A Transport Protocol For Supporting Multimedia Streaming in Mobile Ad Hoc Networks

Zhenghua Fu, *Student Member, IEEE,* Xiaoqiao Meng, *Student Member, IEEE,* and Songwu Lu, *Member, IEEE*

***Abstract*— Transport protocol design for supporting multimedia streaming in mobile ad hoc networks is challenging because of unique issues, including mobility-induced disconnection, reconnection, and high out-of-order delivery ratios; channel errors; and network congestion. In this work, we describe the design and implementation of a TCP-friendly transport protocol for ad hoc networks. Our key design novelty is to perform multi-metric joint identification for packet and connection behaviors based on end-to-end measurements. Our ns-2 simulations show significant performance improvement over wired TCP friendly congestion control and TCP with explicit-link-failure-notification (ELFN) support in ad hoc networks.**

I. Introduction

The proliferation of wireless communication technology has spurred growing interests in the use of mobile ad hoc wireless networks. Ad hoc networks offer users convenient wireless communications without any infrastructure support. With the advent of IEEE 802.11a/b technology, the wireless channel typically supports a data rate up to 11Mbps or even higher, which makes real-time multimedia streaming possible. Potential applications of such a streaming technology include radio broadcasting, voice/video conferencing, real-time environmental monitoring, and distributed gaming.

Most existing research on multimedia streaming over ad hoc networks focuses on lower-layer design such as service differentiation in MAC layer [27], QoS-aware routing, admission control [25] and adaptive packet scheduling [26]. In this paper, we address the transport layer issues. We adapt the popular slowly responsive congestion control protocol – TCP Friendly Rate Control (TFRC) – proposed for wired multimedia transport, to mobile ad hoc networks. To this end, we devise a suite of novel end-to-end mechanisms to effectively address the wireless and mobility issues.

In early proposals, multimedia streaming is carried by UDP flows [33]. Since UDP does not have any congestion control mechanism, these unresponsive multimedia flows will compete unfairly with other responsive TCP flows. Consequently, network congestion may significantly degrade the network performance. Using TCP to carry multimedia traffic can prevent such congestion collapse. However, TCP's retransmission scheme may be too expensive or unnecessary for real-time loss-tolerant multimedia streaming applications. Moreover, TCP halves its transmission rate upon any congestion event. Such a dramatic oscillation in transmission rate could be detrimental to these applications.

Recent research on multimedia congestion control has focused on developing a TCP friendly transport protocol that does not react to any single congestion event dramatically (such as halving the sending rate) and slowly adapts to the changes of network condition [28] [30] [31] to better serve multimedia streaming applications. A particularly visible proposal is the TCP Friendly Rate Control (TFRC) [29], which theoretically characterizes a TCP friendly throughput given the RTT and congestion probability measurements, and increases/decreases the current sending rate accordingly.

In wired networks, it has been shown [32] that TFRC serves multimedia streaming reasonably well in terms of stability and bandwidth efficiency. However, there are several technical challenges in order for it to function well in mobile ad hoc networks. Mobile ad hoc networks exhibit a rich set of packet loss behaviors. It is well known that for TCP friendly flows, uniformly applying congestion control upon every packet loss leads to unacceptable performance in ad hoc networks [1] [8] [9] [7]. In fact, in addition to reacting to network congestion, TFRC protocol in an ad hoc network must handle other types of network events such as mobility induced disconnection and re-connection, route-change induced out-of-order delivery and error/contention-prone wireless transmissions[1]. In order to support multimedia streaming efficiently, these events require TFRC to respond differently from congestion control. For example, it makes sense to simply ignore a packet loss due to random channel errors than to multiplicatively decrease the current sending rate [4]; and it is more appropriate to periodically probe the network during disconnection period for a prompt recovery than to slow down and exponentially increase the retransmission timer [1]. On the other hand, even if the correct action is executed in response to each type of these events, it is not immediately obvious how to construct an engine that will accurately detect and differentiate events in the first place. Packet loss as the sole detector used by conventional TCP/TFRC flows cannot detect and differentiate all these new network events [15].

In this paper, we take an end-to-end approach, in order to adapt TFRC to mobile ad hoc networks. We rely on measurements at the end hosts to differentiate different network events. However, end-to-end measurement in ad hoc networks is noisy and may consequently lead to false detections. How to detect events in an ad hoc network in a *robust* manner

Manuscript received Oct 1, 2002; revised Apr 10, 2003. This work was supported by the NSF.

Z.Fu, X.Meng and S.Lu is with the Computer Science Department of UCLA. Email:{zfu,xqmeng,slu}@cs.ucla.edu.

[1]Even with link-layer re-transmissions of 802.11 MAC, packet loss still occurs during bursty channel error or MAC-layer contentions.

using noisy measurements poses a great design challenge. Previous study [15] shows that single-metric based approach, e.g., round-trip time (RTT) or packet inter-arrival time, is not encouraging in detecting network congestion mainly because of measurement noise. False detection comes up in two forms. Take congestion for example. Network congestion may go undetected, or conversely congestion may be inferred when the network is in fact *not* congested. Using measurements at the end host, the probability that congestion will go undetected is very low. Measurement metrics such as RTT or inter-arrival time indeed increase when network is congested. However, using single metric measurements, the probability of false congestion detection in a non-congested ad hoc network is quite high due to noisy end-to-end observations. This sort of false detection can lead to severe throughput degradation. The key innovation of this work is the use of multi-metric, joint identification instead of single-metric identification. By exploiting the degree of independence in the measurement noise of individual metrics, the probability of false identification is significantly reduced by cross verification among the multiple metrics.

In this paper, we first describe the necessary network states in an ad hoc network to be identified by the adapted TFRC, and then examine metrics that can be measured end-to-end. In particular, we devise two metrics to detect congestion, i.e., IDD (Inter Delay Difference) and STT (Short Term Throughput). They each exhibit a unique pattern upon congestion; and in non-congestion states, they are influenced by different network conditions in a way that the measurement noise of these two metrics is largely independent. Our multi-metric joint identification approach is then able to reduce false detection by cross validation. Extensive simulations show that this technique is effective by achieving reasonably accurate detection ratio and improves TFRC performance significantly in ad hoc networks.

Two main contributions are summarized as follows:

- We implement a joint detection approach based on measurements of *multiple metrics* at the receiver side. If each metric is noisy and does not provide accurate detection, we combine four metrics including throughput, delay, packet loss and out-of-order delivery ratio to collaboratively identify the network state. Extensive simulations show that this approach can significantly reduce the probability of false detection.
- We modify the TFRC state machine so that it reacts adaptively to the detection feedback from the receiver side. The resultant ADTFRC protocol is implemented in NS-2 simulator, and the performance of ADTFRC is extensively evaluated in mobile ad hoc networks. The results show that, without compromising the TCP-Friendliness property, ADTFRC outperforms TFRC and TCP NewReno with ELFN support in terms of both throughput and smoothness in rate adaptation behavior. The performance gain does not come from stealing bandwidth from standard TCP or TFRC flows, instead it comes from a better network state detection and appropriate ADTFRC reactions.

