Congestion Control in Multihop Wireless Networks

Kun Tan, *Member, IEEE*, Feng Jiang, Qian Zhang, *Senior Member, IEEE*, and Xuemin (Sherman) Shen, *Senior Member, IEEE*

***Abstract*—Conventional congestion control protocols, such as the Transmission Control Protocol (TCP), suffer from performance degradation in multihop wireless networks, due to the unique characteristics of shared wireless media. On the other hand, lack of precise congestion indication and coordination among nodes that compete for the shared wireless channel makes TCP fail to allocate resource efficiently and fairly among flows. In this paper, we present a novel Explicit Wireless Congestion Control Protocol (EWCCP) for stationary multihop wireless networks. By exploiting explicit coordination and multibit explicit feedback from routers, EWCCP gains fine-grain control and is robust to the dynamics of the wireless channel. EWCCP stabilizes at a smaller but more optimal sending window size as compared to TCP and achieves low buffer occupation and low delay. With explicit coordination, EWCCP allocates resource fairly among flows that compete for the shared channel. Extensive analysis and simulations demonstrate that EWCCP achieves proportional fairness and is a viable congestion control scheme for multihop wireless networks.**

***Index Terms*—Congestion control, multihop wireless network.**

I. Introduction

THIS PAPER considers the problem of congestion control over multihop wireless networks. In such networks, radio-equipped nodes can communicate with their neighbors directly when they are within the radio transmission range. Two nodes that are far away from each other may rely on intermediate nodes to relay traffic. Our focus is on multihop wireless networks based on IEEE 802.11 (WiFi) technology since it is a standard for wireless local area networks. Many commercial applications have been built on multihop WiFi networks. One such example is "community mesh networks" [20], [21].

Congestion control is a critical issue to ensure the efficient and fair allocation of network resource among communication flows. In the Internet, this issue has been resolved by applying an end-to-end congestion control protocol, i.e., Transmission Control Protocol (TCP) [19]. However, it has been reported that TCP does not perform well in a wireless, especially multihop wireless environment [14], which usually results in inefficient and unfair bandwidth allocation among different flows.

Manuscript received April 17, 2005; revised November 2, 2005 and January 16, 2006. The work of Q. Zhang was supported in part by the National Basic Research Program of China (973 Program) under Grant 2006CB303000, HKUST Nansha Grant NRC06/07.EG01, and MRA05/06.EG03. The review of this paper was coordinated by Prof. T. Hou.

K. Tan is with Microsoft Research Asia, Beijing 100080, China (e-mail: kuntan@microsoft.com).

F. Jiang is with Corporate Technology, Siemens Ltd., Beijing 100102, China.

Q. Zhang is with the Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong (e-mail: qianzh@cs.ust.hk).

X. Shen is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L3G1, Canada (e-mail: xshen@bbcr.uwaterloo.ca).

Digital Object Identifier 10.1109/TVT.2007.891405

This is because the wireless channel is a spatially shared resource. Wireless nodes within the interference range compete for the same wireless channel, and only one transmission is allowed at the same time. Conventional TCP, using packet loss as implicit congestion feedback, adversely interacts with this characteristic of multihop wireless network. First, TCP greedily increases the sending rate until packet loss occurs. This behavior keeps high-level buffer occupation on wireless nodes and, therefore, increases the contention in the multihop wireless network, which consequently causes "wastage" in the wireless broadcast medium [15]. Second, TCP observes congestion only by the packet loss on its own path. However, in a multihop wireless network, two TCP flows, even without any common node, may compete for the same wireless channel. With different locations, these competing flows may have largely different perceptions on the congestion (i.e., packet loss/delay). Without any coordination mechanism, some unlucky flows may always experience more packet losses (or delay) and reduce the sending rate more frequently, while others get a significant large share of bandwidth [8] and [9].

