TCP-BuS : Improving TCP Performance in Wireless Ad Hoc Networks[1]

Dongkyun Kim†, C.-K. Toh‡, and Yanghee Choi†

Department of Computer Engineering†
Seoul National University
San 56 - 1, Shilim-dong, Kwanak-ku, Seoul, Korea

School of Electrical and Computer Engineering‡
Georgia Institute of Technology, Atlanta, Georgia 30332-0250 U.S.A.
E-mail : {pretty,yhchoi}@mmlab.snu.ac.kr, cktoh@ece.gatech.edu

Abstract

Reliable data transmission over wireless multi-hop network, called ad hoc network, has proven to be non-trivial. TCP(Transmission Control Protocol), widely used end-to-end reliable transport protocol in wired network, is not entirely suitable when applied to wireless ad hoc network due to TCP's congestion control schemes. In particular, TCP at the source considers the network is congested when detecting packet losses or timeouts. However, in a wireless ad hoc network, when a route disconnection occurs because of node movement, the network mistakens this as a congestion. Therefore, the conventional TCP congestion control mechanism cannot be applied, because a route disconnection must be handled differently from network congestion. In this paper, we propose a new mechanism that improves TCP performance in a wireless ad hoc network where each node can buffer packets during a route disconnection and reestablishment. Additionally, we incorporate new measures to deal with the reliable transmission of important control messages. Our simulation results further confirmed these advantages.

I. INTRODUCTION

Wireless ad hoc network is a new mobile network infrastructure that can be used when the deployment of wired network is expensive and time-consuming. This applies to battlefield, emergency rescue operations, and large-scale wireless conferencing situations where all the nodes are mobile. Recently, several proposals related to routing protocols were suggested for wireless ad hoc networks[1-8]. However, reliable data transmission problem has been overlooked.

TCP is widely used in the current Internet as the reliable end-to-end transport protocol. However, earlier research work had confirmed that TCP cannot be applied to wireless networks without modifications to take into account the time-varying link characteristics and node mobility issues associated with wireless networks. If a TCP source does not receive the acknowledgement packets from the destination, timeout events for the transmitted segments will occcur. TCP assumes that congestion has occured within the network and initiates the congestion control procedures. In general, TCP implements two phases for congestion control ; (a) slow start, and (b) congestion avoidance.

TCP slow start is a process through which the source initiates data transmission. However at a certain point, the buffering and processing limits of intermediate routers in the route are reached and then packets will be dropped. Congestion avoidance, on the other hand, provides the means for the source to deal with lost packets. The source then detects congestion occurrences according to two indications of packet losses: (a) when a timeout occurs, and (b) when duplicate ACKs are received. Currently, there are two widely used variants of TCP which can handle the abovementioned problems.; TCP Tahoe and TCP Reno[17].

Recently, most works for improving TCP performance have focused on the cellular-style wireless network (last-hop wireless network)[9,13,14] where base stations play a significant role in providing wireless access by mobile users to the fixed network. Distinct mobile connections are partitioned into segments, one between the mobile host and base station, and the other between the base station and the correspondent host. I-TCP[10] and SNOOP[11] are examples of such schemes which employ the concept of distinct connection segments. In these approaches, the base station buffers the transferred TCP segments and masquerades the mobile hosts from the fixed side of the network.

However, since all nodes are movable in a wireless ad hoc network, the route reconstructions in this multi-hop wireless network are frequently invoked during data transmission due to node movements and it is impossible that the buffer capability for masqurading is performed by the node detecting the route disconnection every time the route disconnection occurs.

Moreover, route failure is unavoidable due to the adherent nature of the wireless ad hoc network. If unmodified TCP used in existing wired networks is applied to wireless ad hoc networks, TCP performance will be degraded because it cannot differentiate congestion from route failure. Thus, in this paper, we propose a novel scheme for improving the TCP performance in wireless ad hoc networks by introducing the buffering capability in mobile nodes. Our proposed scheme takes advantage of the feedback information for detecting the route disconnection , which is also used by TCP-F [12], as well as the features of underlying ad hoc routing protocol. In addition, we include intelligent buffering techniques at mobile nodes. We selected the ABR (Associativity-Based Routing)[15,16] protocol as the underlying routing protocol based on source-initiated on-demand protocol [19]. ABR advocates for stable and long-lived routes.

The rest of this paper is organized as follows. Section II presents

[1]This work was supported by the Brain Korea 21 Project and Agency for Defence Development and ACRC(Automatic Control Research Center), Korea.

the basic operation of routing protocol used in this paper. Our proposed scheme is described in detail in Section III, followed by discussions on simulations performed in Sections IV. Finally, conclusions are made in Section V.

