Improving Transport Layer Performance in Multihop Ad Hoc Networks by Exploiting MAC Layer Information

Hongqiang Zhai, Xiang Chen, and Yuguang Fang

***Abstract*— The traditional TCP congestion control mechanism encounters a number of new problems and suffers a poor performance when the IEEE 802.11 MAC protocol is used in multihop ad hoc networks. Many of the problems result from medium contention at the MAC layer. In this paper, we first illustrate that severe medium contention and congestion are intimately coupled, and TCP's congestion control algorithm becomes too coarse in its granularity, causing throughput instability and excessively long delay. Further, we illustrate TCP's severe unfairness problem due to the medium contention and the tradeoff between aggregate throughput and fairness. Then, based on the novel use of channel busyness ratio, a more accurate metric to characterize the network utilization and congestion status, we propose a new wireless congestion control protocol (WCCP) to efficiently and fairly support the transport service in multihop ad hoc networks. In this protocol, each forwarding node along a traffic flow exercises the inter-node and intra-node fair resource allocation and determines the MAC layer feedback accordingly. The end-to-end feedback, which is ultimately determined by the bottleneck node along the flow, is carried back to the source to control its sending rate. Extensive simulations show that WCCP significantly outperforms traditional TCP in terms of channel utilization, delay, and fairness, and eliminates the starvation problem.**

***Index Terms*— Medium access control (MAC), congestion control, fairness, transmission control protocol (TCP), multihop ad hoc networks.**

I. Introduction

WIRELESS ad hoc networks have found many applications in battlefield, disaster rescue and conventions, where fixed communications infrastructures are not available and quick network configurations are needed. To provide reliable transport service over and hence fully exploit the potential of ad hoc networks, efficient congestion control is of paramount importance.

Unfortunately, the traditional TCP congestion control mechanism performs very poorly, as shown in recent studies ([4], [5], [7], [8], [10], [11], [20], [24] and reference therein). TCP congestion control has an implicit assumption, i.e., any packet loss is due to network congestion. However, this assumption is no longer valid in the ad hoc networks as packet losses may well be due to channel errors, medium contention, and route failures.

Several works have pointed out that greedy TCP can result in severe congestion in ad hoc networks and hence performance degradation. Link-RED [8] was proposed to mark or drop TCP packets according to observed packet collisions. Subsequently the TCP source will reduce congestion window size before it becomes excessively large. To avoid congestion, Chen et al. dynamically adjusted the congestion window limit according to path length of TCP flows [4]. In [19], a neighborhood RED scheme was proposed to alleviate TCP fairness problem by adjusting marking/dropping probability in light of observed channel information.

Meanwhile, to alleviate the adverse impact of mobility, several schemes were proposed, such as those in [3], [12], [15], [16]. The design philosophy is to distinguish route failures from topology changes and network congestion through explicit route failure notification. Other schemes like [6], [18], instead of using the network layer feedback, keep the TCP states unchanged when the source first detects out-of-order packets and retransmission timeout.

In this paper, we mainly focus on the problems arising from medium contention when the IEEE 802.11 MAC protocol is used in the multi-hop ad hoc networks. In Section II, we show that a rate based congestion control protocol is more appropriate than its window based counterpart in multihop ad hoc networks. We illustrate the close coupling between congestion and medium contention, which explains the instability of TCP. We also find that the optimum congestion window size of TCP may be less than one even in a very simple topology, say chain topology, in order to maximize the end-to-end throughput and minimize the end-to-end delay. Thus TCP tends to overshoot the network capacity and its granularity of sending rate adjustment is too coarse because the minimum increase in window size is the size of one packet upon each TCP acknowledgment or during each round trip time. Then we further show that medium contention also results in severe unfairness and starvation problems for TCP flows. Therefore, we conclude that congestion control, fairness and medium contention are all closely coupled in multihop ad hoc networks.

Manuscript received May 17, 2005; revised December 1, 2006; accepted January 28, 2007. The associate editor coordinating the review of this paper and approving it for publication was Q. Zhang. This work was supported in part by the National Science Foundation under grant DBI-0529012 and under Faculty Early Career Development Award ANI-0093241, and by the U.S. Office of Naval Research under Young Investigator Award N000140210464.

H. Zhai is with Philips Research North America, Briarcliff Manor, NY 10510 (e-mail: hong.zhai@philips.com).

X. Chen is with Motorola Labs, Schaumburg, IL 60196 (e-mail: a00131@motorola.com).

Y. Fang is with the Department of Electrical & Computer Engineering, University of Florida, Gainesville, Florida 32611-6130 (e-mail: fang@ece.ufl.edu).

Digital Object Identifier 10.1109/TWC.2007.05306.

Nevertheless, it is not an easy task to conduct accurate end-to-end rate control in the ad hoc environment, despite the fact that the explicit and precise congestion feedback for end-to-end control has been extensively studied in the Internet and ATM networks [1], [9]. This is because each node is in dire need of a robust and easily measured metric to adjust the feedback for each passing packet. While packet loss, queue length, and link utilization are good measures for these wired networks, they cannot be directly applied to ad hoc networks for two main reasons. First, the occurrence of packet loss and large queue length may indicate that severe congestion has already happened due to medium contention, thereby leaving no time for the network to react promptly. Second, unlike a wired link normally between two nodes, a wireless link is shared by all the neighboring nodes. Consequently, any change in the status of the wireless link is much harder to trace, which in turn renders accurate control extremely difficult. To overcome this difficulty, we propose to use a novel measure to reflect wireless link status. The channel busyness ratio, as shown in Section III-A, is a timely and accurate metric for the network utilization as well as congestion.

In Section III, we propose a new wireless congestion control protocol (WCCP) based on the channel busyness ratio. In this protocol, each forwarding node determines the inter-node and intra-node fair channel resource allocation and allocates the resource to the passing flows by monitoring and possibly overwriting the feedback field of the data packets according to its measured channel busyness ratio. The feedback is then carried back to the source by the destination, which copies it from the data packet to its corresponding acknowledgment. Finally, the source adjusts the sending rate accordingly. Clearly, the sending rate of each flow is determined by the channel utilization status at the bottleneck node. In this way, WCCP is able to approach the max-min fairness ([2]) in certain scenarios.

We compare WCCP with TCP through extensive simulations in Section IV. We observe that WCCP significantly outperforms TCP in terms of channel utilization, delay, and fairness. Especially, it solves the starvation problem suffered by TCP.

As a final remark, we note that WCCP is not meant to address the problems caused by mobility and it is more effective in static multihop ad hoc networks. However, one can combine some schemes proposed in [3], [6], [12], [15], [16], [18] and WCCP to alleviate the performance degradation due to mobility. This will be further explored in our future work. Conclusions are given in Section V.

II. Medium Contention and Its Impact

A. TCP Performance Degradation Due to Coupling of Congestion and Medium Contention

To illustrate the coupling of congestion and medium contention, we use ns 2.27 ([17]) to conduct a set of simulations over a 9-node chain topology as shown in Fig. 1(a). One or more TCP flows with 1000 bytes payload traverse from node 1 to node 9. The pre-computed shortest path is used, so there is no routing overhead. The channel bandwidth (channel transmission rate) is 2 Mbps. Simulations run for 300 seconds.

