packet but out-of-sequence: It means that some packets are lost due to congestion in WN1. So BS1 receives discontinuous packets. Like (1), BS1 will copy and forward this packet to wireless link, and set the clock for the local round-trip time of this packet.

(3) Get a duplicate packet which was received by BS1 before: It means that the source node detect timeout of the packet, and make a retransmission. This may be caused by congestion in WN1 or WN2, or caused by packet loss in wireless link. In this case, packet_monitor_module will put the packet it receives into packet pool no matter whether the packet is still in packet pool or not. The new packet will replace the old packet if the old one is still in packet pool. Then BS1 forwards this packet to wireless link with higher priority.

ack_monitor_module (Figure 5): This is another module in BS1. ack_monitor_module is in charge of dealing with the local acknowledgments from BS2. As we describe above, there are two kinds of local acknowledgments received by BS1, CACK and EACK.

(1) Get CACK from BS2: It means that the packet arrives by BS2 correctly. So BS2 send CACK to BS1. In such case, BS1 will drop the packet from packet pool.

(2) Get EACK from BS2: It means that the packet arrives by BS2 with errors due to bad channel quality of wireless link. In such situation, BS1

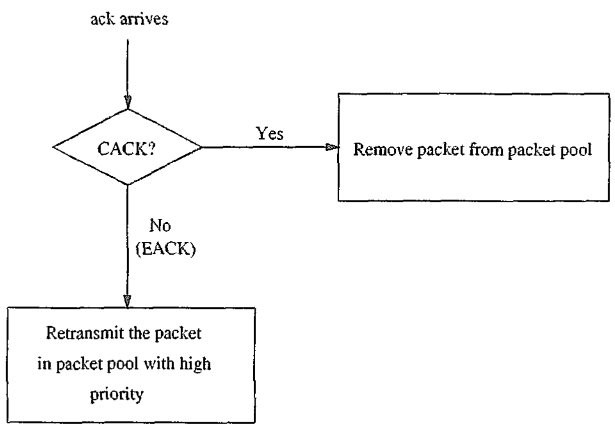
ack arrives
CACK?
Yes
Remove packet from packet pool
No
(EACK)
Retransmit the packet in packet pool with high priority

Figure 5: The flow chart of ack_monitor_module in BS1

will retransmit the packet with higher priority.

(3) Timeout of local acknowledgment: When waiting time of a local acknowledgment exceeds the maximal response time, BS1 will retransmits the packet to wireless link with higher priority. The operation is the same as (2).

3.3. Modules and Database in Base Station 2

In base station 2 we add packet_check_module (Figure 6) to deal with the packets from wireless link. According to the result of the checksum, BS2 will send CACK or EACK to BS1. When a received packet is correct, BS2 will send CACK to BS1. Otherwise, BS2 will send EACK to request retransmission. In addition, we use a variable number, p_number, to represent the next local packet serial number which BS2 will receive from wireless link. Those packets whose local packet serial numbers are larger than p_number must be put in packet buffer and are not forwarded to WN2. BS2 can only forward the packet whose local packet serial number is equal to p_number. packet_check_module works as following:

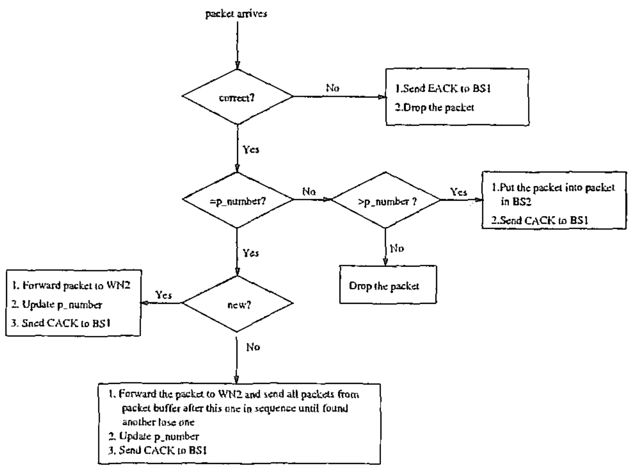

Figure 6: The flow chart of packet_check_module in BS2

(1) Get an error packet: BS2 will drop this packet and send EACK to BS1 to request retransmission.

(2) Get a correct new packet but its local packet serial number is larger than p_number: It will be put into packet buffer and BS2 will send a CACK to BS1.

(3) Get a correct packet but its local packet serial number is less than p_number: This means that the packet was received correctly before. CACK for the packet may be lost or be damaged in wireless link. If CACK is lost in wireless link, BS1 would detect timeout and retransmit the packet. At this time, BS2 will do nothing but only drop the packet and send CACK again.