Previous works tried to resolve the two preceding issues separately by adding explicit feedback control, i.e., random early detection (RED)/explicit congestion notification (ECN) [5] and [16], into multihop wireless networks [12], [15]. However, the RED/ECN scheme provides only 1-bit information about the congestion state. It is well known that with 1-bit feedback, TCP is prone to be unstable with large feedback delay [2]. As indicated in [12], the feedback delay of ECN in a multihop wireless network is generally large, i.e., from 100 ms to 1 s, in order to avoid too rugged estimations or reduce overhead. Large oscillation in the sending rate may cause corresponding oscillation in network queues. This instability may cause many potential problems in multihop wireless networks. For example, it may cause underutilization of the wireless channel. As the network queues jump from empty to near full, the state of wireless channels also jump from idle to heavy congestion. Both cases reduce the throughput. Moreover, coordination is required among flows that may interfere with each other to ensure fairness.[1] The key issue is to make these flows aware of the congestion state of each other. In [15], implicitly coordinating congestion observation using passive monitoring of neighbors' transmission is suggested. However, accurately monitoring channel utilization is difficult in practice. Not only does conventional hardware usually disclose critical information from the upper layer but the noisy environment of wireless networks, e.g., wireless loss and collisions, also brings challenges to precise channel measurement.

[1]We may "abuse" the term *interfere* here and say that two flows to interfere with each other if any two links belong to them interfere with each other.

We argue that efficient and fair bandwidth allocation can be achieved by applying *explicit multibit feedback* and *explicit signaling* in multihop wireless networks. By *explicit multibit feedback*, the congestion control protocol is more robust to feedback delay. Multibit feedbacks give fine-granularity control for senders and are able to stabilize the sending rate at a value that can efficiently utilize the wireless channel resource while achieving low buffer occupation and queuing delay. By *explicit signaling*, wireless links that mutually interfere *explicitly* exchange congestion information, i.e., average queue size information, with management packets. With a properly chosen signaling interval, explicit signaling of the queuing information can provide a simple yet precise view of wireless congestion with acceptable overhead.

In this paper, we present a systematic design of a novel Explicit Wireless Congestion Control Protocol (EWCCP) for multihop wireless networks. The protocol achieves both efficient and fair allocation of bandwidth among flows by using 1) explicit multibit congestion feedback from routers, which is computed based on the coordinated congestion information, and 2) an explicit signaling protocol to coordinate flows that contend for the shared wireless channel. We derive an analytical model of EWCCP by extending Kelly's model [4] to multihop wireless networks. Our analysis demonstrates that EWCCP achieves a proportional fair allocation among flows. We then conduct extensive packet-level simulations on IEEE 802.11 distributed coordination function (DCF) medium access control (MAC) and show that EWCCP performs well with commercially available IEEE 802.11 hardware. We believe that EWCCP is a viable congestion control scheme for multihop wireless networks.

The rest of this paper is organized as follows: In Section II, the systematic design of EWCCP is presented in detail. An analytical model of EWCCP is developed in Section III. Extensive simulations are conducted to evaluate EWCCP in Section IV. Related work is given in Section V. Some related issues and further extension of this paper are discussed in Section VI. Section VII concludes this paper.

II. EWCCP

We consider a static multihop wireless network based on IEEE 802.11 technology. IEEE 802.11 uses a carrier sense multiple access with collision avoidance (CSMA/CA) MAC. Each packet may be preceded by an exchange of request to send/clear to send (RTS/CTS) control messages to reserve the channel time for sending the data packet and an ACK. Upon overhearing the handshake, the nodes within the interference range of both the sender and receiver defers their transmissions. Therefore, two wireless links will contend with each other if an endpoint of one link is within the interference range of the endpoints of the other link. For instance, in Fig. 1, link 5∼6[2] is interfering with link 1∼2, 3∼4, 7∼8, 9∼10, etc. We term a wireless link's *interference set* as the set of links that interfere with that link. For example, in Fig. 1, link 5∼6's interference set may contain all links shown in the figure.

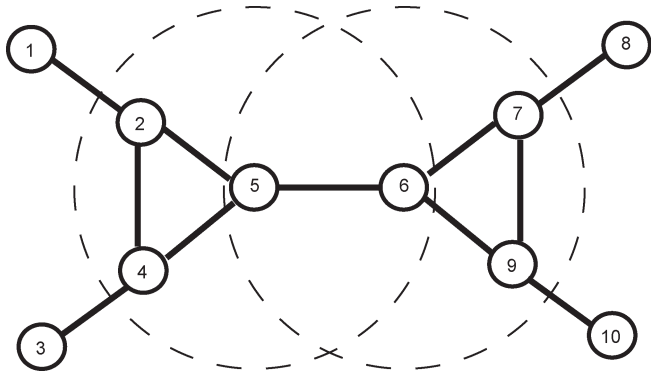


Fig. 1. Illustration of wireless interference of a link. The dashed circles present the interference range.