The remainder of the paper is organized as follows: Section II briefly introduces the related work. Section III describes the detection design. The ADTFRC protocol is presented in Section IV Performance evaluation is given in Section V. Section VI concludes the paper.

II. Related Work

There are two approaches to detecting network congestion in the Internet. One is based on end-to-end measurement, and the other is based on feedback from intermediate gateways in the network. Standard TCP [18] [19] [16] uses end-to-end measurement of RTT and packet losses to infer congestion; RED/ECN [14] [6] provide congestion notification by monitoring the queue length at the network gateways.

For wireless networks, detecting network congestion is critical in improving TCP performance. In cellular networks, [21] [22] [4] directly monitor packet loss at the base station so that wireline packet losses (congestion related) and wireless packet losses (channel error related) are treated differently. As an alternative technique, [3] [20] propose to use receiver-based measurements on packet interarrival time and loss behaviors to distinguish between congestion losses and wireless link losses.

In mobile ad hoc network, mobility-induced link failure may occur. [1] proposes an explicit link failure notification (ELFN) mechanism for each wireless node to inform the TCP sender. This way, the sender can distinguish link failure losses from congestion losses. [8] further shows that ELFN improves standard TCP by as much as seven times in mobile ad hoc network, with about a 5% degradation in static ad hoc networks. For end-to-end measurement based congestion detection, [15] studied the approach of using single metrics measurement such as inter-arrival delay, throughput, or packet losses to identify network congestion, but the results are not encouraging. There is simply too much noise in the measurement, especially when node mobility and channel errors are both present.

In general, the network assisted approach provides a more direct monitoring of congestion, while in the end-to-end measurement approach, the congestion has to be inferred from metric observation. However, end-to-end measurement maintains the end-to-end semantics of TCP and provide a convenient implementation that does not need infrastructure support. In this paper, we further explore the feasibility of end-to-end based congestion detection in mobile ad hoc network using *multi-metric joint identification*.

III. Detection via Multiple Metrics

In this paper, we design a robust detection solution that uses end-to-end measurements only to improve TFRC performance in mobile ad hoc networks. Measurements at the end hosts are used to detect congestion, disconnection, route change, or channel errors, so that the TFRC sender could respond accordingly to achieve better quality of the multimedia streaming over mobile ad hoc networks. The design is easy to implement and deploy for users who want to improve the performance of their multimedia streaming applications in an ad hoc network. It requires only software upgrades at the two end hosts, without assuming any other supporting modules. It provides the flexibility of backward compatibility with standard TFRC and can be incrementally deployed in the entire network. In addition, TFRC derives TCP friendly sending rate based on end to end loss frequency measurement at receiver side. Adopting ad hoc network congestion detection at receiver side fits naturally to the existing TFRC protocol [29].

In the following, we first describe what network states TFRC has to monitor, and then what end-to-end metrics to collect. The robust detection algorithm is presented next.

A. States To Be Detected

Previous research has indicated that identifying the following network states should be necessary to improve the performance of TCP-friendly multimedia streaming over ad hoc networks [28] [29] [31] [30].

- CONGESTION (CONG): We define congestion in ad hoc networks as the signal that the offered load exceeds the network capacity. When congestion occurs, the queue size will grow and the network throughput is reduced. To deal with congestion, the transport protocol should reduce the sending rate, similar to the standard TFRC protocol.
- CHANNEL_ERR (CHERR): When random packet loss occurs, the receiver should not count it as a congestion event. The sender should maintain its current sending rate.
- ROUTE_CHANGE (RTCHG): The delivery path between the two end hosts can change from time to time, but disconnections are too short to result in a retransmission timeout. Depending on routing protocols, the receiver may experience a short burst of out-of-order packet delivery or packet losses. In both cases, the receiver should not treat it as congestion; and the sender should keep the streaming rate unchanged in the next RTT period, waiting for the receiver to feedback more measurement statistics for the new path.
- DISCONNECTION (DISC): When the delivery path is disconnected long enough to cause a retransmission timeout, instead of exponentially slowing down and backing off, the sender should freeze the current congestion window and the retransmission timer. It then performs a periodic probing so that the transmission can be resumed promptly once a new path is established. Once it is recovered, the actions of RTCHG should be followed. This probing technique is also used in [1] [3] [2].

The above states take an action-oriented classification. However, the response actions are not necessarily *optimal*. They simply represent the four necessary states that TFRC uses to improve its performance. In some cases, a packet loss may be caused by the combined effects of multiple network conditions. For example, RTCHG may cause multiple flows to go across a hot spot so that CONG losses occur; or bursty CHERR might cause repeated link layer failures, so that RTCHG or DISC conditions eventually take place. However, since our primary goal is to be TCP friendly, CONG is given the highest priority in detection. The other three states are considered only if the network is not congested.

Metric	Definition
IDD	$A^{i+1} - A^i - (S^{i+1} - S^i)$, where A^i is the arrival time of packet i and S^i is its sending time from the sender
STT	$N_p(T)/T$, where $N_p(T)$ is the # of received packets during interval T,
POR	$N_{po}(T)/N_p(T)$ where $N_{po}(T)$ is # of out-of-order packets during T
PLR	$N_l(T)/N_p(T)$ where $N_l(T)$ is # of lost packets during interval T

TABLE I

DEFINITIONS OF THE FOUR PROPOSED METRICS

B. Devising End-to-End Metrics

End-to-end measurement is widely used in transport protocols. In TCP, the round trip time (RTT) is maintained by the sender to calculate the retransmission timeout. Previous work uses delay-related metrics to measure the congestion level of the network. For example, [3] and [15] use inter packet arrival delay, and [16] uses RTT to estimate the expected throughput. A challenge in ad hoc networks is that packet delay is not only influenced by network queue length, but also susceptible to other conditions such as random packet loss, routing path oscillations, MAC layer contention, etc. These conditions make such measurements highly noisy. Therefore, any single metric may not be reliable. We devise four end-to-end metrics that may observe independent noises.

a) Inter-packet delay difference IDD*:* Metric *IDD* measures the delay difference between consecutive packets (Table I). It reflects the congestion level along the forwarding delivery path by directly sampling the transient queue size variations among the intermediate nodes. In Figure 1, time at the sender and receiver sides is shown by vertical arrows. Upon each packet arrival, the receiver calculates the IDD value. The first figure shows that the onset of congestion and the queue length buildup process is reflected by a series of IDD samples with increasing values.