(4) Get a correct packet, and its local packet serial number is equal to p_number: This is the normal case in BS2. BS2 will forward it to WN2 and increase p_number by 1. In addition, BS2 will send CACK to BS1. If packet buffer is not empty, BS2 gets other packets from packet buffer, forwards them to WN2 according to local packet serial number until another out-of-order packet found. p_number is increased by one at each forwarding.

4. PERFORMANCE

We do some experiments to show that our method is suitable for the wireless environment. This simulator follows all features of TCP, including congestion control algorithm, retransmission for timeout, evaluation of round-trip time and flow control of sliding window. In our simulation, we have two wired networks and a wireless link. BS1 and BS2 are two intermediate nodes. All packets are sent from the source node to the destination node. The wired networks are WN1 and WN2. Each has a bandwidth of 10 Mbps. The bandwidth of wireless link is 2 Mbps. The maximal window size of each TCP connection is 50. Propagation time of wired and wireless link is 4 msec and 20 msec, respectively. The length of data packet and acknowledgment are 4 kbits and 200 bits.

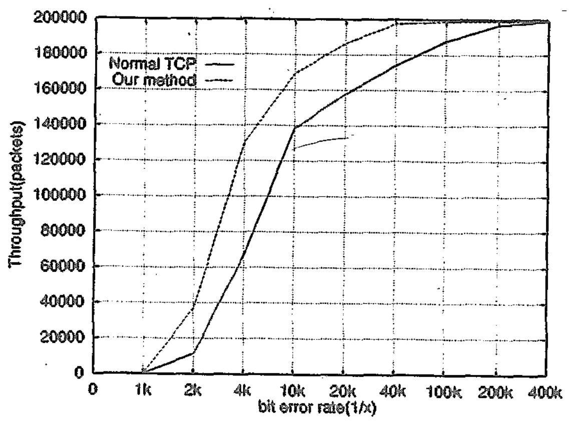

Figure 7: The total packets received by destination node with different bit-error rate

Figure 7 shows the throughput of normal TCP and our method with different bit-error rates. The total simulation time is 200 sec. The y-axis shows the total number of packets arrive destination node, and the x-axis shows the different bit-error rates. From Figure 7, we know that when wireless link has very high bit-error rate, our method has no significant advantage as compared with normal TCP. Consider the case of bit-error rate 1/4k (0.025%) for example. Most of packets are damaged at BS2, and must be retransmitted twice or more by BS1. Thus, the source node always detects timeout of almost all packets and starts unnecessary congestion and avoidance mechanisms. This results in the performance of this TCP connection to be very poor. However, our method can improve the performance. In Figure 7, there are about 65000 packets received by destination node when normal TCP is used, and about 130000 packets received by running our method in base stations. The throughput is increased about 2 times. When bit-error rate is decreased to 1/20k, 1/40k, 1/100k, our method still performs well. It is notable that at very high bit-error rate (i.e. more than 1/1k) or very low bit-error rate (i.e. less than 1/400k) our method almost has the same performance as the normal TCP.

