



Understanding bandwidth-delay product in mobile ad hoc networks

Kai Chen*, Yuan Xue, Samarth H. Shah, Klara Nahrstedt

Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA

Abstract

Bandwidth-delay product (BDP) and its upper bound (BDP-UB) have been well-understood in wireline networks such as the Internet. However, they have not been carefully studied in the multi-hop wireless ad hoc network (MANET) domain. In this paper, we show that the most significant difference of computing BDP and BDP-UB in MANET is the coupling of bandwidth and delay over a wireless link, where only one packet is allowed to be transmitted over the channel at a time. Based on this observation, we prove that BDP-UB of a path in MANET is upper bounded by N , where N is the number of round-trip hops of the path. We then further obtain two tighter bounds of BDP-UB, and verify them through ns-2 simulations.

The understanding of BDP and BDP-UB also contributes to the solution of how to properly set TCP's congestion window limit (CWL) in MANET, in order to mitigate TCP's congestion window overshooting problem. Past studies have shown that using a small CWL improves TCP performance in certain MANET scenarios, however, no quantitative guideline has been given. In this paper, we provide a systematic solution to this problem, by dynamically applying the path's BDP-UB as TCP's CWL. Simulation results show that our solution effectively improves TCP performance in a MANET environment.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Bandwidth-delay product; TCP; Congestion window limit; Mobile ad hoc networks

1. Introduction

Bandwidth-delay product (BDP) is a well-known concept in measuring the capacity of a 'network pipe' [1–3]. When applied to the context of the TCP protocol, the number of outstanding (i.e. in-flight or unacknowledged) data packets cannot exceed the TCP flow's share of BDP:

$$\text{BDP (bits)} = \text{available_bandwidth (bits/s)} \\ \times \text{round_trip_time(s)}, \quad (1)$$

where `available_bandwidth` is the TCP flow's share of bandwidth at the bottleneck router. When there is no competing traffic, the TCP flow should be allowed to obtain all the bandwidth (i.e. total bandwidth) at the bottleneck router. In other words, a TCP flow's BDP should not exceed the following:

$$\text{BDP-UB (bits)} = \text{total_bandwidth (bits/s)} \\ \times \text{round_trip_time(s)}. \quad (2)$$

Therefore, BDP-UB is the *upper bound* of BDP for a flow, and can be considered as a measurement of the *maximum* packet carrying capacity of the path.

BDP and BDP-UB have been well-understood in wireline networks such as the Internet. In order to take advantage of the 'pipelining' effect of packet transmission over a large BDP pipe, TCP's transmission window should be large enough to allow enough in-flight packets to fill the pipe. In fact, the role of TCP's Additive Increase Multiplicative Decrease (AIMD) congestion control algorithm is to dynamically 'probe' the current available bandwidth of the path, in order to reach an optimal congestion window size equaling its share of the BDP. When TCP's congestion window 'overshoots' its share of the BDP, packet dropping and queuing may occur. Without competing traffic, a TCP flow's share of the BDP should equal to the path's BDP-UB; with competing traffic, its share may be lowered. In either case, TCP's congestion window should *never* exceed the path's BDP-UB, because that is the maximum packet carrying capacity of the path; beyond this, no additional throughput can be obtained. We call the upper bound of TCP's congestion window as the Congestion Window Limit (CWL) of a TCP flow. Within the CWL, TCP adjusts its congestion window according to its normal AIMD algorithm.

* Corresponding author.

E-mail addresses: kaichen@cs.uiuc.edu (K. Chen); xue@cs.uiuc.edu (Y. Xue); shshah@cs.uiuc.edu (S.H. Shah); klara@cs.uiuc.edu (K. Nahrstedt).

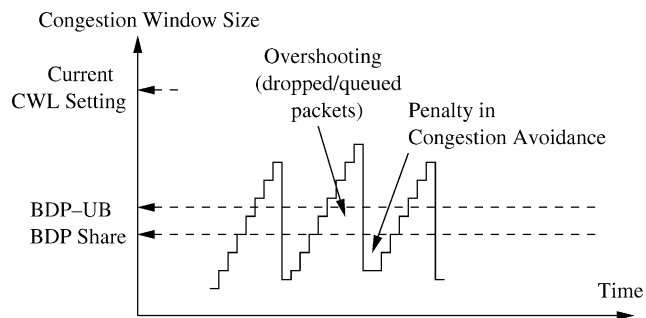


Fig. 1. Effects of CWL and BDP-UB in TCP's congestion window adjustment.

The effect of CWL and BDP-UB on TCP performance can be illustrated in Fig. 1. TCP's congestion window may overshoot its share of BDP, or even the path's BDP-UB, leading to dropped or queued packets, and subsequently paying the penalty of congestion avoidance. Intuitively, if TCP's CWL is limited below the path's BDP-UB, certain overshooting can be prevented, hence it helps to improve the overall TCP performance. In wireline networks such as today's high-speed Internet, a common practice is setting TCP's CWL as unbounded or to a very large value, in order to take advantage of the pipelining effect of packet transmission. This is because, in wireline networks, a router's 'sharp' drop-tail loss behavior is able to convey congestion signal quickly to the end-systems, hence alleviate the window overshooting problem. As we will see below, this is not the case in mobile ad hoc wireless networks.

A multi-hop mobile ad hoc wireless network (MANET) [4] is formed by a collection of mobile nodes connected by wireless links. Past studies have shown that TCP performs poorly in this environment (more details in Section 6). One reason for TCP's poor performance is its congestion window overshooting problem [5–8], which happens commonly in MANET. This is because, in contrast to wireline links, wireless MAC layer's dropping probability increases *gradually* when the network is overloaded [6], which allows a TCP sender's congestion window to 'push' beyond the capacity of the path. This phenomenon underscores the importance of limiting TCP's CWL to mitigate this problem. However, how to properly set this value remains an open problem in current research.

In an early paper by Gerla et al. [5], the authors showed by simulations that TCP performance degrades for CWL greater than 1 or 2 packets, due to medium contention between TCP data and acknowledgment packets. Therefore, using a small CWL alleviates packet contention at the MAC layer. Yet, the paper did not show any quantitative guidelines of how to set this limit. In later studies [9,10], TCP's CWL setting problem was largely ignored; instead, a CWL of eight packets was chosen in their simulations as a 'common' value in MANET. At the same time, other studies [7,8] confirmed

the fact that a small CWL (i.e. 1 or 2 packets) achieves best TCP performance in their simulations.

Two recent studies by Li et al. [11] and Fu et al. [6] shed some new light to the CWL setting problem by considering the transmission interference property of the IEEE 802.11 MAC layer protocol. Their conclusion is that the maximum wireless channel utilization in a chain of ad hoc nodes is 1/4 of the chain length (details in Section 3.1). Therefore, a sensible choice is to set TCP's CWL to that value. Although these two studies offer considerable insights into TCP's CWL setting problem in MANET, especially under the IEEE 802.11 MAC layer protocol, they have not uncovered the fundamental cause to this problem, which is linked to the upper bound of BDP of the path.