Unlike the conventional inter-packet arrival delay (IAD), IDD is unaffected by random channel errors and packet sending behaviors. The second figure of Figure 1 shows that random packet losses can affect the IAD measurement but not IDD. In the third figure, assuming that the network is not congested, the change in packet sending rate also affects the IAD; by subtracting the sending time difference, IDD is not affected.

In an ad hoc network, there are still a number of situations in which *IDD* values might give an inaccurate estimation of congestion. For example, *IDD* can be influenced by non-congestion conditions such as mobility induced out-of-order packet delivery. In section III-D, we evaluate the accuracy of using IDD to detect network congestion. The accuracy decreases from 85% to 55% as node mobility speed increases.

b) Short-term throughput STT*:* Compared with IDD, STT is also intended for network congestion identification. It provides observation over a time interval T, and is less sensitive to short term out-of-order packet delivery compared

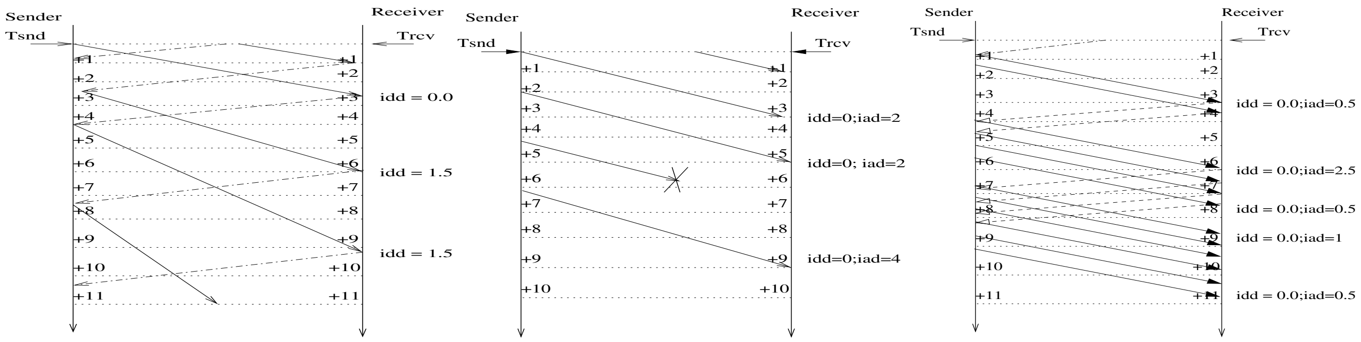

Fig. 1. *IDD and IAD Measurements . From left to right: in incipient network congestion, in channel error, in slow start phase*

with IDD.[2] Therefore, STT is more robust to transient route changes, which can be very frequent in a mobile ad hoc network. However, using STT alone to detect network congestion can be susceptible to measurement noise introduced by bursty channel error, network disconnections or altering source rate. In section III-C, we combine STT and IDD to jointly identify network congestion.

c) Packet out-of-order delivery ratio POR*:* A packet is counted as being out-of-order if it arrives after a packet that was sent later than it (by the same sender). The receiver records a maximum sending time for all the received packets from the connection, denoted by T_{maxtx}. Every received packet that has a sending time-stamp less than T_{maxtx} is added into POR. POR is intended to indicate a route change event. During the route switching period, multiple delivery paths exist. Packets along the new path may catch up, and those along the old path are then delivered out-of-order.

d) Packet loss ratio PLR*:* At each time interval $[t, t+T]$, we compute this metric as the number of missing packets in the current receiving window. POR can be used to measure the intensity of channel error.

C. Detection via Multiple Metrics

This section first describes how to detect the four network states based on multi-metric measurements, and then introduces a sample value classification technique to improve the detection accuracy. The achieved accuracy using our approach is also evaluated through simulations.

For all the simulation results shown in this section, the default setting is as follows unless otherwise specified. We use the NS-2 simulator with CMU wireless extension modules. 30 wireless nodes roam freely in a $400m \times 800m$ topology following a *random waypoint* mobility pattern, in which the pause time is zero so that each node is constantly moving. The wireless link bandwidth is 2M bps, and IEEE 802.11 and DSR are used as MAC and routing layer protocols, respectively. One TFRC flow created with packet size of 1000 bytes. To introduce congestion, three competing UDP/CBR flows, each with source rate 180K bps, are created within the time intervals of [50,250], [100,200] and [130,170], respectively. Each UDP flow transmits at 180Kbps. The simulations last 300 seconds.

[2]The choice of T provides a trade-off between metric accuracy and responsiveness. In our design, we choose T as one RTT.

1) Detecting Network Congestion: To study the relationship between network congestion and IDD/STT, we simulate both static and mobile scenarios. A TFRC flow and three competing UDP/CBR flows are introduced in each simulation and the network is expected to become congested as it is overloaded. The first two figures of Fig. 2 show the simulation results. The first is for the static case without channel errors, and the second is for the mobile case with node mobility speed of 5m/s and 5% channel error. In both figures, we plot the measured IDD/STT values with respect to the instantaneous maximum buffer occupancy of all nodes in the network, which reflects the network congestion level at the sampling time instance[3].

In Fig 2, we observe that when the maximum network queue size exceeds half of the buffer capacity (25 packets), IDD is clearly *high* and STT is clearly *low*. We formalize this observation by defining a value to be HIGH or LOW, respectively, if it is within the top or bottom 30% of all samples[4]. However, when the network queue size is small (non-congestion case), both IDD and STT vary from LOW to HIGH, with the majority of IDD samples being not HIGH and STT samples not LOW. In the left two figures of Fig. 2, when node mobility is present, the two metrics become much more noisy in non-congestion state (i.e., small network queue).

In the single metric-based detection using either IDD or STT, the noise reduces the accuracy significantly when the network is not congested, especially in scenarios with mobility and channel errors. However, in the proposed joint detection approach, we can use both metrics to *verify* each other to improve the accuracy. Specifically, we identify a congestion state when both IDD is HIGH and STT is LOW, and non-congestion state, otherwise. The following shows why the multi-metric approach has better detection accuracy than the single metric approach.

When the network is congested, let P_1 and P_2 be the probabilities that IDD is HIGH and STT is LOW respectively. The single metric accuracy is $acc_{idd}(cong) = P_1$ and $acc_{stt}(cong) = P_2$. For the multiple-metric case, $acc_{multi}(cong) = P_1 \cdot P_2$. Since the simulations show that $P_1 \simeq P_2 \simeq 1$ (see left two figures of Fig. 2), these three metrics are roughly equal in congestion state. On the other hand, when the network is not congested, let P_1' and P_2' be the proba-

[3]The maximum buffer size for each node is 50 packets in our simulations.
[4]This threshold was determined empirically from simulation results and real testbed measurements [24].

