Volume 5, Issue 6, June 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A New Protocol to Provide a Lossless Wireless Link in Manet

[1]Vinod Tiwari, [2]Navdeep Kumar
[1] Student, Haryana Engineering College, Haryana, India
[2]H.O.D (CSE), Haryana Engineering College, Haryana, India

Abstract— The recent activities in MANET (Mobile Adhoc Network) indicate that mobile networks will be a vital part of future networks. In MANET's, a node can dynamically join the network for the communication and also can depart from the network any time. Due to this distinguishing property nodes in the MANET have very high degree of mobility and also route failure is also very common because of link breakage due to high mobility, which affects the data transmission. To increase the trustworthiness of data transmission and to trim down delay due to route re-computation, routing protocols, such as Temporally Ordered Routing Algorithm (TORA), provides various routes between senders and receivers, and use multi-path route to convey packets. In that case, packets arriving from dissimilar paths may not appear at the receiver in correct order.

The TCP receiver is not responsive of this type of multi-path route; it would misunderstand such broken order packet arrivals which is an indication of network congestion. TCP does not work well under these conditions, because TCP was designed for consistent wired network in which almost packet losses are supposed to be due to network congestion.

In this thesis by implementing TCP Recognition of Broken Order and Answer with Time Stamp (TCP-BO) algorithm, the negativity of TCP has been conquering, and it is confirmed that broken order packet is delivered due to multi-path routing protocol. The simulation shows that, TCP-BO algorithm has a higher throughput than TCP-SACK. The TCP with selective acknowledgement scheme (TCP SACK)[4 crc-ubi] improves TCP performance by allowing the TCP sender to retransmit packets based on selective ACKs provided by the receiver. For two different simulation scenarios in TCP-BO, a throughput of 14 % and 8.45 % has been achieved.

Keywords: Mobile ad hoc networks, Network congestion, TCP-SACK, TCP-BO, Broken order packet, MANET, TORA, Multi path route.

I. INTRODUCTION

MANET's [1] can be represented as an intricate distributed system, which consists of wireless nodes, and the nodes can dynamically move and self-organize within the prescribed topology. With these circumstances they figure-out an arbitrary, and transitory ''ad-hoc'' network topology. . It is a concept of communications, which shows that if a person wants to communicate with one other, then they can create a network with out any pre-decided communication infrastructure.

Now a days Mobile ad hoc networks, can be implemented in infrastructure less connection with out the need of conventional fixed wired communications networks. In MANET's, a node can dynamically join the network for the communication and also can depart from the network any time. Due to this distinguishing property nodes in the MANET have very high degree of mobility and also route failure is also very common because of link breakage due to high mobility, which affects the data transmission. MANET's have given new challenges, as a result of its unique characteristics and the dynamic nature of the network.

dhoc networks are uniquely characterized by the several factors that differentiate them from traditional computer networks:

1. Lack of a fixed infrastructure: Due to absence of dedicated routers, mobile hosts in ad hoc networks also serve as peer-to-peer relays for connections in the network.

2. Mobility: All hosts in the network are mobile and, hence, the network topology can be highly dynamic. From the perspective of a single end-to-end connection, not only are the end-hosts mobile, but the intermediate "routers" are mobile, too.

3. Shared channel: Because of the all-wireless nature of ad hoc networks, not only do flows in the same vicinity contend with each other, but part of a flow traversing multiple hops can contend with other parts of the same flow in its vicinity.

4. Limited bandwidth: While mobile hosts in general can be assumed to possess fewer amounts of resources than their static (wire line) counterparts, the wireless channel bandwidth is also scarce, resulting in multihop flows typically enjoying limited Bandwidths of at most a few hundred kilobits per second.

Reliable data delivery is a major requirement for the communication over the network. TCP is the most widely used transport protocol for this purpose, because its congestion control is essential for guaranteeing network stability, and its flow control and end to-end reliability remove the need for loss recovery in the Application layer. These issues are

especially important in the future wireless networks, which will use wireless technology not only in the last hop, but also in the backbone.

However, TCP introduces significant latency and throughput variability in the presence of mobility and frequent handovers. It was developed to work with fixed networks and optimized to allow good bandwidth utilization with congestion control. Therefore, in the case of handovers, TCP considers packet losses as a sign of congestion. This results in unnecessary timeouts and retransmissions.