II. Ad Hoc Routing Protocol

The ABR(Associativity-Based Routing) protocol is a source-based routing protocol where a route is established based on-demand. ABR consists of three phases; (a) Route Discovery Phase, (b) Route Reconstruction Phase, and (c) Route Deletion Phase.

As in most source-based routing protocols, the source generates a route request packet (in ABR, BQ-Broadcast Query message). During a route discovery process, the route request packet is replicated at the intermediate nodes. Therefore, the destination is able to receive multiple route discovery packets containing different path information. Among these collected paths, the destination selects the best route based on a selection criteria. Thereafter, the destination conveys the selected route information to the source by using route reply packet(in ABR, REPLY message). ABR exploits the spatial, temporal and connection stability of mobile hosts to derive long-lived routes. Each node maintains associativity information with its neighbors by recording the number of control beacons received from its neighbors. Each BQ packet includes associativity information of visited intermediate nodes during a route discovery process. Therefore, the destination can select the best long-lived route.

When the association property is violated (i.e., when nodes in a selected route moves outside the radio range of its neighbors), the route reconstruction phase is initiated. Here, an alternative partial route has to be discovered so that data packets can ultimately reach the destination.

Figure 1 illustrates an example of the route reconstruction phase. Due to mobility, a node can detect that the path is broken when it does not receive any beacons from its neighbors after a timeout interval. Therefore, it can initiate an LQ(Localized Query) to discover a new partial route. At the same time, the downstream node of a migrating node can no longer receive the beacon message within a timeout value. This node invalidates the old partial path from itself to the destination by propagating an RN(Route Notification) message. During a route reconstruction process, a new partial path is discovered, allowing the node which has detected the route failure to resume forwarding data packets to the destination.

III. Proposed Scheme : TCP-BuS[2]

A. Features of Proposed Scheme

Our proposed modifications to TCP are :

1. **Explicit notifications:** Two control messages (ERDN and ERSN) related to route maintenance are introduced to notify the source of route failures and route re-establishments. These indicators are used to differentiate between network congestion and route failure as a result of node movement. ERDN (Explicit Route Disconnection Notification) message is generated at an intermediate node (pivoting node, PN) upon detection of a route disconenction, and is propagated towards the source. After receiving ERDN message, the source stops transmission. Similarly, after discovering a new partial path from the PN to the destination, the PN sends an ERSN (Explicit Route Successful Notification) message to the source. On receiving ERSN message, the source resumes transmission.

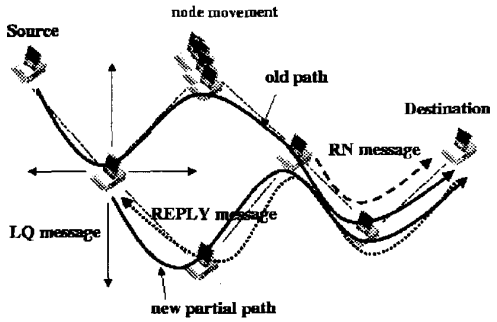

Fig. 1. Route Recovery Procedure of ABR protocol

2. **Extending timeout values:** Packets are buffered along the path from the source to the PN until a new partial route is established. Under route failure conditions, the source will experience repeated timeouts and will begin to retransmit what it construes as packet loss. Although the source does nothing during a route recovery, after the source is notified of a route reestablishment, it is possible that timeout events will occur because it has taken some time to recover the route. Therefore, in this paper, timeout values for buffered packets at the source and nodes along the path to the PN are doubled. To clarify further, although the timeout values should be based on *RTT (Round Trip Time)*, for the simplicity of implementation, we recommended the timeout values to be doubled.

3. **Selective Retransmission of Lost Packets at Receiver Node :** The retransmission of lost packets on the path due to congestion relies on timeout mechanism. Therefore, if the timeout values for buffered packets at the source and nodes along the path to the PN are adjusted to be doubled, the lost packets are not retransmitted until the adjusted timeout values are expired. To early cope with the packet losses along the path from the source to the PN, a request is generated to require the source to retransmit the lost packet selectively before their timeout values are expired to make the lost packets retransmitted by the source.