Links included in one interference set share the same channel resource. Transmissions on one link contribute to the congestion over all links in its interference set. Therefore, feedbacks from congestion control protocols for multihop wireless networks should base on the congestion state in the entire interference set rather than in individual links. The congestion in the interference set can be measured by the aggregated queue size over all links in the entire interference set. We define this aggregated queue as[3] EWCCP routers are responsible for feedback congestion by coordinating among links that mutually interfere with. The congestion feedbacks are provided based on the size of the neighborhood queue defined in the previous section. EWCCP mimics the *additive increase and multiplicative decrease* (AIMD) control scheme. That is, when congestion is detected, different flows get a portion of negative feedback that is proportional to their sending rate. On the other hand, if the network is not congested, every flow will get the same amount of positive feedback, as will be shown later in Section III. This AIMD control scheme converges to a proportional fairness allocation.

A. Framework of EWCCP

The EWCCP sender maintains a congestion window (or simply *window*), which controls the maximum number of packets that are allowed to be sent before receiving an acknowledgment.[4] Every data packet carries a special congestion header, which can be annotated by the intermediate routers. Fig. 2 shows the format of a congestion header. There are two parts. One contains end-to-end congestion control information; the other contains control information for one link. The light grayed

[2]Throughout this paper, we use link #∼# to denote a nondirectional link and link #–# to denote a directional link, where the former one is the sender and the latter one is the receiver.

[3]In multihop wireless networks, there is no strict separation for end systems and core routers. We use these terms just for the sake of description.

[4]EWCCP chooses to use a window-based control scheme rather than a rate-based control. The reason is that wireless channel is a very dynamic medium. Window-based control is more stable because of the beauty of self-clocking. EWCCP also adopts a pacing technique that evenly spreads transmission to avoid burstiness [22].

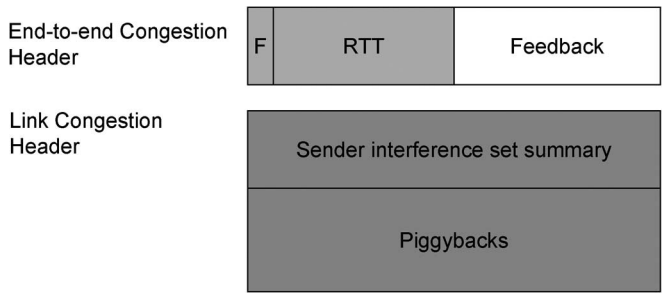


Fig. 2. Congestion header.

field is filled by end systems, and the dark grayed fields are updated by routers. EWCCP senders and receivers only care about the end-to-end congestion header. The link congestion header is used between routers, which will be discussed in Section II-B2.

The intermediate routers calculate the congestion feedback based on the size of the neighborhood queue on each wireless link and add the feedback in the congestion header of the packet. When the packet arrives at the receiver, the feedback field in the congestion header holds the sum of all feedbacks given by all wireless links along the path. This aggregated congestion feedback is echoed back to the EWCCP sender with an acknowledgment from the receiver. Then, the sender can adjust the sending window based on the routers' feedbacks. An *acknowledgment* conveys both positive feedback and negative feedback. However, since EWCCP adopts AIMD, the positive feedback is fixed, which can be easily calculated at end systems. Thus, only negative feedback requires explicit calculation on intermediate routers.

Intermediate EWCCP routers periodically exchange their local queue information with their neighbors. To completely collect the neighborhood queue information, one node may need to know the local queue information of nodes that are multiple hops away. For instance, in Fig. 1, if node 5 wants to calculate congestion feedback, it may need to know the queues on nodes 7 and 8 since link 7-8 and 8-7 interfere with link 5-6. One simple way to do this is to propagate information by rebroadcasting it several times. However, such multiple rebroadcasting increases not only overhead but also delays. In EWCCP, we have developed a technique called *separated control*, which allows EWCCP routers to correctly calculate congestion feedback without complete knowledge of *neighborhood queue*, so that local queue information only needs to be broadcasted within one-hop neighbors. We will discuss it in detail in Section II-B2.