ATP is Adhoc transport protocol that is specifically designed for Adhoc wireless network. It is the advancement of TCP's design [1]. Its different components are designed in such a manner so as to solve the problem experienced by TCP [2] [3] [4] [5] [6] over Adhoc network. It is a rate based transmission protocol with functionalities that resides at one of three entities namely ATP sender, intermediate node and ATP receiver.

ATP sender is responsible for connection management, reliability, congestion control, and initial rate estimation. The intermediate node assists the node in its operation by providing network feedback with respect to congestion control and initial rate estimation. ATP receiver is responsible for collating the feedback information provided by intermediate nodes before sending the final feedback to ATP sender for reliability, rate and flow control.

Due to mobility of nodes route failure may occur, hence packets coming from TCP sender may not arrive in order at the TCP receiver, which means routes in MANETs, are short-lived due to frequent link breakages. To reduce delay due to route re-computation and node mobility, some routing protocols such as Temporally Ordered Routing Algorithm (TORA) maintain multiple routes between a sender-receiver pair.

In such a case, packets coming from different paths may not arrive at the receiver in order. Being unaware of multi-path routing, TCP receiver would consider such out-of-order packet arrivals as a sign of congestion. The receiver will thus generate duplicate ACKs that cause the sender to invoke congestion control algorithms like fast retransmission (upon reception of three duplicate ACKs), which degrades TCP's performance a lot.

In this thesis by implementing TCP Recognition of Broken Order and Answer with Time Stamp (TCP-BO) algorithm, the negativity of TCP has been conquering, and it is confirmed that broken order packet is delivered due to multi-path routing protocol.

II. PROPOSED WORK

In this thesis, I have proposed TCP-BO algorithm. It is a lengthwise approach implemented in TCP-SACK in order to improve the performance of TCP by dealing with broken-order packet delivery due to multi-path routing protocol. I have used Time Stamp option for the mechanism of detection. TCP-BO mainly deal with detection of broken-order packet and removal of the problem of packet loss in the MANET; it is providing a mechanism for the detection and packet loss.

TCP-BO algorithm is an extension of TCP-SACK. I have used NS-2 to implement this algorithm and for its evaluation. The implementation of this algorithm is compared with TCP-SACK.

III. RESULTS AND DISCUSSION

Out-of-order packet is delivered due to the implementation of Multi-path routing protocol TORA, and TCP-BO have detected and responded in order to improve the TCP performance by avoiding unnecessary invoking of congestion control and retransmission algorithms, which reduce the congestion window size that was seen by "TCP-SACK". A detail simulation study is presented below.

1 Performance evaluation of TCP with TCP-SACK

First we have evaluated the performance of TCP with out implementing TCP-BO algorithm that is with TCP-SACK at the sender side and TCP-SINK-SACK at the receiver side. The simulation was done for thirty five times by changing the position and movement of the 20 nodes and average was taken. Figure 1 shows the simulation results. Packet sending was started at 20 simulations second, and a maximum of 1551 packets were sent by the TCP sender (TCP-SACK) over 200 simulated times.

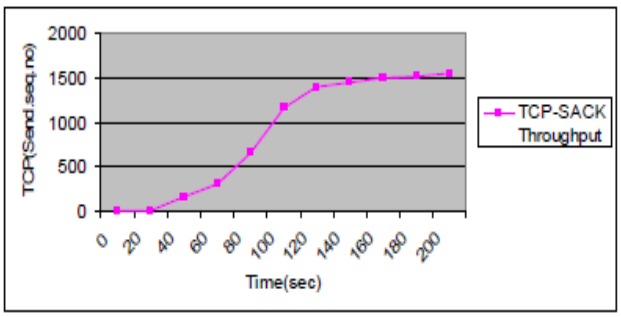

Figure 1. Maximum number of packets sent by TCP sender over 200 simulation seconds

And as shown in figure 2, the TCP receiver received maximum of 1549 packets. Theoretically, since TCP protocol is working on clock-based acknowledgment, new packet is sent only if the previously sent packet is acknowledged. The simulation results in figure 1 and 2 reveal this fact; the TCP receiver receives almost the entire packet sent by the TCP sender.