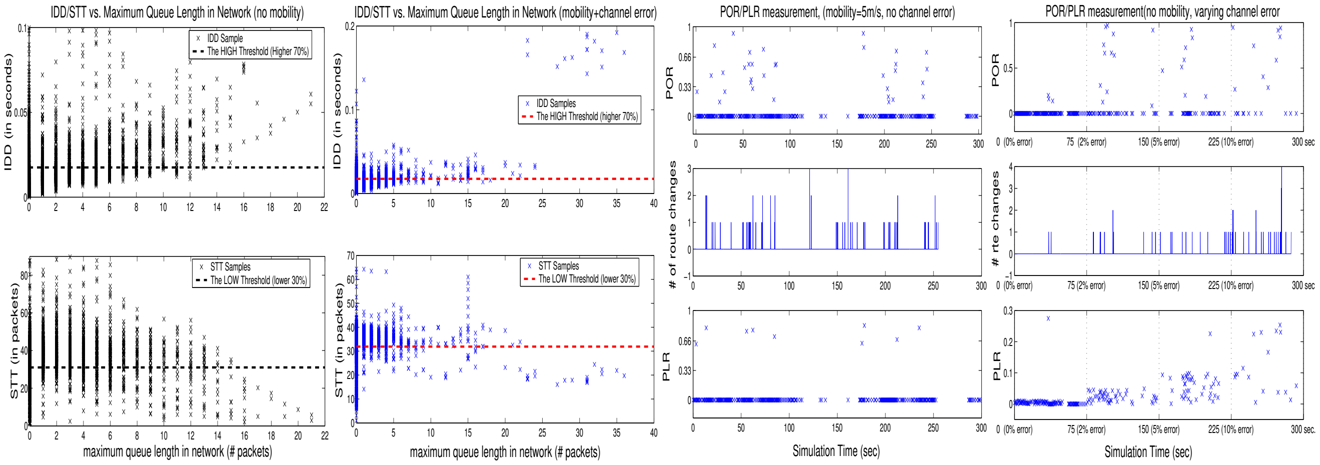

Fig. 2. *End To End Metric Measurement. Left two: the IDD and STT measurement w.r.t. instantaneous maximum queue occupation of all wireless nodes in the network. First figure shows simulation with static nodes; second figure shows simulation with mobile nodes (5m/s). The simulations run 300 seconds, 3 competing UDP/CBR flows are introduced in [50,250],[100,200] and [130,170] respectively. Right two: POR and PLR measurements w.r.t. the number of route changes. The third figure shows simulation with mobile nodes (5m/s), no channel error. The fourth one shows simulation with static nodes, progressively increasing channel error (0%,2%,5%,10%) during the entire run. These two simulations are run with single TFRC flow.*

bilities that IDD is still HIGH and STT is still LOW. Similarly, we have $acc_{idd}(non\_cong) = 1 - P_1'$, $acc_{stt}(non\_cong) = 1 - P_2'$ and $acc_{multi}(non\_cong) = 1 - P_1' \cdot P_2'$. Since each noise probability, $0 < P_1', P_2' < 1$, is non-negligible, multiple metrics thus achieve higher accuracy. Combining these two cases, multi-metric identification improves the accuracy in non-congestion states while maintaining a comparable level of accuracy in the congestion state. Therefore, it achieves better identification performance over a variety of network conditions.

The key insight here is that in the non-congestion state, IDD and STT are influenced differently by various network conditions, such as route change and channel error; while in congestion state, they are both dominated by prolonged queuing delay. Thus, the two noise probability P_1' and P_2' become largely independent. Effective verification across multi-metrics is possible as long as these conditions do not co-exist during the measurement time interval. Although this joint identification technique cannot achieve perfect accuracy, it does increase the accuracy significantly as we show in Section III-D

2) Detecting Non-Congestion States: If the network state is not congestion, we next seek to detect whether it is RTCHG or CHERR. The third figure of Fig 2 shows data from a simulation run with node mobility speed being 5m/s. A single TFRC flow is created without any competing flows that might cause network congestion. We plot POR and PLR sample values together with the number of route changes in the forwarding path over time. A clear correlation is seen between route change events and bursts of high POR measurement. During the changing period, packets arrive at the receiver from multiple paths and consequently may lose their ordering. Although not all route changes result in out-of-order packet delivery, we only count those observable changes, which would have an impact upon TFRC. POR can be used to identify RTCHG state and PLR can be used for CHERR state.

Moreover, since there is no congestion or channel errors in these simulations, PLR remains stable with a few significant outliers. These anomalies correspond to situations in which packets along the old path are excessively delayed or lost.

In the simulation shown in the fourth figure of Fig 2, nodes are immobile and four channel error rates (0%, 2%, 5% and 10%) are introduced into four identical time intervals (75 seconds). In this case, packet loss is proportional to the channel error rate and the PLR gradually increases as the channel error rate increases. Note that a high channel error rate can also create route change in the network that will in turn result in bursts of high POR measurements. The routing layer interprets any MAC-layer transmission failures (in this case, channel error) as a sign of a broken link and consequently will seek to repair/re-establish the delivery path, which may causes route changes.

In summary, a burst of high POR sample values is a good indication of a route change and a high PLR is a good indication of a high rate of channel error. It should be noted that the network may be both in a state of high channel error and route change, which can be identified by high values in both PLR and POR.

We next consider the disconnection state. Disconnection happens when packet delivery is interrupted (for non-congestion reasons) long enough to trigger a retransmission timeout at the sender. Multiple network conditions can trigger such a timeout at the sender including frequent route changes, heavy channel error, and mobility-induced network partition. If the timeout is triggered by congestion, then previous state feedback should reflect the transient queue buildup period by increasing IDD and STT measurement at the receiver; if not, the timeout was due to non-congestion conditions in the network. Therefore, a DISC state is identified at the sender if the current state estimation is non-congestion but retransmission timeout is triggered.

Table 2 summaries the metrics patterns in five different network states. They are the identification rules used in our ADTFRC. We show later in this section that such an identifi-

	IDD and STT	POR	PLR
CONG	(High, Low)	*	*
RTCHG	NOT (High, Low)	High	*
CHERR	NOT (High, Low)	*	High
DISC	(*, ≈ 0)	*	*
NORMAL	default		

TABLE II

Metrics patterns in 5 network states. High: top 30% values; Low: bottom 30% values; '': do not care*

cation method, combined with a simple sample classification technique, achieves an accuracy above 80% on average in all simulations scenarios.

D. Detection Accuracy

We now study the accuracy of the proposed congestion detection techniques. In particular, we compare the single-metric (using only IDD or STT) and multiple-metric (using both) approaches. We run two sets of simulations under non-congested and congested cases (Figure 3). In the first non-congested case, a single TFRC flow is created within the topology. In the second congested case, two competing UDP flows are created as before. In both cases, 1% random channel error is introduced and the mobility speed is varied from 0 to 20 m/s. We repeat simulations 50 times at each speed to reduce the impact of random topology factors.