4. **Avoiding unnecessary requests for fast retransmission:** Using the ABR protocol, packets along the path from the PN to the destination may be discarded by intermediate nodes after receiving a RN (Route Notification) message. When the best partial path is re-established, the destination can notify the source of the information on the packets lost along the path. When notified, the source simply retransmits the lost packets. However, in this process the packets buffered along the path from the source to the PN may arrive at the destination earlier than the retransmitted packets. Therefore, the destination continues to send acknowledgement packets containing an expected sequence number until the retranmsitted packets arrive at the destination (via the fast retransmit method adopted by TCP-Reno). In this paper, these unneces-

[2]TCP with BUffering capability and Sequence information

sary request packets for fast retransmission are avoided.

5. **Reliable transmission of control messages:** After a PN detects a route disconnection, the node will notify the source of the route failure by using ERDN message. However, the source can only take action on the route failure only if it receives the ERDN message reliably. In addition, each intermediate node receiving the ERDN message stops transmission of its buffered packets.

The reliable transmission of ERDN message depends on the link layer and network layer. If a node A (including PN) reliably sends an ERDN message to its upstream node B, the ERDN message forwarded by its upstream node B can be heard by node A. Thus, if a node has sent an ERDN message but cannot hear any ERDN message forwarded by its upstream node during ERDN_RET_TIMER period, it concludes that the ERDN message is lost and will try to retransmit the message. Similarly, after a PN acquires a new partial path from itself to the destination, it notifies the source of the successful route re-establishment by using ERSN message. However, the ERSN message can again be discarded in the network because of congestion. Consequently, only if the source receives the ERSN message from a PN, it can resume its transmission. Therefore, it is very important for a PN to send the ERSN message reliably to the source. There are two possible approaches. One way is to have the source generate Probe messages periodically to check if the PN has found a new partial route successfully until it receives the ERSN message or it times out. The other approach has the characteristics where intermediate node after receiving an ERSN message will retransmit it towards its upstream node if it does not receive any data packets or an echoed ERSN message during the ERSN_RET_TIMER time period.

B. Proposed Control Messages and Parameters

In this paper, the following control messages are introduced to improve TCP performance (shown in Table 1). In addition, the system parameters related to the various timers are shown in Table 2.

In source-initiated on-demand routing scheme like ABR, "on-the-fly" packets on the nodes along the partial path from the PN to the destination are discarded during route reconstruction while other packets are buffered at intermediate nodes. We improve TCP performance by using sequence information and buffering capability of mobile hosts. The parameters and control messages used are :

- ERDN_Time_SEQ: When an intermediate node detects a route failure and it cannot forward the buffered data packets, ERDN_Time_SEQ is defined as the sequence number of the TCP segment pending in the head of line (HOL) of the node's transmit queue. ERDN_Time_SEQ information is propagated to the source by using ERDN message.
- ERDN_Recv_SEQ: When the source is transmitting TCP segments, if the source receives an ERDN message from the network, the source stops sending TCP segments. ERDN_Recv_SEQ is defined as the sequence number of the last TCP segment sent up to when the TCP source receives an ERDN control packet.
- Last_ACK: During route reconstruction, the destination responds to LQ message with a REPLY message. Last_ACK is therefore defined as the last sequence number of the segment which the destination has received successfully.

Table 1 : TCP-BuS Control Messages and their Parameters.

Related to Route Discovery		
Type of Control Message	**Parameters**	**Remarks**
ERDN	ERDN_Time_SEQ	To notify the source of route failure
ERSN	Last_ACK	To notify the source of successful route re-establishment

Related to Routing Protocol		
Type of Control Message	**Parameters**	**Remarks**
RN	ERDN_Time_SEQ	To invalidate the route/buffered packets towards the destination
LQ	ERDN_Time_SEQ	Broadcasted in the network for finding a new partial path
REPLY	Last_ACK	The destination responds to LQ with REPLY packet

Table 2 : System Parameter related to timer.

Type of System Parameter	Remarks
ERDN_RET_TIMER	Durations set to reliably transmit
ERSN_RET_TIMER	ERSN control message towards the source

By using ERDN_Time_SEQ and ERDN_Recv_SEQ mentioned above, the folllowing information can be inferred:

- The unacknowledged segments, buffered at the source, up to the segment whose sequence number is (ERDN_Time_SEQ - 1), may be forwarded along the path from the node after the PN towards the destination.
- Unacknowledged segments, buffered at the source, whose sequence numbers are from ERDN_Time_SEQ to ERDN_Recv_SEQ may be buffered at intermediate nodes from the source to the PN.

C. Operation of TCP-BuS

Since ad hoc routes can be invalidated by nodes movements, we shall discuss the actions taken by TCP-BuS at the source, destination and intermediate nodes to cope with host mobility.

