TCP-BuS: Improving TCP Performance in Wireless Ad Hoc Networks

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

Abstract: **Reliable data transmission over wireless multi-hop networks, called ad hoc networks, has proven to be non-trivial. TCP (Transmission Control Protocol), a widely used end-to-end reliable transport protocol designed for wired networks, is not entirely suitable for wireless ad hoc networks due to the inappropriateness of TCP congestion control schemes. Specifically, the TCP sender concludes that there is network congestion upon detecting packet losses or at time-outs. However, in wireless ad hoc networks, links are broken as a result of node mobility and hence some time is needed to perform route reconfiguration. During this time, packets could be lost or held back. Hence, the TCP sender could mistake this event as congestion, which is untrue. A route disconnection should 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 ongoing packets during a route disconnection and re-establishment. In addition to distinguishing network congestion from route disconnection due to node mobility, we also incorporate new measures to deal with reliable transmission of important control messages and exploitation of TCP fast recovery procedures. Our simulation compares the proposed TCP-BuS approach with general TCP and TCP-Feedback. Results reveal that TCP-BuS outperforms other approaches in terms of communication throughput under the presence of mobility.**

Index Terms: **TCP, Ad Hoc networks, transport protocols.**

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. In such networks, each host acts as a router to forward packets sent by the source to the receiver. Recently, several proposals on efficient routing protocols were suggested for wireless ad hoc networks [1]–[6]. However, reliable data transmission problem has not yet been examined thoroughly.

TCP (Transmission Control Protocol) [7] is widely used in the current Internet as the reliable end-to-end transport protocol. However, earlier research works [8]–[11] had confirmed that TCP cannot be directly applied to wireless networks due to the presence of the time-varying link characteristics and node mobility issues. If a TCP source does not receive the acknowledgement packets from the destination in a timely fashion, timeout events for the transmitted segments will occur. TCP assumes that congestion has occurred within the network and initiates the congestion control procedures. In general, TCP implements two phases for congestion control, namely: (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 packets will then be dropped. Congestion avoidance, on the other hand, provides the means for the source to deal with lost packets. The source 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 that can handle the abovementioned problems: (a) TCP Tahoe [12], and (b) TCP Reno [13].

Recently, most proposals for improving TCP performance have focused on the cellular-type wireless network (last-hop wireless network) [11], [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 [8] and SNOOP [9] 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 reconstruction procedures are frequently invoked during data transmission due to node movements. It is impossible that the buffering capability for masquerading is performed by the node detecting the route disconnection every time a route disconnection occurs. Moreover, route failure is unavoidable due to the inherent nature of the wireless ad hoc network. If the TCP used in existing wired networks is applied to wireless ad hoc networks, TCP performance will be degraded as it cannot distinguish congestion from route failure. Thus, in this paper, we propose a novel scheme for improving TCP performance in wireless ad hoc networks by introducing buffering capability in mobile nodes. Our proposed scheme takes advantage of the feedback information for detecting a route disconnection (which is also used by TCP-F [10]) as well as the features of the underlying ad hoc routing protocol. In particular, we include in-

Manuscript received November 17, 2000; approved for publication by Jaiyong Lee, Division III Editor, March 7, 2001.

D. Kim and Y. Choi are with School of Computer Science and Engineering, Seoul National University, Seoul, Korea, e-mail: pretty@mmlab.snu.ac.kr, yhchoi@mmlab.snu.ac.kr.

C.-K. Toh is with School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia, U.S.A., e-mail: cktoh@ece.gatech.edu.

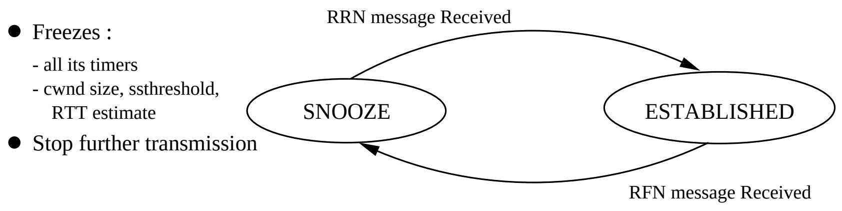

Fig. 1. Finite state machine of TCP-feedback.

telligent buffering techniques at mobile nodes. We selected the ABR (Associativity-Based Routing) [15], [16] protocol as the underlying routing protocol, which is a form of source-initiated on-demand protocol [17]. In addition, ABR advocates for stable and long-lived routes.

The rest of this paper is organized as follows. Section II describes the TCP-F scheme. Section III presents the basic operation of the ad hoc routing protocol referred to in this paper. Our proposed TCP-BuS scheme is then described in detail in Section IV, followed by simulation works in Section V. Some discussion is presented in Section VI. Finally, conclusions are made in Section VII.

II. RELATED WORK: TCP-FEEDBACK

Since our proposed scheme is based on TCP-Feedback (denoted by TCP-F) proposed by University of Texas at Dallas, we shall briefly describe its operation in this section. TCP-F allows the source to be informed of a route disconnection as a result of node mobility (via the use of RFN: Route Failure Notification message), so that the source does not unnecessarily invoke congestion control procedures, which can refrain the source from sending any further packets. This avoids the drop in communication throughput. The source then enters SNOOZE state. When the route is repaired, an RRN (Route Re-establishment Notification) message is sent back to the source to inform that communication can now resume.

When the source enters SNOOZE state, it performs the following steps:

1. The source stops transmitting data packets (be it new or retransmitted).
2. The source freezes: (a) all of its timers, and (b) the current congestion window size and values of other state variables such as the retransmission timer value. The source then starts a Route Failure Timer, whose value will depend on the worst case route re-establishment time (i.e., the maximum time needed to perform a successful route repair).
3. When the RRN message is received, data transmission is resumed, and all timers and state variables are restored.

Although the above-mentioned steps perform well in a wireless ad hoc network, TCP-F is also a timer-based TCP implementation. Once the route reconstruction is completed, it is possible that the data packets are retransmitted after timeout because it takes a discrete amount of time to get the segment acknowledgements back from the receiver. The slow start threshold, ssthreshold, is also set to one-half of the frozen window size value, and the cwnd is set to one TCP segment value [18]. This, therefore, results in a decrease in throughput performance. In addition, TCP-F does not consider the possible loss of RFN and RRN messages, which can influence TCP performance. Section IV will discuss this in detail.

III. AD HOC ROUTING PROTOCOL

Existing routing protocols for mobile ad hoc networks can generally be categorized into two classes: (a) proactive, and (b) reactive. In proactive scheme [1], all nodes maintain their routing tables for all possible destinations irrespective of the need of routes. However, in reactive scheme [15], routes in the source-receiver pair are acquired in on-demand fashion by the source. Therefore, it does not have to constantly maintain routing tables when there is no desire for routes. In [4], [6], some hybrid approaches are presented to take advantages of both reactive and proactive schemes. In this paper, we focus our discussion on reactive-based routing protocols. We have developed a variation of TCP suitable for ABR (Associativity-Based Routing) [15], [16], which belongs to a class of reactive routing protocols.