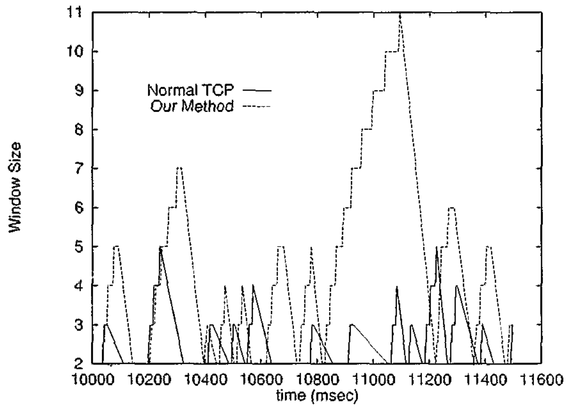

Figure 8: The window size change with bit-error rate = 1/4k

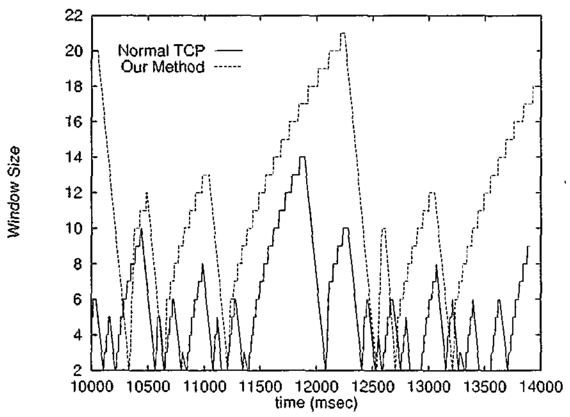

Figure 9: The window size change with bit-error rate = 1/10k

Figure 8 to Figure 12 show the window size change with different bit-error rates (1/4k, 1/10k, 1/20k, 1/40k and 1/100k). As shown in these figures, when bit-error rate is less than 1/20k, the window size will increase as time goes by, and it can reach maximal window size (i.e. 50) or go down to 0 (when packet loss is detected). Each decrease-window event may be caused by congestion in wired networks or caused by packet error or loss in wireless link. When bit-error rate is 1/10k (Figure 9), the maximal window size of normal TCP can only reach 14. In such situation, the maximal window size of our method is up to 21. Because of high bit-error rate (1/10k) in wireless link, our method can not deal with all lost packets in time by fast retransmission in BS1 such that it is hard to reach the optimal window size (50). When bit-error rate is 1/20k or less, the window size of our method can reach the optimal value 50 while the maximal window size of normal TCP is only 35 and average window size is 15.

Figure 13 to Figure 17 show the total packets which destination node receives. Obviously, our

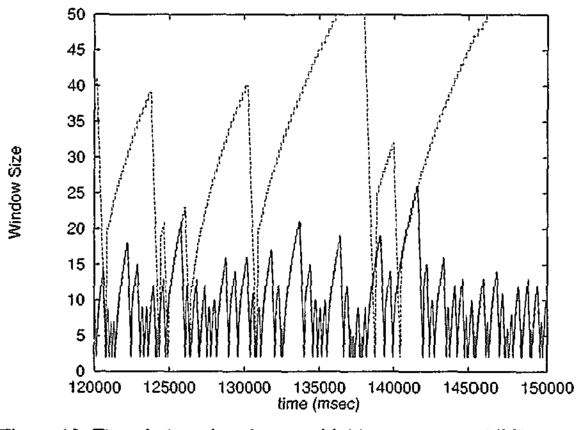

Figure 10: The window size change with bit-error rate = 1/20k

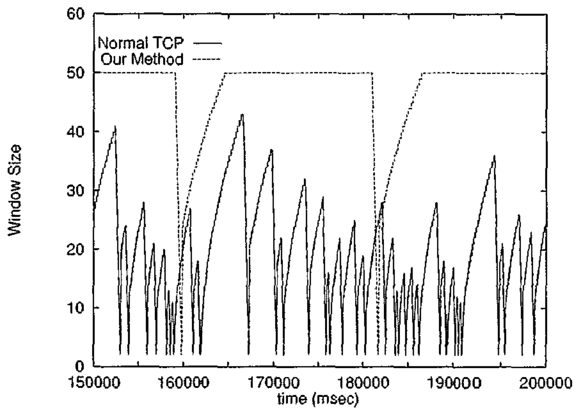

Figure 11: The window size change with bit-error rate = 1/40k

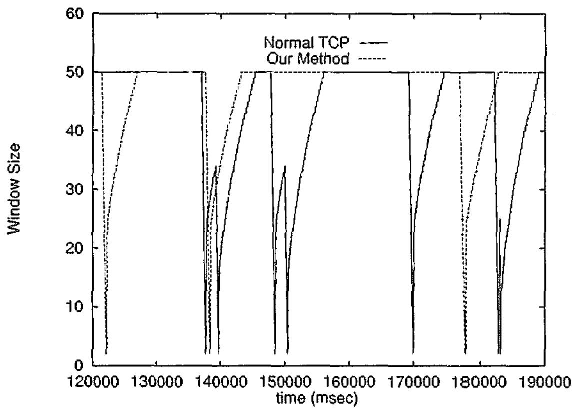

Figure 12: The window size change with bit-error rate = 1/100k

method can receive more packets than normal TCP. It is due to our fast retransmission in BS1 such that the damaged packet detected by BS2 can be retransmitted in time. With high bit-error rate, destination node receives more damaged packets and source node detects more timeout of packets when TCP is run. Such events may start unnecessary congestion control or avoidance mechanism. For our method, BS2 will find such kind of packets, do necessary operations to recover error packets, and make sure that all packets will be sent to WN2 correctly. This can make the number of error packets as few as possible. Especially when bit-error rate of wireless link is high (e.g. 1/4k), our method performs significantly better than original TCP.