During the simulation, upon each packet loss, we compare the identified network state and the actual network state to determine the accuracy of detection[5]. In particular, if a packet is lost due to network congestion, but the algorithm gives non-congestion estimation, we count it as an incompatible error because this error in detection (and only this one) causes ADTFRC to be more aggressive than a TCP-friendly flow and consequently TCP-incompatible.

Figure 3 shows the percentage of inaccurate identification in both cases. In the single TFRC flow case (the left figure), mobility and channel error are the dominant reasons for packet loss. The increase in mobility speed reduces the accuracy of the single-metric identification quickly. However, from the simulations, the multi-metric approach results in only 10% to 30% inaccurate identification. This is achieved by the cross verification between IDD and STT measurements to eliminate false congestion alarms. Meanwhile, the incompatible error remains less than 2%.

In the multi-flow cases (the right figure), congestion happens more frequently. For multi-metric identification, more than 95% accuracy is observed in all simulations with less than 2% incompatible errors. For the single metric approach, accuracy is only about 70% to 80%.

In summary, we have demonstrated that multiple metrics is a feasible approach to detect network events using end-to-end measurements only.

[5]The real network state is obtained by a global monitor implemented in NS-2 simulator. See [7] for implementation details.

IV. ADTFRC Protocol Design and Implementation

We now incorporate the design of Section III in our ADTFRC protocol to improve the performance of TFRC in ad hoc networks.

ADTFRC seeks to maintain backward compatibility with conventional TFRC. It uses identical connection establishment and connection teardown processes. It estimates the RTT and derives the sending rate identical to TFRC. To improve the performance of TFRC in ad hoc networks, ADTFRC makes several extensions at both the sender side and the receiver side.

Upon each packet arrival at the receiver, besides the normal operations, values for the four previously discussed metrics are calculated and network states are estimated. In ADTFRC, the congestion probability is calculated based on the outcomes of our multi-metric identification in stead of the packet loss events. The receiver then passes this congestion frequency measurement together with state estimations, i.e., CONG, CHERR and RTCHG, to the sender in every feedback packet. Besides regular feedbacks each RTT, the receiver generates *Urgent* state update packet as soon as a congestion event is detected and feedback to the sender immediately. This way, information about persistent network conditions will likely be relayed to the sender in multiple feedback packets, providing robustness against possible losses of state report. The sender maintains the most current state report received, and proceeds with normal TFRC operations until either of the following two events happens: the reception of feedback packet, or the re-transmission time out. A modified TFRC state diagram is shown in Figure 4 for the sender. A

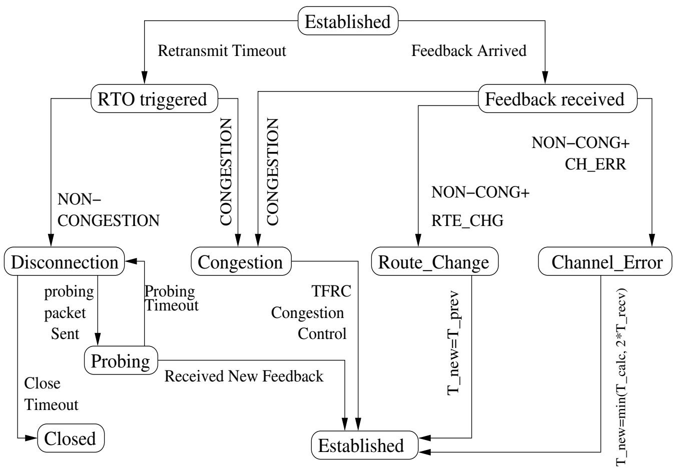

Fig. 4. *ADTFRC state diagram for sender in* NS-2 *implementation*

feedback report or retransmission timeout triggers ADTFRC to take different control actions according to the current network state estimation. In particular, a probing state is introduced to explicitly handle network disconnection. When a non-congestion induced retransmission timeout occurs at the sender, ADTFRC freezes its current transmission state and enters a probing state. The sender leaves the probing state when a new acknowledgment is received or the probing is timed out[6]. The ADTFRC connection is closed after multiple

[6]A similar probing mechanism was proposed by [1]

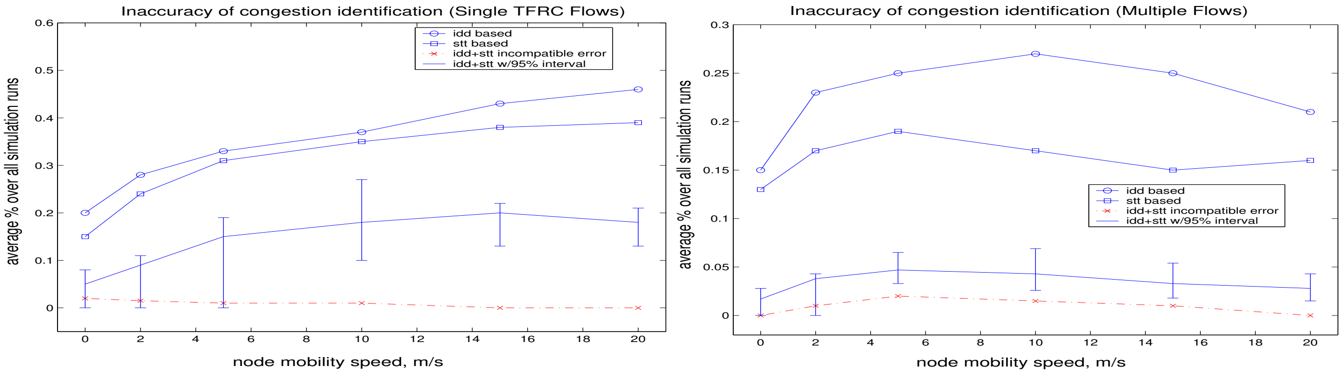

Fig. 3. *Identification Accuracy. Left: Percent of inaccurate identifications in a non-congested case, Right: Inaccuracy ratio in a congested case*

probing attempts fail.

The pseudo-code is presented in the following to illustrate the actions taken at both the sender and the receiver.

Algorithm 1 Sender Side: Upon Receiving Feedback

if $probing\_state$ **then**
  $probing\_state = 0$
  $probing\_count = 0$
  resume next_packet_timer
end if
measure RTT
calculate new sending rate as T_{calc}
check the network state from the latest feedback;
if CONG **then**
  $T_{new} = min\{T_{calc}, T_{recv}\}$ {decrease the rate}
else if ROUTE_CHANGE **then**
  $T_{new} = T_{prev}$ {maintain current rate}
else if CHANNEL_ERR **then**
  $T_{new} = min\{T_{calc}, 2 * T_{recv}\}$ {without slowing down}
end if
if $T_{new} \geq T_{prev}$ **then**
  decrease current rate
else
  increase current rate
end if

The sender maintains its normal TFRC operation until the specific events are triggered. The detailed behavior responding to each event is designed following the description in Section III-A. Note that in our design, receiver-side identification is treated as an enhancement to TFRC in ad hoc networks that helps the sender to take more appropriate control and improve transport performance significantly. Without these enhancements, ADTFRC would behave exactly as TFRC.