The ABR protocol is a source initiated on-demand routing protocol where a route is established in on-demand fashion. ABR consists of three phases; (a) Route Discovery Phase, (b) Route Reconstruction (RRC) 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 routing path information. Among these collected paths, the destination selects the best route based on selection criteria. In ABR, longevity of a route is the main metric used for route selection. 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 move 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. Fig. 2 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

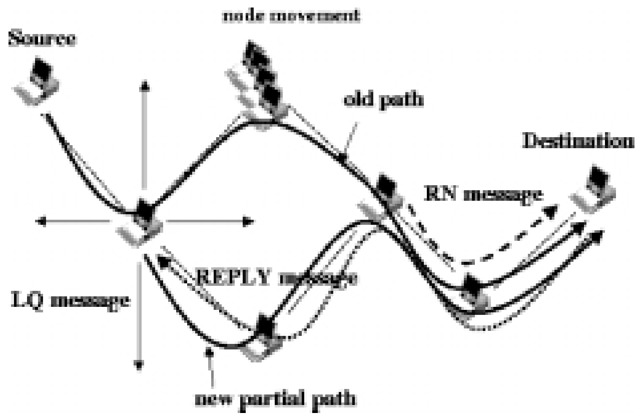

Fig. 2. An illustration of route recovery in ABR.

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 upstream node which has detected the route failure to resume forwarding data packets to the destination.

IV. PROPOSED SCHEME: TCP-BUS[1]

A. Key Features

We propose five features of enhancements to TCP and these are discussed below.

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 disconnection, and is propagated towards the source. After receiving an ERDN message, the source stops transmission. Similarly, after discovering a new partial path from the PN to the destination, the PN returns an ERSN (Explicit Route Successful Notification) message back to the source. On receiving ERSN message, the source resumes data transmission.
2. *Extending Timeout Values:* During the RRC phase, 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 re-establishment, it is possible that timeout events will occur because some time is needed 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 simplicity of implementation, we recommended the timeout values to be doubled.
3. *Selective Retransmission Requested by Receiver Node for Lost Packets:* Currently, in TCP, the retransmission of lost packets on the path due to congestion relies on a 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 expire. To cope with packet losses along the path from the source to the PN, an indication is made to the source so that it can retransmit the lost packet selectively before their timeout values expire 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 an RN (Route Notification) message. When a partial path is re-established, the destination can notify the source about the packets lost along the path. When notified, the source simply retransmits only the reported lost packets. However, in this process, other packets already 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 expected in-sequence packets arrive at the destination (via the fast retransmit method adopted by TCP-Reno). In our approach, these unnecessary 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 take action on the route failure only if it receives the ERDN message reliably. In addition, each intermediate node receiving the ERDN message should stop transmitting its buffered packets.
   The reliable transmission of ERDN message will depend on the link and network layers. If a node A (including PN) reliably sends an ERDN message to its upstream node B, the ERDN message subsequently forwarded by node B can be overheard by node A. Thus, if a node has sent an ERDN message but cannot overhear any ERDN message relayed by its upstream node during the ERDN_RET_TIMER period, it concludes that the ERDN message is lost and will attempt 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 the ERSN message. However, the ERSN message can also be discarded in the network due to congestion. Consequently, only if the source receives the ERSN message from a PN, will it resume data transmission. Therefore, it is very important for the PN to send the ERSN message reliably to the source.

[1]TCP with BUffering capability and sequence information

Table 1. Control messages and their parameters.

Related to Route Maintenance Invoked by Mobility		
Type of Control Message	**Parameters**	**Remarks**
ERDN	ERDN_GEN_SEQ	To notify the source of route failure
ERSN	Last_ACK	To notify the source of successful route re-establishment

Related to Existing ABR Protocol		
Type of Control Message	**Parameters**	**Remarks**
RN	Nothing additional	To invalidate the route/buffered packets towards the destination
LQ	ERDN_GEN_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 ERSN_RET_TIMER	Duration set to reliably transmit ERSN (or ERDN) control message towards the source

We examine 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. This probing can continue until the source receives the ERSN message or it times out. The other approach relies on the intermediate node to retransmit the ERSN message if it does not receive any data packets or an echoed ERSN message during ERSN_RET_TIMER after relaying the message to its upstream node.

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 (such as ABR), "on-the-fly" packets transiting 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 additional buffering capability at mobile hosts.

Two approaches are considered to implement TCP-BuS. One is that in order to conserve the end-to-end semantics of TCP protocol, ABR routing protocol is modified to generate additional control messages such as ERDN and ERSN with the parameters of segment information and some messages such as LQ and REPLY are extended to convey the segment information, and TCP-BuS (sender or receiver) performs the corresponding functions enumerated in Section IV-A by dealing with events invoked by ABR protocol using several types of primitives used at the interface between transport and network layers. The other is that each node (including intermediate nodes) is equipped with TCP-BuS (i.e., transport layer), since each node acts as a router and an end-host in wireless ad hoc network even if a tranport protocol such as TCP is an end-to-end protocol and conventional routers used in the wired networks do not have a transport protocol. Therefore, on detecting route failure, each TCP-BuS of intermediate nodes along the existing path performs the corresponding function enumerated in Section IV-A with the help of ABR routing protocol in which some existing messages such as LQ and REPLY are modified to contain the segment information. In order to minimize the functions of ABR protocol, TCP-BuS of each intermediate node is invoked to deal with the route failure and congestion in this paper. The parameters and control messages are:

- ERDN_GEN_SEQ: When an intermediate node detects a route failure and it cannot forward the buffered data packets, ERDN_GEN_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_GEN_SEQ information is propagated from the PN to the source via the ERDN message.
- ERDN_RCV_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_RCV_SEQ is defined as the sequence number of the last TCP segment sent until the TCP source receives an ERDN control packet.
- Last_ACK: During route reconstruction, the destination responds to the LQ message with a REPLY. Last_ACK is therefore defined as the sequence number of the last segment which the destination has received successfully.

