Performance Analysis of TCP over Static Ad Hoc Wireless Networks

Venkatesh Ramarathinam and *Miguel A. Labrador*
Department of Computer Science and Engineering
University of South Florida, Tampa, FL 33620, U.S.A
{rv,labrador}@csee.usf.edu

Abstract

Most research in mobile ad hoc networks has been devoted to routing protocols since they are fundamental to the technology. Performance analyses of these protocols have shown advantages and disadvantages and the effectiveness usually depends on the type of application or environment they are meant to run on. Lately, some studies have also included the performance of transport level protocols over ad hoc networks, in particular TCP. However, these studies provide general insights about the performance of TCP over these networks but no specific application is associated with them. In this paper we study the performance of the most important TCP versions and ad hoc routing protocols in a closed environment simulating a very likely scenario such as a classroom, conference room, or the like and show that due to the static nature of the setting these types of networks are a good solution, in particular when TCP Reno and DSR routing protocols are combined.

I. Introduction

Ad hoc wireless networks have been proposed as the networking solution for those situations where the network set up time is a major constraint and/or a networking infrastructure is either not available or not desirable. Ad hoc networks allow mobile devices to exchange information using their wireless interfaces without the need of the fixed infrastructure and the attached specialized devices commonly found in wired networks such as routers, switches, gateways, etc. As a result, every device in an ad hoc wireless network can take the role of an end system, a server, a router, gateway, etc., or all of them at the same time.

One of the most important functions in ad hoc networks is routing. Performing the routing function in an environment where the routers move and run on batteries is very challenging. Mobility makes good routes now to break some time later or not to be optimal anymore. In addition, mobile devices run on batteries and can't afford to have a routing function that consumes a big portion of the energy. This is perhaps why most research on ad hoc wireless networks has focused on developing efficient routing protocols. Nowadays several standard routing protocols for ad hoc networks exist [3], [15], [14], [12] and performance evaluations comparing them have shown the advantages and disadvantages of each one [4].

Once the ad hoc network is up and running using a particular routing protocol, the next step is to assess the performance of the transport layer protocols. We are particularly interested in the performance of TCP and UDP as they are the transport layer protocols we commonly use in wired networks. The performance of TCP over wireless networks has been studied extensively and the main problems and issues are well known and have already been addressed. However, TCP confronts new issues and problems when running over ad hoc wireless networks and performance evaluations are not widely available. Only a few studies have considered transport protocols like TCP in their performance evaluations over ad hoc networks [7], [8] and few studies have proposed improvements [10], [5]. It is expected that the performance of TCP will be affected considerably in ad hoc networks not only due to the effects of the wireless environment but also due to specific issues only found in ad hoc networks like mobility, routing, and energy. Even less research work has been done on UDP and the effects of running UDP traffic on power consumption and network performance when mixed with TCP. We expect to find a worse TCP-friendliness problem in ad hoc networks. However, these issues are not considered in this paper.

Two main problems commonly found in the available performance analyses of TCP over wireless ad hoc networks are lack of completeness and generalization. Available performance evaluations usually analyze the performance of a specific TCP version using the different routing protocols available. Or vice versa, all TCP versions over a specific routing protocol. Another important factor is the network scenario these protocols are meant to work on. So far, general scenarios with random mobility patterns have been analyzed but not many studies have focused on finding the best combination of ad hoc routing protocol and TCP version for a particular setting. In this paper we address these two aspects, analyzing the performance of the most important TCP versions and ad hoc routing protocols over a static ad hoc wireless network that we consider as a very likely scenario: a room where the participants are meant to do some collaborative work of some sort, such as a classroom, a conference, or a business meeting in a hotel room.

The rest of the paper is organized as follows. Next section briefly describes the related literature in routing protocols, TCP versions, and performance of TCP over ad hoc networks. Section III describes our specific environment and simulation settings. In Section IV we present our simulation results and in Section V, we present our conclusions and directions for additional research.