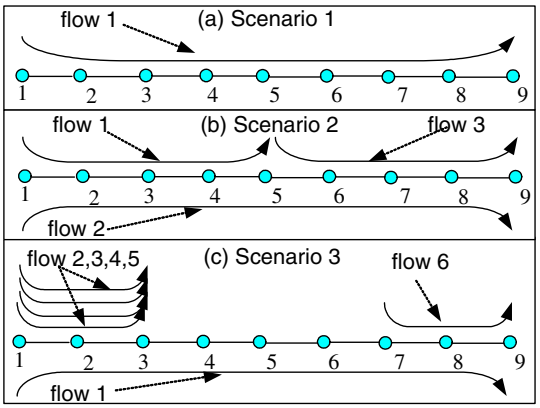


Fig. 1. 9-node chain topology with different traffic distribution

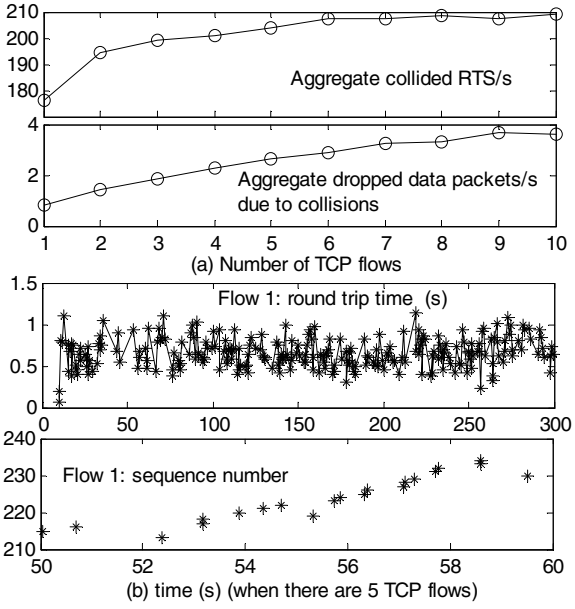


Fig. 2. Simulation results for 9-node chain topology

We can see from Fig. 2(a) that TCP traffic introduces a lot of collisions. Though there is a retransmission mechanism for RTS and DATA frames at the MAC layer ([13]), there are still many TCP packets dropped at a rate of $0.83 \sim 3.63$ pkts/s because of medium contentions. Note that no packet loss is observed for queue overflow.

Fig. 2(b) demonstrates that TCP traffic is unstable in the wireless multihop environment. The round trip time (RTT) oscillates dramatically, and so does the instantaneous throughput, which can be obtained by differentiating the number of delivered packets (sequence number) with respect to time.

All of these observations can be attributed to the greedy properties of TCP and the coupling of congestion and contention. TCP will continually increase the congestion window size until it detects a packet loss. When the sending rate of TCP sources surpass the channel capacity, packets start to cumulate along the path. When the neighboring nodes all have packets to transmit, they keep contending for the channel. Consequently, more collisions happen and hence the channel contention delay increases, slowing down the forwarding

rate and exacerbating congestion. Thus the congestion and collision form a positive feedback loop until there are some packets dropped due to continual collisions and such losses are detected by TCP sources from retransmission timeouts or delayed duplicate ACKs.

If dynamic routing schemes are used in multihop ad hoc networks, the situation will become worse. Since the MAC layer can not distinguish whether the losses are due to collision or unreachable next hops, it will report false link/route failures when packets are dropped due to collisions. Then, the routing layer will launch time-consuming route search or re-routing, thereby increasing end-to-end delay.

B. Optimal Congestion Window Size for TCP and Ideal Sending Rate

In this subsection we show that how the nature of medium contention in multihop ad hoc networks dictates the optimal sending rate per RTT and why the window based congestion control mechanism for TCP performs poorly.

Li et al. [14] have shown that, in chain topology like that in Fig. 1(a), the maximum channel utilization is achieved by scheduling the nodes four hops away to transmit simultaneously. And the optimal sending rate R_o from the source cannot be higher than that to make the above schedule feasible. This is because that higher sending rate will result in packet collisions and losses, leading to low throughput and long delay. At rate R_o, the packet is delivered to the destination in the shortest time without encountering much medium collision and long queueing delay. Also, RTT, denoted by RTT_o, is small. Assuming there are N TCP flows from node 1 to node 9, the optimal sending rate of each TCP flow and the number of packets sent by each TCP source per RTT_o are given by

$$R_{oeachtcp} = R_o/N, \tag{1}$$

$$K_{eachtcp} = RTT_o \times R_o/N \tag{2}$$

respectively.

According to the above optimal schedule, we can find out the optimal aggregate sending rate. Here again, we use simulation to illustrate R_o when the 802.11 MAC is used. CBR/UDP traffic with the same packet length as TCP DATA packets flows from node 1 to node 9. CBR/UDP traffic with the same packet length as TCP ACK packets flows in the reverse direction. The CBR traffics in both directions have the same packet sending rate (pkt/s). We gradually increase the sending rate until the DATA packet dropping ratio due to collision is larger than $0.1\ pkt/s$ in the 300 seconds simulation. Notice that further increasing the sending rate is followed by dramatic increase in collision and packet dropping ratio. The results are summarized in Table I, where the performance for five TCP flows is included for comparison.

The corresponding aggregate sending rate is about $24.3\ pkt/s$ given that the DATA packet dropping ratio due to collision is less than $0.1\ pkt/s$. And $RTT_o = 0.139s$ and $K_{eachtcp} = 0.676pkt/RTT$ when N = 5. Apparently, the more TCP flows there are, the smaller $K_{eachtcp}$ is. Since the optimal sending rate per RTT is less than one packet/RTT, we can see that window based congestion control protocols such as TCP tend to overshoot the network capacity as the minimum increase in window size is one packet. In other words, the granularity of window based congestion control mechanism is too coarse. In this sense, window based protocols are not appropriate for supporting stable and reliable transport service in multihop ad hoc networks. Therefore, to provide high throughput, short delay and stable performance with few packet collisions, we opt for an efficient rate-based congestion control algorithm detailed in Section III.

TABLE I
SIMULATION RESULTS FOR TCP AND UDP FLOWS

Traffic type	UDP (node 1 to 9)	5 TCP flows
Aggregate throughput (Kbps)	198	196
Average end-to-end delay (s)	0.0695	0.431
RTT(s)	0.139*	0.738
Dropped packets / s due to collision	0.0931	2.90
* the sum of average end-to-end delay of the two UDP flows		

C. Unfairness Problem Due to Medium Contention

Medium contention is also a major source for unfairness in two aspects. First, different flows may traverse through different geographical regions. They may encounter different level of contention due to the various number of contending nodes in each region, and get different allocation of the shared wireless channel. Second, unfairness problem exists for flows with various path lengths since the flow with longer path consumes more network resource and is likely to encounter more medium contention and more packet drops.

Starvation is a severe unfairness problem suffered by TCP flows in multihop ad hoc networks, which can be attributed to the medium contention. The hidden terminal and receiver blocking problems ([26], [27]) are common in multihop ad hoc networks. Together with the greediness of TCP flows, they contribute to flow starvation as well as packet collision. For example, as shown in Fig. 1(a), suppose that there are two TCP flows passing through the links 1 to 2 and 4 to 5 separately. Notice that carrier sensing range is normally a little larger than twice of transmission range ([17] [21] [22]). When node 4 is transmitting packets to node 5, node 2 is a blocked receiver of node 1 since node 2 senses the busy channel and cannot respond to node 1. As a result, node 1 keeps doubling its contention window and retransmitting the RTS packet until dropping it. After node 4 finishes the transmission, it resets its contention window and hence has higher priority than node 1 in capturing the channel. Furthermore, node 4 can initiate a new transmission to node 5 when node 1 is transmitting to node 2 since node 4 senses the idle channel. Thus the flow passing through the link 1 to 2 will be starved if there is a greedy flow passing through the link 4 to 5.

