

On Setting TCP's Congestion Window Limit in Mobile Ad Hoc Networks

Kai Chen, Yuan Xue, Klara Nahrstedt
Computer Science Department
University of Illinois at Urbana-Champaign
Urbana, IL 61801, U.S.A.
Email: {kaichen,xue,klara}@cs.uiuc.edu

Abstract—Improving TCP performance has long been the focus of many research efforts in mobile ad hoc networks (MANET). In this paper, we address one aspect of this endeavor: how to properly set TCP's congestion window limit (CWL) to achieve optimal performance. Past research has shown that using a small CWL improves TCP performance in certain scenarios [1], [2], however, no comprehensive study has been given.

To this end, we turn the problem of setting TCP's optimal CWL into identifying the bandwidth-delay product (BDP) of a path in MANET. We first show and prove that, independent of the MAC layer protocol being used, the BDP of a path in MANET cannot exceed the round-trip hop-count (RTHC) of the path. We further refine this upper bound based on the IEEE 802.11 MAC layer protocol, and show that in a chain topology, a tighter upper bound exists which is approximately $1/5$ of the RTHC of the path. Based on this tighter bound, we propose an adaptive CWL setting strategy to dynamically adjust TCP's CWL according to the current RTHC of its path. Using ns-2 simulations, we show that our simple strategy improves TCP performance by 8% to 16% in a dynamic MANET environment.

I. INTRODUCTION

Mobile ad hoc network (MANET) consists of many mobile nodes connected by wireless links. In MANET, TCP remains a de facto standard for reliable data transfer due to its wide acceptance over the Internet. However, recent research has shown that TCP performs poorly in MANET [1]–[7].

TCP's degraded performance in MANET can be attributed to a number of factors. At the MAC layer, wireless medium contention and hidden terminal problem can cause channel capturing, leading to link-layer packet dropping and unfairness. At the routing layer, mobility often leads to route breakage and re-routing, which in turn disrupts TCP transmissions. At the TCP layer, exponential retransmission time-out (RTO) backoff amplifies the route breakage effect, and may result in prolonged periods of TCP transmission blackouts. In response to these problems, many studies have been conducted to improve TCP performance in MANET.

In this paper, we focus on another aspect of improving TCP performance: how to properly set TCP's *congestion window limit* (CWL) to achieve optimal performance. TCP's CWL is the upper bound of its congestion window size that cannot be surpassed. Within this limit, TCP adjusts its congestion window according to its normal congestion control algorithm.

It has been observed that setting TCP's CWL to a large value in MANET would adversely affect its performance [1],

[2], [6], [7]. With a large CWL, TCP's congestion control algorithm often over-shoots, leading to network overloads and heavy contention at the MAC layer. In an early paper by Gerla et al. [1], the authors showed by simulations that TCP performance degrades for CWL greater than 1 packet when the MAC layer offers no ACK protection; with link-layer ACK protection (such as the one used in IEEE 802.11), certain performance gain can be realized by allowing a slightly larger CWL. The reason for a small CWL is to limit the contention between TCP data packets along the forward path, as well as data and acknowledgment packets along the forward and return path. Yet, no quantitative guideline of how to properly set this limit was given. In later studies [3], [5], this problem was largely ignored; instead, a CWL of 8 packets was chosen in their simulations as a "conventional" value in MANET. During the same time, other studies [6], [7] confirmed that a small CWL (e.g., 1 or 2 packets) achieves best TCP performance in their simulations.

Two recent studies by Li et al. [8] and Fu et al. [2] shed some new light to this problem by considering the spatial reuse property of the IEEE 802.11 MAC layer protocol in a chain topology. Their conclusion is that the maximum utility of a chain of nodes is $1/4$ of the chain length, due to transmission interference in a neighborhood area (details in Section III-A). Therefore, a sensible choice is to set TCP's CWL at $h/4$, where h is the length of the chain [2], because TCP tends to under-utilize the channel if its window size is below this value, and TCP cannot further increase the channel utilization if its window size is larger than that. Although this observation offers considerable insight into TCP's CWL setting problem, it is tied to the IEEE 802.11 MAC layer protocol, and the impact of the return path is not considered.