In this paper, we solve TCP's CWL setting problem by identifying BDP-UB of a path in MANET. We first examine the important difference in computing BDP-UB of a path in MANET and wireline networks. Specifically, we show that because a single wireless link cannot carry several packets 'back-to-back' in one transmission, the delay of transmitting a packet from a node to its next-hop neighbor is tightly coupled with the bandwidth of the wireless link. Based on this observation, we prove that BDP-UB of a path in MANET is tied to the number of round-trip hops of a path, i.e. it is bounded by $N \times S$, where N is the number of round-trip hops and S is the size of the TCP data packet, independent of the bandwidth of the wireless links.¹ We then further obtain two tighter bounds of BDP-UB, one based on the interference property of the IEEE 802.11 MAC layer protocol (called 'hop-based bound'), and the other based on the knowledge of per-hop transmission delays (called 'delay-based bound'), and verify them through ns-2 simulations. These two bounds provide a solid ground for our systematic solution to TCP's CWL setting problem. Finally, we compare the performance improvements of applying these bounds to TCP's CWL, and show that the hop-based bound is a simple and effective solution.

Our contribution in this paper is two-fold. At the theoretical front, we show the fundamental difference of how to compute BDP and BDP-UB in the MANET domain, and prove two bounds for BDP-UB. At the practical front, we provide a solid and systematic solution to TCP's CWL setting problem in MANET, which effectively improves TCP performance.

The rest of the paper is organized as follows. In Section 2, we discuss the difference in computing BDP and BDP-UB in MANET and wireline networks, and give a loose bound for BDP-UB. Two tighter bounds are then introduced in Sections 3 and Section 4. In Section 5, we show the performance improvement of TCP by applying the BDP-UB bounds. Section 6 discusses additional related work. Section 7 concludes the paper.

¹ For simplicity, we sometimes use N to represent BDP-UB, which always means " N times the size of the TCP data packet".

2. Bandwidth-delay product and its upper bound in MANET

2.1. Computing BDP-UB in MANET

We carry over the same concept of BDP-UB from wireline networks (in Eq. (2) in Section 1), as a measurement of the maximum packet carrying capacity of a path. The important difference of computing BDP-UB in MANET lies in the special property of the wireless MAC layer. In MANET, mobile nodes are connected by wireless links. Before transmitting a packet, a node has to contend for the channel following the MAC layer protocol, which is responsible for resolving the conflict in accessing the shared channel. Here we do *not* assume any particular MAC layer protocol. We only assume the following property of wireless packet transmission: 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. For instance, in the IEEE 802.11 MAC layer protocol, the sender can only send a data packet and get an acknowledgment back, before it contends for the channel again. This packet transmission property is clearly very different from that in wireline networks, where multiple packets can be pushed into a pipe, such as a long high-speed link, without waiting for the first packet to reach the other end of the link.

The difference of these two types of packet transmission is illustrated in Fig. 2. Over a wired link, the delay of sending a packet (from a sender sitting at one end of the link to a receiver sitting at the other end) includes not only the time to push the packet into the link (i.e. transmission delay), but also the time for the signal to propagate through

the link (i.e. propagation delay). Between the sender and the receiver, the link can hold many packets back-to-back. Whereas over a wireless link, this is not the case, because the receiver must have received the packet before the sender can start to transmit another one. In other words, the wireless channel cannot hold multiple packets in the air. As a result, the delay (d) of sending out a packet over the wireless link is tightly *coupled* with the link’s bandwidth, as in the following equation: $d = S/b$, where S is the size of the packet, and b is the wireless link’s *effective* bandwidth in sending out that packet. Note that we refer to the link’s bandwidth b as its effective bandwidth, which has considered the channel contention overhead. In a more heavily contended channel, the link’s effective bandwidth is smaller. In contrast, a wired link’s bandwidth and delay cannot be completely correlated to each other, because S/b is only the time to *inject* a 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.

This special property of wireless packet transmission makes the computation of BDP-UB very different in MANET. In wireline networks, BDP-UB can be calculated based on the bottleneck link’s bandwidth and the round-trip delay. For instance, over a trans-continental T1 fiber link, BDP-UB can be calculated as $1,544,000 \text{ bits/s} \times 0.1 \text{ s} = 19,300 \text{ bytes}$. The round-trip delay (0.1 s) is dominated by the light signal’s propagation delay inside the fiber. In MANET, because of the tight coupling effect of a wireless link’s bandwidth and delay, a very different result is that, BDP-UB of a path is tied to the number of round-trip hops of a path, *independent* of each link’s bandwidth along that path. We will prove this result formally in the following section.

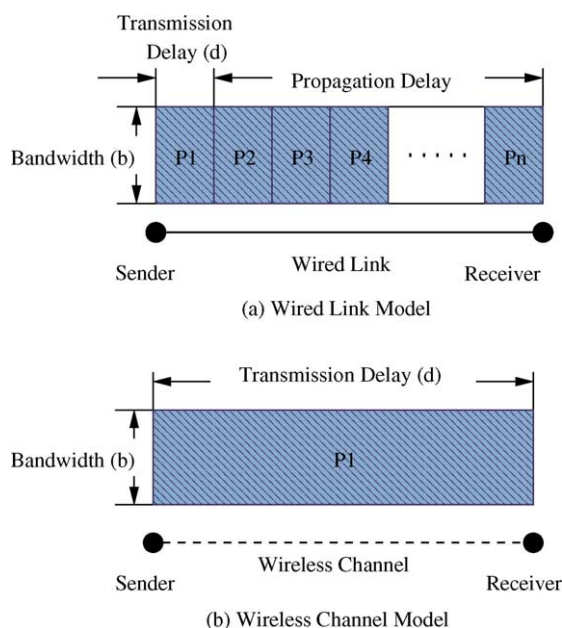


Fig. 2. The difference of packet transmission over wired link and wireless channel.

2.2. Loose upper bound of BDP-UB in MANET

We claim that in MANET, BDP-UB of a path with N number of round-trip hops cannot exceed N . A model of the end-to-end TCP transmission over MANET is shown in Fig. 3, where only one packet is allowed over a wireless link. When exactly N TCP data packets are allowed to be outstanding in Fig. 3(a), at least one packet will be queued at the bottleneck router. This always keeps the bottleneck saturated, which means pushing more packets into the path (as in Fig. 3(b)) cannot further increase the TCP flow’s throughput. It only increases the backlog at the bottleneck router. Below we give a formal proof of this result. Note that in the proof, the special property of wireless packet transmission is reflected in the correlation of a link’s effective bandwidth and its packet transmission delay (i.e. $d = S/b$).