C.1 TCP-BuS Functions at Source Node

TCP-BuS at source transmits its segments in the same manner as general TCP when there are no feedback messages (such as ERDN and ERSN messages). The slow start and congestion avoidance mechanisms function as normal. However, when the source receives the ERDN feedback message from the network, it stops sending data packets. In addition, it freezes all timer values and window sizes in a manner to TCP-F.

Next, the ERDN_Time_SEQ is acquired after receiving the ERDN message and the ERDN_Recv_SEQ is also calculated at the same time. As shown in Figure 2, a pivoting node detects a route failure when it wants to send the segment whose sequence number is 10. The PN generates ERDN message containing the sequence number(10) in the head of line of its queue. Therefore, when the source receives the ERDN message, EDRN_Time_SEQ value can be obtained. Meanwhile, the source has been sending segments up to the segment whose sequence number is 14. From this, we can calculate ERDN_Recv_SEQ. Additionally, node N, the next downstream node from PN will send RN message towards the destination, which invalidates the old partial path and flushes buffered segments along that path. Finally, after a successful route reconstruction, the source knows up to which segments the destination had received when it receives ERSN message. Since

ERDN message indicates that there is a route failure in the network, the source just waits for an ERSN message. On receiving this message, the source interprets that route re-establishment is successful. The source is then able to resume data transmission according to the window-based mechanism.

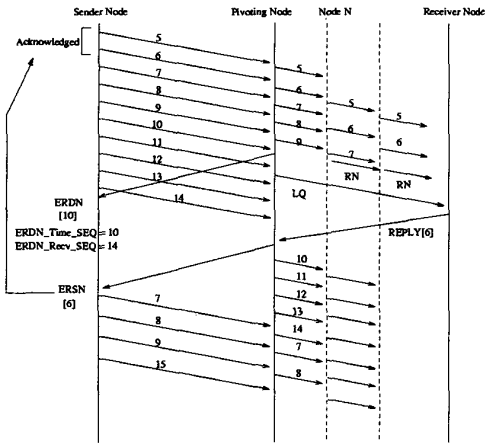

Fig. 2. An example illustrating ERDN_Time_SEQ and ERDN_Recv_SEQ mechanisms.

Meanwhile, the ERSN message includes the information on successfully received packets at the destination (Last_ACK). Therefore, the source can increase the congestion window (cwnd) by the amount of the acknowledged packets. After receiving the ERSN message, the source can safely assume that the segments whose sequence numbers range from (Last_ACK + 1) to (ERDN_Time_SEQ - 1) were discarded on the path from PN to the destination in the network. It is obvious that these discarded packets should be retransmitted by the source. However, it depends on the congestion situation over the path from the source to the PN. An ERSN message can include congestion state information notifying the status of queues at the intermediate nodes. One can make use of ICMP message such as *Source Quench* to indicate the presence of congestion and the source will have to stop transmissions for a short period of time during which the intermediate node can catch up.

Note that the timeout values at the TCP source for the unacknowledged and non-retransmitted segments from ERDN_Time_SEQ to ERDN_Recv_SEQ should be adjusted because of the expected increase in the packet arrival time at the destination due to the presence of route re-establishment.

However, if the packet losses are experienced on the partial path from the source to the PN due to congestion, the source reacts to the congestion and retransmits the lost packets selectively on receiving the selective retransmission request packet issued by the receiver. Therefore, it performs a congestion control procedure and shrinks congestion window size accordingly.

C.2 TCP-BuS Functions at Intermediate Nodes

After a node (PN) detects a route failure, it sends the ERDN message to notify the source of route failure and initiates partial route discovery using the LQ-REPLY process. While the ERDN message is propagated towards the source, each intermediate node stops further transmission of data packets and buffers all pending packets to defer transmission. After receiving the REPLY message, the PN notifies the source of successful route re-establishment via the ERSN message, which also includes Last_ACK information. At each intermediate node receiving the ERSN message, transmission of buffered packets resumes. This is illustrated in Figure 3.

In addition, intermediate nodes perform the process of reliable transmission of control messages like ERDN and ERSN messages (Refer to section III.D).

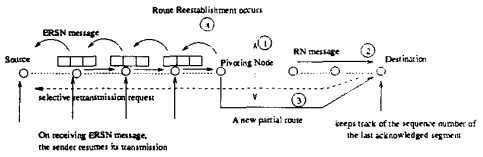

Fig. 3. Sequence of events occurring after a successful route reestablishment.