II. Related Work

In this section, we briefly describe the functionality and most important characteristics of the most relevant ad hoc routing protocols and TCP versions and the current performance evaluations. In particular, we consider DSDV, DSR, and AODV ad hoc routing protocols and TCP Tahoe, Reno, New Reno, and SACK versions.

Ad hoc routing protocols can be classified as reactive and proactive. Reactive routing protocols find routes only when needed while proactive ones always try to maintain valid routes in their routing tables. It can be easily seen that an important tradeoff exists in terms of performance and power consumption. In general, reactive protocols tend to consume less power at the expense of having not valid information in their routing tables. On the other hand, proactive protocols incur in a higher overhead to keep their routing tables up to date, consuming more energy. However, this is not the whole story as having valid routes all the time could translate in having to send fewer packets to complete the required transactions.

Several routing protocols have been developed for ad hoc networks. The Destination-Sequenced Distance Vector (DSDV) protocol was introduced in [15]. DSDV is a distance vector routing protocol that requires each node to advertise to its neighbors its whole routing table, including the next hop information to reach all other destinations in the network. DSDV sends updates periodically (proactive) or in the event of a link change and use sequence numbers in order to use the most recent information. Dynamic Source Routing (DSR) [3] is a reactive routing protocol based on source routing, so packets leave the source node with the complete route in their headers. DSR relies on a flooding route discovery mechanism to find out appropriate routes, and a route maintenance mechanism to know whether existing routes can still be used. The main advantage of DSR over DSDV is that intermediate nodes don't need to maintain routing tables. DSR doesn't need to send routing table updates periodically but need to implement a controlled flooding mechanism instead. Temporally-Ordered Routing Algorithm (TORA) [12] uses a controlled flooding mechanism to discover multiple routes from a source to a destination on a demand basis. In order to reduce overhead, TORA sometimes utilizes sub-optimal paths instead of triggering a new flooding procedure. Ad Hoc On-Demand Distance Vector (AODV) [14] is a source routing protocol based on DSDV and DSR. It utilizes the sequence numbers of DSDV and the on-demand route discovery and maintenance mechanisms of DSR.

In [4], a thorough performance evaluation of DSR, DSDV, AODV and TORA is presented analyzing the percentage of packets received and the routing overhead in relation to the pause time, or the time the mobile user is not moving before taking a new direction. They analyze the four routing protocols using different values of pause times, from continuous movement to no motion at all. One of the main conclusions and the one most related to our study is that DSR is the protocol that under no movement provides the highest percentage of packets delivered and the lowest routing overhead. These results are somehow expected because DSR, as a source routing protocol, doesn't need routing updates if nodes don't move and connections don't break. In addition, source routing is very effective if a valid route is available all the time. In the same paper, the authors show that TORA is the worst performing routing protocol in terms of overhead (over all pause times considered) and percentage of packets delivered (from medium to large pause times). These are the main reasons why we didn't include TORA in our performance evaluations.

Since its initial design, TCP has suffered many modifications, which have caused different versions to appear. The first version is TCP Tahoe introduced in 1990 [9]. In TCP Tahoe, the known slow start, congestion avoidance, and fast retransmit mechanisms are included. All these new mechanisms not only improve TCP's performance but also are responsible of the stability of the Internet. They are meant to probe the network slowly and reduce the sending rate drastically in the event of congestion. The fast retransmit mechanism allows for the retransmission of a missing packet faster and it is activated after the reception of three duplicate acknowledgements for the same TCP segment. TCP Reno includes two main additions, the fast recovery mechanism and the window inflation/deflation mechanism. The fast recovery mechanism is activated after fast retransmit and doesn't reduce the congestion window to one but only to half its current value. The reception of three duplicate acknowledgements is an indication that the network is not heavily congested, therefore TCP throughput can be improved not reducing the congestion window so drastically. The inflation and deflation mechanism allows TCP to continue sending new segments during the fast retransmit/fast recovery phase, avoiding timeouts. TCP New Reno was introduced to avoid timeouts in the case of multiple segments lost from the same window by staying in the fast retransmit/fast recovery phase until all the segments are correctly received. TCP SACK recuperates all segments lost from the current window faster than TCP NewReno by having specific information about the packets in flight and the ones correctly received. Other TCP versions have been published but not widely implemented as they are in research stages yet or are more specific for certain environments [16], [1], [17], [13], [2]. These TCP versions are not considered in this paper.