Theorem 1. *In MANET, the upper bound of BDP of a path cannot exceed $N \times S$, where N is the number of round-trip hops and S is the size of the TCP data packet, assuming*

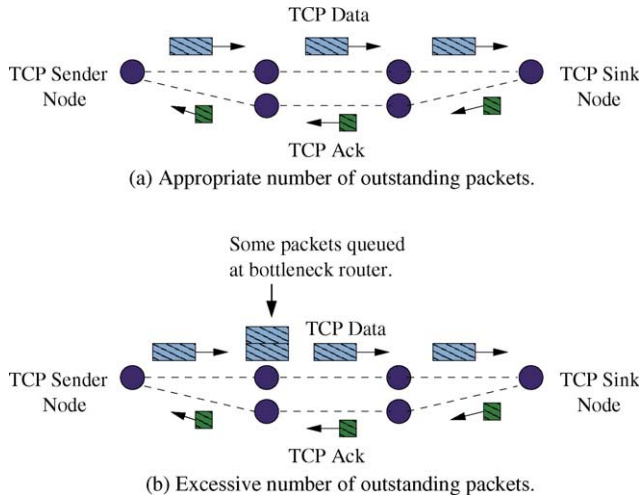


Fig. 3. A conceptual model of TCP transmission over MANET.

similar bottleneck bandwidth along the forward and return paths.

Proof. Consider a pair of sender and receiver nodes. The forward path has n hops of wireless links with bandwidth b_1, b_2, \dots, b_n ; the return path has m hops of wireless links with bandwidth 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

$$\frac{S}{b_1} + \dots + \frac{S}{b_n} \leq \frac{S}{B_{\min}} + \dots + \frac{S}{B_{\min}} = n \frac{S}{B_{\min}}.$$

Note that router's queuing delay should not be included in computing BDP. 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 [1–3], the upper bound of the bandwidth-delay product (BDP-UB) of the path is computed as the bottleneck bandwidth of the forward path, times the round-trip delay:

$$\begin{aligned} \text{BDP-UB} &= B_{\min} \left(\frac{S}{b_1} + \dots + \frac{S}{b_n} + \frac{S'}{b'_1} + \dots + \frac{S'}{b'_m} \right) \\ &\leq B_{\min} \left(n \frac{S}{B_{\min}} + m \frac{S}{B'_{\min}} \right) = S(n+m) \left(\frac{B_{\min}}{B'_{\min}} \right). \end{aligned}$$

Although the forward and return paths do not necessarily travel along the same set of nodes (i.e. symmetric), they are typically geometrically close to each other. Therefore, we can reasonably assume that their bottleneck bandwidths should be similar: $B_{\min} \approx B'_{\min}$. As a result, BDP-UB of the path cannot exceed $S(n+m)$, which is $S \times N$.

Remark 1. An implicit assumption in the proof is that concurrent transmissions are allowed between neighboring nodes at the MAC layer. This is not the case with the use of omni-directional antennas, due to the signal interference within a neighborhood area. We will exploit this interference property based on the popular IEEE 802.11 MAC layer in Section 3 to derive a tighter bound for BDP-UB.

Remark 2. In the proof, we have used the maximum delay of the forward path (i.e. S/B_{\min}) to bound the round-trip delay, leading to a loss of precision in the computation. If individual per-hop delays are available, the computation of BDP-UB may be made more precise. We will explore this possibility in Section 4 to derive another tighter bound for BDP-UB.

3. Hop-based upper bound of BDP-UB

In this section, we derive a tighter bound of BDP-UB based on the IEEE 802.11 MAC layer. We call it 'hop-based' bound because it depends on the number of round-trip hops of the path.

3.1. Transmission interference under IEEE 802.11

TCP data packets may encounter self-interference along the forward path, caused by IEEE 802.11 MAC layer's channel interference 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 [6,11]. Below we give a brief explanation of this result. Consider a chain of nodes separated by the *transmission range* of the wireless signal, as shown in Fig. 4. Transmission range is the maximum distance more than which a wireless 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 wireless signal, which largely depends on the physical

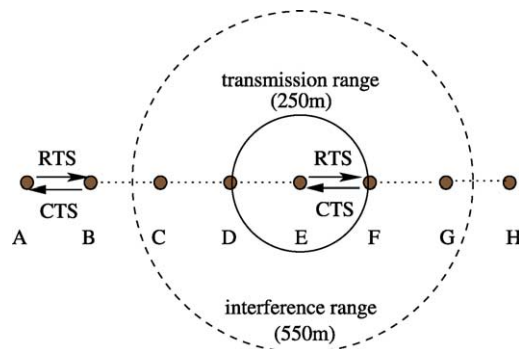


Fig. 4. Transmission interference under IEEE 802.11 in a chain topology.

environment and the propagation model. For instance, using the ‘Two-Ray-Ground’ signal propagation model in the ns-2 simulator [12], the default transmission range is 250 m and the interference range is 550 m. In Fig. 4, 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 the RTS packet from node B. Therefore, the maximum spatial reuse is 1/4 of the chain.² 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. Here we assume the TCP receiver acknowledges every data packet it receives. 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 concurrent transmission of an acknowledgment packet along the return path in the same neighborhood, and vice versa. In this case, if we reduce the number of packets to more than half, certain spatial reuse will be forfeited. Therefore, to accommodate this type of interference, BDP-UB of the path should be reduced by less than half.

Combining these two types of interference, i.e. 1/4 reduction of BDP-UB due to MAC layer interference, and 1/2 reduction due to TCP’s data and acknowledgment packets traveling along different directions, 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 cannot exceed kN , where N is the number of round-trip hops, and $1/8 < k < 1/4$ is a reduction factor due to transmission interference at the MAC layer.

Remark 3. In Corollary 1, k indicates the degree of interference by a TCP flow’s acknowledgment packets. A larger k value (i.e. closer to 1/4) means that the interference from the acknowledgment packets is smaller, and the chain of nodes can accommodate more in-flight packets. Since an exact k value depends on the scheduling of packets along the forward and return paths, it is very difficult to be theoretically derived. In the next section, we will resort to simulation to obtain an empirical k value.

Remark 4. Corollary 1 is obtained in a best-case chain topology, where spatial reuse has been maximized. In a random topology, spatial reuse may be reduced. As

² Note 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.

a result, the packet carrying capacity of a path under a random topology should be *smaller* than that in the chain topology. Therefore, the upper bound of BDP-UB in Corollary 1 still holds for a path with the same number of round-trip hops in a random topology.

3.2. Validation of Corollary 1

We validate Corollary 1 using the ns-2 simulator [12]. Specifically, we want to show that BDP-UB of a chain cannot exceed kN , where k is bounded between 1/4 and 1/8, and we want to obtain an empirical k value from simulations.

The simulated chain topology consists of 16 stationary nodes (from 0 to 15), each separated by the transmission range (250 m) of the IEEE 802.11MAC layer capable of 2 Mbps transmission rate. In each simulation, a TCP sender at node 0 transmits a TCP flow³ to a receiver at node h ($1 \leq h \leq 15$). There is no other background traffic, i.e. the TCP flow can obtain the maximum packet carrying capacity of the chain. TCP’s data packet size is set to 1460 bytes. Each simulation run lasts for 1000 s. At the end of each run, we obtain the *average throughput* of a TCP flow over the entire course of the simulation, in terms of the number of successfully (i.e. acknowledged) transmitted packets per second.