C.3 TCP-BuS Functions at Destination Node

A receiver performs the normal TCP end-to-end procedure on the acquired path in case that there is no route disconnection. Also, a selective retransmission mechanism as in TCP-SACK can be applied to the source and receiver's procedure. We proposed an additional selective retransmission scheme to cope with the l ost packets due to a congestion on the partial path from the source to the receiver. A request for selective retransmission of lost packets is generated on the receiver detecting the hole of consecutive segment sequence. It requires the source to react to the congestion.

Under the above-mentioned approach, it is still possible that there are many requests for fast retransmission in the backward direction. Consider the case where segments having sequence numbers ranging from (Last_ACK+1) to (ERDN_Time_SEQ-1) will arrive at the destination later than those packets having sequence numbers ranging from ERDN_Time_SEQ to ERDN_Recv_SEQ. As a result, the destination continues to request for fast retransmission by sending duplicated ACK packets for each incoming packets received due to discrepancies in the sequence order. To avoid this problem, an additional procedure at the destination (Figure 4) is required after receiving the RN or LQ message. When receiving an LQ packet during a route recovery process which includes also ERDN_Time_SEQ, the destination keeps the ERDN_Time_SEQ.

With the consideration of selective retransmission and avoiding the unnecessary requests for fast retransmission, the destination sends duplicated ACKs and requests for missing packets selectively according to the following rule. Here, we denote the sequence number of the incoming segment as incoming_SEQ. Pivot_value is the sequence number whose next some segments are lost due to congestion. Therefore, when the receiver notifies the source of lost segment information selectively, the receiver sends the source the Pivot_value and the distance of segment sequence which denotes how many segments are lost from the Pivot_value.

- On receiving LQ message for route extension, Pivot_value =

ERDN_Time_SEQ.

- If incoming_SEQ $\geq$ ERDN_Time_SEQ, then the transmission of duplicated ACKs for fast retransmission is refrained. If incoming_SEQ > Pivot_value, notify the source of the information on missing segments.
- Pivot_value = incoming_SEQ.
- Otherwise, the transmission of duplicated ACKs is permitted.

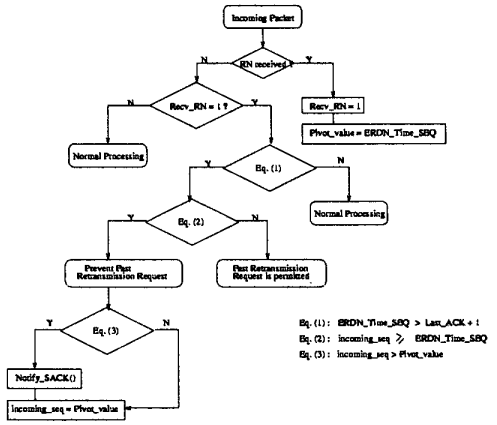

Fig. 4. Procedure to avoid unnecessary fast retransmissions performed at the destination.

D. Reliable Transmission of Control Messages

There exists a case where an ERSN packet can be discarded due to the congestion before it can make its way to the source. The ERSN packet is one of the most important control messages for notifying the source to transmit further data packets according to its current window. In order for the source to receive the ERSN packet reliably despite congestion, we have identified two possible approaches :

- A source periodically sends Probe messages to check if a PN has successfully acquired a new partial path to the destination (see Figure 5(a)).
- Each intermediate node is responsible for sending ERSN message reliably to its upstream node until it receives data packets or hears the echoed ERSN message from its immediate upstream node during the ERSN_RET_TIMER period (see Figure 5(b)).

From Figure 5(a), if ERSN message is discarded due to congestion at intermediate nodes, a new ERSN message will not be generated by the PN unless it successfully receives a Probe message sent by the source. Note that the Probe message itself can be discarded as a result of network congestion.

In the second approach (see Figure 5(b)), if an intermediate node forwards an ERSN message reliably to its upstream node, it will be able to receive data packets from the upstream node because the upstream node was buffering "on-the-fly" packets and will resume transmitting these packekts on receiving an ERSN message. Because there might be no buffered data packets at the upstream node, these data packets cannot be used as the sole indicator of successful ERSN transmission. To overcome this limitation, one can apply the passive acknowledgement technique used in packet radio. Basically, if intermediate node receives an ERSN message and forwards it to its upstream node, the forwarded ERSN message can also be heard by its downstream node. From this context, if an intermediate node cannot hear any packets or echoed ERSN message from its upstream node during ERSN_RET_TIMER period, it can initiate retransmission of the ERSN message.