It is important to note that there is a tradeoff between fairness and aggregate throughput. It is known that spatial reuse of the channel can be achieved by scheduling simultaneous transmissions whose regions are not in conflict. However, as said above, different flows may experience contention of different degree. Achieving fairness among those flows requires allocating the channel to flows with heavy contention

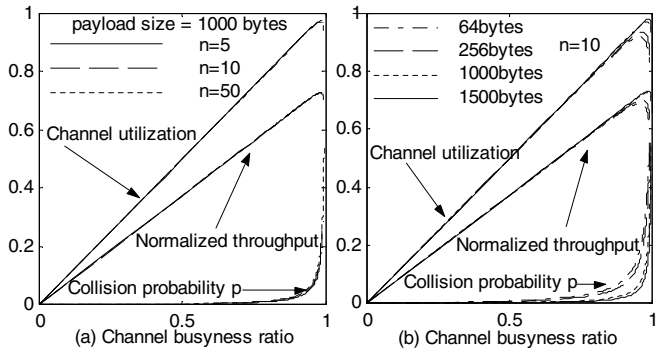


Fig. 3. The relationship between channel busyness ratio and other metrics

for a long time share, which correspondingly reduces the channel reuse and hence the aggregate throughput. In addition, while maximum throughput is the channel transmission rate for a one-hop flow, it reduces to one half, one third, and one fourth of the channel transmission rate for a two-, three-, and four-hop flow, respectively. Therefore, fair throughput allocation for flows with different path lengths in the same region has to be achieved at the expense of the aggregate throughput.

Fig. 1 shows a few more examples of unfairness. In Fig. 1(b), flow 2 traverses more hops and suffers more medium contention. Consequently, it has to drop more packets and suffers more serious throughput degradation than flow 1 and 3. In Fig. 1 (c), flow 6 suffers no hidden terminal and receiver blocking problems while flow 1 does and could be starved. Assigning channel resource to flow 1 will result in the decrease of aggregate throughput due to the shared channel resource. With perfect scheduling, the throughputs for the six flows are $(1/12, 1/12, 1/12, 1/12, 1/12, 1/3)$ of the channel bandwidth for max-min fair allocation ([2]) and $(0, 1/8, 1/8, 1/8, 1/8, 1/2)$ for maximizing the aggregate throughput and maintaining fairness among flow 2, 3, 4, and 5 at the same time. The aggregate throughputs for these two cases are $3/4$ and 1 of the channel bandwidth, respectively. Clearly, this demonstrates the tradeoff between the aggregate throughput and fairness. We will present the simulation results for these scenarios in Section IV and show WCCP approaches the max-min fairness in certain scenarios.

III. Wireless Congestion Control Protocol (WCCP)

As mentioned in the previous section, TCP's congestion control suffers from a coarse granularity when applied to the multihop ad hoc environment. To overcome this problem, we propose a rate based wireless congestion control protocol (WCCP). There are two components in WCCP. One is at the transport layer. It replaces the window adjusting algorithm of TCP with a rate control algorithm to regulate the sending rate. The other is between the networking layer and the MAC layer. It monitors and possibly modifies the feedback field in TCP data packets when it passes the outgoing packets from the networking layer to the MAC layer and the incoming packets in the reverse direction.

In this section, we first discuss how to characterize the channel status and measure the available bandwidth in the shared channel. Based on the estimate of the available bandwidth, the inter-node and intra-node resource allocation schemes are proposed to determine the available channel resource for each node and for each flow passing through that node and accordingly modify the MAC layer feedback. Then an end-to-end rate control scheme is proposed to carry the feedback from the bottleneck node to the source node which accordingly adjust the sending rate to make full and fair utilization of the channel resource at the bottleneck node without causing severe medium contention and packet collision.

A. Channel Busyness Ratio: a More Accurate Metric for Congestion and Available Bandwidth

In the rate-based congestion control algorithm, to calculate the ideal sending rate, the source is in dire need of a timely and easily measured metric which should satisfy two requirements. First, as mentioned in previous discussion, since MAC contention is tightly coupled with congestion, a candidate of congestion signal should reflect the condition of MAC contention and collision. Second, in order to fully utilize the shared channel without causing severe congestion and packet collision, the candidate should indicate the available bandwidth.

In our previous work [23], we have shown that the channel busyness ratio r_b, which is defined as the ratio of time intervals when the channel is busy due to successful transmission or collision to the total time, meets these two requirements. And the main results are shown in Fig. 3. In the figure, the channel utilization c_u indicates the ratio of the channel occupancy time for successful transmissions to the total time, the normalized throughput s indicates the achievable data rate for payload divided by the channel data rate and is proportional to c_u, and the collision probability p indicates the probability that each transmission encounters a collision.

Several important results can be observed from Fig. 3. Firstly, before channel utilization c_u reaches its peak, r_b is almost the same as c_u and hence can be used to represent the normalized throughput. Secondly, after r_b exceeds a threshold where c_u reaches its peak, small increase in r_b will cause p to increase very fast until saturated status is reached. This case is certainly undesirable since p, the queue size, and the queue waiting time will all become unacceptably large, as indicated in [23], [25]. Finally and most importantly, the above observations are almost independent of the total number of nodes in the neighborhood. This is a very nice feature since changes in the number of neighbors will not affect a node's perception of the channel utilization or network congestion, as long as it relies on the observed channel busyness ratio.

Since the channel busyness ratio r_b provides a good early signal of network congestion, we can feedback the observed r_b to the end-to-end control mechanism to control TCP sources and hence avoid overloading the network. To do so, we choose the threshold, denoted by th_b, for r_b to indicate the inception of congestion. Obviously, th_b should be chosen such that

$$r_b \approx c_u (r_b \leq th_b) \tag{3}$$

Since the performance of r_b is not sensitive to n, we can fix n and observe the effect of the payload size. Fig. 3(b) shows c_u, s, and p as a function of r_b, with different average payload size of DATA packets when $n = 10$. It can be observed that the smaller the average payload size is, the smaller th_b should be, usually, i.e., $90\% \sim 95\%$. Since the payload size of $1000 \sim 1500$ bytes is commonly used in ad hoc networks, we set th_b to 92% accordingly and leave 3% space to avoid entering saturation.

After choosing th_b, according to Equation (3), we can estimate the available bandwidth of each node, denoted by BW_a, as follows:

$$BW_a = \begin{cases} BW(th_b - r_b)\overline{data}/T_s & ,(r_b < th_b) \\ 0 & ,(r_b \geq th_b) \end{cases}, \quad (4)$$

where BW is the transmission rate in bits/s for the DATA packets, $\overline{data}$ is the average payload size in unit of channel occupancy time, and T_s is the average time of a successful transmission at the MAC layer. Therefore, as long as the channel busyness ratio does not exceed the threshold, the network works in non-saturated status and the available bandwidth could be used to accommodate more traffic without causing severe MAC contention. Note that the available bandwidth can be shared by all the nodes in the neighborhood including the observing node.