We obtain the ‘true’ BDP-UB of a path based on the following observation: when there is no competing traffic, best TCP performance can be achieved only when its CWL is set to the path’s BDP-UB. This is because the TCP flow’s share of BDP equals to the path’s BDP-UB without competing traffic, and TCP’s best performance can be obtained only when its CWL is set to its share of the BDP. Specifically, if CWL is smaller than the path’s BDP-UB, increasing CWL will allow more pipelining effect, which leads to better performance; if CWL is larger than BDP-UB, it leads to more congestion window overshooting, which decreases TCP’s overall performance. The ‘optimal’ CWL should correspond to the true BDP-UB of the path. Therefore, we are able to obtain the true BDP-UB of a chain through simulation as follows: for each receiver located at node h , a TCP flow runs each time with a different CWL (from 1 to 20 packets). Among these runs, we select the TCP flow with the best throughput, and consider its CWL as the optimal CWL, which reflects the true BDP-UB of the path: $\text{BDP-UB} = \arg \max_{\text{CWL}}(\text{Throughput}(\text{CWL}))$.

The simulation result in Fig. 5 shows that for a given chain with 1–15 hops, a TCP flow’s performance varies with its CWL. For instance, in the longest chain with 15 hops, the TCP flow achieves the best performance when its CWL is set to five packets; hence we consider five packets as the BDP-UB for the 15-hop chain. One observation from Fig. 5 is that for long chains (3–15 hops), TCP performance

³ We use TCP-Reno as the TCP version in our simulations.

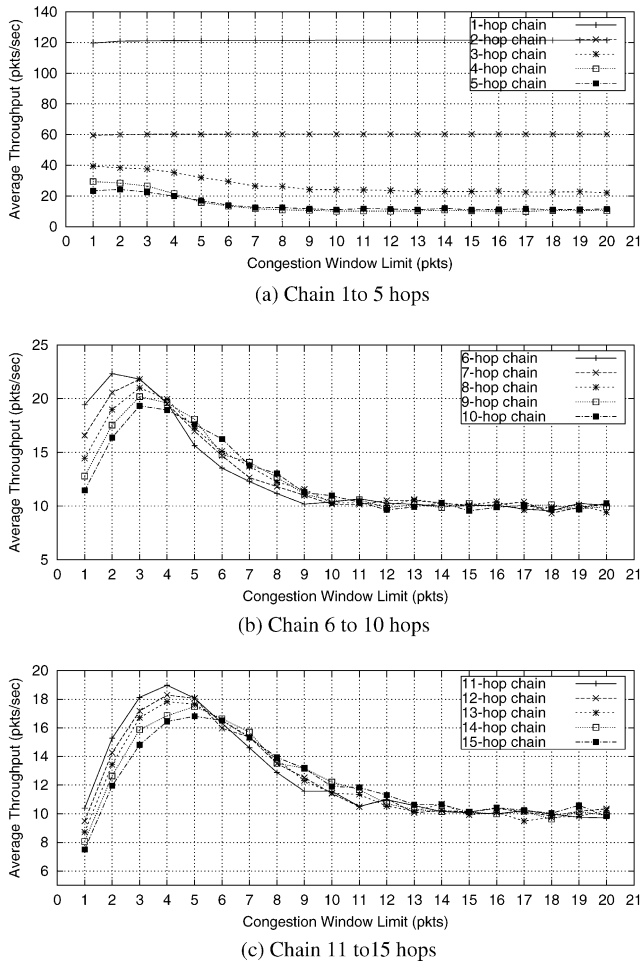


Fig. 5. TCP flow’s number of successfully transmitted packets varies with its CWL.

improves initially with the increase of CWL, then degrades after the optimal CWL (or the path’s BDP-UB) has been reached. However, for short chains (1 and 2 hops), TCP performance appears to stay unchanged (or very minimally changed) with the increase of CWL. This is because in a short chain, the self-interference problem by TCP’s data and acknowledgment packets is less severe due to the small number of contending nodes. Therefore, in a short chain, a large CWL does not have the same negative impact on TCP performance as in longer chains. From the result in Fig. 5(a),

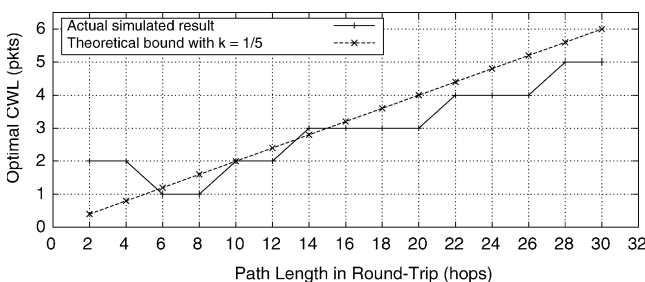


Fig. 6. Optimal CWL of TCP flow over a chain topology.

Table 1
Simulation result of TCP flow’s optimal CWL

Round-Trip Hops (N)	Optimal CWL
$N \leq 4$	2
$4 < N \leq 8$	1
$8 < N \leq 12$	2
$12 < N \leq 20$	3
$20 < N \leq 26$	4
$26 < N \leq 30$	5

we choose the optimal CWL as 2 packets for the 1 and 2 hop chains, although its TCP performance is only *slightly* better than using other CWLs.

From Fig. 5, we are able to identify the optimal CWL (or true BDP-UB) of each path. We then plot the relation of BDP-UB with the round-trip hop-count of the path (which is twice the chain length), in Fig. 6. It shows that BDP-UB can be bounded by kN with $k = 1/5$, where N is the number of round-trip hops, especially in the long chain cases. This result validates our analytical prediction in Corollary 1. Fig. 6 also suggests a CWL setting strategy based on the number of round-trip hops of the path, as shown in Table 1. The results for longer chains can be obtained through similar simulations.

To further understand TCP’s behavior with different CWL settings, we examine the *average congestion window size* of a TCP flow in each run. Fig. 7 shows that except the short chain cases (e.g. 1 and 2 hops), the average congestion window size of a TCP flow increases initially with CWL, then flattens out. That means TCP initially gains more throughput by the pipelining effect of packet transmission, but the congestion window continues to grow even after the optimal CWL is reached (corresponding to the path’s BDP-UB). The packet queue size is kept relatively large (25 packets) in these simulations, therefore, packet loss is mainly due to MAC layer contention loss, which is similar to the observation in Ref. [6]. This is an example that TCP’s congestion window can easily overshoot in MANET, and hence underscores the importance of properly limiting TCP’s CWL to mitigate this problem.

4. Delay-based upper bound of BDP-UB

As mentioned earlier in Remark 2 (following Theorem 1), if individual per-hop delays along the path are available, the computation of BDP-UB may be made more precise. In this section we will derive a tighter bound of BDP-UB based on this observation (called the ‘delay-based’ bound).

4.1. Consideration of Per-Hop delays

Suppose the individual per-hop delays are d_1, d_2, \dots, d_n along the forward path, and d'_1, d'_2, \dots, d'_m along the return path.