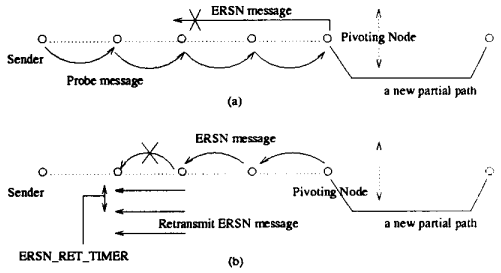

Fig. 5. Reliable transmission for control message: (a) Sender retransmits Probe message periodically, and (b) Intermediate node retransmits ERSN message periodically if unsuccessful.

IV. Simulation & Performance Evaluation

A. Simulation Environment

The underlying routing protocol used in ad hoc wireles networks can have an impact on the performance of reliable transport protocols because of the latency associated with recovery from route disconnection. In our simulator, a source-initiated on-demand routing protocol (ABR) is employed where link disconnection in a route is handled in a speedy manner using the localized query approach. As for a reliable transport protocol, TCP Reno is used in our simulation. Our simulator is written using a discrete-event simulation language, SMPL (Simulation Model Programming Language) [20]. For simplicity, TCP segments are not numbered as byte sequences but rather as segment count. Based on [14], the transmission rate of wireless link is set to be 19.2 kbps. The maximum window size is set to 60 segments and ssthreshold is set to 30 segments, initially. We employ TCP timeout estimation mechanism with parameters ($\alpha = 0.9, \beta = 2$). Additionally, in order to avoid the well-known *retransmission ambiguity problem*, we use *Karn's Algorithm*[18] for estimating RTT.

During the simulation, we assume that the path from a source to a destination consists of 20 nodes and these nodes are separated from each other by 50 meters. In order to simulate link disconnection, one of the links in the path is randomly selected and disconnected, periodically. In addition, we make use of a model of congestion along the path from the source to the PN. In our simulations, random links over the path experience the congestion during a given period. We assume that a source always has data packets to be sent. We simulated the TCP connection for 1000 seconds. Additionally, we assume that the errors at wireless link level can be recovered by using retransmission or selective ARQ control, therefore we focused our attention on the effect of route failures on TCP performance in a wireless ad hoc network.

B. Simulation Results & Observations

We measured and compared TCP throughput performance of three different variations of TCP implementations : General TCP, TCP-F, and TCP-BuS. In our simulation, throughput is defined as the amount of TCP segments successfully acknowledged by the destination during a given simulation time. Our first simulation test evaluates TCP throughput with respect to the frequency of route failures. In this simulation, a congestion has occurred 20 times during the simulation time. we set the duration of congestion as 10 seconds at random links. As shown in Figure 6, the throughput of each TCP implementation decreases as the route failures occur more frequently. In case of TCP-F and TCP-BuS, more frequent route failures cause the TCP source to stay longer in IDLE state. However, for general TCP, slow start procedures due to timeout events were invoked many times. These events have a negative effect on the throughput performance. However, among all the three TCPs examined, TCP-BuS shows the best performance, especially at high frequency of route failures. This is so due to the enhanced features incorporated in TCP-BuS.

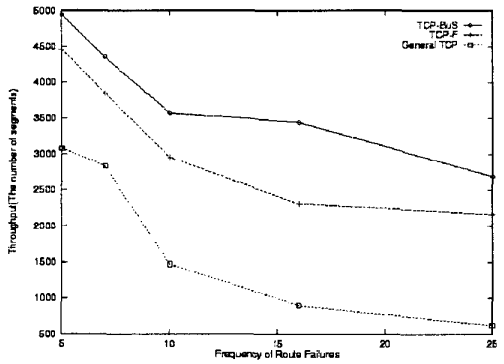

Fig. 6. TCP Throughput with respect to Frequency of Route Failures.

In the second simulation test, we measured throughput with respect to the delay incurred during route discovery. Because more time is incurred to complete the route discovery process, general TCP experienced more timeout events, resulting in substantial throughput degradation due to false activation of congestion control mechanisms. In TCP-F, although the TCP source stops sending further data packets during the route reconstruction, it invalidates every packets buffered at nodes on the old route, resulting in severe retransmissions of these packets. In addition, although the TCP source freezes all the variables related to timer values, it may still experience timeout events because it takes some time to perform route reconstruction. These events can lower communication throughput. Because TCP-BuS deals with the buffered data packets according to above mentioned four modifications, it yields better performance than the other schemes as shown in Figure 7.

In the third simulation test, we measured the impact of the duration of congestion on TCP throughput. Thanks to the selective retransmission mechanism for the lost packets after route reestablishment, TCP-BuS shows better performance than any other scheme. The longer a congestion exists, the better performance shows TCP-BuS due to early selective retransmission instead of relying on the timeout mechanism used in other schemes (see Figure 8).