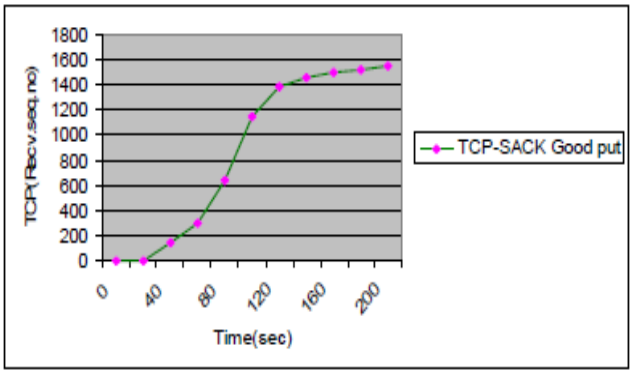

FIGURE 2. Maximum number of packets received by TCP

Figure 3 and 4 tell about the congestion window size [CWND] with the 200 simulated times, as per the previous discussion, CWND means the maximum number of window size that can be sent by the TCP sender at a time. In figure 3 between 0 and 50 simulation seconds the sender sent a maximum of CWND 12. A maximum of 30 CWND was achieved at a simulation time of 100. Whenever the CWND reduces drastically, it means packets are reached out-of-order in which the TCP receiver produces either three duplicate acknowledgment or the expected acknowledgment is not come with in the retransmission time out period, If time out is occurred, slow start threshold is reduced to half of the current congestion window size, the CWND is reduced below SSTRHESHOLD that is the TCP sender enters in to slow start and start sending packet from one and continuous sending exponentially until it reaches to slow start threshold (SSTRHESHOLD). If the CWND is reduced but not below SSTRHESHOLD, then the TCP sender has received three DUPACKs, so it will start from congestion avoidance phase. From the simulation result, about the time of 85sec the TCP sender receives three DUPACKs, and start sending from congestion avoidance phase because as shown in figure 6.4.a at this 85 second the SSTHRESHOLD was found to be 8, which is equal to half of the CWND.
At about 30,110 and between 100 and 150, and 160 seconds, retransmission timer has expired, and The TCP sender enters into slow start phase in which it start sending form one packet. At the above simulation times as shown in figure 5 the slow start threshold was found to be 2.

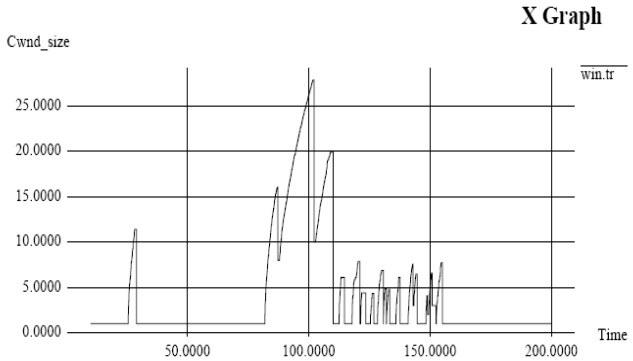

Figure 3. Congestion window Vs simulated time with TCP-SACK

In Figure 4, between 100 and 200 simulation seconds, the congestion window size reduced a lot, which is the effect of out-of-order delivery of packets. This reduces the TCP's sender CWND that affects the maximum number of packets that can be sent by the TCP sender (throughput).
The slow start threshold of "TCP-SACK" has been shown in figure 6. It shows the boundary between the slow start phase and congestion avoidance phase in order to maintain the consistency of TCP sender – receiver pair.