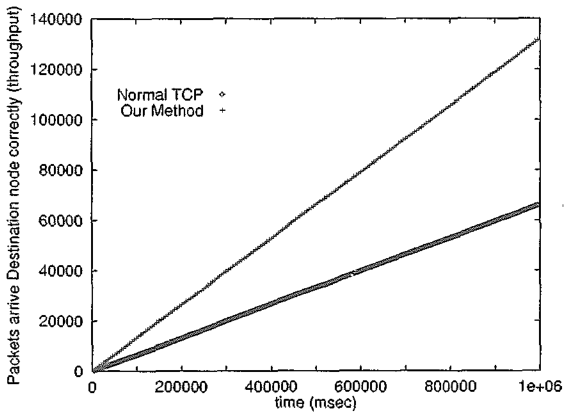

Figure 13: Throughput change with bit-error rate = 1/4k

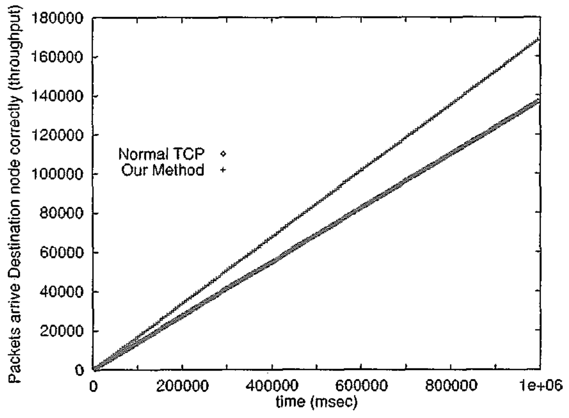

Figure 14: Throughput change with bit-error rate = 1/10k

5. CONCLUSION

Wireless media is known to have high bit-error rates, so as to lead to significant packet loss. TCP is a reliable transport protocol which performs well in traditional networks with links of low bit-error rate [3]. It assumes that the networks are relatively error-free, and the loss of packets is due to congestion. Thus, TCP reduces its window size to alleviate network congestion. This scheme is effective when loss is only due to congestion. However, it is not effective for the unreliable wireless media [5, 6]. When we use TCP protocol in networks with high bit-error-rate links, problems will occur. Many papers showed that regular TCP is impractical in networks with wireless links [5]. If we use regular TCP in such an environment, the window size at sender will never reach its optimal value, and throughput will be decreased.

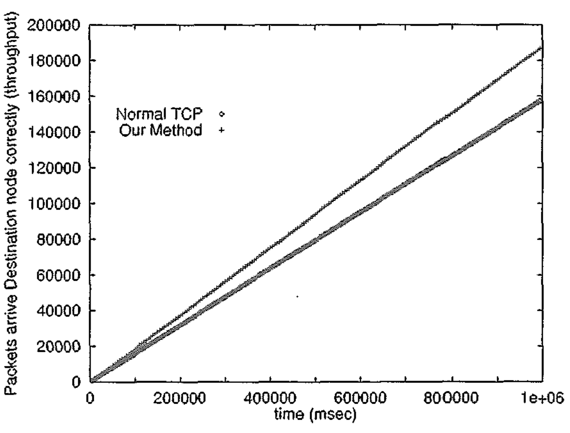

Figure 15: Throughput change with bit-error rate = 1/20k

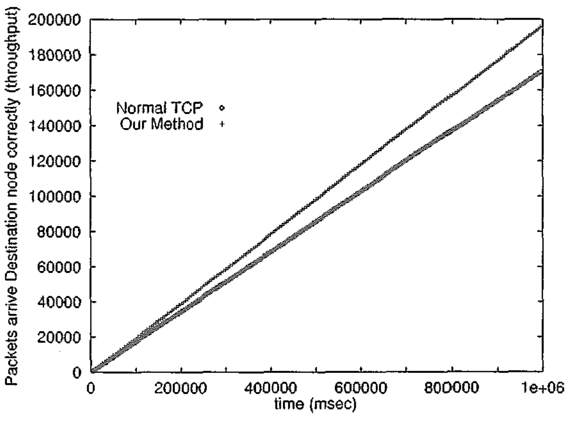

Figure 16: Throughput change with bit-error rate = 1/40k

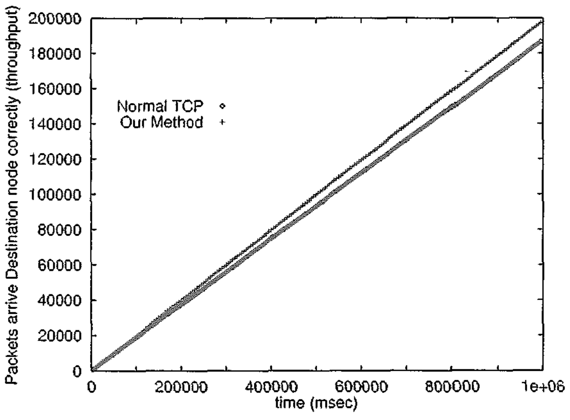

Figure 17: Throughput change with bit-error rate = 1/100k

In this paper, we show a simple mechanism to improve the performance of TCP protocol in networks with wireless link. This mechanism works by adding some modules in base stations without changing anything in fixed hosts elsewhere in the network. Our simulation results show that the mechanism we present significantly improves the performance of TCP.