In this paper, we attempt to provide a more comprehensive study to this problem. To this end, we turn the problem of setting TCP's CWL into identifying the bandwidth-delay product (BDP) of a path in MANET. Based on this methodology, we first show and prove that, independent of any specific MAC layer protocol, the upper bound of a path's BDP cannot exceed its round-trip hop-count (RTHC). By considering the transmission interference of the IEEE 802.11 MAC layer protocol, we derive a tighter upper bound of BDP, which is approximately $1/5$ of the RTHC in a chain topology. Based on this result, we propose an adaptive CWL setting strategy to

dynamically adjust TCP's CWL according to its current path in MANET. Simulation results show that our simple strategy can improve TCP performance by 8% to 16%.

The rest of the paper is organized as follows. In Section II we show and prove an upper bound of BDP for a path in MANET independent of the MAC layer protocol. In Section III, we show that a tighter bound exists for the IEEE 802.11 MAC layer protocol. In Section IV we apply the adaptive CWL setting strategy in a dynamic MANET and show its performance improvements.

II. UPPER BOUND OF BANDWIDTH-DELAY PRODUCT

In this section we discuss our methodology in setting TCP's CWL, and prove a loose upper bound of BDP for a path in MANET, independent of the MAC layer protocol.

A. Background

TCP's congestion control is window-based, i.e., it keeps a congestion window to signal the availability of network bandwidth, and adjusts the window size according to the AIMD (Additive Increase Multiplicative Decrease) algorithm. In a typical TCP, the congestion window size is unbounded, which allows a TCP flow to obtain as much bandwidth as the network permits.

In steady state, TCP's AIMD algorithm converges to an average sending rate which takes up the available capacity of the network path measured by its *bandwidth-delay product* (BDP). By definition, a path's BDP is the bottleneck bandwidth of the forward path, times the packet transmission delay in a round-trip [9]. For instance, a T1 line crossing the continental U.S. has a maximum bandwidth-delay product of 1,544,000 bits/sec * 0.06 sec = 11,580 bytes. To ensure sufficient "pipelining" effect, TCP's congestion window size should be large enough to fill the "pipe", but it should never exceed the *upper bound* of the path's BDP. Further increase of congestion window will only cause network congestion, i.e., packet buffering and dropping, along the path.

As a result, a TCP flow should set its CWL to the upper bound of BDP of the current path (if that information is available), to ensure sufficient pipelining and to avoid the risk of overloading the network. Therefore, our basic methodology in this paper, is to identify the upper bound of the BDP of a path in MANET, and to use that as the CWL for a TCP flow.

B. Bandwidth-Delay Product in MANET

We carry over the same concept of BDP from wireline networks to MANET. The only difference is the MAC layer property. In MANET, in order to transmit a packet, a node has to "grab" the wireless channel via certain MAC layer protocol to resolve the contentions. Here we do not assume any particular MAC layer protocol; the only MAC layer property we assume is that, a channel cannot hold multiple packets "back-to-back" in one transmission. After transmitting a packet, the sender has to contend for the channel again for the next transmission.¹ This MAC layer property is

clearly very different from that in the wireline networks, where multiple packets can be pushed into a pipe, such as a long fiber link, back to back, without waiting for the first packet to reach the receiver.

This special property of wireless transmission makes the computation of BDP in MANET very different. In wireline networks, the bottleneck link's bandwidth needs to be uncovered by looking at the network topology map, and the round-trip delay can be measured by the ping tool. In MANET, as we shall see below, BDP of a path is tied to the path's *round-trip hop-count* (RTHC), independent of each link's bandwidth along the path. This is because in a wireless transmission, the delay of sending out a packet can be computed as S/b , where S is the size of the packet and b is the link's bandwidth in sending out the packet²; while in wireline networks, this is *not* the case because S/b is only the time it takes to *inject* the packet into the wireline pipe, not the delay that the receiver actually receives the packet. Before that, the sender may have injected multiple packets into the pipe.