Not many performance studies of transport layer protocols over ad hoc networks exist. In [7], the authors rather focus on the interactions between TCP and different MAC layer protocols. They don't investigate the performance of different TCP versions over available ad hoc routing protocols but instead vary the MAC layer protocol. In [8] the authors investigate the throughput of TCP connections as a function of the node mobility and find that the throughput decreases considerably because of TCP's inability to differentiate between network congestion and link failures. The authors show that the inclusion of the Explicit Link Fail-

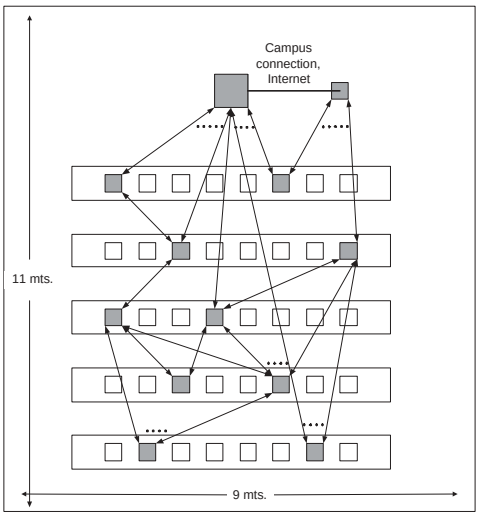

Fig. 1. The classroom ad hoc network

ure Notification (ELFN) mechanism improves TCP performance considerably. However, the study is restricted to DSR and TCP Reno. Ad Hoc TCP (ATCP [10]) introduces a thin layer between IP and TCP that receives network information in the form of Explicit Congestion Notification (ECN) messages and ICMP messages to control the behavior of TCP. Depending upon the received message, ATCP put TCP into persistent state, congestion control state, or retransmit state. However, the emphasis is on ATCP and its comparison against TCP and not TCP versions under different ad hoc routing algorithms.

III. Simulation Environment

In this section we describe our simulation environment and settings. Figure 1 shows a typical classroom, a 9x11 Mts. area with five rows of desks each with capacity to seat eight students. At the front is the professor desk with a wired connection to the school network. We assume that not all students have laptops and used a 4:1 ratio; only ten students and the professor will be connected by means of the ad hoc network. We also assume that once the network is set up users don't move until the end of the lecture.

We utilize the ns-2 simulator [11] and present simulation results focusing on two main performance metrics, TCP throughput and energy level. Simulations are run for a total of 500 seconds each. The mobility pattern is static; therefore once the initial position of the nodes is set no further node motion is allowed. In terms of energy, the initial energy for all nodes is set to 200 Joules, which is a realistic value taken from the specifications of a Li-ion rechargeable battery of a popular laptop. The battery is drained 0.5 Watts for every received packet and 0.3 Watts for every transmitted one. Packet sizes are limited to 512 bytes and acknowledgements are 40 bytes long. As mentioned in Section II, the same experiment will be performed making all possible combinations between TCP Tahoe, Reno, New Reno and SACK, and DSDV, DSR, and AODV so we can say which TCP version and routing protocol is the best combination for this environment. All the scripts used to run the simulations can be found at http://www.csee.usf.edu/˜labrador.

