TCP Performance Enhancement in Wireless Mobile Ad Hoc Networks

Md. Mohsin Ali‡, A. K. M. Sazzadul Alam†, and Md. Shohan Sarker†

†Department of Electronics & Communication Engineering (ECE)
‡Department of Computer Science and Engineering (CSE)
Khulna University of Engineering & Technology (KUET), Khulna – 9203, Bangladesh
mohsin_kuet_cse@yahoo.com, sazzad_ece@yahoo.com, shohan.ece@gmail.com

***Abstract*—Transmission Control Protocol (TCP) is the dominating end-to-end transport layer protocol which provides secure and reliable data transfer together with some other protocols in the protocol stack. Its performance is good in wired networks where packet losses are due to congestion. If standard TCP is applied on wireless networks its performance degrades. In this paper performance of the Newreno and Sack1 have been analyzed through simulation for wireless topology and then the new solution called Sack2 has been proposed and tested through simulations. Results from the simulation show that Sack2 is more stable and consistent than others. Its link utilization is also more efficient than the other protocols**.

***Keywords*— TCP, Newreno, Sack1, Mobility, Bit error rate.**

I. INTRODUCTION

Popularity of the internet packet data services for applications like e-mail, file transfer, web surfing etc. are increasing rapidly. And hence, TCP [1] which is the dominating end-to-end protocol on the internet today carrying more than 90% of the total traffic. It provides a secure and reliable connection between two hosts in a multi-network environment appeared in numerous clones (e.g. *Tahoe, Reno, Newreno, Sack* etc.). All these are with different features and advantages, but with maximal throughput as main objective. The performance of a protocol for different network environment and topology can effectively be evaluated through simulations. It has been evaluated that these clones perform reliably in traditional wired networks where losses are mostly caused by network congestion. But the performances are degraded when these clones are applied in wireless networks as they misinterpret every packet loss as a sign of congestion in the link. As a result, it invokes congestion control mechanism resulting in reduction of the link utilization and eventually a significant degradation in performance occurs in the form of poor throughput. The different principles that can be applied to handle these situations are classified into three categories [2] and some of the approaches are discussed in [3].

In this paper we have simulated two existing end-to-end protocols named *Newreno* and *Sack1*. In addition with this, we have proposed and simulated a new protocol named *Sack2*. Finally, we have compared and analyzed these three protocols.

The rest of the paper is organized as follows. Section II provides an overview of the related recent research work. Proposed approach is described in Section III. Section IV gives the simulation environment. Simulation results and comparison among different methods are described in Section V. Section VI provides a concluding remark of this paper.

II. RELATED WORK

A. Newreno

TCP *Newreno* is an improved version of TCP *Reno*. It improves the performance of the TCP *Reno* in case multiple packet losses [4], [5]. It stays in the fast recovery mode if the first new partial acknowledgement received after a fast retransmission. Such partial acknowledgements indicate multiple packet losses within the single window of data. From the three duplicate acknowledgements, the sender infers a packet loss and retransmits the indicated packet. Remaining packets are retransmitted on arrival of single duplicate acknowledgement. In the *Newreno*, there is a variable called recover that keeps the record of highest sequence number transmitted before the start of fast retransmission. By waiting in fast recovery mode, TCP recovers from losses at the rate of one segment per round-trip time rather than waiting until a timeout to occur as TCP *Reno* does often. However, the sender still assumes that losses are due to the result of congestion and invokes congestion control procedures by shrinking its congestion window. TCP *Newreno* applies only for those TCP connections that are unable to use the TCP Sack.

B. Sack1

Sack is a type of selective acknowledgements for the TCP to provide the sender with sufficient information to recover quickly from multiple packet losses within a single transmission window [5], [6], and [7]. Each acknowledgement contains information about up to three noncontiguous blocks of data that have been received successfully by the receiver. Each block of data is described by its starting and ending sequence numbers describing the left and right edges of blocks of received data. The congestion control actions are performed at the sender whenever losses occur. *Sack* uses *Reno*'s *Fast Recovery* algorithm and each packet loss leads to congestion avoidance as compared to *Newreno* (once for all in single window). TCP with *Sack* option performs better than standard TCP in situations where there are multiple packet losses within a window of outstanding data [8]. However, this scheme is not good when the sender's window size is small. Moreover, there is another approach called *delayed acknowledgement* which is discussed in [9].

The receiver waits for receiving the data by means of retransmissions to fill the gaps between received blocks. When missing segments are received, data receiver acknowledges the data by advancing the left window edge in the Acknowledgement Number Field (ANF) of the TCP header. Sender receiving Acknowledgment (ACK) with *Sack* option builds a mask to keep track of all the packets sent, packets lost, and packets retransmitted.