By using ERDN_GEN_SEQ and ERDN_RCV_SEQ mentioned above, the following information can be inferred:

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

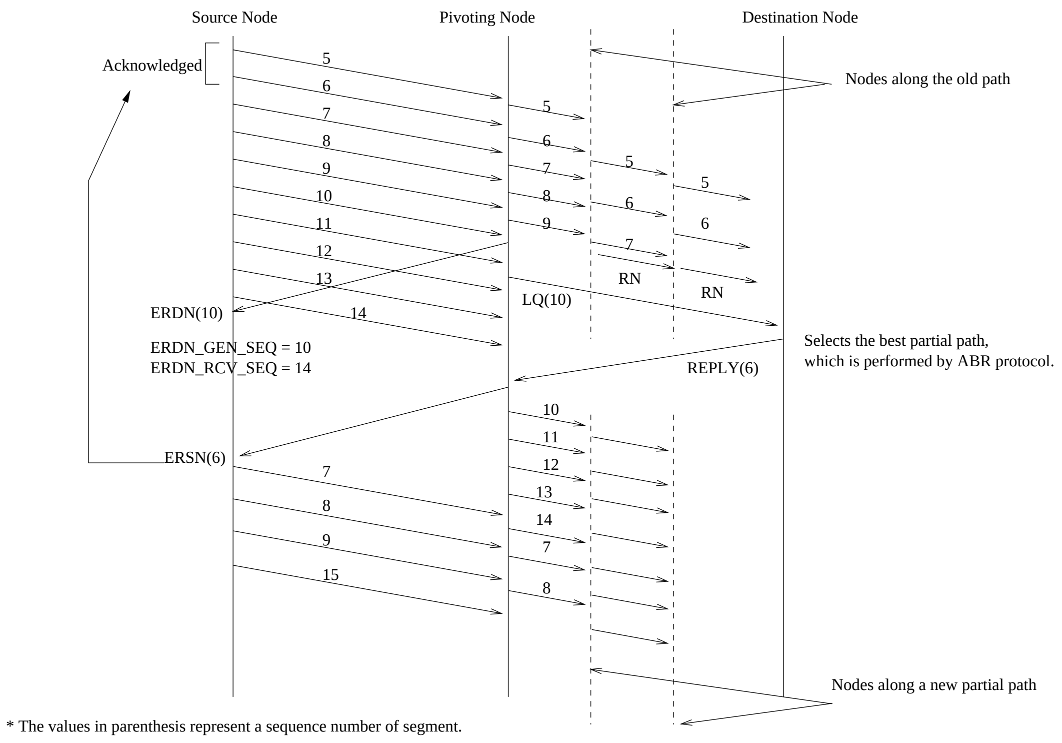
Source Node
Pivoting Node
Destination Node
Acknowledged
Nodes along the old path
RN
LQ(10)
ERDN(10)
ERDN_GEN_SEQ = 10
ERDN_RCV_SEQ = 14
Selects the best partial path, which is performed by ABR protocol.
REPLY(6)
ERSN(6)
Nodes along a new partial path
* The values in parenthesis represent a sequence number of segment.

Fig. 3. An example illustrating ERDN_GEN_SEQ and ERDN_RCV_SEQ mechanisms.

from the source to the PN.

C. *Operation of TCP-BuS*

Since ad hoc routes can be invalidated by node 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

At the source, TCP-BuS 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 usual. 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 as in TCP-F.

Next, the ERDN_GEN_SEQ value is extracted from the ERDN message and the ERDN_RCV_SEQ is calculated. As an example (see Fig. 3), a pivoting node detects a route failure when it has a segment(10) to transmit. The PN generates ERDN message containing the sequence number(10) of the segment in the head of line of its transmit queue. Therefore, when the source receives the ERDN message, ERDN_GEN_SEQ is set to 10. Meanwhile, the source has been sending segments up to the segment(14). From this, we can calculate ERDN_RCV_SEQ which is set to 14. Additionally, the next downstream node from PN will send RN message towards the destination, which invalidates the old partial path and flushes buffered packets along that path. Since ERDN message indicates that there is a route failure in the network, the source just waits for an ERSN message. On receiving this ERSN message, the source interprets that route re-establishment is successful. The source is then able to resume data transmission according to the TCP window-based mechanism.

On receiving ERSN message after a successful route reconstruction (i.e., after the PN node receives the REPLY message used in ABR, it generates the ERSN message with the Last_ACK parameter(6) that is extracted from REPLY message.), the source can increase the congestion window (cwnd) by the amount of the acknowledged packets. At the same time, the source can safely assume that the segments whose sequence numbers range from (Last_ACK + 1) to (ERDN_GEN_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. In Fig. 3, since ERSN message includes the segment sequence number(6) up to which the destination has received successfully, the source becomes aware that it should retransmit the segments from sequence number(7) to sequence number(9) which have been discarded along the old partial path. 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 router's 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_GEN_SEQ to ERDN_RCV_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 packet losses are experienced on the partial path

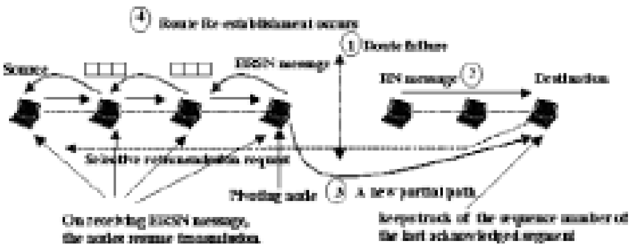
4 Route Re-establishment occurs
1 Route failure
Source
ERSN message
LQ message 2
Destination
Selective retransmission request
Pivoting node
3 A new partial path
On receiving ERSN message, the nodes resume transmission.
keeps track of the sequence number of the last acknowledged segment

Fig. 4. Sequence of events occurring after a successful route reestablishment.

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 reduces the congestion window size accordingly.

C.2 TCP-BuS Functions at Intermediate Node

After a 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 their transmission. After receiving the REPLY message, the PN notifies the source of successful route reestablishment via the ERSN message, which also includes the Last_ACK information. At each intermediate node receiving the ERSN message, transmission of buffered packets resumes. This is illustrated in Fig. 4. In addition, intermediate nodes perform the process of reliable transmission of control messages, i.e., ERDN and ERSN messages (Refer to Section IV-D).

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 for efficient flow control [18]. We proposed an additional selective retransmission scheme to cope with lost packets due to congestion on the partial path from the source to the receiver. A request for selective retransmission of lost packets is generated at the receiver on detecting the absence of consecutive segment sequence. This requires the source to react to the congestion.

Given 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_GEN_SEQ-1) will arrive at the destination later than those packets having sequence numbers ranging from ERDN_GEN_SEQ to ERDN_RCV_SEQ. As a result, the destination continues to request for fast retransmission by sending duplicated ACK packets for each incoming packet received due to discrepancies in the sequence order. To avoid this problem, an additional procedure at the destination (See Fig. 5) is required after receiving the LQ message. When receiving an LQ packet during a route recovery process, the destination extracts the ERDN_GEN_SEQ value piggybacked in the LQ packet.

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 subsequent segments are lost due to congestion. Therefore, the receiver notifies the source of lost segment information selectively (notify_SACK()[2] in Fig. 5).

- On receiving LQ message for route extension, Pivot_value = ERDN_RCV_SEQ.
- If incoming_SEQ $\geq$ ERDN_GEN_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.