B. EWCCP Router Behavior

An EWCCP router collects the neighborhood queue information and calculates congestion feedbacks. The router makes the control decision once with control interval d, which is called a control *round*. As mentioned previously, routers broadcast their local queue information periodically, say, in interval d_s. From the control theory, a control interval should not be less than a signaling interval, i.e., one must see the results of a control before making another control decision. Therefore, we set $d >= d_s$.[5]

1) Congestion Detection: Congestion is detected on one wireless link if the size of the neighborhood queue is greater than a predefined threshold in the last control interval. In other words, congestion is detected if the following is satisfied:

$$\hat{Q}_{i,j} = \sum_{l \in \mathrm{IS}(i-j)} Q_l > \gamma \tag{1}$$

where $\hat{Q}_{i,j}$ stands for the neighborhood queue on wireless link $(i-j)$, Q_l is the average queue size (in bytes) on link l in the last *round*, IS(.) is the *interference set* of the given link, and γ is the desired queuing level. The rule of thumb for setting γ is to let it be equal to the pipe size of the wireless link. For IEEE 802.11, it is a stop-wait protocol as it adopts a DATA/ACK sequence. The pipe size over one hop is therefore, at most, one packet, which is irrelevant to the link speed [23], [25]. In addition, it is desirable to keep only one link active within an interference set. Therefore, keeping one packet backlogged in the neighborhood queue should efficiently utilize the channel with minimal interference, and we set $\gamma =$ one maximal segment size (MSS).

2) Negative Feedback Calculation: EWCCP is a window-based protocol. The event of receiving an acknowledgment is regarded as a positive feedback, and the "feedback" field in the congestion header conveys the negative feedback. The window is updated based on both types of feedbacks, as will be shown in detail in Section II-C1.

When a control interval is passed, the negative feedback on a wireless link is calculated if congestion is detected according to (1) in the last *round*. Assuming that the neighborhood queue size is $\hat{Q}_{i,j}$, we choose the total negative feedback on the interference set to be proportional to $\hat{Q}_{i,j}$. This is reasonable because higher $\hat{Q}_{i,j}$ means heavier congestion, and therefore, we may decrease the sending rate more quickly. We denote $\varphi_{i,j} = \beta \hat{Q}_{i,j}$ to be the total negative feedback and allocate it to each flow that passes the interference set. Note that in EWCCP, each data packet carries a congestion header, and routers provide congestion feedback by updating the corresponding field in each congestion header. Therefore, we need to calculate the negative feedback for each packet.

First, we derive the total negative feedback on a wireless link. As EWCCP applies *multiplicative decrease* control, the negative feedback for each flow should be proportional to its throughput. Consider the link $(i-j)$, and denote the aggregated flow rate over the link as $x_{i,j}$. Then, the total amount of negative feedback for link $(i-j)$ can be calculated by

$$\phi_{i,j} = \varphi_{i,j} \cdot \frac{x_{i,j}}{\sum_{(k-l) \in \mathrm{IS}(i-j)} x_{k,l}} \tag{2}$$

where IS(.) is the *interference set* of the given link.

[5]The control interval should also be larger than the rtt of any flow. Here, we assume that the control interval is much larger than rtt. Our simulations will verify that EWCCP usually keeps rtt at a very small level.

Second, we divide the total negative feedback on the link, i.e., $\phi_{i,j}$, into each flow that transverses the link. The total decrease in the aggregated sending rate on the wireless link is $(\phi_{i,j}/d)$. Consider flow f that traverses link $(i-j)$, and denote the *window* of f as w_f and the average round-trip time (RTT) of f as rtt_f. Then, the sending rate of f can be approximately represented as $r_f = w_f/rtt_f$. The desired decrease rate of flow f in the next interval is

$$\Delta r_f = \frac{\Delta w_f}{rtt_f} = \frac{\phi_{i,j}}{d} \cdot \frac{r_f}{x_{i,j}} = \frac{\phi_{i,j} \cdot w_f}{d \cdot x_{i,j} \cdot rtt_f}. \tag{3}$$

Last we divide the feedback into each packet. The total number of packets from flow f in one interval d is approximately

$$N_f = w_f \cdot d/rtt_f/s_f \tag{4}$$

where s_f is the packet size of flow f and d is the control interval.