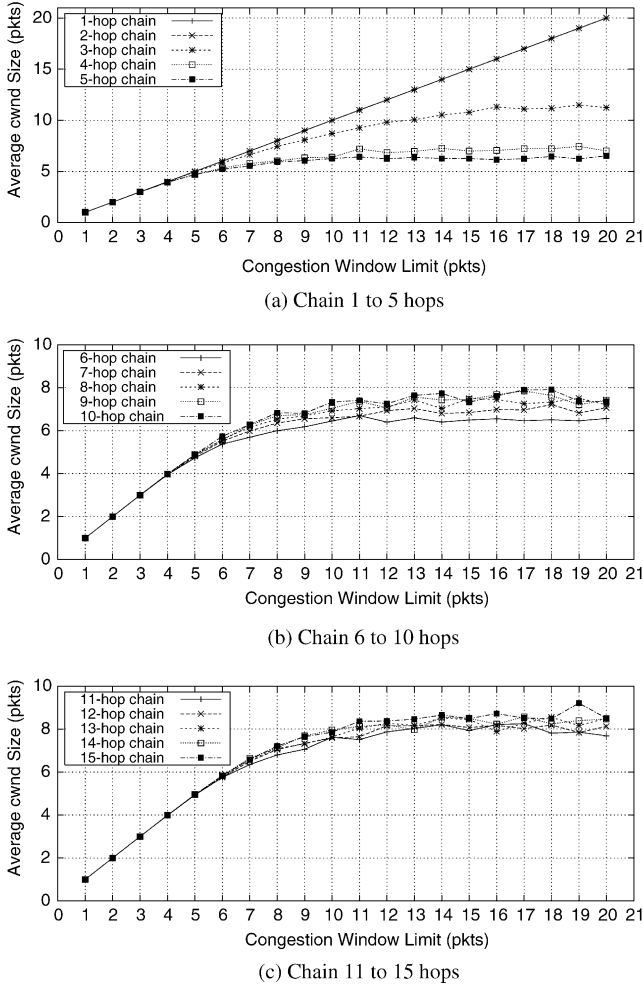


Fig. 7. Relation of TCP's average congestion window size with its CWL settings.

The maximum per-hop delay of the forward path is $d_{\max} = \max(d_1, d_2, \dots, d_n)$, which corresponds to the bottleneck bandwidth as: $B_{\min} = (S/d_{\max})$, where S the TCP data packet size. By definition, BDP-UB of the path is:

$$\begin{aligned} \text{BDP-UB} &= B_{\min} \times \text{round_trip_delay} = \frac{S}{d_{\max}} \left(\sum_{i=0}^n d_i + \sum_{i=0}^m d'_i \right) \\ &= S \left(\frac{\sum_{i=0}^n d_i + \sum_{i=0}^m d'_i}{d_{\max}} \right). \end{aligned}$$

Therefore, BDP-UB of the path is bounded by a factor determined by the round-trip delay and the maximum delay of the forward path. This bound is lower than N in Theorem 1, because the *maximum* per-hop delay d_{\max} is used as the denominator in the computation. Finally, due to the concurrent transmission interference under the IEEE 802.11 MAC layer (as in Section 3.1), this bound should be

reduced by a factor of 1/4. Therefore, we arrive at the following conclusion.

Corollary 2. In a IEEE 802.11-based MANET, the upper bound of bandwidth-delay product of a chain of nodes cannot exceed

$$\frac{\sum_{i=0}^n d_i + \sum_{i=0}^m d'_i}{4d_{\max}},$$

where d_i and d'_i are the per-hop packet transmission delays along the forward and return paths.

Remark 5. Corollaries 1 and 2 are both derived from Theorem 1 which gives a loose upper bound of N . Corollary 1 directly applies the neighborhood interference property of IEEE 802.11 to obtain a tighter bound. Corollary 2, on the other hand, obtains per-hop delay information to compute a more precise and lower BDP-UB, and then applies IEEE 802.11's interference property. Therefore, the relations of these bounds are: $\text{BDP-UB}_{\text{Theorem1}} > \text{BDP-UB}_{\text{Corollary1}} > \text{BDP-UB}_{\text{Corollary2}}$.

Remark 6. The deployment of Corollary 2 is more complicated than Corollary 1, because it requires a MAC layer delay estimation mechanism, and two additional IP header fields to 'probe' the maximum and total delay information along the path. Therefore, although Corollary 2 is theoretically viable, it depends critically on the accuracy and robustness of the packet delay estimation mechanism.

4.2. Validation of Corollary 2

We have validated the use of hop-based CWL in Section 3.2. In this section, we validate the delay-based CWL by comparing its result with the hop-based CWL over the same network topology.

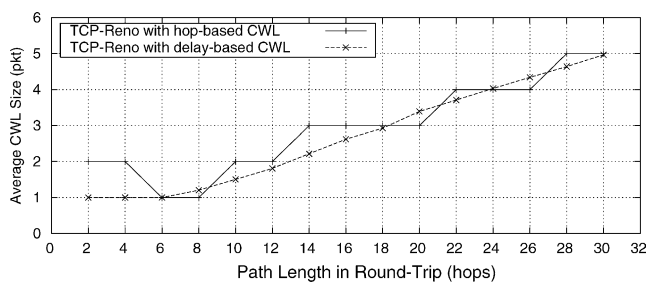
In our simulation, delay information is probed with each TCP data packet using two additional IP header fields: a 'total-delay' field to accumulate the per-hop delays, and a 'maximum-delay' field to carry the maximum per-hop delay. Each router along the path modifies these two fields according to the estimated packet transmission delay to the next-hop neighbor of the packet, which is available from the MAC layer [14]. The maximum delay of the forward path is then returned to the TCP sender in the TCP ACK packets. Upon receiving a TCP ACK packet, the TCP sender computes the bound for BDP-UB according to Corollary 2, and uses that as its CWL. Note that the CWL should be set at least to 1 packet, even when the result from Corollary 2 suggests a smaller bound, to avoid stalling the TCP flow.

Over the chain topology, a TCP flow is transmitted from node 0 to node h ($1 \leq h \leq 15$) in each simulation run.

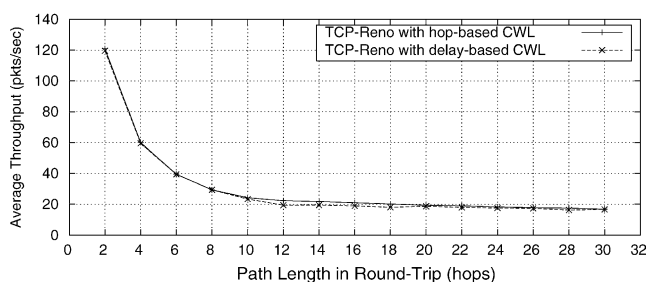
All other parameters remain the same as in Section 3.2. At the end of each run, we collect the following three metrics for the TCP flow: (1) average throughput; (2) average congestion window size; and (3) average CWL size. We then compare these metrics with those obtained in Section 3.2 to illustrate the differences between the hop-based bound and the delay-based bound.

We first compare the average CWL size in Fig. 8(a). The result shows that the CWL size obtained from the hop-based and delay-based bounds are very close to each other, especially in the long chain cases (18–30 round-trip hops). In the medium chain cases (8–16 round-trip hops), hop-based CWL is slightly larger than the delay-based CWL. In short chain cases (2 and 4 hops), the hop-based CWL is much larger, because we have selected CWL to be 2 packets for these cases, although its theoretical value should be $2/5$ and $4/5$, respectively. Overall, the two bounds sufficiently agree with each other.