B. Measurement of Channel Busyness Ratio

The channel busyness ratio r_b is an easily measured metric at the location of each node under the current architecture of the IEEE 802.11 standard. Notice that the IEEE 802.11 is a CSMA-based MAC protocol, working on the physical and virtual carrier sensing mechanisms. There is already a function to determine whether the channel is busy or not, i.e., the channel is determined busy when the measuring node is transmitting, receiving, or its network allocation vector (NAV) ([13]) indicates the channel is busy, and is idle otherwise. The channel busyness ratio can be simply calculated as the ratio of the total lengths of busy periods to the total time during a time interval, which is the average control interval in WCCP as discussed in the next subsections.

C. Inter-node Resource Allocation

According to equation (4), each node could calculate the total available bandwidth for its neighborhood based on the measured channel busyness ratio in a period called *average control interval*, denoted by $\overline{T_c}$. The details on determining $\overline{T_c}$ will be given in Section III-E.

To determine the available bandwidth for each node, WCCP accommodates the channel resource ΔS for each node proportionally to its current traffic load S in $\overline{T_c}$. Notice the linear relationship between BW_a and BW in equation (4), we have

$$\Delta S = \frac{th_b - r_b}{r_b} \times S. \quad (5)$$

Because both the incoming traffic and outgoing traffic of each node consume the shared channel resource, S should include the total traffic (in bytes), i.e., the sum of the total incoming and outgoing traffic. In Fig. 1 (b), for example, there are three flows at node 5, and the total traffic $S = r_1 + r_3 + 2 \times r_2$, where $r_i (1 \leq i \leq 3)$ is the traffic of flow i.

Next, we elaborate more on Equation (5). There are two cases when we compare the observed r_b with th_b, i.e., $r_b < th_b$ and $r_b \geq th_b$. When $r_b < th_b$, ΔS is positive, meaning we increase the traffic. As shown in Fig. 3, in this case the collision probability is very small and all the traffic gets through, thus the total throughput is approximately equal to the total traffic rate. Since the available bandwidth is proportional to $th_b - r_b$ according to equation (4), we may increase S by such an amount that after the increase ΔS, S is proportional to th_b, which is the optimal channel utilization. Actually, equation (5) achieves our desired increase as it can be easily seen that

$$\frac{\Delta S + S}{th_b} = \frac{S}{r_b} \quad (6)$$

Therefore, r_b will approach th_b after one average control interval $\overline{T_c}$ when all the nodes in the neighborhood increase the total traffic rate according to equation (5).

When $r_b \geq th_b$, ΔS is negative, meaning we decrease the traffic. In this case, however, the linear relationship between the available bandwidth and r_b no longer exists, and the collision probability increases dramatically as the total traffic rate increases. In addition, when the node enters saturation, both collision probability and r_b amount to their maximum values and do not change as the traffic increases, although the total throughput decreases. It thus appears that ideally, WCCP needs to aggressively decrease the total traffic rate. However, since it is difficult to derive a simple relationship between the traffic rate and r_b when $r_b \geq th_b$, WCCP uses the same linear function as for the case $r_b < th_b$. This will not affect the performance of WCCP significantly as long as we guarantee the increase in the total traffic rate is appropriate as suggested by th_b and the choice of th_b is a little conservative as discussed in the previous section. Indeed, this brings two advantages. First, as the increase and decrease use the same law, it is simple to implement at each node. Second, opting out of aggressive decrease helps achieve smaller oscillation in channel utilization.

D. Intra-node Resource Allocation

After calculating ΔS, the change in the total traffic or the aggregate feedback at each node, WCCP needs to apportion it to individual flows traversing that node in order to achieve both efficiency and fairness.

WCCP relies on an *Additive-Increase Multiplicative-Decrease (AIMD)* policy to converge to efficiency and fairness: If $\Delta S > 0$, all flows increase the same amount of throughput. And if $\Delta S < 0$, each flow decreases the throughput proportionally to its current throughput.

Before determining the feedback when $\Delta S >= 0$, WCCP needs to estimate the number of flows passing through the considered node. Again, since the channel is shared by both incoming and outgoing traffic, the number of flows J used by WCCP should be different from the real number of flows. For those flows that either originate or terminate at the node, the node counts each as one flow, whereas for those flows only passing through the node, the node counts each as two flows,

i.e., one in and one out. This is because that flows passing through the node occupy twice channel resource of that for flows originating or terminating at the node. For instance, in Fig. 1 (b), $J = 4$ for node 2 to 8, while $J = 2$ for node 1 and 9. Let r_{pk} denote the packet sending rate (pkt/s) of the flow which the kth observed packet during the period $\overline{T_c}$ at node i belongs to. Since $r_{pk}\overline{T_c}$ is equal to the number of packets which are observed from the corresponding flow at node i during $\overline{T_c}$, we can convert the summation over each flow to the summation over each packet by the fact that, for all the packets of each flow, the summation of $\frac{1}{r_{pk}\overline{T_c}}$ is equal to 1. Thus J can be calculated at node i as

$$J = \sum_{k=1}^{K} \frac{1}{r_{pk}\overline{T_c}} \tag{7}$$

where K is the total number of packets seen by node i in $\overline{T_c}$. Notice that packets originating or terminating at node i, e.g. packets belonging to flow 1 and 3 at node 5, are calculated once in the summation, and packets passing through node i, e.g. packets belonging to flow 2 at node 5, are calculated twice. Thus each node only needs to do the summation for each received and transmitted packet.

Therefore, if $\Delta S >= 0$, the increasing amount of traffic rate for each flow C_p, and per packet feedback pf_k will be

$$C_p = \frac{\Delta S}{\overline{T_c}J} \tag{8}$$

$$pf_k = \frac{C_p}{r_{pk}\overline{T_c}}, \tag{9}$$

If $\Delta S < 0$, WCCP should decrease each flow's throughput proportionally to its current throughput. Also notice that the per packet feedback is inversely proportional to the expected number of packets seen by the node in $\overline{T_c}$. Thus

$$pf_k = C_n \frac{r_{pk}}{r_{pk} \times \overline{T_c}} = \frac{C_n}{\overline{T_c}}, \tag{10}$$

where C_n is a constant and

$$\sum_{k=1}^{K} pf_k = \frac{\Delta S}{\overline{T_c}}, \tag{11}$$

$$C_n = \frac{\Delta S}{K} \tag{12}$$

WCCP aims to make full use of the channel resource while not introducing severe medium contention, i.e., r_b should be as close to th_b as possible but never exceed th_b too much. Therefore, when the aggregate feedback of previously passing packets is equal to ΔS, the node sets local feedback value as zero until the next control interval starts. With this mechanism in place, the channel busyness ratio r_b should be around th_b at the bottleneck nodes and be smaller at other nodes.