Then, the per-packet negative feedback on *window* n_f (in bytes) is given by

$$n_f = \frac{\Delta w_f}{N_f} = \frac{\phi_{i,j}}{d^2 \cdot x_{i,j}} \cdot rtt_f \cdot s_f = \beta \frac{\hat{Q}_i}{d^2 \cdot \sum\limits_{j \in \mathrm{IS}(i)} x_j} \cdot rtt_f \cdot s_f. \tag{5}$$

Note that when there is congestion and queues are built up, $\sum_{k-l \in \mathrm{IS}(i-j)} x_{k,l}$ can be approximated by ψC, where C is the link speed of the wireless channel and ψ is a constant that counts for the MAC layer overhead. Therefore, we can simplify (5) as

$$n_f = \beta \frac{\hat{Q}_{i,j}}{\psi \cdot C \cdot d^2} \cdot rtt_f \cdot s_f \tag{6}$$

where d is the control interval, and rtt_f is the RTT carried in the packet congestion header.

Using (6) to compute the congestion feedback, all factors, except $\hat{Q}_{i,j}$, are constant to a specific flow. Furthermore, $\hat{Q}_{i,j}$ is a linear sum of all queues in the interference set. Recalling the example in Fig. 1, it does not require node 5 to know the whole neighborhood queue information. Node 5 can compute the negative feedback based on the partial knowledge of the neighborhood queue (e.g., the queue on links 1∼2, 3∼4, 2∼3, etc.). When the packet is received by node 6, it can further compute another part of the negative feedback (i.e., containing the queue information of links 7∼8, 9∼10, 7∼9, etc). The sum of these two partial results is the complete negative feedback from link 5-6.

We name this technique *separated control*, which allows nodes to perform feedback calculation with only partial knowledge of the neighborhood queue. This reduces the propagation scope of the queuing information, which greatly reduces the signaling overhead.

The link congestion header, in Fig. 2, is used to carry the information for *separated control*. More specifically, the link congestion header contains a hashed summary vector for the IDs of links in the interference set that are known to the link sender (Note that there is also a "piggyback" field, which will be detailed in Section II-D.). The summary vector can be generated using Bloom filters [24]. When the link receiver receives a data packet, it checks all the links it knows in the interference set against the summary in the packet. If the link is contained in the summary, the receiver just skips it, as the link sender has already counted this link in its feedback. All other links are then considered, and the link receiver feedbacks are calculated according to (6). Finally, the feedbacks are added back to the end-to-end congestion header to complete the feedback of the link.

3) Start-Up Probe: When a connection is started, EWCCP initializes the *window* value to be one MSS, and only one packet is allowed to be sent out. The first data packet also serves as a *probe packet* with a mark in the *flag* field in the congestion header. Probe packets are treated specially by routers. The feedback field of a probe packet is filled by routers for a proper initial *window*. In addition, when an acknowledgment for a probe packet is received, the sender directly sets the current window to the value suggested by the routers, instead of slowly probing through linear increase. For any flow, only one probe packet is allowed to be sent at any time.

The EWCCP router uses a very simple yet effect heuristic to set the proper starting window. It is still based on the observation that the pipe size of an 802.11 link is, at most, one, which is irrelevant to the link speed. Therefore, if the router does not detect congestion, it will increase the feedback of a probe packet by one. However, if local congestion is detected, the router should turn off the flag field and send congestion feedback, as usual.

C. End-System Behaviors

1) EWCCP Sender: As mentioned previously, an EWCCP sender maintains a *window* and a smoothed rtt. The sending rate of the sender can be estimated as *window/rtt*. With a timer, the sender evenly spreads a window of data packets within rtt.

When a connection is started, EWCCP initializes the *window* to be one MSS and sends the first packet. When an acknowledgment for this packet is received, the sender may directly set the current window to the value suggested by the router (see Section II-B3).