We claim that in MANET, the BDP of a path with N round-trip hops cannot exceed N (times of the size of the data packet)³. Intuitively, when more than N packets are pushed into the path, there is at least one node (the bottleneck node) holding more than one packet, because the wireless medium cannot hold any packets. This always keeps the bottleneck node saturated. That means pushing more packets into the path cannot further increase the TCP flow's throughput. Below we give a formal proof of this result.

Theorem 1: In MANET, the bandwidth-delay product of a path cannot exceed N (times the size of the data packet), where N is the round-trip hop-count of the path, assuming similar bottleneck bandwidths along the forward and return paths.

Proof: Consider a pair of sender and receiver. The forward path has n hops of wireless links with bandwidths b_1, b_2, \dots, b_n ; the return path has m hops of wireless links with bandwidths b'_1, b'_2, \dots, b'_m . The bottleneck bandwidth of the forward path is $B_{min} = \min(b_1, b_2, \dots, b_n)$, and of the return path is $B'_{min} = \min(b'_1, b'_2, \dots, b'_m)$.

When a data packet with size S travels from the sender to the receiver along the forward path, the one-way delay is $\frac{S}{b_1} + \dots + \frac{S}{b_n} \leq \frac{S}{B_{min}} + \dots + \frac{S}{B_{min}} = n \frac{S}{B_{min}}$.⁴ Similarly, the one-way delay of traveling along the return path for a TCP acknowledgment packet (with size $S' \leq S$) is $\frac{S'}{b'_1} + \dots + \frac{S'}{b'_m} \leq m \frac{S'}{B'_{min}} \leq m \frac{S}{B'_{min}}$.

By definition, the bandwidth-delay product of the path is computed as the bottleneck bandwidth of the forward path, times the round-trip delay: $B_{min}(\frac{S}{b_1} + \dots + \frac{S}{b_n} + \frac{S'}{b'_1} + \dots + \frac{S'}{b'_m}) \leq B_{min}(n \frac{S}{B_{min}} + m \frac{S}{B'_{min}}) = S(n + m \frac{B_{min}}{B'_{min}})$. Although the forward and return paths do not necessarily travel along the same set of nodes (or symmetric), they should be geometrically

²Note that the channel contention time has been figured into the link's bandwidth b . In a heavily contended channel, the bandwidth b is smaller.

³In this paper, when we use N to represent BDP, we always mean " N times the size of the data packet".

⁴Note that queuing delay is not considered in computing BDP.

¹IEEE 802.11 MAC protocol clearly possesses this property.

close to each other in most cases. Therefore, we can reasonably assume that their bottleneck bandwidths should be similar: $B_{min} \simeq B'_{min}$. As a result, the bandwidth-delay product of the path cannot exceed $S(n + m)$, which is the round-trip hop-count $N(= n + m)$ times the size of the data packet. ■

III. A TIGHTER BOUND IN IEEE 802.11 MAC LAYER

In Theorem 1, we have not assumed any particular MAC layer protocol, nor considered any transmission interference between neighboring nodes. In this section, we derive a tighter upper bound of BDP for a path based on the IEEE 802.11 MAC layer protocol.

A. IEEE 802.11 Transmission Interference

Transmission interference of a TCP flow includes two parts: 1) TCP data packets' self interference along the forward path, and 2) TCP data packets and TCP acknowledgment packets along the forward and return paths.