However, it is hard to converge to a fair resource allocation since the adjustment by the multiplicative-decrease law is limited if r_b is well controlled and always close to th_b. Thus WCCP manages to transfer resource from high throughput flows to low throughput flows by employing both increase law and decrease law when $\mid \Delta S \mid < \alpha(\Delta S + S)$. Let ΔS^+ and ΔS^- denote the increased and decreased traffic amount, respectively. Also, let pf_k^+ and pf_k^- denote the positive and negative feedback calculated by the increase law with ΔS^+ and the decrease law with ΔS^-, respectively. Specifically,

$$\begin{aligned} &\Delta S = \Delta S^+ + \Delta S^-, pf_k = pf_k^+ + pf_k^- \\ &\text{If } 0 < \Delta S < \alpha(S + \Delta S), \quad \Delta S^+ = \alpha(S + \Delta S) \\ &\text{If } 0 > \Delta S > -\alpha(S + \Delta S), \ \Delta S^- = -\alpha(S + \Delta S) \end{aligned} \tag{13}$$

where the adjustment by the decrease law is about $\alpha(S+\Delta S)$ in each control interval when r_b is around th_b. We set $\alpha = 10\%$ as a tradeoff between the convergence speed of fairness and throughput.

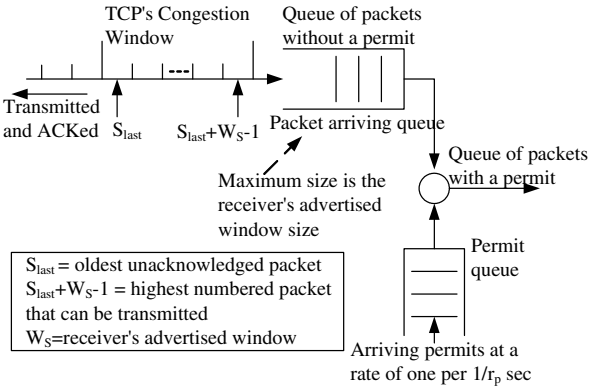


Fig. 4. Rate control mechanism

E. End-to-end rate-based congestion control scheme

The rate control mechanism of WCCP is illustrated in Fig. 4. A leaky bucket (permit queue) is attached to the transport layer to control the sending rate of a WCCP sender. The permit arrival rate r_p of the leaky bucket is dynamically adjusted according to the explicit feedback fb carried in the returned ACK whenever a new ACK arrives (henceforth, ACKs refer to the transport layer acknowledgments). Namely,

$$r_p = r_p + fb. \tag{14}$$

where the setting of fb will be given below.

To enable this feedback mechanism, each WCCP packet carries a congestion header including three fields, i.e., r_p, T_c, and fb, which is used to communicate a flow's state to the intermediate nodes and the feedback from the intermediate nodes to the source. The field r_p is the sender's current permit arrival rate, and the field T_c is the sender's currently used control interval. They are filled in by the sender and never modified in transit. The last field, fb, is initiated by the sender and all the intermediate nodes along the path may modify it to directly control the packet sending rate of the source.

The WCCP sender maintains an estimate of the smoothed round trip time $srtt$ and calculates the control interval T_c as

$$T_c = \max(srtt, \beta/r_p). \tag{15}$$

When r_p is large, i.e., $r_p > \beta/srtt$, $T_c = srtt$. Otherwise, this period equals β/r_p. The value of the control interval thus ensures that, on average, there are at least β data packets being transmitted in this period. If the period is too long, the adjustment of the sending rate is sluggish to respond to

the load change along the path. If the period is too short, the estimation of the feedback over short intervals at the nodes along the path will lead to erroneous estimates, and sometimes there may be no feedback received in one control interval. The choice of β is the tradeoff between these two considerations. In our study, we choose $\beta = 5$.

Initially, when the WCCP sender sends out the first packet of a flow, $r_p = 0$, and $T_c = 0$, indicating to the intermediate nodes that the sender does not yet have a valid estimate of the smoothed round trip time $srtt$. The sender also initializes the fb field to such that if bandwidth is available, this initialization allows the sender to reach the desired rate after one T_c. When the first ACK returns, the sender sets $r_p = 1/rtt$, calculates T_c, and sends out the second data packet. Thereafter, a WCCP sender sends out a data packet only when a permit is available.

All the nodes along the flow's path, including the WCCP sender and receiver, keep monitoring the channel busyness ratio r_b, maintain a per-node estimation-control timer that is set to the most recent estimate of average control interval $\overline{T_c}$, and calculate the local per packet feedback pf_k according to the rules specified in III-C and III-D. With the same argument used in deriving Equation (7), we have

$$\begin{aligned}\sum_j \overline{T_c} &= \sum_j \left(\sum_{k:\ packet\ k\ belonging\ to\ the\ jth\ flow} \frac{1}{r_{pk}} \right) \\ &= \sum_k \frac{1}{r_{pk}} \\ \sum_j T_{cj}\overline{T_c} &= \sum_j \left(\sum_{k:\ packet\ k\ belonging\ to\ the\ jth\ flow} \frac{T_{ck}}{r_{pk}} \right) \\ &= \sum_k \frac{T_{ck}}{r_{pk}}\end{aligned} \tag{16}$$

where j is the index for each flow, k is the index for each packet observed in $\overline{T_c}$, T_{cj} is the packet sending rate of the jth flow and T_{ck} is the packet sending rate of the kth observed packet at the considered node during $\overline{T_c}$. Thus $\overline{T_c}$ can be updated by $\overline{T_{c,new}}$ at the end of each $\overline{T_c}$ by the following equation:

$$\overline{T_{c,new}} = \sum_j T_{cj} \times \overline{T_c} / \sum_j \overline{T_c} = \sum_k \frac{T_{ck}}{r_{pk}} / \sum_k \frac{1}{r_{pk}} \tag{17}$$

If $pf_k < fb$, the node will set fb field in the congestion header with the value of pf_k. Ultimately, the packet will contain the feedback from the bottleneck node along the path. When the feedback reaches the WCCP receiver, it is returned to the sender in an ACK packet. Notice that a WCCP receiver is similar to a TCP receiver except that when acknowledging a packet, it copies the congestion header from the data packet to its ACK.

As for the overhead, WCCP does not require each node to keep per-flow state information and hence can scale well to any number of flows. Moreover, the feedback calculation at each intermediate node is quite simple, only requiring a few CPU cycles for each packet.

Retransmission timer RTO will expire when there is a packet loss. Note that in ad hoc networks, queue overflow rarely happens for TCP flows [8]. And packet losses mainly result from the failed transmission attempts at the MAC layer due to contention, collision, wireless channel error, or route failures due to mobility. Subsequently, the link breakage will be reported to the routing protocol, which may further drop subsequent packets. Notice in this case, the original route is broken, thus the timeout signals not only the packet loss, but also the route breakage. To avoid long periods of pausing and hence waste of channel capacity, it is wise for WCCP to send out a probe message or just retransmit the lost packet in periodic intervals to detect whether a new route is established.

Therefore, the response of WCCP to timeout is the following. For the first timeout, the WCCP sender retransmits the corresponding packet, double the retransmission timer, and reset r_p to $1/RTO$, where RTO denotes the retransmission timeout. Note that the retransmitted packets have higher priority than normal packets, in other words, the retransmitted packets will be transmitted when the next permit arrives, no matter whether there are any other packets in the window. For the subsequent back-to-back timeouts before a new acknowledgments arrives, WCCP does not double its retransmission timer again, nor does it reset r_p. It also records the time when the retransmission timer expires to differentially treat the feedback information carried by the ACKs that arrive after the timeout and route repair. The feedback in those ACKs that acknowledge the packets that are sent prior to the timeout is simply ignored, because it is very likely the feedback was calculated before the route failure and hence becomes outdated. By contrast, the feedback in those ACKs, acknowledging the packets sent later than the timeout, is qualified to be used to adjust the permit arrival rate.