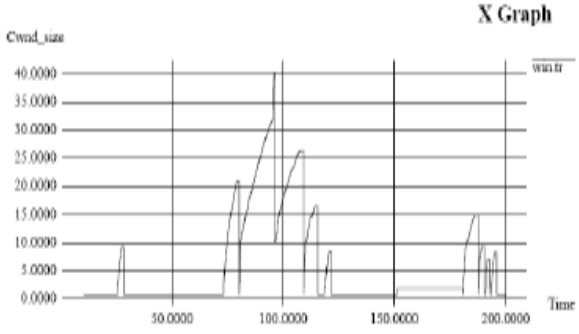

Figure 4 Congestion window Vs simulated time with TCP-SACK

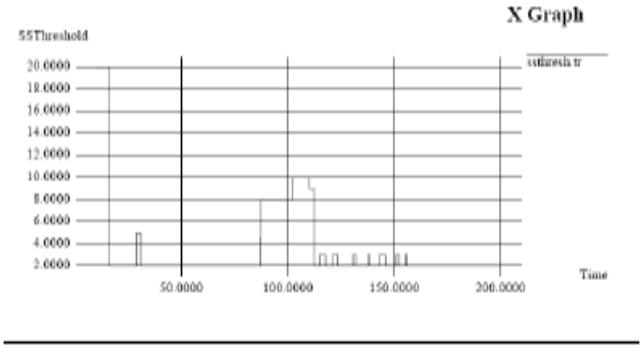

Figure 5 Slow start threshold over 200 simulated times with TCP-SACK.

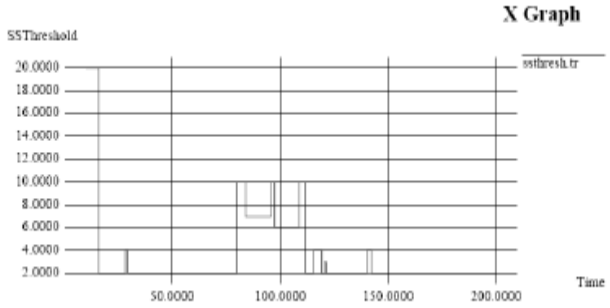

Figure 6. Slow start threshold over 200 simulated times with TCP-SACK.

2. Performance evaluation of TCP by implementing TCP-BO algorithm

In this evaluation we have varied the disabling period T to be 1 and 2 times RTT (retransmission timer).

2.1 Disabling period T = RTT

Once again the simulation was done for thirty five times by changing the position and movement of the 20 nodes and their speeds and average was taken. When the disabling period T is equal to one time the retransmission timer, as shown in figure 6. the maximum number of packets sent by TCP sender over 200 simulated time were found to be 1754.With the above result an average improvement of 13 % was found by implementing TCP-BO algorithm. In figure 6.6 the good put effect has been shown and the maximum number of packets received by the TCP receiver was about 1738, which improves the TCP good put by 12.7%. The above percentages improvement implies that out-of-order packets are really delivered due to multi-path route in case of "TCP-SACK" which degrades the performance of TCP, and implementing TCP-BO algorithm have improved the effect of out-of-order problem.

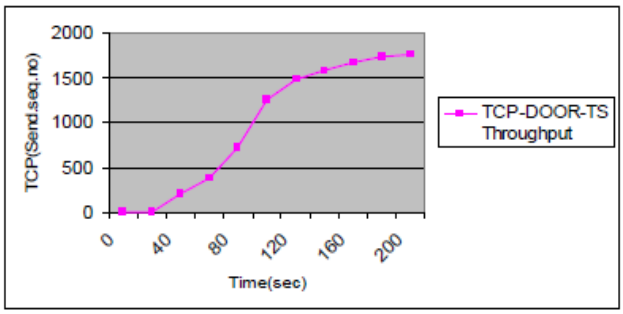

Figure 7. The maximum number of packets sent by TCP sender over 200 simulated seconds (throughput), with TCP-BO (T = RTT)

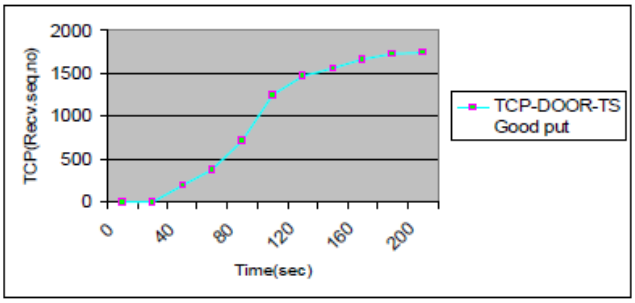

Figure 8. The maximum number of packets received by TCP sender over 200 simulation seconds (good put), with TCP-BO (T = RTT)

In figure 9 and 10 TCP-BO shows larger congestion window size than "TCP-SACK", this indicates better utilization of available bandwidth. The slow start threshold is shown in figure 11 and 12.