We consider several types of TCP connections and transmissions during the lecture time but the traffic in the network is introduced in a controlled manner so that we can assess the performance of the TCP connections under three different network conditions: low, medium, and high loads. The most important connections are the file transfers that the students perform to the Internet during the class time using the professor node as the main router. These connections allow us to vary the network load to the required level, as they are scheduled in the following order: Node number 1 (student 1) begins a file transfer connection at t=25 seconds that last for 250 seconds. Student number two begins its transmission 25 seconds later and last 250 seconds. The same pattern is repeated until the last student begins his/her transmission at 250 seconds and ends at 500 seconds (simulation end time). The result of this traffic pattern is the staircase pattern shown in Figure 2 showing the number of concurrent FTP connections to the Internet. Also, at time 5 seconds and until the simulation end time, all students establish file transfer connections to all other students. This is a mesh-type of situation created to establish kind of constant background traffic in the network. Each student transmission is generated using an exponential distribution with mean values of 0.2 and 0.8 seconds of duration for the transmission and quite time (ON and OFF mean times). During the ON time, the file transfer generates data at 64Kbps. Finally, the professor makes three file transfers to all the students during the lecture. At the beginning of the class, the professor transfers the lecture notes to all students. A similar file is transferred in the middle of the lecture, and another file is sent towards the end of the class. From Figure 2 it can be seen that the first file transfer occurs when the network is lightly loaded as there is a maximum of three concurrent FTP connections to the Internet; the second transfer occurs when the network is heavily loaded as there are 8, 9, or 10 simultaneous connections; finally, during the third transfer the network is moderately loaded having 4, 5, or 6 concurrent connections. We plot segment sequence numbers as an indication of TCP throughput and energy results for two nodes only, the professor node and node number 4. TCP sequence numbers are shown for the three file transfer connections from the professor to student number 4 and the file transfer of student number 4 to the Internet. (This connection begins at 100 seconds and ends at 350 seconds, and it is considered a variable load scenario - during this connection's life time, the number of concurrent connections from the students to the Internet goes from 4 to 10 and then from 10 to 7.) Energy results are shown for both, the professor and student number four nodes. Figure 1 only shows a few of these connections and Figure 2 describes all these connections.

TCP	Low Load		Medium Load		High Load		FTP Node 4	
Version	**Protocol**	**Seq. #**	**Protocol**	**Seq. #**	**Protocol**	**Seq. #**	**Protocol**	**Seq. #**
Tahoe	**AODV**	**500**	AODV	350	DSR	300	DSDV	950
	DSR	300	DSDV	350	AODV	285	DSR	850
	DSDV	250	DSR	300	DSDV	225	AODV	850
Reno	**DSR**	**400**	**DSR**	**370**	**DSR**	**300**	**DSR**	**1000**
	AODV	200	AODV	300	AODV	225	AODV	1000
	DSDV	200	DSDV	300	DSDV	190	DSDV	1000
New	AODV	450	DSDV	310	DSR	200	DSDV	1200
Reno	DSR	325	DSR	275	AODV	170	DSR	1000
	DSDV	325	AODV	250	DSDV	150	AODV	1000
SACK	DSR	475	DSDV	360	AODV	300	**AODV**	**1400**
	AODV	280	AODV	340	DSDV	240	DSR	900
	DSDV	225	DSR	250	DSR	210	DSDV	800

TABLE I
PERFORMANCE OF TCP VERSIONS USING DIFFERENT AD HOC ROUTING PROTOCOLS UNDER LOW, MEDIUM, AND HIGH LOADS IN A STATIC SETTING