D. Reliable Transmission of Control Messages

There can exist a case where an ERSN packet can be discarded due to the presence of congestion before it can make its way to the source. The ERSN packet is one of the most important control messages used to notify the source to resume data transmission 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 Fig. 6(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 Fig. 6(b)).

From Fig. 6(a), if the 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 Fig. 6(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 packets 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 an 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

[2]The receiver sends the source the Pivot_value and the distance of segment sequence that denotes how many segments are lost from the Pivot_value.

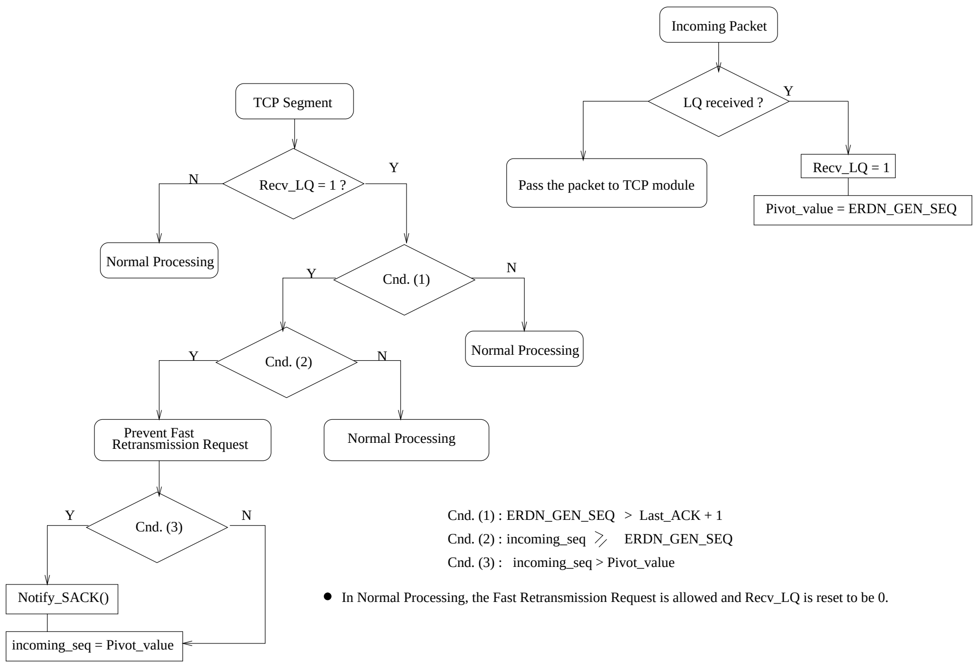
Incoming Packet
LQ received ?
Y
Pass the packet to TCP module
Recv_LQ = 1
Pivot_value = ERDN_GEN_SEQ
TCP Segment
Recv_LQ = 1 ?
N
Y
Normal Processing
Cnd. (1)
Y
N
Normal Processing
Cnd. (2)
Y
N
Prevent Fast Retransmission Request
Normal Processing
Cnd. (3)
Y
N
Notify_SACK()
incoming_seq = Pivot_value
Cnd. (1) : ERDN_GEN_SEQ > Last_ACK + 1
Cnd. (2) : incoming_seq ≯ ERDN_GEN_SEQ
Cnd. (3) : incoming_seq > Pivot_value
● In Normal Processing, the Fast Retransmission Request is allowed and Recv_LQ is reset to be 0.

Fig. 5. Procedure to avoid unnecessary fast retransmissions, performed at the destination.

during the ERSN_RET_TIMER period, it can initiate retransmission of the ERSN message.

V. SIMULATION & PERFORMANCE EVALUATION

A. Simulation Environment

The underlying routing protocol used in wireless ad hoc 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. Our simulator didn't implement MAC (Medium Access Control) protocol and Physical layers. Instead, we focused on the routing and transport protocols. When it comes to MAC and physical layers, an omnidirectioanl antenna radios are assumed and an ad hoc MAC based on CSMA/CA is assumed. 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) [19]. For simplicity, TCP segments are not numbered as byte sequences but rather as segment count. We assume that the transmission rate of wireless link is 19.2 kbps, a very narrow-band rate. We employ TCP timeout estimation mechanism with parameters, $\alpha = 0.9$ and $\beta = 2$. Additionally, in order to avoid the well-known *retransmission ambiguity problem*, we use *Karn's Algorithm* [20] for estimating RTT.

We assume that all nodes are separated from one another by 50 meters. In order to simulate link disconnection, one of the links in the path is randomly selected and disconnected periodically. We assume that a source always has data packets to send. Furthermore, we used a small segment size of 640 bytes. Additionally, we assume that errors occurred over the wireless link level can be recovered by using retransmission or selective ARQ

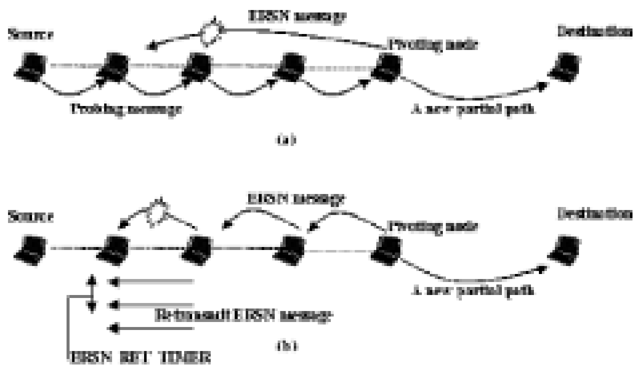

Fig. 6. Reliable transmission techniques for control messages: (a) sender retransmits Probe message periodically, and (b) intermediate node retransmits ERSN message periodically if unsuccessful.

(Automatic Repeat Request). We, therefore, focus 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 implementations of TCP: (a) General TCP, (b) TCP-F, and (c) TCP-BuS. In our simulation, throughput is represented by "Delivery Rate," which is defined as the ratio of the number of segments sent successfully to the receiver and the number of segments buffered and waiting to be sent at the source.

First, we examine the impact of route length on TCP throughput. For simplicity of simulation, the congestion of a partial path from the source to the PN is ignored in this test. In addition, the maximum segment size is initially set to 40 segments (ssthreshold is set to 20 segments). During the simulation,

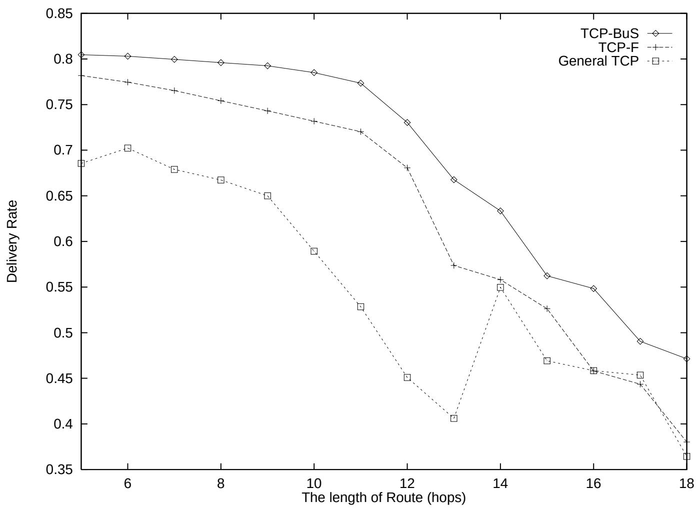

Fig. 7. TCP throughput with respect to route length.