After initial start-up, the EWCCP sender updates the *window* according to the acknowledgment received. An discussed before, the event of receiving an acknowledgment conveys a fixed positive feedback, while the feedback field in the congestion header contains the negative feedback from routers. Therefore, the *window* should be updated according to

$$w(k+1) = w(k) + p - n \tag{7}$$

where

$$p = \alpha \cdot rtt_f \cdot s_f^2/(w(k) \cdot d) \tag{8}$$

and n is the feedback carried in the packet congestion header.

Note that the "increase" part calculated in (8) implements *additive increase*. However, (8) is different from the one used in TCP, in which the additive increase is scaled with rtt, and therefore, flows with longer rtt are treated with bias. EWCCP makes the increase scale with the control interval of the routers

Src ID	Dst ID	Type	Value(kB)	Seq #
5	6	O	0.1	6433
2	5	P	0.4	2201
2	1	B	1	1000

Fig. 3. Example of the neighbor queue table.

and, thus, removes the rtt unfairness. α is the parameter that controls the increase step. If $\alpha = 1$, it means that the flow increases one MSS for each control interval.

2) EWCCP Receiver: An EWCCP receiver is rather simple. It only receives data packets and copies the end-to-end congestion header from data packets to acknowledgments.

D. Signaling Protocol

EWCCP uses explicit signaling to coordinate with neighbor nodes. It periodically broadcasts the average queue information. The broadcasting is limited only within the one-hop neighbors.

The broadcast packet contains the average queue size in the last round of all links of a node. It includes not only the outgoing links but also the incoming links of the node. Note that the queue information of a link is usually available only at the link sender, and we make it also available at the link receiver by using the piggyback field in the link congestion header, as shown in Fig. 2. When a data packet is transmitted, the queue information of the link is also delivered to the link receiver. Then, the receiver can propagate it out with broadcasting.

Each node maintains a neighbor queue table. One example is shown in Fig. 3. A link is uniquely identified by the IDs of the sender node and receiver node. Each entry has the following three types: 1) *O* (own), presenting a queue on a downstream link; 2) *P* (piggyback), presenting a queue that is piggybacked from an upstream link; and 3) *B* (broadcast), presenting the other information learned from receiving broadcast packets. The sequence number is used to judge the freshness of the entry. A node can only increase the sequence number of the entries that marks *O*.

There is a tradeoff in the interval between two successive broadcasting. Frequent broadcasting will give more timely information but also consumes more bandwidth. A signaling interval with a magnitude of hundreds milliseconds should be an acceptable overhead. We evaluate this with a simple calculation. Assume that one node may have an average of ten neighbors in one hop and that the signaling interval is 100 ms. Each entry in the queue table may cost 6 bytes (i.e., 2 bytes link id, 2 bytes queue size, 1 byte sequence number, and 1 byte flag). The total bandwidth for signaling is $(10+1)*(2*8*10*6)/100 = 105.6$ kb/s, which is about 1% of the 11-Mb/s channel of 802.11b without counting the MAC overhead. Note that multiplier 2 is applied in preceding calculation because wireless links may be bidirectional; therefore, both incoming and outgoing queues are broadcasted.

The signaling protocol is summarized as follows.

1) When a node sends a data packet along a downstream link, it fills the average queue size of that link in the piggyback field in the link congestion header.
2) When a signaling interval is passed, a broadcast packet is generated and sent. The broadcast packet contains all the entries in the table that marks *O* and *P*.
3) When receiving a broadcast or a data packet with link congestion header, a node updates the neighbor queue table with the information contained. If an entry already exists, the one with the larger sequence number wins.

III. Analytical Model of EWCCP

In this section, we will focus on EWCCP in an analytical way. In Section III-A, we will first brief the network model. In Section III-B, we will formulate the congestion control based on Kelly's framework and extend it to the wireless network, where the wireless interference between links is considered. Finally, we show that the EWCCP control law given in (6)–(8) is the solution to the problem that we have formulized in Section III-C.