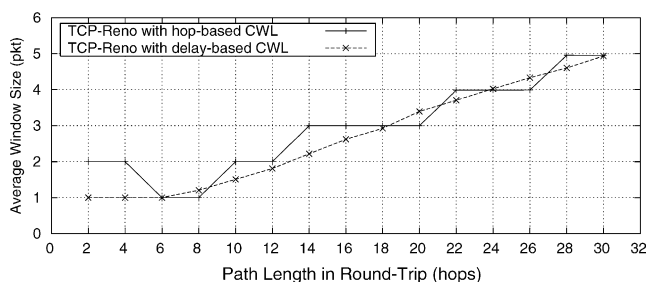
Next, we compare the overall TCP performance (i.e. average throughput) of using hop-based CWL and delay-based CWL. The result in Fig. 8(b) shows that their



(a) Average CWL Size



(b) Overall Performance



(c) Average Congestion Window Size

Fig. 8. Comparison of using hop-based CWL and delay-based CWL in chain topology.

overall performance is very close to each other. This is not surprising because their CWLs are very similar. We further examine each flow's average congestion window size in Fig. 8(c). It shows that the average congestion window sizes are almost identical to their corresponding CWL sizes in Fig. 8(a). That means both hop-based and delay-based CWL are able to mitigate congestion window overshooting, and hence improve TCP performance.

5. Setting TCP's CWL using BDP-UB bounds in MANET

5.1. Methodology

So far we have proved and validated two upper bounds of BDP-UB, which can be applied to TCP's CWL setting in a dynamic MANET environment. Hop-based CWL is derived from the round-trip hop count of the current path. In MANET, round-trip hop count of a path can be obtained from the routing protocol if source routing is being used (e.g. DSR [13]); alternatively, each packet's IP header can be augmented to include a TTL-like counter to carry the hop count of the path. A TCP flow's CWL is set according to the simulation result in Table 1, because it is a little bit lower than the theoretical bound of BDP-UB from Corollary 1.

Delay-based CWL of a TCP flow is computed from the maximum per-hop delay of the forward path and the total round-trip delay, according to Corollary 2. As in the chain topology, we implement a packet delay estimation mechanism at the MAC layer. If the computed result is less than 1 packet, the CWL is set to 1 packet.

5.2. Evaluation

We evaluate our approach in ns-2 (version ns-2.1b9a) using the following simulation network: there are 50 nodes moving around in a $1500\text{ m} \times 300\text{ m}$ space using the 'random way-point' mobility model with maximum speed of 5 m/s and pause time of 0 s.⁴ This creates a moderately dynamic network. In this environment, we make sure that the whole network is *not* partitioned at any time during the simulations. Moreover, in order to limit the impact of 'false link failure'⁵, we set the maximum re-transmission time-out (RTO) to a relatively short 2 s, to let the TCP sender recover from the failure quickly. Each simulation run lasts 1000 s.

⁴ The random way-point mobility model is recently shown to decrease nodal speed over time [15]. This is not a significant problem in our analysis because we are not concerned about the exact mobility speeds.

⁵ False link failure is a phenomenon at the MAC layer where two nodes cannot talk to each other temporarily, even though they are within each other's transmission range, due to the hidden terminal problem or the capturing of the wireless channel by other transmissions.

Dynamic source routing (DSR [13]) is used as the routing protocol.

We create several levels of traffic intensity in the network, each with a different number of concurrent TCP flows (5, 10, 15, 20 and 25), and between a set of randomly selected source and destination pairs. In each simulation, we use one of the following types of TCP: (1) TCP with a large CWL of 256 packets, (2) TCP with hop-based CWL, and (3) TCP with delay-based CWL. The performance comparison in Fig. 9(a) shows that TCP with hop-based CWL has the best performance, i.e. 8–16% more throughput than TCP with a large CWL. TCP with delay-based CWL comes in second and has 3–12% more throughput than the TCP with large CWL. That means the hop-based and delay-based BDP-UB bounds can effectively improve TCP performance in a dynamic mobile ad hoc network environment. Recall that in a chain topology, the performance of hop-based CWL and delay-based CWL are very close to each other (Fig. 8(b)); whereas here in a dynamic topology, hop-based CWL performs better than the delay-based CWL. This is because the measurement of packet transmission delay in a dynamic MANET environment is not as accurate as in

a static chain topology, while the hop count measurement is not subject to such noise. Therefore, we conclude that the hop-based CWL is a better solution for setting TCP's CWL in a dynamic MANET.

Two added benefits of using hop-based and delay-based CWL can also be observed in our simulations: (1) smaller end-to-end delay due to shorter router queues (in Fig. 9(b)); and (2) improved network efficiency due to fewer dropped packets (in Fig. 9(c)). These results further suggest that both the hop-based and delay-based CWLs are beneficial in improving end-to-end delay and network efficiency for TCP flows in MANET.

5.3. Impact of pacing

TCP *pacing* [16] is a simple technique to smooth TCP traffic in wireline networks. The basic idea is to let the TCP sender pace a window worth of packets over the current estimated round-trip time, hence avoids possible bursts of packets. In MANET, in addition to smoothing the traffic flow, TCP pacing has the potential benefit of mitigating the degree of channel contention at the MAC layer, which may help to improve a TCP flow's overall performance. Note that TCP pacing is *orthogonal* to its CWL setting: CWL decides the congestion window limit, while pacing smooths out the flow shape. Here we want to explore the potential benefit of TCP pacing with an appropriate CWL setting.

We use the same simulation network and traffic pattern as in Section 5.2. Pacing is implemented in TCP's packet sending routine, to evenly pace out the packets, according to the current congestion window size and the estimated round-trip time.

The simulation result in Fig. 10 shows that pacing has very limited impact on the overall performance of the TCP flows, i.e. Fig. 10 is almost identical to Fig. 9(a). This result coincides with the TCP pacing result obtained in wireline networks [16]. The possible reason for this is two-fold. First, as noted in [16], a paced TCP misses the sending opportunities and may incur higher loss rate compared to a nonpaced TCP. Second, in this set of simulations, TCP's congestion window is never opened up to a large size, limiting the possible benefits of pacing. To isolate this problem, we repeat TCP with pacing in a 15-hop chain topology, where TCP's window size can open up to eight packets. The result shows that pacing still has only marginal effect on TCP performance. Therefore, we conclude that pacing does not have significant impact on TCP performance in MANET.