As shown in figure 9, between 100 and 150 simulation seconds, the congestion window size (CWND) is better than that of shown in figure 10, which is the effect of detecting out-of-order packet and responded accordingly. In figure 10 between 150 and 200 simulation seconds only one packet has been sent at a time. But as shown in figure 10 between 150 and 200 seconds the congestion size is increased up to a maximum of 15 and many packets have been sent which is the effect of avoiding unnecessary calling of congestion control algorithm.

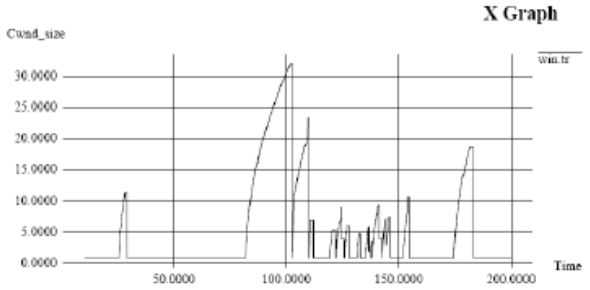

Figure 9. Congestion window Vs simulated time with TCP-BO algorithm (T=RTT)

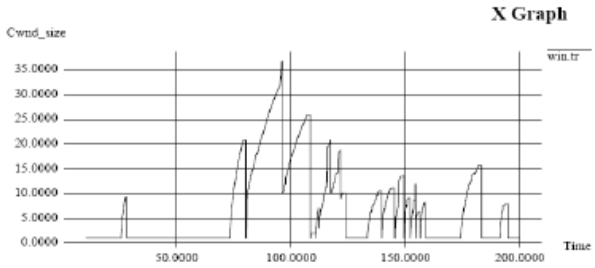

Figure 10. Congestion window Vs simulated time with TCP-BO algorithm (T=RTT)

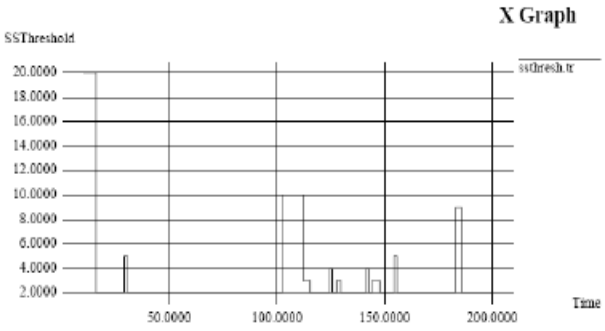

Figure 11. Slow start threshold over 200 simulated time with TCP-BO (t=rtt)

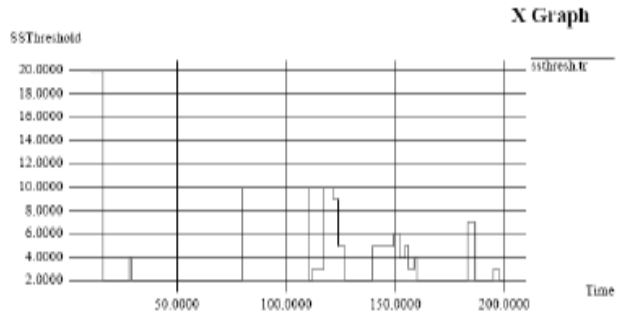

Figure 12. Slow start threshold over 200 simulated time with TCP-BO (t=rtt)

2.2. Disabling period T = 2 x RTT

For disabling period T = 2 x RTT, as shown in figure 13, a maximum of 1668 packets have been sent by the TCP sender over 200 simulated times (throughput), which is compared to the base TCP-SACK, it improves the throughput by 7.54% but compared to disabling period of T = RTT, the throughput is decreased by 5.46%.

When we see the good put effect, as shown in figure 14 for T = 2 x RTT, about 1668 packets have been received by the TCP receiver, which indicates that all the packets sent by the sender have been received by the receiver. The good put result compared with TCP-SACK is about 7.54%, which shows improvement compared to TCP-SACK and it decreases by 5.46% compared to T = RTT of TCP-BO.