IV. Performance Evaluation

In this section, we demonstrate through extensive simulations that WCCP outperforms TCP in the multihop ad hoc networks. In contrast to TCP, the new protocol dampens the oscillations of channel utilization, quickly converges to high utilization, short round trip time, and fair bandwidth allocation.

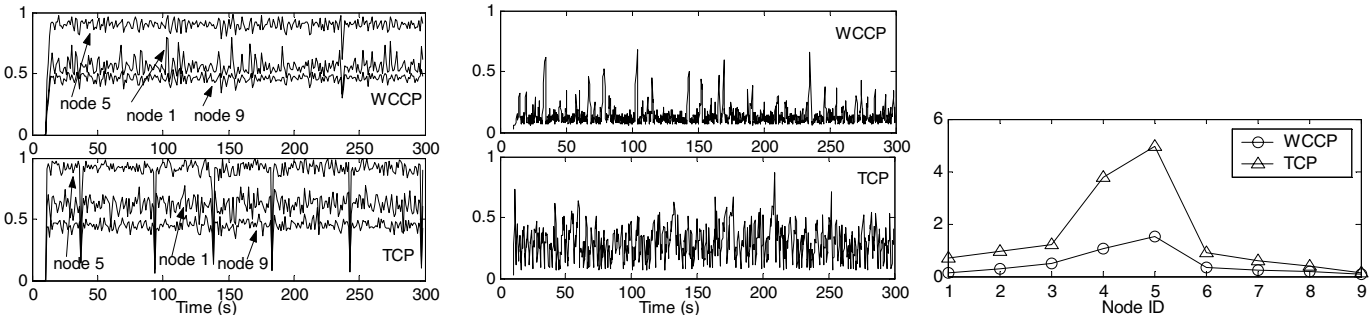
We use network simulator ns 2.27 to conduct the simulations. The transmission range is about 250m and the sensing range is about 550m. We set the channel bandwidth as 2Mbps and use 1000bytes as the payload size of each DATA packet.

In the simulations, we first consider the chain topology in Fig. 1 where nodes are separated by 200m, which is simple and allows us to clearly demonstrate the advantages of WCCP over TCP. Then, we consider a random network topology with a large number of flows, in an effort to model a more realistic network environment. The pre-computed shortest paths are used unless otherwise indicated.

A. Chain Topology

Channel utilization and packet collision: In the scenario of Fig. 1(a), as compared with TCP, WCCP greatly reduces the number of dropped packets from 0.834*pkt/s* to 0.120*pkt/s* by 86% and shorten end-to-end delay from 0.1844*s* to 0.0757*s* by 59%. The throughput of WCCP is not sacrificed and even improved from 194.6*Kbps* to 209.9*Kbps* by 8%.

In Fig. 5(a), the channel busyness ratio is presented. Each point in the curves is an average value during each second. It can be observed that WCCP converges to high link utilization and stabilizes in a narrow range, while TCP frequently oscillates in a large range. In fact, the stable and high channel



(a) Channel busyness ratio (b) Round trip time (s) (c) Average queue length

Fig. 5. Simulation results for the 9-node chain topology with one flow in the scenario Fig. 1(a)

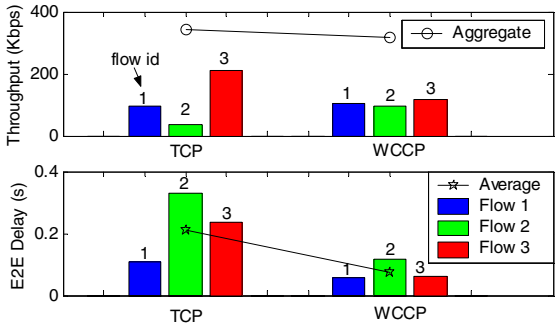


Fig. 6. Performance of scenario Fig.1(b)

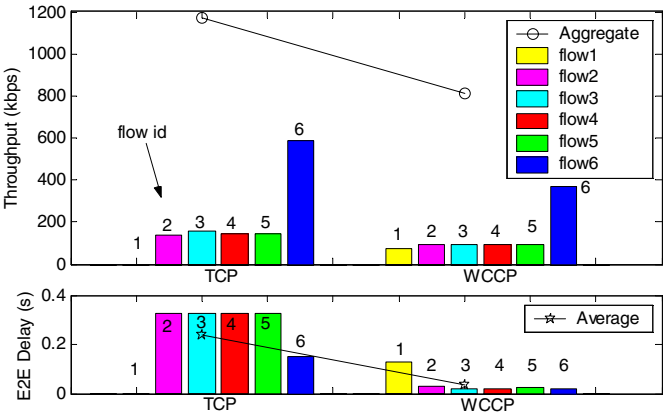


Fig. 7. Performance of scenario Fig.1(c)

utilization results in the improvement of throughput. We also observe that different nodes see different channel busyness ratio. Since node 5 is in the middle of the chain and thus encounters the heaviest collisions, its channel busyness ratio is the largest. On the other hand, compared to node 1, node 9, as a destination, does not transmit any DATA packets, so it observes the smallest channel busyness ratio.

Fig. 5(b) demonstrates that WCCP has a much smaller round trip time, rtt, than TCP. For WCCP, the average value of rtt is $0.1228s$, as opposed to $0.2646s$ for TCP. Fig. 5(c) shows that WCCP maintains a much smaller queue size at all the nodes than TCP. In addition, as pointed out earlier, a large queue size keeps node busy with contending the channel, which increases contention and causes packets to be dropped. Thus, a small queue size is desirable. This also explains why TCP has a much larger packet dropping rate (in pkts/s) than WCCP.

Fairness: In this simulation, we illustrate how WCCP addresses the unfairness problems illustrated in Section II-C. The simulation uses the scenarios in Fig. 1(b) and (c).

In Fig. 6, we observe that TCP completely fails to guarantee fairness for the flows. Especially, flow 2 takes the smallest share and flow 3 takes the largest share in terms of throughput. To simplify the explanation, we only consider the forward path for data packets, since the transmission of data packets is much longer than that of short ACKs. In the 9-node chain, node $i+3$ $(1 \leq i \leq 6)$ is the hidden terminal of node i, because the former can not sense the transmission from the latter but will interfere with the latter's intended receiver. Obviously, these three flows have different numbers of hidden terminals. Along the path of flow 3, only node 8 is a hidden terminal of node 5, while the other two flows, especially flow 2, suffer severe interference due to multiple hidden terminals. Accordingly, those flows have different throughput as shown in the above, since TCP is unable to ensure fairness.

By contrast, WCCP is able to allocate fair throughput to each flow. The reason is that, by monitoring the channel busyness ratio and each flow's traffic, WCCP can accurately calculate the available bandwidth of the channel and fairly assign it to each flow. Also, since WCCP controls each flow's incoming traffic and hence the channel utilization, it successfully reduces the MAC collision. Therefore, we also discover flow 2 has less dropped packets than in the case of TCP.