III. PROPOSED METHOD: *Sack2*

TCP *Sack* increases the efficiency on large networks with multiple packet losses and it is suitable for wireless mobile ad hoc networks. But conventional *Sack* (*Sack1*) cannot perform satisfactorily for wireless mobile ad hoc networks as it interprets every packet loss as a sign of congestion and works aggressively on timeouts and packet losses. Moreover, it is frequent in wireless mobile ad hoc networks due to mobility and bit error. So it is necessary to design a modified version of *Sack1* suitable for mobile ad hoc networks.

Sack2 is a modified version of *Sack1* agent which takes some special steps in case of packet losses and timeouts. It considers lost of last ACK and increasing delay of getting ACK as an outcome of mobility in case of wireless mobile ad hoc networks. In case of fast recovery retransmit timeout, it considers an indication of a real congestion in the link. It never makes the slow start threshold value too low. It sets some integer multiple of the initial window if congestion window is less than the initial window in the networks during the quick start process. At timeout, it sets a calculated value other than one if congestion window is less than one.

In this algorithm *Sack2* performs the following actions.

1. If timeout occurs and congestion window is less than one, then it sets

$$cwnd = (pipe\_prev\_)/2$$

2. If retransmit timeout occurs, then it sets

$$sstheshold = (3 \times half\ window/4)$$
$$cwnd = ssthreshold$$

3. If spurious timeout occurs, then it sets

$$cwnd = 1 + (t\_seqno\_ - prev\_highest\_ack\_)/2$$

IV. SIMULATION ENVIRONMENT

We used the discrete event simulator ns-2 [10], [11] in order to simulate our proposed *Sack2* agent and other two existing agents named *Newreno* and *Sack1*. The version used for creating a wireless mobile ad hoc network was ns-2.32. In simulation study there were 10 maximum numbers of nodes and simulation time was 150 seconds. Nodes were arranged in a 600x600 meters grid. We used MAC type 802.11, Omni-directional antenna and AODV routing protocol.

V. SIMULATION RESULTS AND COMPARISON

A. *Number of Mobile Nodes vs. Packet Drops*

We have changed the number of mobile nodes and measured the performance of all protocols under the stated simulation environment. Increasing the number of mobile nodes increases the number of hops in the transmission path and hence the number of packet drops increases for all protocols. Fig. 1 and Fig. 2 show the number of mobile nodes vs. total packet drops and number of mobile nodes vs. percentage of packet drops among the three protocols, respectively. From these two Figs. it is observed that the total packet drops and percentage of packet drops for *Sack2* get lowered for small number of mobile nodes because of its flexible action in case of timeout and multiple packet loss. On the other hand, *Newreno* is not suitable for multiple packet loss and performance of *Sack1* is degraded whenever timeout occurs frequently because it forces the protocol to go to the slow start phase.

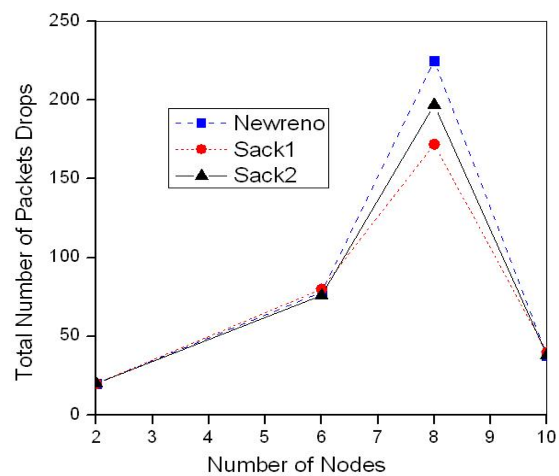

Fig. 1. Number of mobile nodes vs. total number of packets drops.

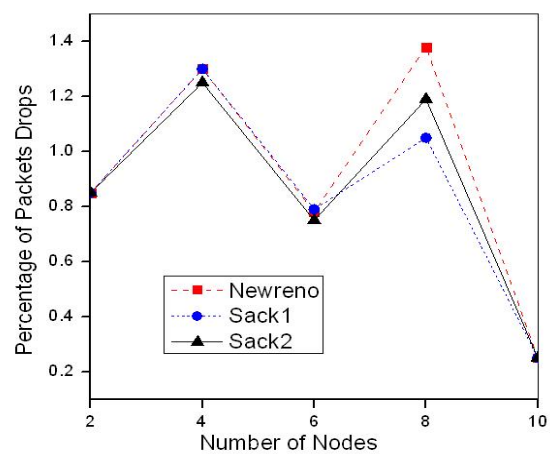

Fig. 2. Number of mobile nodes vs. percentage of packets drops.