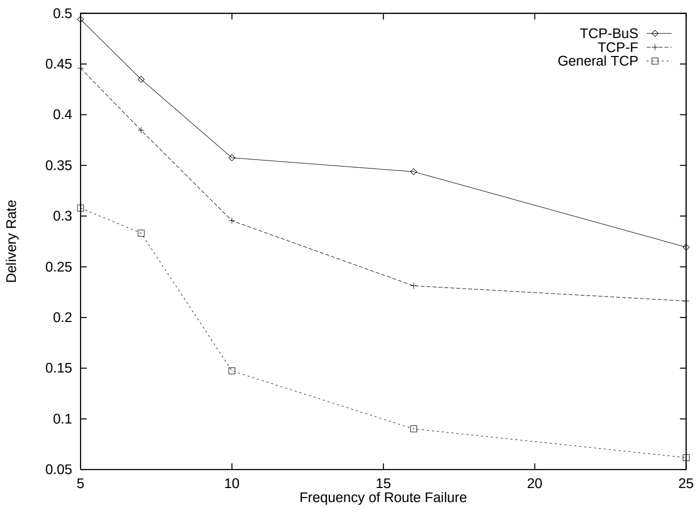

Fig. 8. TCP throughput with respect to frequency of route failures.

route failures occurred 20 times, where each route failure lasts for 10 seconds. As shown in Fig. 7, a longer route makes a decrease in TCP throughput. In addition, compared to TCP-F and General TCP, TCP-BuS achieves improvements of average 15% and 30%, respectively. TCP-F and General TCP have similar throughput at longer routes due to the operation of timeout mechanism and the limitation of sliding window mechanism.

After this, we assume a long path from a source to a destination consisting of 20 nodes, which can be applied to wide area ad hoc network. Our second simulation test evaluates TCP throughput with respect to the frequency of route failures. In this test, we do not ignore the possible presence of congestion along the path from the source to the PN. In our simulation, random links over the path experience congestion. We assume that congestion has occurred 20 times during the simulation time. We set a congestion duration of 10 seconds over randomly selected links. To better differentiate the difference of throughput of TCP schemes, initially, we choose a maximum window size and the ssthreshold of 60 segments and 30 segments, respectively. As shown in Fig. 8, the throughput of each TCP implementation decreases as the frequency of route failures increases. In case of TCP-F and TCP-BuS, 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 schemes examined, TCP-BuS shows the best performance, especially at high frequency of route failures (average 10% and 30% improvement over TCP-F and General TCP, respectively).

Furthermore, because delay is incurred to complete the route recovery process, General TCP experienced many timeout events, resulting in substantial throughput degradation due to false activation of congestion control mechanisms. In TCP-F, although the TCP-F source stops sending further data packet on detection of route failure, it invalidates every packets that are buffered at nodes on the old route, resulting in heavy retransmissions of these packets. In addition, although the TCP-F source freezes all the variables related to timer values, it may still experience timeout events because some time is needed to perform

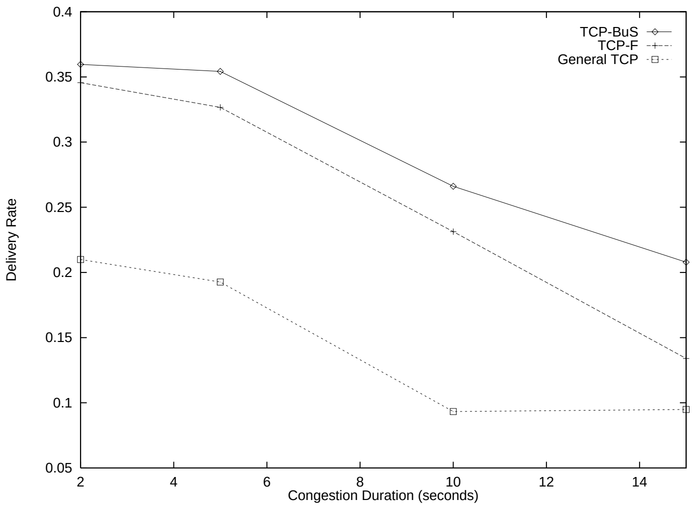

Fig. 9. TCP throughput with respect to congestion duration.

route reconstruction. These events can lower communication throughput. Because TCP-BuS deals with buffered data packets according to previously mentioned four modifications, it yields better performance than other schemes.

In the third simulation test, we measured the impact of congestion duration on TCP throughput. Thanks to the selective retransmission mechanism for the lost packets after route reestablishment, TCP-BuS shows better performance than other schemes. The longer the congestion exists, the resultant delivery rate is lower. TCP-BuS shows the better performance by using the early selective retransmission scheme instead of relying on the timeout mechanism used by other schemes (see Fig. 9).

We also traced TCP segments sequences observed at the source to study the behavior of TCP-BuS, TCP-F, and General TCP (see Fig. 10). In TCP-F and TCP-BuS, the discontinuities of the curves shown in Fig. 10 represent cases of route failures, where the source stops further transmission of data packets. In General TCP, even if an intermediate node is performing a route recovery process, because the source is not notified of the route failure, it still continues to transmit data packets. This results in lower throughput than TCP-F and TCP-BuS. Discontinuity periods are observed because the source uses a 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. It can be observed that TCP-F also experiences some segment retransmissions after route reconstruction as mentioned above. Overall, TCP-BuS exhibits fast increments in segment sequences which makes an improvement in TCP throughput.

In addition, the congestion window size for each TCP scheme was also traced. As shown in Fig. 11, the congestion window size of TCP-BuS has higher value than those of TCP-F and General TCP. TCP-BuS is thus capable of sending more data packets than others. Note that each curve shows the phases of slow-start, congestion avoidance as well as the AIMD (Additive Increase and Multiplicative Decrease [7]) effect.