The first part of interference is caused by IEEE 802.11 MAC layer's transmission interference which prevents concurrent transmissions within a neighborhood area. Past research has shown that the maximum spatial reuse of a chain of nodes is only $1/4$ of the chain length [2], [8]. Below we give a brief explanation of this result. Consider a chain of nodes separated by the *transmission range* of the radio, as shown in Figure 1. Transmission range is the maximum distance over which a radio signal cannot be correctly decoded, due to signal loss in propagation. Within certain distance beyond the transmission range, although the signal cannot be correctly received, it can still cause interference to other signals, preventing those signals from being correctly decoded. This longer distance is called the *interference range* of the radio, which largely depends on the physical environment and the propagation model. For instance, using the "Two-Ray-Ground" signal propagation model in the ns-2 simulator, the radio transmission range is 250m and the interference range is 500m. In Figure 1, when node E is transmitting a packet to node F, the nearest possible concurrent transmission is between A and B, because E's interference range covers node C, which prevents node C from correctly receiving a RTS packet from node B. Therefore, the maximum spatial reuse is $1/4$ of the chain.⁵ In a "perfect" scheduling scenario, all the data packets should be paced out evenly along the path, allowing concurrent pipelining transmission of the data packets.

The second part of interference is caused by TCP data packets and TCP acknowledgment packets along the forward and return paths. Although the forward and return paths do not necessarily overlap, they are usually close enough to cause contention for the wireless channel. The transmission of a data packet along the forward path will *prevent* the transmission of an acknowledgment packet along the return path in the same neighborhood, and vice versa. Because TCP's data and

⁵Note that this analysis depends on the interference range; a shorter interference range (e.g., less than 500m) may allow B and C to correctly exchange their RTS-CTS handshake, increasing the spatial reuse to $1/3$ of the chain length.

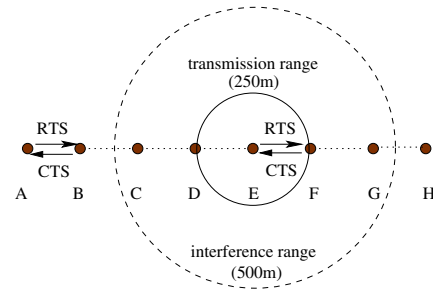


Fig. 1. 802.11 MAC layer interference of concurrent transmission in a chain topology, where the maximum spatial reuse is $1/4$ the chain length.

acknowledgment packets have to "clash" somewhere along the path, a contention-free "perfect" scheduling is not possible. In this case, if we reduce the number of packets to half ($1/2$), certain spatial reuse will be forfeited. Therefore, to accommodate this type of interference, the BDP reduction should be less than half.

Combining these two parts of interference, we arrive at the following conclusion:

Corollary 1: In a IEEE 802.11-based MANET, the upper bound of bandwidth-delay product of a chain is kN , where N is the round-trip hop-count of the path, and $1/8 < k < 1/4$ is a reduction factor due to transmission interference at the MAC layer.

B. Validation of Corollary 1

In this section, we validate Corollary 1 using the ns-2 simulator (v2.1b8a) [10]. Specifically, we want to show that BDP of a chain is bounded by kN , and k is approximately $1/5$ in our simulations.

The simulated chain topology consists of 16 nodes (from 0 to 15) each separated by the transmission range (250m) of the IEEE 802.11 MAC layer. In each simulation, a TCP sender at node 0 transmits a TCP-Reno flow to a receiver at node h ($1 \leq h \leq 15$). There is no other background traffic. The queuing space at each node is 25 packets, which is enough to accommodate the TCP traffic. TCP's data packet size is set to 1460 bytes, and each simulation run lasts for 1000 seconds.

The simulation data are collected as follows. For each receiver located at node h , a TCP-Reno flow is run each time with a different CWL (from 1 to 20 packets). Among these runs, we select the one with the best performance, i.e., largest number of successfully transmitted packets, and consider its CWL the achievable BDP of the path.