Both the ADTFRC sender and its receiver are able to work correctly with conventional TFRC end hosts. An ADTFRC sender communicating with a conventional receiver will receive no explicit network state information in feedback packets and will consequently fall back to standard TFRC congestion control operations. On the other hand, a conventional TFRC sender that receives the state information bits sent by an ADTFRC receiver in an option field of feedback will simply ignore this information. This backward compatibility requires

Algorithm 2 Sender Side: Upon Retransmission Timeout

check the network state from the latest feedback
if $probing\_state$ **then**
  increase $probing\_count$
  if $probing\_count \geq max\_probing\_limit$ **then**
    close connection
  end if
  set_next_rtx_timeout($probing\_period$)
  transmit the probing packet
  return
end if
if not CONGESTION **then**
  set probing_state {DISCONNECTION state is assumed here, enter into period probing state}
  freeze current states
  pause next_packet_timer {The probing state is cleared and TFRC state restored upon receiving a new feedback}
  set_next_rtx_timeout($probing\_period$)
  retransmit the packet
else {default case, invoke TFRC congestion control}
  exponential slow down and backoff timer
end if

Algorithm 3 Receiver Side: Upon packet arrival

compute sample value for four metrics
classify for each metric into HIGH/LOW
network state identification
if $packet\_lost$ AND CONG **then**
  $Urgent = 1$
end if
update congestion frequency $flost$
measure T_{recv}
if $Urgent$ OR $now - last\_feedback > feedback\_interval$ OR $probing\_packet$ **then**
  fill in $flost$ and set state bits in option field
  transmit feedback
end if

the use of an option field in the packet header. We discuss this aspect in the next section.

V. Performance Evaluation

This section evaluates the performance of ADTFRC through extensive *NS-2* simulations. We also compare it to TFRC and TCP with ELFN [1] support. Instead of using end-to-end measurements, TCP ELFN collects link state information directly from the network and is expected to be more accurate. It is used as a reference system; a throughput close to ELFN indicates the effectiveness of ADTFRC.

We further measure the rate oscillations experienced at the receiver for each of these three protocols. To effectively support best-effort multimedia streaming, dramatic rate variations are highly undesirable. At the end of this section, we examine where the ADTFRC's performance improvement comes from. ADTFRC should not steal a competing TCP flow's fair share of bandwidth and the TCP friendliness should be maintained in all cases.

A. Performance Improvement

Figure 5 shows the single flow throughput of ADTFRC, TFRC and TCP-NewReno with ELFN support. The simulation parameters for TFRC flows are set according to Section III. For TCP ELFN flows, we set the packet size to be 1000 bytes and maximum window size to be 8 packets. In all three cases, ADTFRC offers significantly better throughput than TFRC. When nodes are mobile, ADTFRC achieves a throughput improvement from 100% to 800% over TFRC. Furthermore, it is surprising to see that ADTFRC outperforms TCP+ELFN. The reason is that the ACK packet traffic on the reverse path is much heavier in TCP+ELFN than ADTFRC. Due to the broadcast nature of the wireless link, such ACK flows contend channel access with forward data flows, incurring additional delay in RTT and throughput decrease.

In Figure 5, it can be seen that TFRC's performance is more sensitive to node mobility than ADTFRC. This is because as the mobility speed increases, network disconnection becomes more common. By correctly identifying non-congestion packet losses, ADTFRC is able to recover from such interruptions quickly and achieve higher throughput.

The presence of channel error slightly decreases the performance gap between ADTFRC and TCP+ELFN (second of Figure 5). The gap comes from the identification inaccuracy of ADTFRC. When both mobility and channel error are present, metric samples such as IDD and STT become highly noisy, which makes end-to-end network state detection, especially for non-congestion states, more difficult. In the third figure where 3 competing UDP/CBR flows are introduced, congestion becomes more frequent and available network bandwidth to each flow is reduced. Therefore, the throughput gap between ADTFRC and TCP+ELFN further narrows. However, in all cases, ADTFRC outperforms TFRC and TCP+ELFN in term of throughput.

B. Rate Adaptation

Graceful rate adaptation is a key feature for TFRC to support multimedia streaming applications. In this section, we study the rate variations for ADTFRC in ad hoc networks. In Figure 6 we show receiver-side throughput measurements in two environment settings. The left figure is the simulation result in static network, with 2% random channel error. It shows that the ADTFRC (the middle one) maintains a stable transport rate. The rate-changing range for TFRC flows is much larger than ADTFRC flows, because they are more susceptible to channel error and wireless link losses. TCP-ELFN flow suffers from periodic interruption (throughput almost becomes 0) caused by repeated packet losses. Large amount of packet losses is due to the fact that TCP's congestion avoidance tends to overload the network, intensify wireless channel contention and trigger delivery-path oscillation. ADTFRC, on the other hand, does not rely on such bandwidth probing; it gradually adapts its sending rate according to the RTT and congestion probability measurements. This mild behavior yields better performance in ad hoc network as shown in Figures 5 and 6.

When mobility induced path disconnections and reconnections are frequent in the network, ADTFRC's probing mechanism can reduce the recovery waiting time. The right one of Figure 6 corresponds to simulations with node mobility 5m/s and random channel error 2%. In this figure, the rate variation of ADTFRC becomes much larger. However, if we further examine the rate adaptation behavior of TFRC flow under the same mobility pattern, much more frequent transmission interruptions and longer disconnection period are perceived. For a TCP+ELFN flow, although its disconnection period is short, due to its aggressive bandwidth probing mechanism, it again encounters more frequent interruptions than ADTFRC.

C. TCP Friendliness

So far we have shown that ADTFRC performs better than TFRC in terms of transport throughput and rate adaptation behavior. This is because it determines why packets have been lost and reacts appropriately. Such behavior does not cause other competing TCP-Friendly flows to suffer. Using simulations, we show that ADTFRC indeed is TCP friendly. This TCP friendliness comes from the low rate of *incompatible* identification from our algorithm. An *incompatible* identification is defined as an identification of a non-congestion state when congestion exists. Such an incompatible identification could cause ADTFRC to act too aggressively. Figure 3 shows that the incidence of such *incompatible* identifications is extremely low.