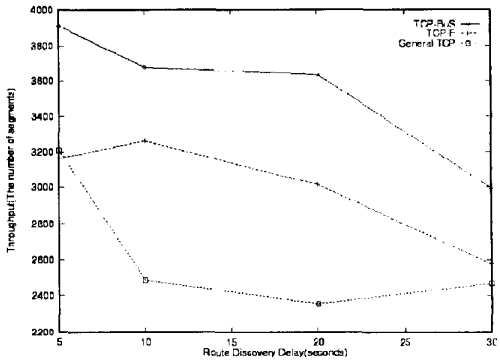

Fig. 7. TCP Throughput with respect to Route Discovery Delay

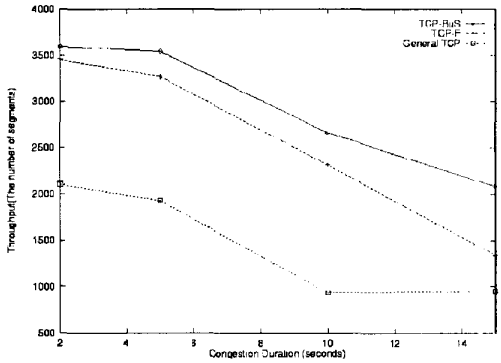

Fig. 8. TCP Throughput with respect to Congestion Duration

In this measure, we traced TCP segments sequences at the source to observe the behavior of the source (see Figure 9). In TCP-F and TCP-BuS, the discontinuities of the curves shown in Figure 9 represent causes of route failures, where the source stops further transmitting data packets. In general TCP, even if an intermediate node are performing a route recovery process, because the source is not notified of the route failure, the source continues to transmit data packets. This results in lower throughput than TCP-F and TCP-BuS. Discontinuity periods are observed because the source behaves based on the window mechanism and the limit of maximum window is reached. The "dip" in general TCP curve represents the occurrences of segment retransmissions which are more severe than TCP-F and TCP-BuS. Overall, TCP-BuS exhibits the fast increments in segment sequences which results in improvement of TCP throughput.

We also traced the congestion window size for each TCP scheme. As shown in Figure 10, the congestion wimdow size of TCP-BuS has the higher values than those of TCP-F and general TCP. It means that TCP-BuS is able to send the data packets more than others.

Finally, we examine the impact of control message (ERSN) reliability on TCP performance. we simulated the two approaches discussed in section III.A. In the case when the source generates periodic probe messages to acertain if a route reconstruction is successful, if the ERSN message is discarded at the intermediate nodes, the pivoting node can only generate an ERSN message after successfully receiving a probe message. Hence, this approach

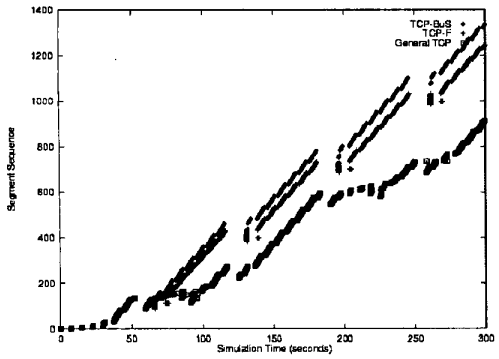

Fig. 9. Trace of Segement Sequence

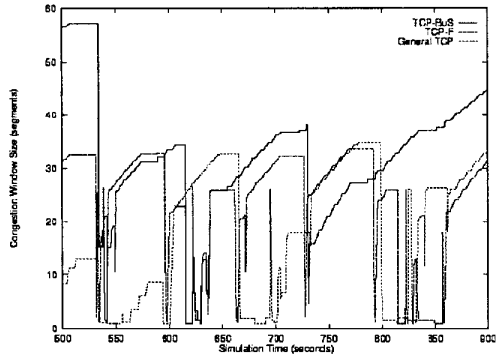

Fig. 10. Trace of Congestion Window Size

requires the source to wait for ERSN message. Therefore, as the probability of ERSN message being discarded increases, the degree of throughput degradation increases as shown in Figure 11. In the second approach, after the ERSN message is forwarded from an intermediate node to its upstream node, it can infer the successful delivery of ERSN message through the echoed ERSN message received from its upstream node. Because of the presence of ERSN retransmissions performed at the intermediate node which fails in transmitting the forwarded ERSN message to its upstream node, the ERSN message can be propagated to the source earlier than the source probing scheme, explaining why the latter performance is inferior to the former.