The simulation result shows that for a given chain with 1 to 15 hops, a TCP flow's successfully transmitted packets vary with its CWL (Figure 2). For instance, in the longest chain with 15 hops, the TCP flow achieves the best performance when its CWL is set at 5 packets; hence we consider 5 packets as the achievable BDP for the 30-hop round-trip path over the 15-hop chain. One observation from Figure 2 is that for long chains (i.e., 3 to 15 hops), TCP performance improves initially with the increase of the CWL, then degrades after the maximum spatial reuse has been reached. However, for

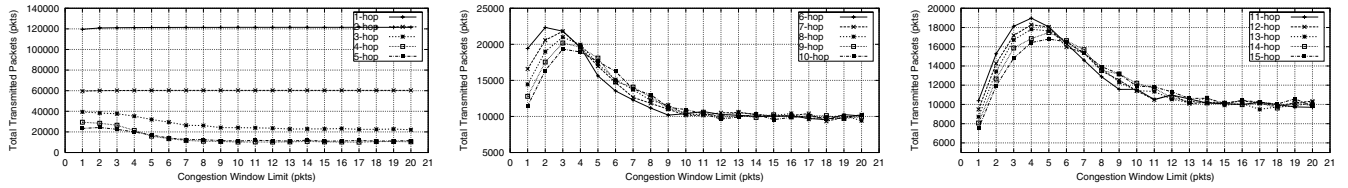


Fig. 2. Relation of a TCP flow's number of successfully transmitted packets with its congestion window limit in a chain topology of 1 to 15 hops.

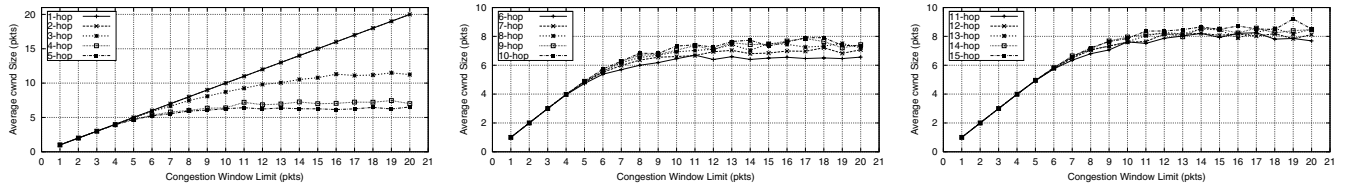


Fig. 3. Relation of a TCP flow's average congestion window size with its CWL setting in a chain topology of 1 to 15 hops.

short chains (i.e., 1 and 2 hops), TCP performance appears to stay unchanged with the increase of the CWL. This is because in a short chain, the MAC layer transmission self-interference problem (in Section III-A) does not exist. Therefore, in a short chain, a large CWL does not have the same negative effect on TCP performance as in longer chains. To further understand TCP's behavior with a different CWL setting, we examine the *average congestion window size* of a TCP flow in each run. Figure 3 shows that with the exception of the 1 and 2 hop chains (whose curves are overlapping with each other in the figure), the average congestion window size of a TCP flow increases initially with CWL, then flattens out. That means TCP initially improves performance with the increase of CWL because it gains more spatial reuse of the channel over the chain. After the maximum spatial reuse is reached, the average window size continues to increase a little bit more, because MAC layer's packet dropping probability increases gradually with the overloading of traffic [2]. When CWL is further increased, the window size flattens out, signaling heavy MAC layer packet dropping due to severe overloading of traffic.

Using this method, we are able to identify the achievable BDP of each path. The result in Figure 4 shows that the achievable BDP of each path can be bounded by kN with $k = 1/5$, where N is the round-trip hop-count of the path. This result validates our analytical prediction in Corollary 1, except the short chain cases (i.e., 1 and 2 hops) where MAC layer transmission self-interference does not exist. Figure 4 also suggests an adaptive CWL setting strategy according to the round-trip hop-count (N) of the current path:⁶

```

if (N <= 4) CWL = 2;
else if (N <= 8) CWL = 1;
else if (N <= 12) CWL = 2;
else if (N <= 20) CWL = 3;
else if (N <= 26) CWL = 4;
else if (N <= 30) CWL = 5;

```

⁶For a path longer than 30 round-trip hops, the optimal CWL can be obtained via similar simulations.