In order to further demonstrate ADTFRC's TCP-friendliness, we run simulations with four transport level flows (Figure 7). We keep the senders and the receivers of these flows static and place them at the edges of the network topology, so that they all run across some bottleneck area of

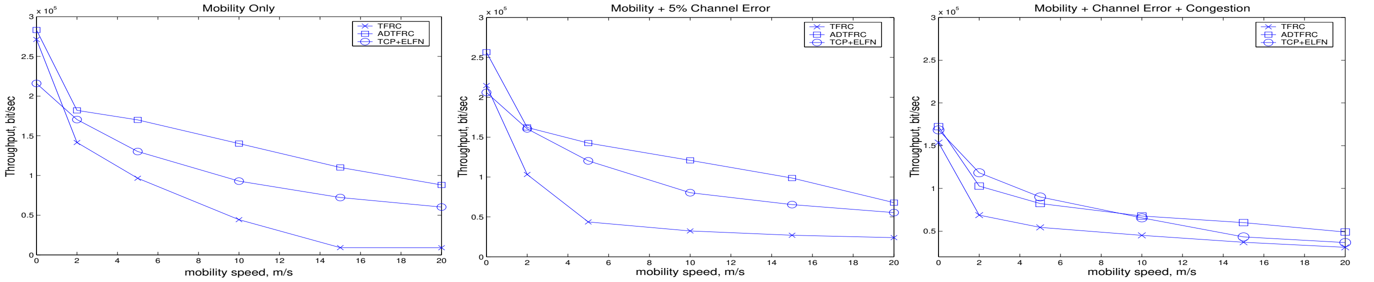

Fig. 5. *Performance Improvement of ADTFRC. From left to right:1)mobility only, 2)mobility+5% channel error, 3) mobility+5% channel error + 3 competing UDP/CBR flows*

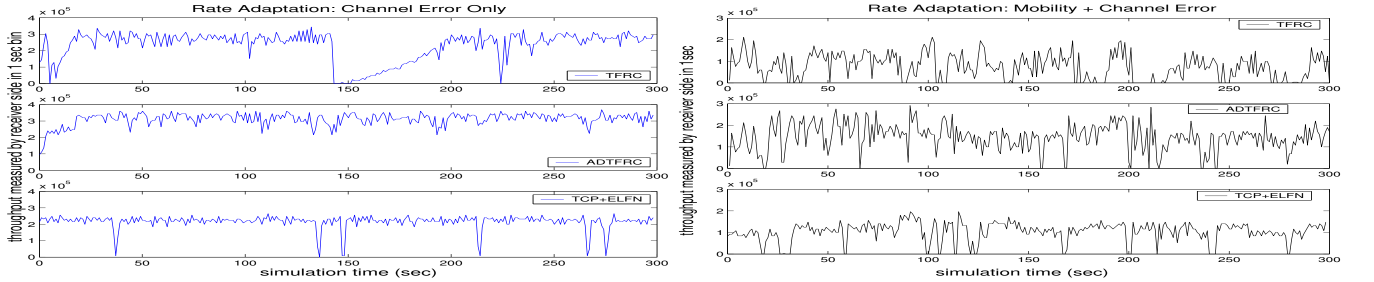

Fig. 6. *Smoothness of Rate Adaptation. From left to right:1)Single flow in static ad hoc network, 2% random channel error. 2) Single flow with node mobility 5m/s, 2% random channel error.*

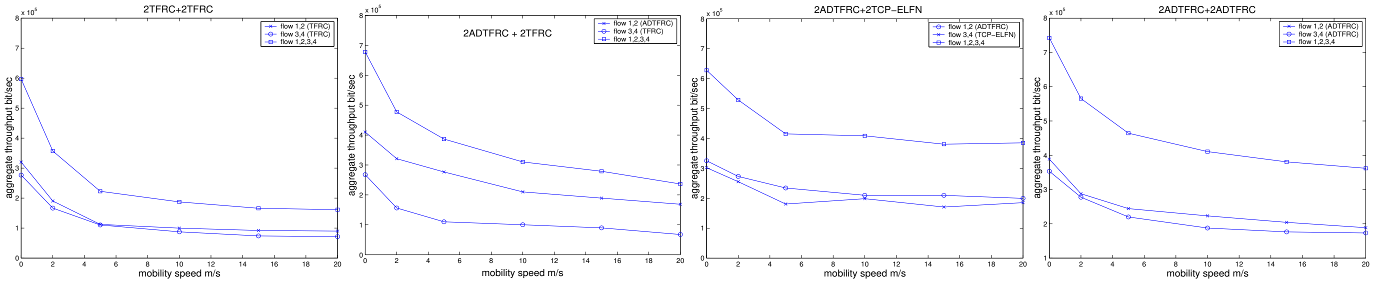

Fig. 7. *TCP Friendliness of ADTFRC (5m/s Mobility+5% Channel Error). From left to right: a)4 TFRC flows, b) 2 ADTFRC+ 2 TFRC flows, c) 2 ADTFRC + 2 TCP-NewReno w/ ELFN support, d) 4 ADTFRC flows*

the network and share the bandwidth with each other[7].

The first figure shows the aggregate throughput of TFRC flows 1,2 and TFRC flows 3,4. The aggregate throughputs of all four flows are also shown in the figure. The fourth figure is for four ADTFRC flows. These two sets of simulations show that ADTFRC improves the throughput at the network aggregate level. The bandwidth sharing of identical flows is not perfectly fair in these two cases. This is mainly due to the topology edge effects and MAC-layer unfairness [9]. In the second figure, we run simulations with flows 1,2 being ADTFRC and flows 3,4 being TFRC. Comparing the first and the second set of simulations, we observe that the aggregate throughput of two TFRC flows does not decrease in the two traffic environments. The same TCP-Friendly property exists for ADTFRC flows when they co-exist with TCP flows in the third figure. The above simulations show that ADTFRC does not steal the fair share bandwidth of competing TCP-Friendly flows; the performance gain comes from ADTFRC's more efficient utilization of the network resources.

[7]According to our simulations, the mobile nodes tend to gather around center of the topology with the random way point mobility pattern. Therefore randomly choosing a sender and receiver often results in short delivery paths (1 or 2 hops), which reduces the chance of bandwidth sharing among competing flows

VI. Conclusion

The fundamental problem of transport protocol design for multimedia applications in mobile ad hoc networks is that such networks exhibit a rich set of network behaviors, including congestion, channel error, route change and disconnection that must be reliably detected and reacted. Detection of such behaviors is challenging because measurement data is noisy. Existing approaches typically rely on the network-layer notification support at each router. This paper explores an alternative approach that relies solely on end-to-end mechanisms. To robustly detect network states in the presence of measurement noise, we propose a multiple-metric based joint detection technique. In this technique, a network event is signaled only if all the relevant metrics detect it. The simulations show that ADTFRC is able to significantly reduce the false detection probability while keeping the incompatible detection errors low, thus greatly improving the transport performance in a TCP friendly way. This demonstrates that the end-to-end approach for multimedia transport is also viable for ad hoc networks.