We also simulate the scenario in Fig. 1(c). Fig. 7 demonstrates that TCP favors short flows, especially the one or two-hop flows, and penalizes long flows. For the one or two-hop flows, since each node along the path can sense other node's transmission, there is no hidden terminal within the path. If there is no other competing one or two-hop flow in the neighborhood, they turn out to seize all the bandwidth and obtain high throughput. Flow 6 is such a two-hop flow and achieves the maximum throughput as if there were no other flows in the neighborhood. As a victim, flow 1 encounters severe contention from flow 6 and gains no throughput at all, although there is a pre-computed shortest route available. Other four two-hop flows compete with each other along the

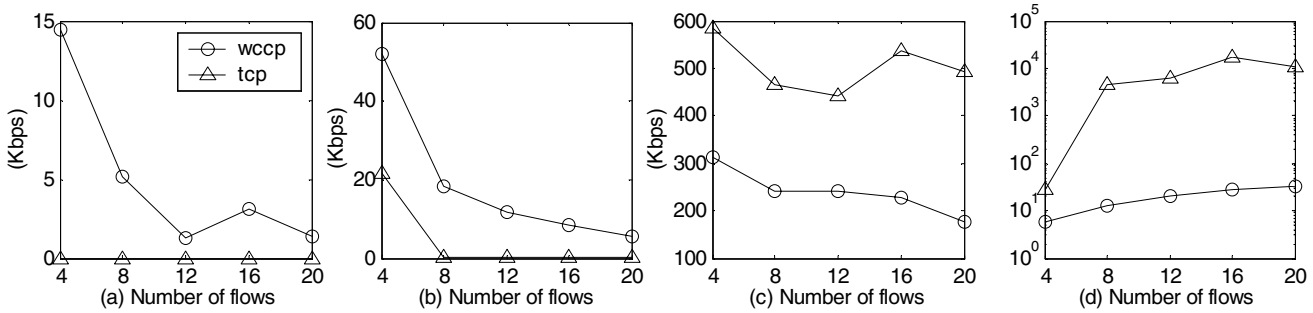


Fig. 8. Simulation results for random topology with precomputed paths: (a) minimum flow throughput in 20 runs, (b) minimum flow throughput averaged over 20 runs, (c) maximum flow throughput averaged over 20 runs, (d) ratio of averaged maximum flow throughput to averaged minimum flow throughput.

same path and approximately fairly share the channel with a little variation, as seen from their throughput.

With WCCP, we see that the starving problem for long flows is resolved. Flow 1 achieves almost the same throughput as flow 2-5, which share the same bottleneck, namely, the node with the maximum number of flows. Also, flow 6 takes all the channel capacity except flow 1's share. Therefore, WCCP approaches the max-min fairness for this scenario as discussed in Section II-C.

Tradeoff between Throughput and Fairness: The difference in the aggregate throughput for TCP and WCCP shown in Fig. 7 confirms that there is a tradeoff between throughput and fairness: fairness is improved at the expense of the aggregate throughput when the one or two-hop flows and the flows with longer flows coexist in the network. Since long flows consume more resource than short flows do when transmitting the same amount of traffic, if we grant all the flows the same throughput, some resource has to be taken from short flows to supply long flows. Thus, short flows will suffer throughput loss. Furthermore, long flows mean more hidden terminals and more MAC collision, and hence incur more nodes into the MAC contention. An additional amount of resource is thus consumed by the coordination of the channel access. In this scenario, max-min fairness is approached with a sacrifice of $1/4$ of aggregate throughput as discussed in Section II-C.

End-to-End Delay: All the above simulation results demonstrate that WCCP always achieve significantly shorter end-to-end delay than TCP does. We also observe that in WCCP, the end-to-end delay is proportional to the flow length. This illustrates that WCCP maintains a very small queue size at each node and greatly alleviates the MAC contention. As a result, the queueing delay and the delay caused by channel contention is very small, compared with those of TCP.

B. Random Topology

In this simulation, a random network topology is used. 50 nodes are randomly deployed in a 300m by 1500m field. The results are averaged over 20 runs.

Fig. 8 shows there are always some TCP flows having been starved while all WCCP flows can obtain a certain amount of throughput. The ratio of average maximum flow throughput to average minimum flow throughput is decreased by up to 1000 times. Clearly, WCCP completely eliminates the starvation problem. Furthermore, Fig.9(a) shows that WCCP improves

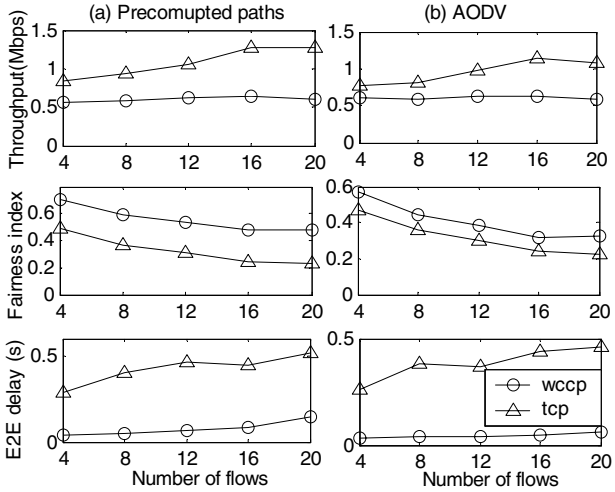


Fig. 9. Simulation results averaged over 20 runs in the random topology

the Jain's fairness index by about 0.1 at a price of $20\% \sim 45\%$ drop in aggregate throughput, and the end-to-end delay is decreased by $8 \sim 10$ times. Similar results are also observed if the on demand routing protocol AODV is used, as shown in Fig. 9(b).

V. Conclusions

Congestion control is critical to reliable transport service in wireless multihop ad hoc networks. Unfortunately, traditional TCP suffers severe performance degradation and unfairness. Realizing that the main reason is the poor interaction between traditional TCP and the MAC layer, we propose a systematic solution named Wireless Congestion Control Protocol (WCCP) to address this problem in both layers. WCCP uses channel busyness ratio to allocate the shared resource and accordingly adjusts the sender's rate so that the channel capacity can be fully utilized and fairness is improved. We evaluate WCCP in comparison with TCP in various scenarios. The results show that our scheme outperforms traditional TCP in terms of channel utilization, end-to-end delay, and fairness, and solves the starvation problem of TCP flows.

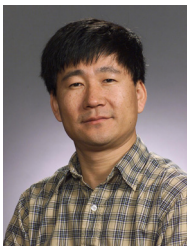
References

[1] Y. Afek, Y. Mansour, and Z. Ostfeld, "Phantom: a simple and effective flow control scheme," *ACM SIGCOMM*, 1996.