Wireless links cause packet loss due to bad channel quality. In this case, the sender should not reduce its sending rate, since the loss of packets is not caused by congestion. So our scheme is useful for such a case. Base station retransmits the error packets in time to prevent the source from reducing its window size. Thus we guarantee the end-to-end performance, and have high throughput when the bit error rate in the wireless link is high.

REFERENCES

[1] E. Ayanoglu, S. Paul, T.F. laPorta, K.K. Sabnani, and R.D. Gitlin, "AirMAIL: A Link-Layer Protocol for Wireless Networks", *ACM Wireless Networks*, pp. 47-60, February 1995.

[2] A. Baker and B.R. Badrinath, "I-TCP: Indirect TCP for Mobile Hosts", *15th International Conference on Distributed Computing Systems*, May, 1995.

[3] H. Balakrishnan, S. Seshan, E. Amir and R.H. Katz, "Improving TCP/IP Performance over Wireless Networks", *Proceedings of ACM International Conference on Mobicom*, November, 1995.

[4] H. Balakrishnan, V.N. Padmanabhan, S. Seshan and R.H. Katz, "A Comparison of Mechanisms for Improving TCP/IP Performance over Wireless Links", *Proceedings of ACM SIGCOMM '96*, August, 1996.

[5] R. Caceres and L. Iftode, "Improving the Performance of Reliable Transport Protocols in Mobile Computing Environments", *IEEE Journal on Selected Areas in Communications*, pp. 850-857, June 1995.

[6] M.C. Chuah, O.C. Yue, A. DeSimone, "Performance of Two TCP Implementations in Mobile Computing Environments", *IEEE Globecom' 95*, pp 339-344 vol.1 1995.

[7] D. C. Clark, V. Jacobson, J. Romkey and H. Salwen, "An Analysis of TCP Processing Overhead", *IEEE Communication Magazine*, June 1989.

[8] J.A. Cobb and P. Agrawal, "Congestion or Corruption? a Strategy for Efficient Wireless TCP Sessions", *IEEE Symposium on Computers and Communication*, pp. 262-268, 1995.

[9] D. Cohen, J. Postal, and R. Rom, "IP Addressing and Routing in a Local Wireless Network", *IEEE INFOCOM' 92*, 1992.

[10] A. DesSimone, M.C. Chuah and O.C. Yue, "Throughput Performance of Transport-Layer Protocols over Wireless LANs", *Proceedings of Globecom' 93*, December 1993.

[11] V. Jacobson, "Congestion Avoidance and Control", *Proceedings of ACM SIGCOMM' 88*, August 1988.

[12] P. Karn and C. Partridge, "Improving Round-Trip Time Estimates in Reliable Transport Protocols", *ACM SIGCOMM' 87*, August 1987.

[13] P. Manzoni, D. Ghosal, and G. Serazzi, "A Simulation Study of the Impact of Mobility on TCP/IP", *IEEE International Conference on Network Protocol*, 1994.

[14] P. Manzoni, D. Ghosal, and G. Serazzi, "Impact of Mobility on TCP/IP: an Integrated Performance Study", *IEEE Journal on Selected Areas in Communications*, pp. 858-867, June 1995.

[15] S. Nanda, R. Ejzak and B. T. Doshi, "A Retransmission Scheme for Circuit-Mode data on Wireless Links", *IEEE Journal on Selected Areas in Communications*, 12(8), October 1994.

[16] C.E. Perkins, "Simplified Routing for Mobile Computers Using TCP/IP", *Proceedings IEEE Conference on Wireless LAN Implementation*, pp 7-13, September, 1992.

[17] C.E. Perkins, "Providing Continuous Network Access to Mobile Hosts Using TCP/IP", *Computer Networks and ISDN Systems*, Vol. 26 pp. 357-369.

[18] C.E. Perkins and P. Bhagwat, "A Mobile Networking System based on Internet Protocol", *IEEE Personal Communication*, pp 32-41, First Quarter 1994.

[19] F. Teraoka, Y. Yokote and M. Tokoro, "A Network Architecture Providing Host Migration Transparency", *ACM SIGCOMM' 91*, 1991.