In a broader context, the robustness of multimedia transport, as well as TCP, has not been equally well explored in the current research. Ad hoc networks offer a good example to demonstrate its importance. The issue of robust detection of network states in ad hoc networks in the presence of transient dynamics and measurement noise deserves further attention. In fact, this issue is not only valid in the context of our end-to-end approach, it also holds for network-oriented design. Each router still suffers from imperfect monitoring and observation noise, and thus may wrongly send out notification signals. Our multi-metric technique is in principle also applicable in this context. This will be our future work.

REFERENCES

[1] G. Holland and N. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," *MOBICOM'99*.

[2] V. Tsaoussidis, H. Badr, "TCP-Probing: Towards an Error Control Schema with Energy and Throughput Performance Gains", The 8th IEEE Conference on Network Protocols, Nov. 2000.

[3] P. Sinha, N. Venkitaraman, R. Sivakumar and V. Bharghavan, "WTCP: A reliable transport protocol for wireless wide-area networks," *MOBICOM'99*.

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

[5] H. Balakrishnan, and R. Katz, "Explicit loss notification and wireless web performances," *Globecom'98*.

[6] S. Floyd, "TCP and explicit congestion notification," *ACM CCR*, 1994.

[7] Zhenghua Fu, Xiaoqiao Meng, Songwu Lu "How bad TCP can perform in wireless ad hoc network" IEEE ISCC (IEEE Symposium on Computers and Communications) 2002, Italy, July 2002.

[8] J. Monks, P. Sinha and V. Bharghavan, "Limitations of TCP-ELFN for ad hoc networks," *MOMUC'00*.

[9] M. Gerla, K. Tang, and R. Bagrodia, "TCP performance in wireless multihop networks," *WMCSA'99*.

[10] Eliason, Scott R., "Maximum Likelihood Estimation: Logic and Practice." *Newbury Park:Sage.*

[11] W.R.Stevens, "TCP/IP Illustrated, Vol. 1 and 2".

[12] N.Reynolds and D.Duchamp. "Measured Performance of a Wireless LAN.", *In Proc. 17th Conf. on Local Computer Networks, IEEE, Sep 1992.*

[13] Kevin Lai and Mary Baker, "Measuring Bandwidth", *Proceedings of IEEE INFOCOMM 1999*

[14] Sally Floyd and Van Jacobson, "Random Early Detection Gateways for Congestion Avoidance.", *IEEE/ACM Transactions on Networking, 1(4):397-413, August 1993.*

[15] S. Biaz and N.H. Vaidya, "Distinguishing congestion losses from wireless transmission losses" *IEEE 7th Int. Conf. on Computer Communications and Networks*, October 1998.

[16] L. Brakmo, S. O'Malley, and L. Peterson "TCP Vegas: New techniques for congestion detection and avoidance" *ACM SIGCOMM 1994*

[17] C. Casetti, M. Gerla, S. Mascolo, M. Y. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links", *ACM Mobicom 2001*

[18] V. Jacobson and M.J.Karels, "Congestion Avoidance and Control" *ACM Sigcomm 1988*

[19] M.Allman, V.Paxson, and W. Stevens, "TCP Congestion Control" *RFC 2581* April 1999.

[20] S. Biaz, N. Vaidya, "Discriminating congestion losses from wireless losses using inter-arrival times at the receiver" *IEEE Symp. ASSET'99* March 1999

[21] A. Bakre and B. Badrinath, "I-TCP:indirect TCP for mobile hosts", *Proc. 15th International Conf. on Distributed Computing Systems (ICDCS)* May 1995

[22] A. Bakre and B.Badrinath "Implementation and performance evaluation of Indirect TCP" *IEEE Trans. Computers*, Vol. 46, March 1997

[23] The NS-2 Simulator available at $http://www.isi.edu/nsnam/ns/$

[24] Z.Fu, B.Greenstein, X.Meng, S.Lu "Design and Implementation of a TCP-Friendly Transport Protocol for Ad Hoc Wireless Networks", *In Proceedings of ICNP 2002, Paris, France.*

[25] G.Ahn, A.Campbell, A.Veres and L.Sun "SWAN: Service Differentiation in Stateless Wireless Ad Hoc Networks", *In Proc. IEEE INFOCOM'2002, New York, 2002*

[26] Haiyun Luo, Songwu Lu, Vaduvur Bharghavan, Jerry Cheng and Gary Zhong, "A Packet Scheduling Approach to QoS Support in Multihop Wireless Networks," *to appear in ACM Journal of Mobile Networks and Applications (MONET), Special Issue on QoS in Heterogeneous Wireless Networks, 2002*.

[27] H.Wu, P.Chuang "Dynamic QoS Allocation for Multimedia Ad Hoc Wireless Networks", *Mobile Networks and Applications, Vol 6, Issue 4, August 2001*

[28] S.Floyd, M.Handley, J.Padhye and J.Widmer, "Equation-Based Congestion Control for Unicast Applications" In *Proceedings of ACM Sigcomm 2000*

[29] M.Handley, S.Floyd, J.Padhye and J.Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", *RFC 3448.*

[30] D.Bansal and H.Balakrishnan, "Binomial Congestion Control Algorithms." In *Proceedings of IEEE Infocom, 2001.*

[31] I.Rhee, V.Ozdemir and Y.Yi, "TEAR: TCP Emulation at Receivers-Flow Control for Multimedia Streaming.", *Technical Report, NCSU, April 2000.*

[32] D.Bansal, H.Baladrishnan, S.Floyd and S.Shenker, "Dynamic Behavior of Slowly-Responsive Congestion Control Algorithms", In *Proceedings of ACM Sigcomm 2001*

[33] H.Schulzrinne, S.Casner, R.Frederick, V.Jacobson, "RTP: A Transport Protocol for Real-Time Applications", *RFC 1889, 1996*

[34] J.Li, C.Blake, D.De Couto, H.Lee and R.Morris, "Capacity of Ad Hoc Wireless networks",*In Proceedings of ACM MOBICOM 2001*

Zhenghua Fu is currently a Ph.D. student in the University of California, Los Angeles (UCLA). He receives his B.S. degree from Fudan University, China in 1995 and M.S. degree from University of California, Irvine in 2000, both in Computer Science. His research interests include wireless networks, distributed systems and performance evaluation.

Xiaoqiao Meng is currently a Ph.D. student in UCLA. He received his B.S. degree in Automatic control from the University of Science and Technology of China in 1998, and his M.S. degree in Pattern Recognition and Intelligent Control from the Institute of Automation, Chinese Academy of Sciences, P.R.China. His interests include QoS, wireless networks and communications.

Songwu Lu received both his M.S. and Ph.D. from University of Illinois at Urbana-Champaign. He is currently an assistant professor at UCLA Computer Science. He received NSF CAREER award in 2001. His research interests include wireless networking, mobile computing, wireless security, and computer networks.