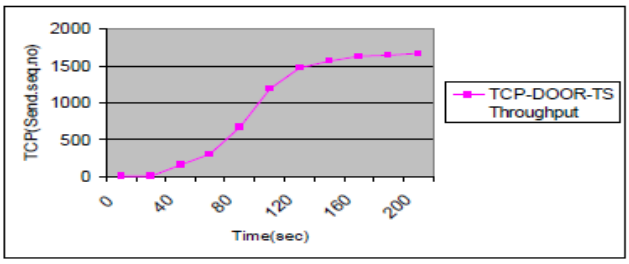

Figure 13. TCP throughput measured for TCP-BO for T =2 x RTT

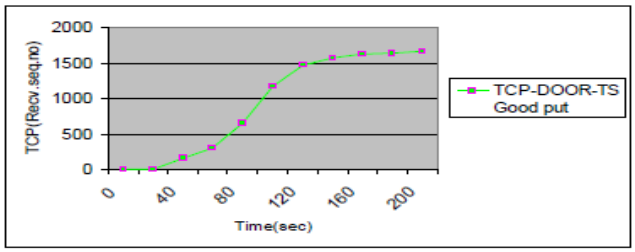

Figure 14. Good put result found by implementing TCP-BO with t=2xrtt

Figure 15, 16, 17 and 18 show the congestion window and slow start threshold found by making the disabling period T = 2xRTT. The size of congestion window with in simulated time is some how good relative to base TCP-SACK, and drops to some extent compared to T = RTT.

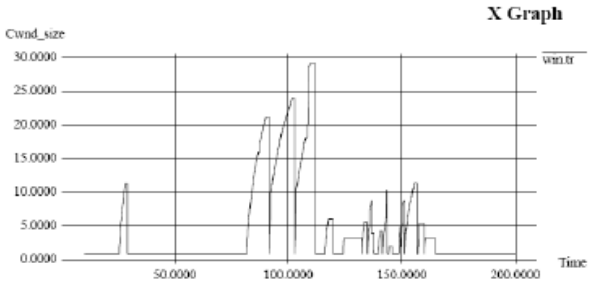

Figure 15. A congestion window Vs simulated time for t = 2* rtt

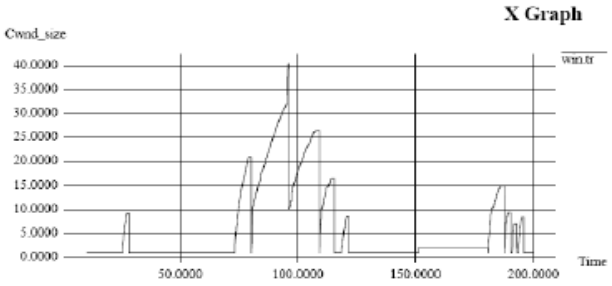

Figure 16. congestion window Vs simulated time for t = 2xrtt

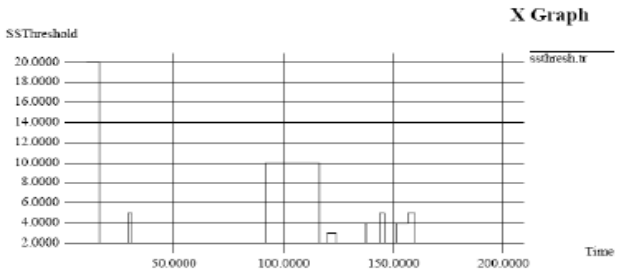

Figure 17. Slow start threshold Vs simulated time for t = 2xrtt

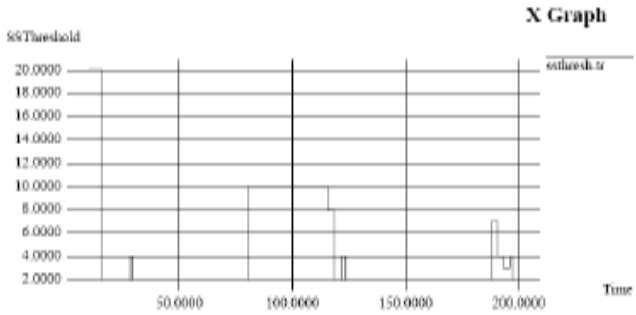

Figure 18. Slow start threshold Vs simulated time for t = 2xrtt

Figure 19 and figure 20 show comparison of throughput and good put measurement for both TCP-SACK, and TCP-BO with disabling period T=RTT and T=2XRTT. All measures taken by the TCP-BO improve the throughput and good compared to TCP-SACK. For disabling period T = RTT, a throughput and good put improvement ratio of 13% and 12.7% have been achieved, respectively. And for a disabling period of T = 2xRTT, for both throughput and good put improvement ratio of 7.54% has been found.