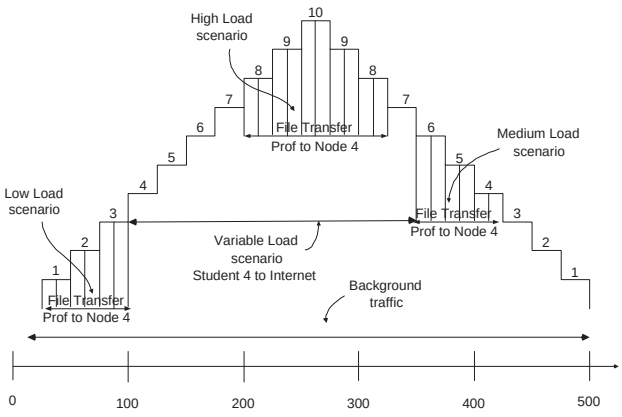

Fig. 2. The traffic pattern

IV. Simulation Results

We made experiments running all considered TCP versions (TCP Tahoe, Reno, New Reno, and SACK) over the different ad hoc routing protocols (DSR, DSDV and AODV). In this section we present all our simulation results showing the best combination when the network is lightly, medium, and highly loaded. Another important way of looking at our experiments is from the duration of the connection point of view. The FTP connections established to analyze the low, medium, and high load scenarios are of short duration (100 seconds each). We also present the best combinations for the FTP connection that student number 4 makes to the Internet. This can be considered a long-lasting connection since it is on during 250 seconds out of the 500 seconds simulation time. According to Figure 2, over the lifetime of this connection the network load varies from medium to high and back to medium load. As a result, we will also be able to say which combinations are the best for short and long-lived connections.

Table I summarizes the throughput results showing the sequence number that each TCP version achieved at the end of the simulation. (Best results are shown in bold for easier identification in the table.) The higher the sequence number the better the throughput as more segments are sent during the same period of time. We are looking for consistency and performance. Consistency refers to the combination that performs better in the majority of the scenarios. Consistency is very important because systems usually have one TCP stack and one routing protocol installed, therefore it is better to install the combination that performs the best under most scenarios. From the table it can be seen that TCP Reno along with DSR is the most consistent combination. In terms of the routing protocol, this result agrees with the one found in [4] in the case of a very long pause time (no mobility). For the short-lived connections, no matter the network load, TCP Reno along with DSR is always better than Reno with any of the other routing protocols. For long-lived connections the performance received by TCP Reno is independent of the routing protocol.

The fact that TCP Reno with DSR is the most consistent combination doesn't mean that it is always the best performing one. In fact, from the table, it can be seen that for the low load scenario TCP Tahoe with AODV is the best; TCP Reno with DSR is the best one in the medium load scenario; for the high load scenario TCP Tahoe with DSR, TCP Reno with DSR, and TCP SACK with AODV are the best performing combinations. In addition, TCP SACK with AODV is the best combination for long-lasting connections. In Figures 3, 4, 5, and 6 we show these best performing combinations.

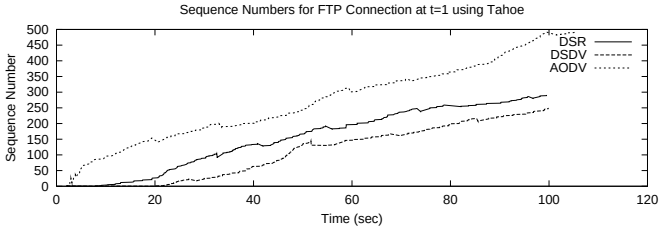

Fig. 3. TCP Tahoe and AODV is the best performing combination for low loads

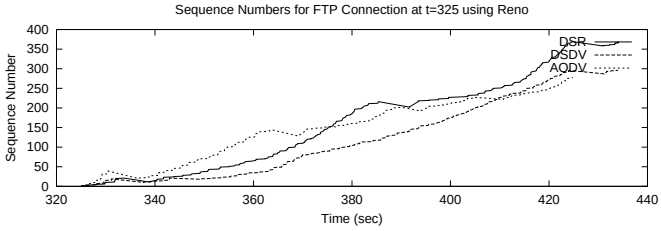

Fig. 4. TCP Reno and DSR is the best performing combination for medium loads

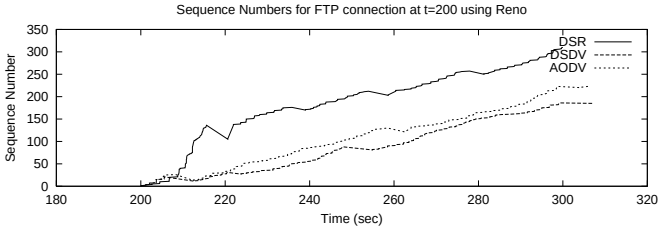

Fig. 5. TCP Reno and DSR is one of the best performing combinations for high loads