[2] D. Bertsekas and R. Gallager, *Data Networks*. Prentice-Hall, 1992.
[3] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash, "A feedback-based scheme for improving TCP performance in ad hoc wireless networks," *IEEE Personal Commun.*, vol. 8, no. 1, pp. 34–39, Feb. 2001.
[4] K. Chen, Y. Xue, and K. Nahrstedt, "On setting TCP's congestion window limit in mobile ad hoc networks," in *Proc. IEEE ICC 2003*.
[5] X. Chen, H. Zhai, J. Wang, and Y. Fang, "TCP performance over mobile ad hoc networks," *Canadian J. Electric. Comput. Engin.*, vol. 29, no. 1/2, pp. 129-134, Jan./April 2004.
[6] T. D. Dyer and R. V. Boppana, "A comparison of TCP performance over three routing protocols for mobile ad hoc networks," in *Proc. ACM Mobihoc 2001*.
[7] Z. Fu, X. Meng, and S. Lu, "How bad TCP can perform in mobile ad-hoc networks," in *Proc. IEEE Symposium on Computers and Communications 2002*.
[8] 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*.
[9] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," in *Proc. ACM SIGCOMM 2002*.
[10] M. Gerla, R. Bagrodia, L. Zhang, K. Tang, and L. Wang, "TCP over wireless multihop protocols: simulation and experiments," in *Proc. IEEE ICC 1999*.
[11] M. Gerla, K. Tang, and R. Bagrodia, "TCP performance in wireless multihop networks," in *Proc. IEEE WMCSA 1999*.
[12] G. Holland and N. H. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *Proc. ACM MOBICOM 1999*.
[13] IEEE standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, ISO/IEC 8802-11: 1999(E), Aug. 1999
[14] J. Li, C. Blake, D. S. J. De Couto, H. I. Lee, and R. Morris, "Capacity of ad hoc wireless network," in *Proc. ACM MobiCom 2001*.
[15] J. Liu and S. Singh, "ATCP: TCP for mobile ad hoc networks," *IEEE J. Sel. Areas Commun.*, vol. 19, pp. 1300–1315, July 2001.
[16] J. P. Monks, P. Sinha, and V. Bharghavan, "Limitations of TCP-ELFN for ad hoc networks," in *Proc. MOMUC 2000*.
[17] The network simulator ns-2: http://www.isi.edu/nsnam/ns.
[18] F. Wang and Y. Zhang, "Improving TCP performance over mobile ad-hoc networks with out-of-order detection and response," in *Proc. ACM MobiHoc 2002*.
[19] K. Xu, M. Gerla, L. Qi, and Y. Shu, "Enhancing TCP fairness in ad hoc wireless networks using neighborhood RED," in *Proc. ACM MobiCom 2003*.
[20] S. Xu and T. Safadawi, "Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks? *IEEE Commun. Mag.*, vol. 39, no. 6, pp. 130–137, June 2001.
[21] X. Yang and N. Vaidya, "On physical carrier sensing in wireless ad hoc networks," in *Proc. IEEE INFOCOM 2005*.
[22] H. Zhai and Y. Fang, "Physical carrier sensing and spatial reuse in multirate and multihop wireless ad hoc networks," in *Proc. IEEE INFOCOM 2006*.
[23] H. Zhai, X. Chen, and Y. Fang, "How well can the IEEE 802.11 wireless LAN support quality of service? *IEEE Trans. Wireless Commun.*, vol. 4, no. 6, pp. 3084–3094, Nov. 2005.
[24] H. Zhai, X. Chen, and Y. Fang, "Rate-based transport control for mobile ad hoc networks," in *Proc. IEEE WCNC 2005*.
[25] H. Zhai, Y. Kwon, and Y. Fang, "Performance analysis of IEEE 802.11 MAC protocols in wireless LANs," *Wireless Commun. Mobile Comput.*, vol. 4, no. 8, pp. 917–931, Dec. 2004.
[26] H. Zhai, J. Wang, and Y. Fang, "DUCHA: a dual-channel MAC protocol for mobile ad hoc networks," *IEEE Trans. Wireless Commun.*, vol. 5, no. 11, pp. 3224–3233, Nov. 2006.
[27] H. Zhai, J. Wang, X. Chen, and Y. Fang, "Medium access control in mobile ad hoc networks: challenges and solutions," *Wireless Commun. Mobile Comput.*, vol. 6, no. 2, pp. 151–170, March 2006.

Hongqiang Zhai (S'03-M'06) received the Ph.D. degree in Electrical and Computer Engineering from University of Florida in August 2006 and the B.E. and M.E. degrees in Electrical Engineering from Tsinghua University, Beijing China, in July 1999 and January 2002, respectively. He is now a senior member of research staff in Wireless Communications and Networking Department of Philips Research North America. He is a recipient of the Best Paper Award at the 14th IEEE International Conference on Network Protocols (ICNP 2006). His research interests include performance analysis, medium access control, and cross-layer design in wireless networks. He is a member of the ACM and the IEEE.

Xiang Chen (S'03-M'05) received his Ph.D. degree in electrical and computer engineering from the University of Florida in 2005, and received his M.E. and B.E. degrees in electrical engineering from Shanghai Jiao Tong University, Shanghai, China, in 2000 and 1997, respectively. He is now a senior staff research engineer with Motorola Labs. His research interests include system coexistence, resource management, medium access control, and QoS in wireless networks. He is a member of Tau Beta Pi.



Yuguang "Michael" Fang (S'92-M'94-S'96-M'97-SM'99) received a Ph.D. degree in Systems, Control and Industrial Engineering from Case Western Reserve University in January 1994 and a Ph.D degree in Electrical Engineering from Boston University in May 1997.

He held a post-doctoral position in Department of Electrical and Computer Engineering at Boston University from June 1994 to August 1995. From June 1997 to July 1998, he was a Visiting Assistant Professor in Department of Electrical Engineering at the University of Texas at Dallas. From July 1998 to May 2000, he was an Assistant Professor in the Department of Electrical and Computer Engineering at New Jersey Institute of Technology. In May 2000, he joined the Department of Electrical and Computer Engineering at University of Florida, Gainesville, Florida, where he got early promotion to Associate Professor with tenure in August 2003 and to Full Professor in August 2005. He holds a University of Florida Research Foundation (UFRF) Professorship from 2006 to 2009. His research interests span many areas including wireless networks, mobile computing, mobile communications, wireless security, automatic control, and neural networks. He has published over 100 papers in refereed professional journals and over 100 papers in refereed professional conferences. He received the National Science Foundation Faculty Early Career Award in 2001 and the Office of Naval Research Young Investigator Award in 2002. He is the recipient of the Best Paper Award in IEEE International Conference on Network Protocols (ICNP) in 2006 and the recipient of the IEEE TCGN Best Paper Award in the IEEE High-Speed Networks Symposium, IEEE Globecom in 2002.

Dr. Fang has actively engaged in many professional activities. He is an Editor for several journals including *IEEE Transactions on Communications*, *IEEE Transactions on Wireless Communications, IEEE Transactions on Mobile Computing*, *ACM Wireless Networks* and *Journal of Computer Science and Technology*, and a Technical Editor for *IEEE Wireless Communications Magazine*. He was also an Editor for *IEEE Journal on Selected Areas in Communications: Wireless Communications Series*, an Area Editor for *ACM Mobile Computing* and *Communications Review*, an Editor for *Wireless Communications and Mobile Computing*, and Feature Editor for Scanning the Literature in *IEEE Personal Communications*. He also served on the Technical Program Committee in many professional conferences such as ACM MobiCom'02 (Committee Co-Chair for Student Travel Award), MobiCom'01, IEEE INFOCOM'07, INFOCOM'06, INFOCOM'05 (Vice-Chair for Technical Program Committee), INFOCOM'04, INFOCOM'03, INFOCOM'00, INFOCOM'98, IEEE WCNC'04, WCNC'02, WCNC'00 (Technical Program Vice-Chair), WCNC'99, IEEE Globecom'04 (Symposium Co-Chair), Globecom'02, and International Conference on Computer Communications and Networking (IC3N) (Technical Program Vice-Chair).