6. Additional related work

Past research in improving TCP performance in MANET has spanned over different layers of the protocol stack: (1) transport (TCP) layer; (2) routing layer;

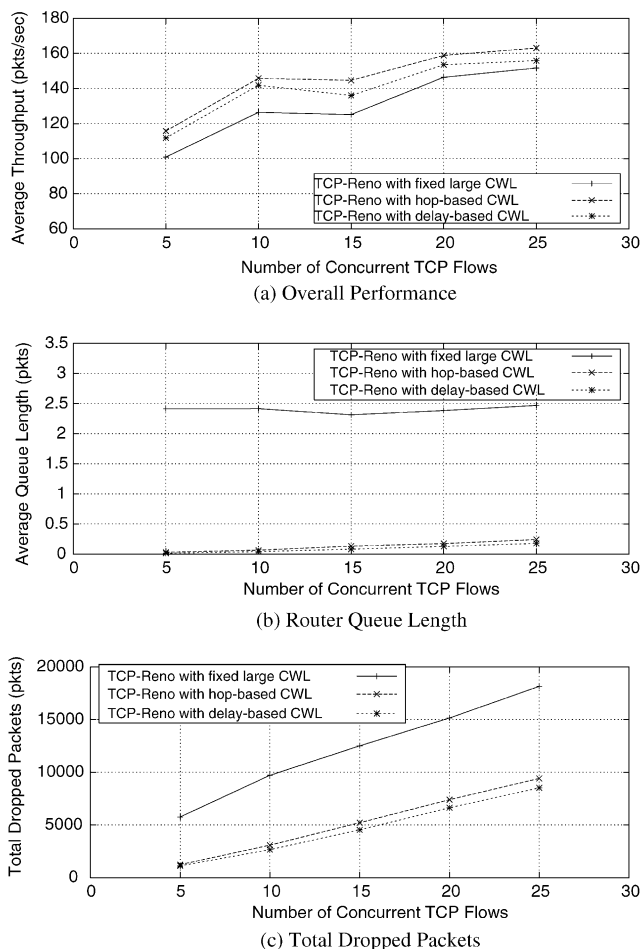


Fig. 9. Comparison of TCP with hop-based CWL and delay-based CWL in dynamic MANET.

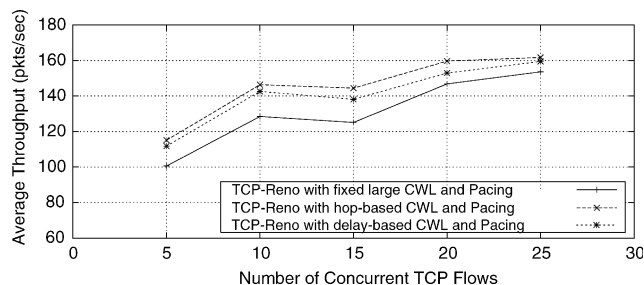


Fig. 10. Performance of TCP pacing with hop-based CWL and delay-based CWL.

and (3) MAC layer. Our study in this paper targets the TCP layer, and has solved TCP's CWL setting problem using a theoretically solid base of BDP-UB. We have also discussed its related work in Section 1. Below we discuss some other related work in improving TCP performance in MANET.

At the TCP layer, one area of work is to achieve accurate congestion detection by distinguishing the cause of packet loss between random wireless loss and congestion loss [17–19]. The basic idea behind these studies is to correlate the nature of packet loss to certain end-to-end measurements, such as inter-arrival time of packets [17], variation of RTT [18], or the joint statistics of inter-packet delay and short-term throughput [19]. If the cause of packet loss is identified as random loss, a graduated loss avoidance action, other than TCP's multiplicative window decrease, can be taken. Note that some of these studies focus on a hybrid wired and wireless network scenario; their results were not necessarily verified in a multi-hop ad hoc network environment.

At the routing layer, TCP performance can be improved by enhancing the cross-layer cooperation between TCP and the routing protocol, with the goal of differentiating the cause of packet loss between route failure and network congestion. To this end, two general approaches have been taken. The first approach requires the routing protocol to notify TCP when route failure occurs (TCP-ELFN [9], TCP-F [20]); the other approach infers route failure by two consecutive TCP re-transmission time-outs (fixed-RTO [10]). With the exception of TCP-F (which relies on explicit route re-establishment notification from the routing layer), all other schemes enter a *probing* state to periodically 'probe' the route until a new route is re-established, indicated by the reception of one or two TCP acknowledgment packet(s). In essence, they share the same principle with the TCP-Probing approach [21], where a 'probing device' is used to let the TCP sender 'sit out' a bad network state, such as network blackout during terminal handoff between base-stations. This cross-layer cooperation enables the TCP sender to recover quicker after the route failure, without exponentially backing off its RTO timer,

and hence avoids unnecessary prolonged periods of transmission blackouts.

At the MAC layer, one area of work is to study the MAC layer protocols and to compare their suitability to support TCP traffic in MANET. Tang et al. [22] compared different MAC layer mechanisms and their combined effects on TCP performance under different network topologies. They conclude that Carrier Sensing Multiple Access with Collision Avoidance (CSMA/CA) with Request-to-Send/Clear-to-Send (RTS/CTS) virtual sensing and link-layer ACK (acknowledge) provides a superior mixture of fairness and aggregate network throughput. Not surprisingly, these are the mechanisms adopted by IEEE 802.11 which has become the de facto standard in connecting mobile nodes in a MANET.

Another area of work at the MAC layer has focused on distributed fair scheduling algorithms, to improve medium access fairness between competing packet transmissions, and to mitigate channel capturing problem. A list of algorithms and their comparison can be found in Refs. [23,24] and references therein.

Finally, for a broader discussion of TCP performance issues in hybrid wired and wireless networks with mobility, interested readers are referred to the overview papers by Balakrishnan et al. [25] and by Tsoussidis and Matta [26].

7. Conclusion

This paper brings the well-known concept of BDP and its upper bound (BDP-UB) from wireline networks into the mobile ad hoc network (MANET) domain. We show that the most significant difference of computing BDP and BDP-UB in MANET is the tight coupling of bandwidth and delay over a wireless link. Based on this observation, we prove that the upper bound of BDP in MANET is tied to the number of round-trip hops (N) of a path. In other words, BDP-UB of a path is bounded by N (in Theorem 1). We then further tighten the BDP-UB bound by considering IEEE 802.11 MAC layer protocol's transmission interference in a neighborhood area, and TCP's DATA and ACK interference ('hop-based' bound in Corollary (1)), and the knowledge of per-hop transmission delays ('delay-based' bound in Corollary (2)). We verify these two bounds both through ns-2 simulations. Furthermore, our finding of the bounds for BDP-UB in MANET provides a systematic solution to the problem of how to properly set TCP's congestion window limit (CWL), which has been an open problem in past research. Simulation results show that both the hop-based and delay-based bounds improve TCP performance in a dynamic MANET, and that the hop-based bound is a better solution due to its effectiveness and simplicity. We also explore the potential benefit of

TCP pacing and find no significant impact on TCP performance.

Acknowledgements

This work was supported by the ONR MURI Grant N00014-00-1-0564 and the NSF EIA Grant 99-72884EQ. Any opinions, findings, and conclusions are those of the authors and do not necessarily reflect the views of the above agencies.