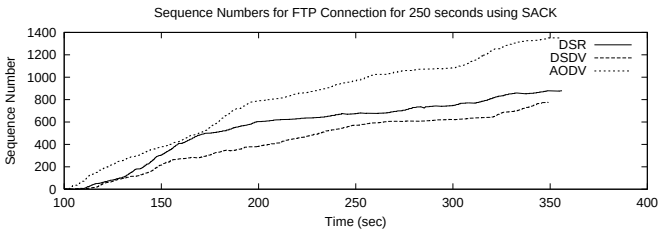

Fig. 6. TCP SACK and AODV is the best performing combination for long-lasting connections

It is worth noticing that our static ad hoc network differs from the scenarios discussed in other papers in two important aspects. First, the distance between nodes is such that each node is always able to hear each other and establish direct connections, therefore, there is no routing of packets. Second, the nodes don't move, therefore once the routes are established, the links do not break and the probability of stale routes in the nodes cache is very small. This latter point suggests that DSDV is not the best option for this kind of scenario, as we would not need frequent route updates. Transmitting routing table updates at regular intervals would consume considerable bandwidth and energy unnecessarily. Our throughput results confirm this as DSDV appears in the worst performing combinations. On the other hand, reactive source routing-based protocols like DSR and AODV, should perform better, as it can also be seen from our results. From the TCP point of view, one of the most important conclusions is that TCP SACK is perhaps the best performing TCP version as it is the second best performing TCP version in the low and medium load scenarios and the best one in the high load and long-connection cases. This result agrees with the findings in [6] where it is shown that TCP SACK performs better than TCP Tahoe, Reno, and NewReno in the presence of one, two, three, and four packet losses.

We also present energy results to show which combination consumes less power. For the energy results it is important to analyze three related values: The time at which the nodes transmit or receive the last packet, the total amount of battery left at the simulation end time, and the total amount of packets transmitted and received by all nodes. Table II shows the time at which nodes 0 and 4 transmit their very last packet. At that time, either the node itself run out of battery or all other active nodes run out of battery. The first observation is that the time for node 0 is always shorter. This is easily explained by the fact that this node is receiving packets from the FTP connections during the whole simulation time and receiving is more expensive than transmitting in terms of energy. The second and most important observation is that there is no major advantage of one combination vs. the others; all combinations perform fairly the same. Similar values (not shown here) were obtained for the total amount of battery left on each node after the simulation finished. These two values are related because a node can still have some energy but if all other nodes are out of battery, this particular node can't do anything else. We also looked at the total number of packets sent and received during the whole simulation time by all nodes using each combination. It is important relate the times for the last packets and the final energy levels to the total number of packets because on equal conditions the combination with the higher number of packets is the most energy efficient. In terms of number of packets we found that AODV is the routing protocol with higher number of packets no matter the TCP version, followed by DSDV and DSR. Two observations must be made though. First, the difference among them is really small. Second, DSDV handles a higher number of packets than DSR because the routing updates packets are included. If they were left out, DSR would have a higher number. We can conclude this part on power consumption saying that for our particular environment, we should avoid proactive routing protocols and could use any reactive protocol with any TCP version.

V. Conclusions and Further Research

In this paper we study the performance of TCP over static ad hoc wireless networks. Two important contributions are included in the paper. First, a complete coverage of the most widely used TCP versions and their performance over the most important ad hoc routing protocols is included in one

	DSR		DSDV		AODV	
TCP Version	**Node 4**	**Node 0**	**Node 4**	**Node 0**	**Node 4**	**Node 0**
Tahoe	465.75	466.6	467.56	468.35	467.07	467.13
Reno	465.7	466.28	468.78	470.00	467.15	468.58
NewReno	465.84	467.24	467.53	470.00	466.76	468.65
SACK	465.92	470.00	467.46	468.20	467.00	468.48

TABLE II
TIME AT WHICH NODE 0 AND NODE 4 TRANSMITTED OR RECEIVED THE LAST PACKET

study, not commonly found in other papers. Second, we focus on a very specific and very likely scenario: a closed room where individuals are meant to work together for a specific amount of time, such as a classroom, a business meeting or a conference in a hotel room, and the like. Usually, these studies have been performed using a general mobility pattern that provide good insights but don't necessarily reflect the real environment.