For the above simulations, we assumed that a route failure could occur at random links over a path between the source and the receiver nodes. In addition, we measured the performance of TCP throughput according to the position of route failure. In this simulation, route failures occurred at the fixed points and we assumed that the maximum window size was 30 segments. To simplify our simulation, the congestion did not occur over the partial path from the source to the PN. As shown in Fig. 12, TCP-BuS yields better performance than other schemes irrespective of the location of link failure. It is observed that TCP-F and General TCP show similar performance when link failure occurs at a location far away from the source. This is because both schemes cannot send further data packets due to the limitation of the sender window size before it is notified of route failure. All schemes show higher throughput when a failed link is farther away from the source. This is because the time needed to discover a shorter partial path is smaller. In particular, TCP-BuS yields higher throughput due to the presence of segment buffering on the path from the source to the PN.

In our simulation, we also evaluate TCP performance by varying the maximum window size. As the maximum window size is increased, the throughput is also increased for all three schemes. It is shown in Fig. 13 that TCP-BuS has the highest performance.

Lastly, we examine the impact of control message (ERSN) transmission reliability on TCP performance. However, even though the errors can be recovered at link layer due to retransmission mechanisms, it is also possible to lose the control messages due to the buffer overflow at network layer caused by congestion. Therefore, we simulated the two possible approaches, as discussed in Section IV-D. In the case when the source generates periodic probe messages to ascertain if a route reconstruction is successful, if the ERSN message is discarded at the intermediate nodes, the PN can only generate an ERSN message after successfully receiving a probe message. Hence, this approach requires the source to wait for an ERSN message. Therefore, when the probability of ERSN message being discarded increases, the degree of throughput degradation also increases, as shown in Fig. 14.

For 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 retransmission mechanism performed at each intermediate node, the ERSN message can be propagated to the source much earlier than the source probing approach, which explains why the latter performance is more inferior.

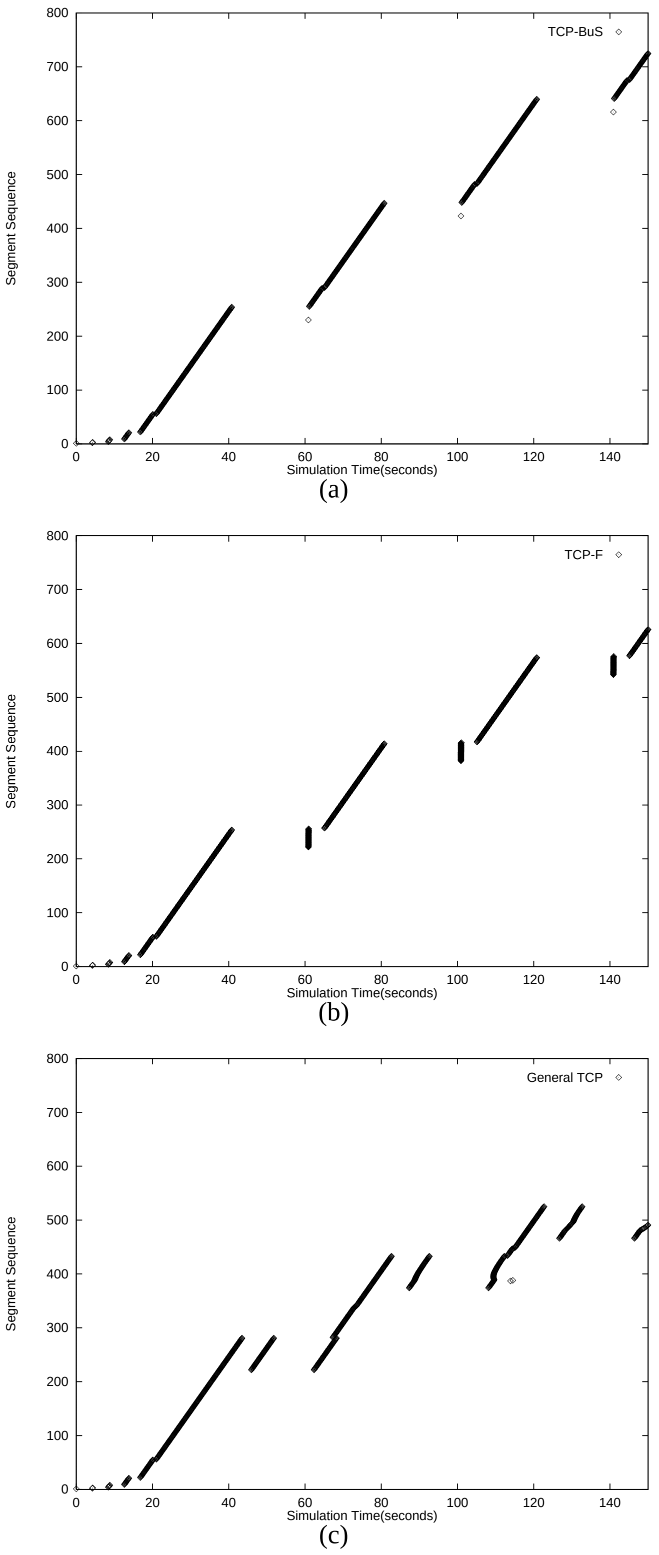

Fig. 10. Trace of segment sequences: (a) TCP-BuS, (b) TCP-F, and (c) general TCP.

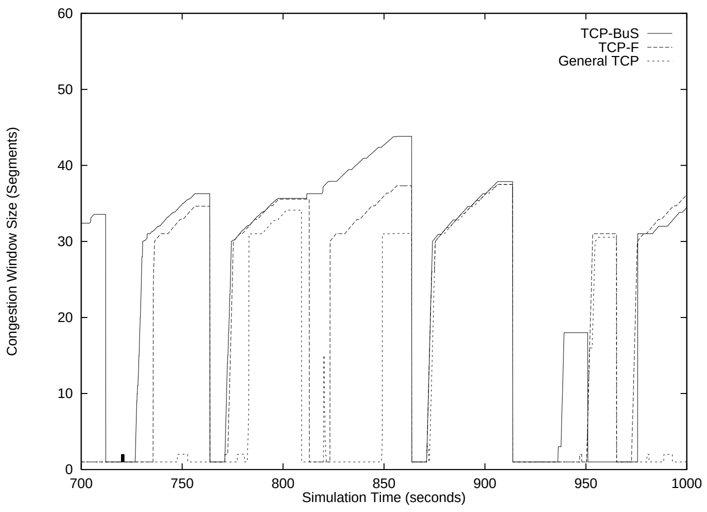

Fig. 11. Trace of congestion window size.