B. *Number of Mobile Nodes vs. Total Number of Packets Sends*

We have changed the number of mobile nodes and measured the performance of all protocols in terms of the number of total packets sent. If the numbers of mobile nodes are limited, the intermediate transmission path of the wireless link becomes unreliable and hence there is huge probability of packet losses and timeouts. As a result, the total number of packets sent is low for all these three protocols. Another reason for small number of packet transmission is limited antenna coverage. For an optimum number of intermediate nodes, packet transmission is good for all these three protocols. It is observed from Fig. 3 that the performance of *Sack2* is a little bit better than the others for increased number of mobile nodes as it uses modified quick start procedure under this packet loss and timeout conditions. But if the number of mobile nodes increases more, the optimum value of congestion and nodal delay also increases and hence the total number of packets sent falls again.

C. *Mobility vs. Packet Drops*

We have changed the speed of the mobile nodes and measured the total number of packet drops and percentage of packet drops. With the increase of this speed, probability of timeout increases as it performs handoff and wrong estimation of Round Trip Time (RTT). We observe from Fig. 4 and Fig. 5 that the number of packet drops increases with the increase of velocity for all these three protocols although the performance is not uniform. But the average performance of *Sack2* is better than other existing approaches because of its improved functional criteria in case of timeout. It is also observed from Fig. 4 and 5 that there is an increase of package drops at 4 and 8 m/s velocity of the simulation. The reason behind this behavior depends on the mobile ad-hoc network's topology pattern in simulation like nodes initial and final positions and their antenna parameters.

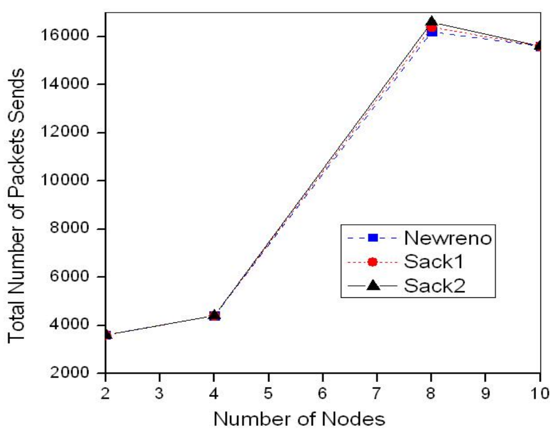

Fig. 3. Number of mobile nodes vs. total number of packets sends.

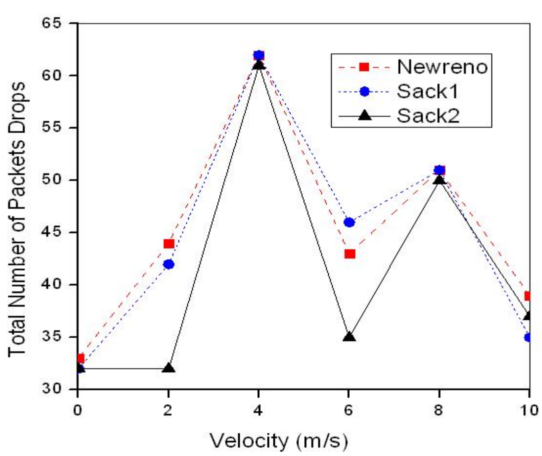

Fig. 4. Velocity vs. total number of packets drops when sender and receiver both are in motion.

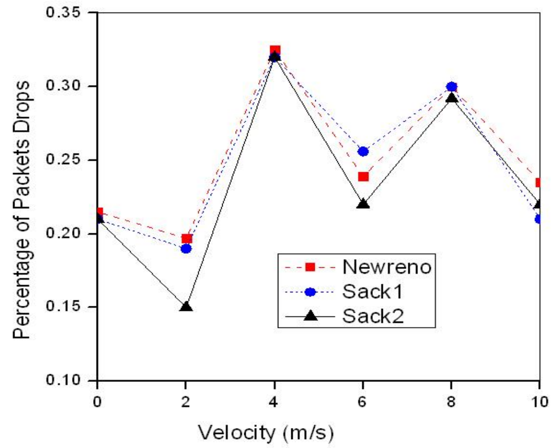

Fig. 5. Velocity vs. percentage of packet drops when sender and receiver both are in motion.

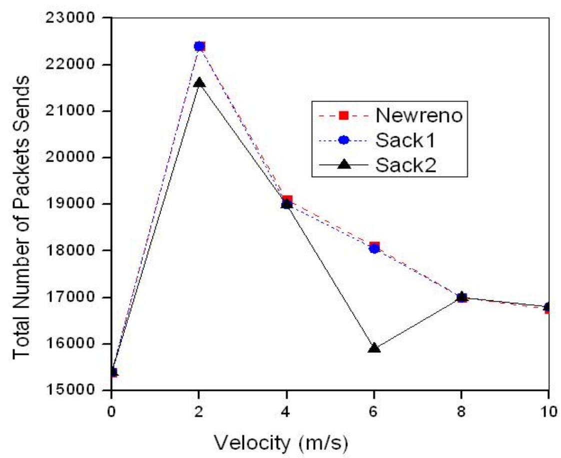

Fig. 6. Velocity vs. total number of packets sends when sender and receiver both are in motion.