We show that TCP Reno along with DSR is the most consistent combination in terms of performance, making it the combination of choice for static ad hoc networks. From the energy consumption point of view, we conclude that proactive routing protocols should be avoided and any reactive protocol can be used along with any TCP version since the difference in energy consumption is almost unnoticeable.

Additional research will be carried out in several directions. First, we plan to perform similar studies on different environments so we can provide insights about what specific combinations of TCP versions and routing protocols are most useful on those settings. Second, we plan to study the TCP-friendliness and congestion collapse problems over ad hoc networks including the effect of UDP on network performance and power consumption.

References

[1] I. F. Akyildiz, G. Morabito, and S. Palazzo. TCP-Peach: A New Congestion Control Scheme for Satellite IP Networks. *IEEE/ACM Transactions on Networking*, Vol. 9, No. 3:307–321, 2001.

[2] L. S. Brakmo and L. L. Peterson. TCP Vegas: End to End Congestion Control on a Global Internet. *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 8:1465–1480, 1995.

[3] J. Broch, D. Johnson, and D. Maltz. *The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks*. IETF Internet draft (draft-ietf-manet-dsr-07.txt), February 2002.

[4] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols. In *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, October 1998.

[5] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash. A Feedback-based Scheme for Improving TCP Performance in Ad Hoc Wireless Networks. In *Proceedings of 18th. International Conference on Distributed Computing Systems*, pages 747–479, May 1998.

[6] K. Fall and S. Floyd. Simulation-based Comparisons of Tahoe, Reno, and SACK TCP. *Computer Communication Review*, pages 6–21, 1996.

[7] M. Gerla, K. Tang, and R. Bagrodia. TCP Performance in Wireless Multi-hop Networks. In *Proceedings of IEEE WMCSA*, 1999.

[8] G. Holland and N. Vaidya. Analysis of TCP Performance over Mobile Ad Hoc Networks. In *Proceedings of ACM Mobile Communications Conference*, pages 219–230, August 1999.

[9] V. Jacobson. Congestion Avoidance and Control. In *Proceedings of SIGCOMM*, pages 314–329, 1988.

[10] J. Liu and S. Singh. ATCP: TCP for Mobile Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications*, Vol. 19, No. 7:1300–1315, 2001.

[11] Network Simulator 2 (ns2). *http://www.isi.edu/nsnam/ns/*.

[12] V. Park and S. Corson. *Temporally-Ordered Routing Algorithm (TORA) version 1: Functional Specification*. IETF Internet draft (draft-ietf-manet-tora-spec-04.txt), July 2001.

[13] C. Parsa and G. Aceves. Improving TCP congestion control over Internets with heterogeneous transmission media. In *Proceedings of IEEE ICNP*, 1999.

[14] C. Perkins. *Ad Hoc On Demand Distance Vector (AODV) Routing*. IETF Internet draft (draft-ietf-manet-aodv-10.txt), January 2002.

[15] C. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance Vector Routing (DSDV) for Mobile Computers. In *Proceedings of the SIGCOMM Conference on Communications Architectures, Protocols and Applications*, pages 234–244, August 1994.

[16] A. Zanella, G. Procissi, M. Gerla, and M. Y. Sanadidi. TCP Westwood: Analytic Model and Performance Evaluation. In *Proceedings of IEEE Globecom*, pages 1703–1707, 2001.

[17] C. Zhang and V. Tsaoussidis. TCP Real: Improving Real-time Capabilities of TCP over Heterogeneous Networks. In *Proceedings of the 11th IEEE/ACM NOSSDAV*, New York, 2001.