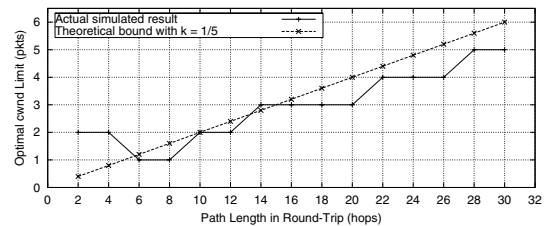


Fig. 4. Simulated result of the achievable BDP of a chain with length 1 to 15 hops. The achievable BDP can be bounded by kN with $k = 1/5$ where N is the round-trip hop-count of the path.

IV. ADAPTIVE CWL SETTING IN A DYNAMIC MANET

In this section, we apply the adaptive CWL setting strategy according to a TCP flow's current path, in a dynamic MANET. We use the simulated result of the achievable BDP in Section III-B, because it is lower and more accurate than the theoretical upper bound.

We claim that the achievable BDP obtained in a chain topology is higher than that of a path with same length in a dynamic MANET, because it has been obtained a) by separating the nodes as far as possible, and b) with no competing cross traffic. Since these conditions cannot be guaranteed in a dynamic MANET, a lower (and better) BDP bound may exist. However, it is extremely difficult to quantify such conditions in a highly dynamic and variable network. Therefore, we can only use the achievable BDP obtained in a chain topology as an approximation of the BDP upper bound of a path in a dynamic MANET.

The simulated network is as follows. There are 50 nodes moving around in a 1500m * 300m space using the "random way-point" mobility model with maximum speed of 5 m/sec and pause time of 0 seconds. This creates a moderately dynamic network. In this environment, we have made sure that the whole network is *not* partitioned at any time during

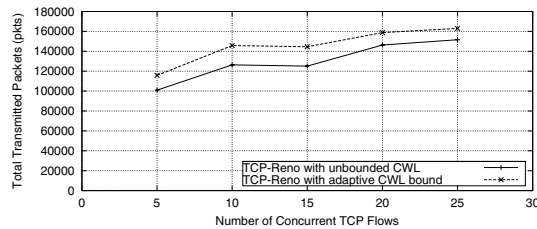


Fig. 5. Performance improvement (8% to 16%) of TCP-Reno with adaptive CWL setting strategy in a dynamic MANET.

the simulations; however, to let TCP probe the route quickly should it break (due to false link failure), we set the maximum retransmission time-out (RTO) to a relatively short 2 seconds. Each simulation run lasts for 1000 seconds. The routing protocol is DSR (Dynamic Source Routing).

We wish to quantify the performance improvement of TCP with the adaptive CWL setting strategy. To this end, we modify TCP-Reno to dynamically adjust its CWL according to the current path's round-trip hop-count.⁷ We create four different levels of traffic intensity in the network, each with a different number (5, 10, 15, 20 and 25) of concurrent TCP flows with random source and destination pairs. As comparison, we also test regular TCP-Reno flows with unbounded CWL (as commonly done) in the same setting. Figure 5 shows that TCP-Reno with adaptive CWL has a performance gain of 8% to 16% over TCP-Reno with unbounded CWL. That means our simple adaptive CWL setting strategy is effective in a dynamic MANET.

Two added benefits of using adaptive CWL can also be observed in our simulations: 1) shorter outgoing queue length and 2) fewer packet drops due to queue overflows. For instance, in the 25 concurrent TCP flows case, the average outgoing queue length (sampled when a packet enters into the queue) of all the 50 nodes is 0.24 packets with adaptive CWL, compared to 2.47 packets with unbounded CWL, which is a 10-fold improvement. At the same time, the total number of packet dropping due to queue overflow is reduced from 23105 packets to 15097 packets, a 53% decrease. These improvements further suggest that our simple adaptive CWL setting strategy is useful in preventing TCP's congestion window overshoots.