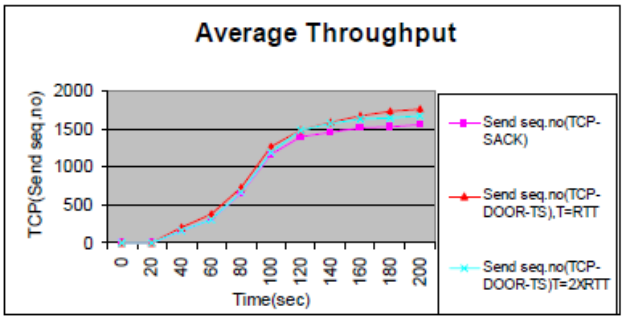

Figure 19. Comparison of throughput, TCP-SACK and TCP-BO

Figure 20. Comparison of good put, TCP-SACK and TCP-BO

IV. CONCLUSIONS

The modification of TCP Detection of Out-of-Order and Response with Time Stamp (TCP-BO) algorithm in TCP-SACK, when multi-path routing protocol (TORA), is used in mobile ad hoc networks has been evaluated. First it is confirmed that out-of-order packet is really delivered due to multi-path route between TCP sender and receiver, which has significant performance degradation effect. TCP-BO algorithm improves the performance of TCP by detecting and responding for out-of-order packet delivery. Unnecessary invoking of congestion control algorithm and retransmission of packets were protected. All measures taken by the TCP-BO algorithm improve the bandwidth utilization and increase the throughput and good put up to 13% and 12.7% respectively, compared to TCP-SACK.

The above percentage improvements implies that out-of-order packet is occurred in case of TCP-SACK which degrades the performance of TCP, and implementing TCP-BO algorithm have improved the effect of out-of-order problem.

For disabling period T = RTT, a throughput and good put improvement ratio of 13% and 12.7% have been achieved, respectively. And for a disabling period of T = 2 x RTT, for both throughput and good put improvement ratio of 7.54% has been found which implies that disabling period of one times the round trip time is optimum for attaining good result. Theoretically the disabling period T = 2 x RTT, the TCP protocol would not respond fairly. Disabling period T = RTT is optimum time for the TCP to recover from out-of-order packet at the receiver side.

The congestion window size and slow start threshold of TCP-SACK, and TCP-BO with T = RTT and T = 2 x RTT have been evaluated, increments of congestion window size for TCP-BO than TCP-SACK has been observed, which implies that unnecessary calling of congestion response algorithm and retransmission of packet have been avoided, which reduces the congestion window size and sending rate a lot.

REFRENCES

[1] K.Sunderesan, V.Anantharaman, R.SivaKumar, "ATP reliable transport protocol for adhoc networks," Proceeding of 4th ACM international Symposium on Mobihoc, June 2003pp 64-75.

[2] B. Bakshi, P. Krishna, N.H. Vaidya, and D.K. Pradhan, "Improving Performance of TCP over Wireless Networks," Proc. 17th Int'l Conf. Distributed Computing Systems (ICDCS), May 1997.

[3] M. Gerla, K.Tang, and R. Bagrodia, "TCP Performance in Wireless Multi Hop Networks," Proc. IEEE Workshop Mobile Computing Systems and Applications, Feb. 1999.

[4] G. Holland and N.H. Vaidya, "Analysis of TCP Performance over Mobile Ad Hoc Networks," Proc. ACM MOBICOM Conf., pp. 219-230, Aug. 1999.

[5] J.P. Monks, P. Sinha, and V. Bharghavan, "Limitations of TCP ELFN for Ad Hoc Networks," Proc. Workshop Mobile and Multimedia Comm., Oct. 2000.

[6] T.D. Dyer and R. Bopanna, "A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks," Proc. ACM MOBIHOC 2001 Conf., Oct. 2001.

[7] /Sami Iren and Paul D Amer, "The Transport Layer tutorial and survey" ACM computing survey, vol 31, No. 4, December 1999.