D. *Mobility vs. Total Number of Packets Sends*

We have changed the speed of the mobile nodes and measured the performance of all protocols in terms of total number of packets sends. It is observed from Fig. 6 that the total number of packets sends decrease when the speeds of the mobile nodes are increased. This is due to the fact of frequent timeout resulted from velocity variation. However, among these three protocols, *Sack2* performs better at higher velocity as it takes special steps in case of mobility. Moreover, it is observed that *Sack2* shows some worst result at velocity 2 m/s and 6 m/s. This type of situation is happened due to the network's topology pattern and antenna parameter in the simulation and behavior of timeout threshold and congestion window at these points for *Sack2*.

VI. CONCLUSION

It is clear to all of us that wireless communication is the best way of communication today and people are badly dependent on it. So it is very important to bring the wireless communication in a better stage. It is worthy noted that many research work is going on in this regard. This paper also presents an approach for improving the TCP performance in wireless mobile ad hoc networks. In this paper, we have proposed a solution of existing TCP agents in the wireless mobile networks. We have proposed a new agent named *Sack2*. From the simulation results it is clear that *Sack2* performs better than the other TCP agents in case of wireless mobile networks considering mobility and bit error rate.

We are currently working with different parameters together with packet loss and timeout to enhance the performance of the TCP agent.

VII. REFERENCES

[1] J. Postel, "Transmission Control Protocol," *RFC 793,* September 1981.

[2] H. Balakrishnan, V. N. Padmanabhan, S. Seshan and R. H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links", *IEEE/ACM Transactions on Networking*, vol. 5, no. 6, pp. 756 – 769, December 1997.

[3] A. Natani, J. Jakilinki, M. Mohsin, and V. Sharma, "TCP for Wireless Networks", *ACM SIGCOMM*, vol. 26, no. 5, pp. 163–243, 2001.

[4] S. Floyd and T. R. Henderson, "The Newreno Modification to TCP's Fast Recovery Algorithm," *RFC 2582,* April 1999.

[5] A. Ahmed, M. H. Zaidi, and K. Sharif, "Enhancing TCP Performance in Mobile Ad hoc Networks," *National Conference on Emerging Technologies, 2004.*

[6] A. Chrungoo, V. Gupta, H. Sarany, and R. Shorey, "TCP k-SACK: A Simple Protocol to Improve Performance over Lossy Links," *Proc. of the IEEE GLOBECOM'01*, San Antonio, Texas, USA, Nov. 2001.

[7] K. Fall and Sally Floyd, "Simulation-based Comparisons of Tahoe, Reno and SACK TCP", *Computer Communications Review*, vol. 26, pp. 5 – 21, 1996.

[8] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgement Options," *RFC-2018,* October 1996.

[9] S. Xu, T. Saadawi, "Evaluation for TCP with Delayed ACK Option in Wireless multi-hop Networks", *IEEE Vehicular Technology Conference,* pp. 267 – 271, 2001.

[10] Network Simulator 2 (ns-2), http://www.isi.edu/nsnam/ns/, (Accessed on 10th December 2009).

[11] P. Meeneghan and D. Delaney, "An Introduction to NS, Nam and OTcl Scripting," April 2004.

Md. Mohsin Ali received the B.Sc. Engg. degree (with honors) in Computer Science & Engineering from Khulna University of Engineering & Technology, Khulna – 9203, Bangladesh, in 2007.

He is currently a graduate student in Department of Computer Science in The University of Western Ontario, London, Ontario, Canada. He has published several international journal and conference papers. His current research interests include computer algebra, mobile computing, algorithms, computer networks, and security in computer networks.

Mr. Ali is an Associate Member of the Institute of Engineers Bangladesh (IEB) and achieved a Champion award from the Cisco Networking Academy.

A. K. M. Sazzadul Alam received the B.Sc. Engg. degree in Electronic & Communication Engineering from Khulna University of Engineering & Technology, Khulna – 9203, Bangladesh, in 2010.

His current research interests include computer networks.

Mr. Alam is now working as a system engineer in GrameenPhone Bangladesh Ltd.

Md. Shohan Sarker received the B.Sc. Engg. degree in Electronic & Communication Engineering from Khulna University of Engineering & Technology, Khulna – 9203, Bangladesh, in 2010.

His current research interests include computer networks.

Mr. Sarker is now working as a system engineer in GrameenPhone Bangladesh Ltd.