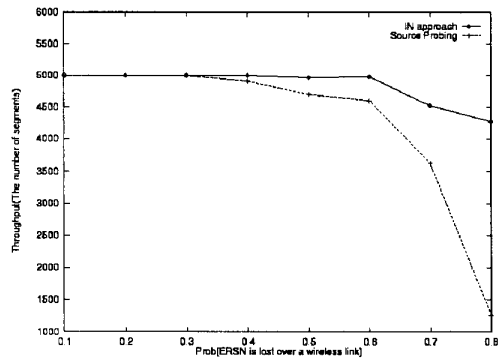

Fig. 11. Comparisons of Two Approaches for Transmitting ERSN Messages Reliably.

V. Conclusions

While some modified TCP approaches are proposed for a wireless cellular-style network(where the base station is able to masquerade the mobile node to the corresponding hosts), there exists little research devoted to the improvement of TCP performance in wireless ad hoc networks. In this paper, we present a novel approach for improving TCP performance by introducing the ERDN_Time_SEQ and ERDN_Recv_SEQ mechanism. In addition, we incorporate new measures to deal with the reliable transmission of important control messages. Our simulation results further confirmed these advantanges.

References

[1] C. Perkins and P. Bhagwat, "Highly Dynamic Destination Sequenced Distance Vector Routing(DSDV) for Mobile Computers," ACM SIGCOMM '94.

[2] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Network," Mobile Computing, edited by Tomasz Imielinski and Hank Korth, Chapter 5, Kluwer Academic Publishers, 1996.

[3] M. Gerla and J.T. Tsai, "Multicluster, mobile, multimedia radio network," Wireless Networks, Vol. 1, pp. 255-265, 1995.

[4] P. Krishna, M. Chatterjee, N.H. Vaidya and D.K. Pradhan, "A Cluster- based Approach for Routing in Ad-Hoc Networks," Second USENIX Symposium on Mobile and Location-Independent Computing, 1995.

[5] D.K.Kim, S.J.Ha and Y.H.Choi, "K-hop Cluster-based Dynamic Source Routing in Wireless Ad-Hoc Packet Radio Network," IEEE VTC,1998.

[6] Z.J.Haas and M.R. Pearlman, "The Zone Routing Procotol (ZRP) for Ad Hoc Networks," draft-ietf-manet-zone-zrp-02.txt, June, 1999.

[7] Z.J. Haas, "A New Routing Protocol for the Reconfigurable Wireless Networks," ICUPC, 1997.

[8] D.K. Kim, S.J. Ha and Y.H. Choi, "Variable-sized Cluster-based Dynamic Source Routing Protocol in Wireless Ad-Hoc Network with Variable Transmission Ranges," IEEE VTC '99, Houston, USA.

[9] Aruna Seneviratne, et al., "Cellular networks and mobile internet," Computer Communications vol. 21, pp. 1244 - 1255, 1998.

[10] A. Bakre, B.Badrinath, "I-TCP:indirect TCP for mobile hosts," ICDCS, 1995.

[11] H. Balakrishnan, S.Seshan, E.Amir, R.Katz, "Improving TCP/IP performance over wireless networks," MOBICOM '95.

[12] K.Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash, "A Feedback Based Scheme For Improving TCP Performance In Ad-Hoc Wireless Networks," ICDCS 1998.

[13] R. Cacares and L. Iftode, "Improving the performance of Reliable Transport Protocols in Mobile Computing Environments," IEEE JSAC, special issue on Mobile Computing Networks, 1994.

[14] B.Bakshi, P. Krishna, N. Vaidya, and D. Pradhan, "Improving Performance of TCP over Wireless Networks," ICDCS, 1997.

[15] C.-K. Toh, "Associativity Based Routing For Ad Hoc Mobile Networks," Wireless Personal Communications Journal, Special Issue on Mobile Networking & Computing Systems, Vol. 4, No. 2, March 1997.

[16] C.-K. Toh, "A Novel Distributed Routing Protocol for Multimedia Wireless LANs," Proceedings of IEEE International Phoenix Conference on Computers & Communications (IPCCC'96), March 1996, Arizona, USA.

[17] K. Fall and S. Floyd, "Comparisons of Tahoe, Reno and SACK TCP," ftp://ftp.ee.lbl.gov, March 1996.

[18] P. Karn and C. Partridge, "Improving Round Trip Time Estimate in Reliable Transport Protocols," ACM SIGCOMM 1987.

[19] E. Royer and C.-K. Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks," To appear in IEEE Personal Communications, 1999.

[20] M.H. MacDougall, "Simulating Computer Systems : Techniques and Tools," The MIT Press, 1987.