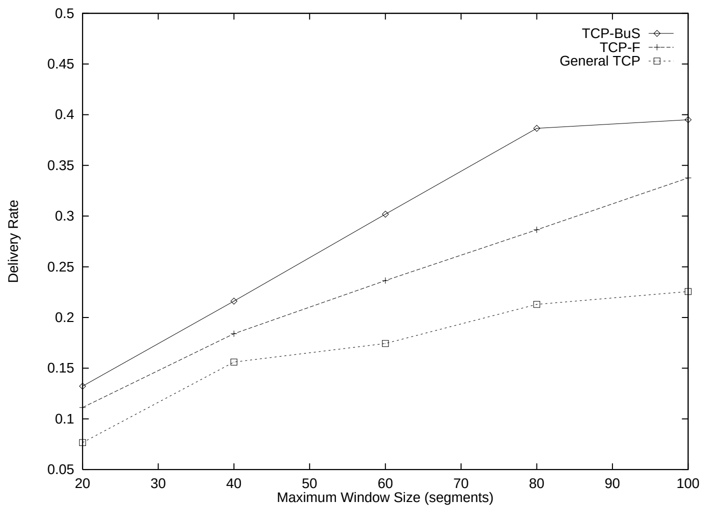

Fig. 13. Comparisons of TCP throughput according to maximum window size.

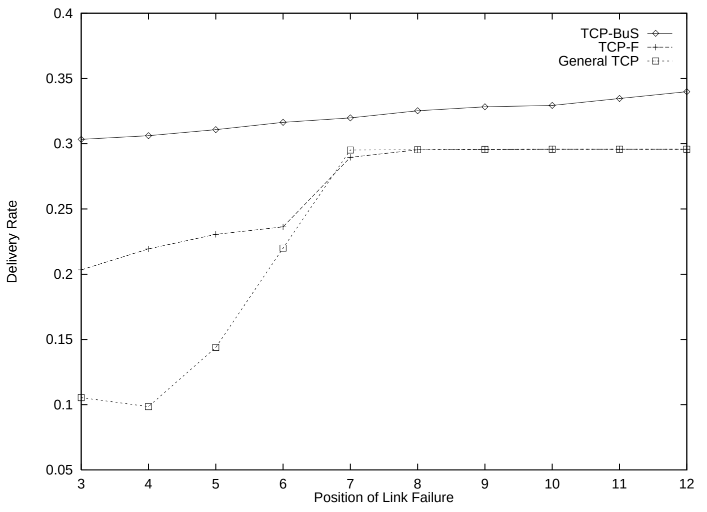

Fig. 12. Comparisons of TCP throughput according to location of link failure.

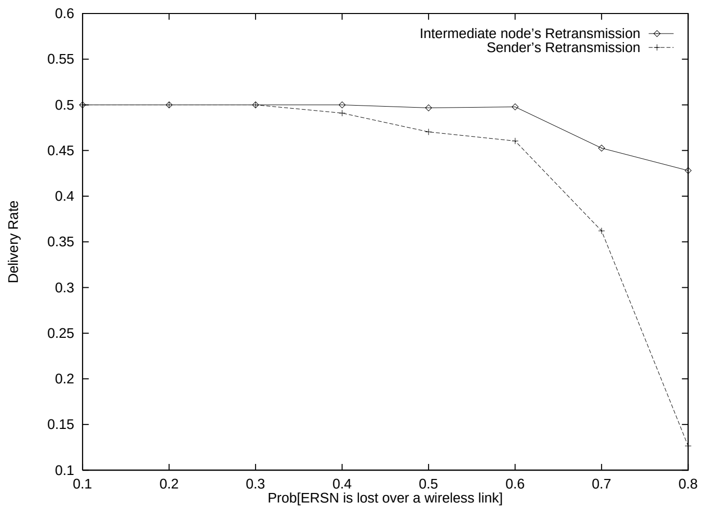

Fig. 14. Impact of control message transmission reliability on TCP throughput.

VI. DISCUSSION

This paper identified the need to communicate routing information to the transport layer in wireless ad hoc networks. Thus, TCP-BuS takes advantage of the underlying routing protocol (especially with source-initiated on-demand routing protocol) efficiently for improving the performance of TCP protocol. TCP has traditionally been used in wired networks where intermediate routes in the route are not mobile. This implies that TCP cannot be used for wireless ad hoc networks since multiple wireless links are now present and routers are mobile. Past research in supporting TCP over wireless last-hop cellular has been to split TCP connections into segments. An example of such work is Indirect-TCP [8]. Even Indirect-TCP breaks the so called "end-to-end semantics" of TCP. One cannot force traditional protocols designed for wired networks to work in mobile networks without modifications or enhancements. TCP-BuS, therefore, is one such protocol that enables TCP applications to be supported in an wireless ad hoc network environment and at the same time enhancing TCP performance. The advantages of TCP-BuS can be exploited by most source-initiated on-demand routing protocols and hence it is not specific to ABR even if this paper applied TCP-BuS to ABR protocol. It is important to realize that as we move into new networking platforms, new or enhanced protocols will be evolved. Future research includes addressing the internetworking issues of a TCP connection that spans from a fixed wired network to an ad hoc wireless network. An internetworking function unit, for example, gateway, may be needed to transform the function of traditional TCP into TCP-BuS and vice verse (see Fig. 15).

VII. CONCLUSIONS

In this paper, we examine issues related to TCP communications over ad hoc wireless networks. In particular, we reveal the confusion faced by a TCP sender - that of delay and packet loss due to route reconstruction as a result of host mobility and that of network congestion. We improve TCP performance by proposing intelligent buffering and sequence checking techniques.

We introduce the ERDN_GEN_SEQ and ERDN_RCV_SEQ mechanism, which adjusts timeout values to compensate for

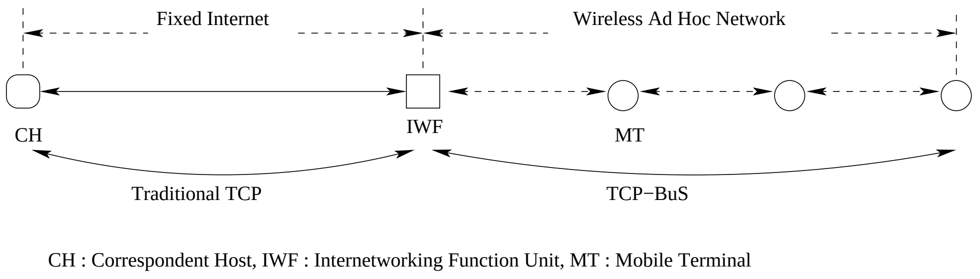
Fixed Internet
Wireless Ad Hoc Network
CH
IWF
MT
Traditional TCP
TCP−BuS
CH : Correspondent Host, IWF : Internetworking Function Unit, MT : Mobile Terminal

Fig. 15. Internetworking with fixed Internet.

route reconstruction time, and avoids unnecessary requests for fast retransmission and selective retransmission of lost packets. In addition, we incorporate new measures to deal with reliable transmission of important control messages used to inform the source about mobility and status of route reconstruction. We compare the performance of our proposed TCP-BuS Scheme with that of other TCP implementations (TCP-F and General TCP) via simulation. Simulation results demonstrate that TCP-BuS outperforms the others in terms of throughput under different conditions (route length, frequency of route failures, congestion duration, and location of link failure).