Note that compared to a small fixed CWL setting (e.g., 1 or 2 packets), although our adaptive CWL setting has better performance in most cases, we do not claim that it outperforms a small fixed CWL in every scenario. This is because our adaptive upper bound may be too high for certain scenarios in a dynamic MANET. Nevertheless, our adaptive CWL rule is generally applicable to *any* path in MANET, and its performance is usually better than that of a small fixed CWL setting.

⁷In MANET, round-trip hop-count of a path can be obtained through the routing protocol if source routing is being used (e.g. DSR). Alternatively, the IP header can be augmented to include a TTL-like counter to carry the hop count of the path.

V. CONCLUSION

Past research has observed that setting TCP's congestion window limit (CWL) to a small value would increase TCP performance in mobile ad hoc networks (MANET). In this paper we address the problem of how to properly set this limit for TCP to achieve its optimal performance. To this end, we turn the problem of setting TCP's optimal CWL into identifying the bandwidth-delay product (BDP) of a path in MANET. We show and prove that, independent of the MAC layer protocol being used, the BDP of a path in MANET cannot exceed the round-trip hop-count (RTHC) of the path. We further refine this upper bound based on the IEEE 802.11 MAC layer protocol, and show by simulations that in a chain topology, the upper bound is $1/5N$, where N is the RTHC of the path. We then propose an adaptive CWL setting strategy to dynamically adjust TCP's CWL according to the RTHC of the current path. Simulations show that our simple strategy is able to improve TCP performance by 8% to 16% in a dynamic MANET.

ACKNOWLEDGMENT

This work was supported by the DoD Multi-disciplinary University Research Initiative (MURI) program administered by the Office of Naval Research under Grant N00014-00-1-0564, and the NSF EIA 99-72884EQ grant. Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the above agencies.

REFERENCES

- [1] M. Gerla, K. Tang, and R. Bagrodia, "Tcp performance in wireless multi-hop networks," in *Proc. IEEE International Workshop on Mobile Computing Systems and Applications (WMCSA'99)*, New Orleans, Louisiana, USA, Feb. 1999.
- [2] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, "The impact of multihop wireless channel on tcp throughput and loss," in *Proc. IEEE Infocom 2003*, San Francisco, California, USA, Apr. 2003.
- [3] G. Holland and N. Vaidya, "Analysis of tcp performance over mobile ad hoc networks," in *Proc. ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99)*, Seattle, Washington, USA, Aug. 1999.
- [4] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash, "A feedback based scheme for improving tcp performance in ad hoc wireless networks," in *Proc. International Conference on Distributed Computing Systems (ICDCS'98)*, Amsterdam, The Netherlands, May 1998.
- [5] T. Dyer and R. Boppana, "A comparison of tcp performance over three routing protocols for mobile ad hoc networks," in *Proc. ACM Symposium of Mobile Ad Hoc Networking and Computing (MobiHoc 2001)*, Long Beach, California, USA, Oct. 2001.
- [6] Z. Fu, X. Meng, and S. Lu, "How bad tcp can perform in mobile ad hoc networks," in *Proc. IEEE International Symposium on Computers and Communications (ISCC'02)*, Taormina, Italy, July 2002.
- [7] K. Xu, S. Bae, S. Lee, and M. Gerla, "Tcp behavior across multihop wireless networks and the wired internet," in *Proc. ACM Workshop on Wireless Mobile Multimedia (WoWMoM'02)*, Atlanta, Georgia, USA, Sept. 2002.
- [8] J. Li, C. Blake, D.S.J.DeCouto, H. Lee, and R. Morris, "Capacity of ad hoc wireless networks," in *Proc. ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2001)*, Rome, Italy, July 2001.
- [9] W. Stevens, *TCP/IP Illustrated (Vol. 1, The Protocols)*. Reading, MA: Addison-Wesley, 1994.
- [10] The network simulator ns-2. [Online]. Available: <http://www.isi.edu/nsnam/ns/>