References

- [1] L.L. Peterson, B.S. Davie, *Computer Networks*, Morgan Kaufmann, San Francisco, CA, 2000.
- [2] W. Stevens, *TCP/IP Illustrated, The Protocols*, vol. 1, Addison-Wesley, Reading, MA, 1994.
- [3] S. Keshav, *An Engineering Approach to Computer Networking*, Addison-Wesley, Reading, MA, 1997.
- [4] Home page of IETF mobile ad-hoc networks (MANET) WG. [Online]. Available: <http://www.ietf.org/html.charters/manet-charter.html>
- [5] M. Gerla, K. Tang, R. Bagrodia, TCP performance in wireless multi-hop networks, in: Proceedings of the IEEE International Workshop on Mobile Computing Systems and Applications (WMCSA'99), New Orleans, Louisiana, USA, February 1999.
- [6] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, M. Gerla, The impact of multihop wireless channel on TCP throughput and loss, in: Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (InfoCom 2003), San Francisco, California, USA, March–April 2003.
- [7] Z. Fu, X. Meng, S. Lu, How bad TCP can perform in mobile ad hoc networks, in: Proceedings of the IEEE International Symposium on Computers and Communications (ISCC'02), Taormina, Italy, July 2002.
- [8] K. Xu, S. Bae, S. Lee, M. Gerla, TCP behavior across multihop wireless networks and the wired Internet, in: Proceedings of the ACM Workshop on Wireless Mobile Multimedia (WoWMoM'02), Atlanta, Georgia, USA, September 2002.
- [9] G. Holland, N. Vaidya, Analysis of TCP performance over mobile ad hoc networks, in: Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99), Seattle, Washington, USA, August 1999.
- [10] T. Dyer, R. Boppana, A comparison of TCP performance over three routing protocols for mobile ad hoc networks, in: Proceedings of the ACM Symposium of Mobile Ad Hoc Networking and Computing (MobiHoc 2001), Long Beach, California, USA, October 2001.
- [11] J. Li, C. Blake, D.S.J. DeCouto, H. Lee, R. Morris, Capacity of ad hoc wireless networks, in: Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2001), Rome, Italy, July 2001.
- [12] The network simulator ns-2. [Online]. Available: <http://www.isi.edu/nsnam/ns/>
- [13] D.B. Johnson, D.A. Maltz, *Dynamic source routing in ad hoc wireless networks*, in: T. Imielinski, H. Korth (Eds.), *Mobile Computing*, Kluwer, Dordrecht, 1996, (Chapter 5).
- [14] S.H. Shah, K. Chen, K. Nahrstedt, Dynamic bandwidth management for single-hop ad hoc wireless networks, in: Proceedings of the IEEE International Conference on Pervasive Computing and Communications (PerCom 2003), Dallas-Fort Worth, Texas, USA, March 2003.
- [15] J. Yoon, M. Liu, B. Nobel, Random waypoint considered harmful, in: Proceedings of the of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (InfoCom 2003), San Francisco, CA, USA, March–April 2003.
- [16] A. Aggarwal, S. Savage, T. Anderson, Understanding the performance of TCP pacing, in: Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (InfoCom 2000), Tel-Aviv, Israel, March 2000.
- [17] S. Biaz, N.H. Vaidya, Discriminating congestion losses from wireless losses using inter-arrival times at the receiver, in: Proceedings of the IEEE Symposium on Application-Specific Systems and Software Engineering Technology (ASSET'99), Richardson, Texas, USA, March 1999.
- [18] D. Barman, I. Matta, Effectiveness of loss labeling in improving TCP performance in wired/wireless networks, in: Proceedings of the IEEE International Conference on Network Protocols (ICNP 2000), Paris, France, November 2002.
- [19] 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 the IEEE International Conference on Network Protocols (ICNP 2000), Paris, France, November 2002.
- [20] K. Chandran, S. Raghunathan, S. Venkatesan, R. Prakash, A feedback based scheme for improving TCP performance in ad hoc wireless networks, in: Proceedings of the International Conference on Distributed Computing Systems (ICDCS'98), Amsterdam, The Netherlands, May 1998.
- [21] V. Tsaoussidis, A. Lahanas, Exploiting the adaptive properties of a probing device for TCP in heterogeneous networks, in: *Computer Communications*, 40(4), Elsevier Science, 2002.
- [22] K. Tang, M. Correa, M. Gerla, Effects of ad hoc MAC layer medium access mechanisms under TCP, *ACM/Kluwer Journal of Mobile Networks and Applications* 6 (4) (2001).
- [23] B. Bensaou, Y. Wang, C. Ko, Fair medium access in 802.11-based wireless ad hoc networks, in: Proceedings of the ACM Symposium of Mobile Ad Hoc Networking and Computing (MobiHoc 2000), Boston, Massachusetts, USA, August 2000.
- [24] H. Luo, P. Medvedev, J. Cheng, S. Lu, A self-coordinating approach to distributed fair queuing in ad hoc wireless networks, in: Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (InfoCom 2001), Anchorage, Alaska, USA, April 2001.
- [25] H. Balakrishnan, V.N. Padmanabhan, S. Seshan, R.H. Katz, A comparison, A comparison of mechanisms for improving TCP performance over wireless links, *IEEE/ACM Transactions on Networking* 5 (6) (1997).
- [26] V. Tsaoussidis, I. Matta, Open Issues on TCP for Mobile Computing, in: *Journal of Wireless Communications and Mobile Computing*, vol. 1(2), Wiley, New York, 2002.

Kai Chen received his BE degree in computer science from Tsinghua University, Beijing, China, in 1995, and MS degree in computer science from the University of Delaware, Newark, Delaware, USA, in 1998. Currently he is a PhD candidate in the department of computer science at the University of Illinois at Urbana-Champaign. From 1998 to 2000, he worked as a research programmer at the National Center for Supercomputing Applications (NCSA) at Urbana, Illinois, USA. His research interests include mobile ad hoc networks, transport layer issues in mobile networks, quality of service, incentive engineering, and pervasive computing.

Yuan Xue received her BS degree in 1998 from Department of Computer Science and Engineering, Harbin Institute of Technology, China, and her MS in 2002 from the Department of Computer Science, University of Illinois at Urbana-Champaign. Currently she is a PhD candidate in the Department of Computer Science at University of Illinois at Urbana-Champaign. Her research interests include wireless network, network-level and application-level Quality of Service provisioning and mobile computing.

Samarth Shah received his BE degree in Computer Science and Engineering from the University of Madras, India in 1998. He is currently a PhD candidate in the Department of Computer Science at the University of Illinois at Urbana-Champaign. His interests include Wireless Networks and Quality of Service (QoS).

Klara Nahrstedt is an associate professor at the University of Illinois at Urbana-Champaign, Computer Science Department. Her research interests are directed towards multimedia middleware systems, quality of service (QoS), QoS routing, QoS-aware resource management in distributed multimedia systems, and multimedia security. She is the coauthor of the widely used multimedia book 'Multimedia: Computing, Communications and Applications' published by Prentice Hall, the recipient of the Early NSF Career Award, the Junior Xerox Award, and the IEEE Communication Society Leonard Abraham Award for Research Achievements. She is the editor-in-chief of ACM/Springer Multimedia Systems Journal, and the Ralph and Catherine Fisher Associate Professor. Klara Nahrstedt received her BA in mathematics from Humboldt University, Berlin, in 1984, and MSc degree in numerical analysis from the same university in 1985. She was a research scientist in the Institute for Informatik in Berlin until 1990. In 1995 she received her PhD from the University of Pennsylvania in the department of Computer and Information Science.