ACKNOWLEDGMENTS

This work was supported in part by the Brain Korea 21 project of Ministry of Education, in part by the National Research Laboratory project of Ministry of Science and Technology, and in part by Agency of Defense Development, 2000, Korea. This paper is an extended version of our paper to be presented at IEEE ICC 2000, New Orleans, 2000.

REFERENCES

[1] C. Perkins and P. Bhagwat, "Highly dynamic destination sequenced distance vector routing (DSDV) for mobile computers," in *Proc. ACM SIGCOMM'94*, 1994.
[2] M. Gerla and J. T. Tsai, "Multicluster, mobile, multimedia radio network," *Wireless Networks*, vol. 1, pp. 255–265, 1995.
[3] D. K. Kim, S. J. Ha and Y. Choi, "K-hop cluster-based dynamic source routing in wireless Ad-Hoc packet radio network," in *Proc. IEEE VTC'98*, 1998.
[4] Z. J. Haas and M. R. Pearlman, "The zone routing protocol (ZRP) for Ad Hoc networks," draft-ietf-manet-zone-zrp-02.txt, June, 1999.
[5] Z. J. Haas, "A new routing protocol for the reconfigurable wireless networks," in *Proc. IEEE ICUPC'97*, 1997.
[6] D. K. Kim, S. J. Ha, and Y. Choi, "Variable-sized cluster-based dynamic source routing protocol in wireless Ad-Hoc Network with variable transmission ranges," in *Proc. IEEE VTC'99*, Houston, USA, 1999.
[7] D. E. Comer, *Internetworking with TCP/IP*, vol. 1, Principles, Protocols, and Architecture, Fourth ed., 2000.
[8] A. Bakre and B. Badrinath, "I-TCP: Indirect TCP for mobile hosts," *ICDCS*, 1995.
[9] H. Balakrishnan *et al.*, "Improving TCP/IP performance over wireless networks," *MOBICOM'95*, 1995.
[10] K. Chandran *et al.*, "A feedback based scheme for improving TCP performance in Ad-Hoc wireless networks," *ICDCS*, 1998.
[11] R. Cacares and L. Iftode, "Improving the performance of reliable transport protocols in mobile computing environments," *IEEE J. Select. Areas Commun.*, (*Special Issue on Mobile Computing Networks*), 1994.
[12] V. Jacobson, "Congestion avoidance and control," in *Proc. ACM SIGCOMM'88*, Stanford, Aug. 1988.
[13] W. Stevens, "TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," RFC 2001, Jan. 1997.
[14] A. Seneviratne *et al.*, "Cellular networks and mobile internet," *Computer Commun.*, vol. 21, pp. 1244–1255, 1998.
[15] C.-K. Toh, "Associativity based routing for Ad Hoc mobile networks," *Wireless Personal Commun. J.*, (*Special Issue on Mobile Networking & Computing Systems*), vol. 4, no. 2, Mar. 1997.
[16] C.-K. Toh, "A novel distributed routing protocol for multimedia wireless LANs," in *Proc. IPCCC'96*, Arizona, USA, Mar. 1996.
[17] E. Royer and C.-K. Toh, "A review of current routing protocols for Ad Hoc mobile wireless networks," *IEEE Personal Commun. Mag.*, pp. 46–55, Apr. 1999.
[18] K. Fall and S. Floyd, "Comparisons of Tahoe, Reno and SACK TCP," Mar. 1996. Available at ftp://ftp.ee.lbl.gov.
[19] M. H. MacDougall, "Simulating computer systems: Techniques and tools," The MIT Press, 1987.
[20] P. Karn and C. Partridge, "Improving round trip time estimate in reliable transport protocols," in *Proc. ACM SIGCOMM'87*, 1987.

Dongkyun Kim received B.S. at Department of Computer Engineering, Kyungpook National University, and M.S. at Department of Computer Engineering, Seoul National University. Now, he is a Ph.D. candidate at School of Computer Science and Engineering, Seoul National University. He has been a visiting scholar to Georgia Institute of Technology, supervised by Dr. Toh since 1999. Also, he is a vice-chair of the IEEE TCPC Subcommittee on Ad Hoc Wireless Networking. His research interest is in High-speed Networks, Wireless Ad Hoc Network, Internet Protocols, and Future Wireless Networks.

C.-K. Toh was educated at the EEE department, University of Manchester Institute of Science & Technology and the Computer Laboratory, Cambridge University, England, in Electrical Engineering and Computer Science. Dr. Toh is the inventor of the Cambridge Ad Hoc mobile routing protocol, which was awarded a US patent. He authored the book on "Wireless ATM & Ad-Hoc Networks" which was published by Kluwer Academic Press in 1996. Dr. Toh is an Editor for IEEE Journal on Selected Areas in Communications (JSAC), an Area Editor on wireless networking for IEEE Communications Survey Journal, a Feature Editor for ACM Mobile Computing & Communications Review and serves as an Editorial Board member for IEEE Network Magazine, Journal of Communications & Networks (IEEE COMSoc), and Springer-Verlag Personal Technologies Journal. He serves as a Technical Chair and Technical Program Committee Member for several IEEE and ACM conferences. As of 1998, he joined Georgia Institute of Technology, School of Electrical and Computer Engineering, directing the Mobile Multimedia and Networking Laboratory. In May 2000, he was named GeorgiaTech Teaching Fellow. He is also founder and director of the Ad Hoc Wireless Networking and Computing Consortium. Prof. Toh is a Chartered Electrical Engineer, of IEE UK and a registered Chartered Engineer (CEng) of the Engineering Council, London. Prof. Toh is acting Chairman of the newly formed IEE (UK) Atlanta Center based in Atlanta, Georgia. Currently, he serves as Chairman of IEEE TCPC Subcommittee on Ad Hoc Mobile Wireless Networks. He is listed in MARQUIS Who'sWho in the World and Who'sWho in Science and Engineering.

Yanghee Choi received B.S. in electronics engineering from Seoul National University, M.S. in electrical engineering from Korea Advanced Institute of Science, and Doctor of Engineering in Computer Science from Ecole Nationale Superieure des Telecommunications (ENST) in Paris, in 1975, 1977, and 1984, respectively. Before joining the School of Computer Engineering, Seoul National University in 1991, he has been with Electronics and Telecommunications Research Institute (ETRI) during 1977–1991, where he served as director of Data Communication Section, and Protocol Engineering Center. He is now leading the Multimedia Comminications Laboratory in Seoul National University. He is also director of Computer Network Research Center in Research Institute of Advanced Computer Technology (RIACT). He is currently editor-in-chief of Korea Information Science Society journals. He was chairman of the Special Interest Group on Information Networking. He is now associate dean of research affairs at Seoul National University. His research interest lies in the field of multimedia systems and high-speed networking.