A. Network Model

We consider a wireless network with N nodes. Let L denote the set of node pairs (i, j) such that transmission from node i to node j is allowed. Such a node pair is also called a link. Due to the shared nature of the wireless media, the capacity of a specific link (i, j), i.e., c_{ij}, depends not only on the transmission power modulation but also on the interference with other links. Without loss of generality, we assume that all nodes use the same power and modulation method for any transmission and that the interference only happens between one-hop neighbors of a node. We denote the feasible rate allocation of links in the wireless network as $\vec{c} = \{c_{ij}, (i, j) \in L\}$ and assume that R, which is the convex hull of $\vec{c}$, is closed and bounded.

B. Congestion Control Problem

We denote the senders of a flow as S and the destinations as D. Furthermore, we assume that each user has only one flow, so that we use s to denote both a user and a flow in the network. Let A denote the routing matrix, i.e., $A_{i-j,s} = 1$, if flow s traverses link $(i - j)$.

Following [4], the congestion control problem can be modeled as a nonlinear programming problem P, i.e.,

$$\max \sum_s U_s(x_s) \tag{9}$$

subject to

$$\sum_s A_{i-j,s} x_s \le c_{i,j} \tag{10}$$

and

$$[c_{i,j}] \in R. \tag{11}$$

$U_s(.)$ is the utility function of user s, and x_s is the rate of user s. However, unlike the wired network, the capacity on a wireless link is not fixed but changes with the rates on the links that can interfere with it. If two links are within interference range, they cannot transmit together. Note that 802.11 DCF MAC is a random access protocol. Without synchronization, DCF cannot schedule well the transmission on each link. Therefore, we only consider the lower bound of $c_{i,j}$ as

$$c_{i,j} \approx C - \sum_{(l-k)\in \mathrm{IS}(i-j)/(i-j)} c_{l,k} \Rightarrow \sum_{(l-k)\in \mathrm{IS}(i-j)} c_{l,k} \approx C \tag{12}$$

where C is the link speed of the wireless channel.

We define G as the interference matrix, i.e., $G_{(i-j),(l-k)} = 1$, if links $(i-j)$ and $(l-k)$ interference with each other. Let $H = GA$. Using (12), the problem P can be rewritten as

$$\max \sum_s U_s(x_s)$$
$$\text{subject to } \sum_s H_{(i-j),s} x_s \le C. \tag{13}$$

We can use the Lagrange relaxation to solve the problem defined in (13). Consider the Lagrangian

$$L(x,\lambda) = \sum_s w_s U_s(x_s) + \sum_{(i-j)\in L} \lambda_{i,j}(H_{(i-j),s} x_s - C). \tag{14}$$

The Lagrange relaxation of problem P can be defined as problem Q

$$\max \sum_s V_s(x_s)$$

where

$$V_s(x_s) = w_s U(x_s) - \sum_{(i,j)\in P_s} \int_0^{\sum_{(k-l)\in \mathrm{IS}(i-j)} x_{k,l_s}} \lambda_{i,j}(y)dy \tag{15}$$

and $\lambda_{i,j}(y)$ is a nonnegative, continuous, convex, and increasing function of rates on links that cross the interference set of link $(i-j)$. P_s is the path that flow s transverses. $\lambda_{i,j}(y)$ can also be interpreted as the penalty function, which is the unit price for transmission over the link $(i-j)$. Equation (15) also states that each flow can maximize its own $V_s(x_s)$, so that the system wise optimal can also be obtained [3].

C. EWCCP Solution

Recall the control law represented by (7) and (8) in Section III-B, which adjusts the *window* based on per-packet feedback (both positive and negative). Translating (7) and (8) into fluid model, we have

$$\dot{x}_s = \alpha - \beta \cdot x_s \sum_{(i-j)\in P_s} \frac{\hat{Q}_{i,j}}{d \cdot C} \tag{16}$$

where $\hat{Q}_{i,j}$ is the neighborhood queue of link $(i-j)$, d is the control interval, and P_s is the path that flow s transverses.

For a specific wireless link $(i-j)$, the average local queue size in the last control interval is approximately

$$Q_{i,j} \approx (x_{i,j} - c_{i,j})^+ \cdot d \tag{17}$$

where $(.)^+$ is defined as max (0, .).

Substituting (17) into (16), we have

$$\dot{x}_s = \alpha - \beta \cdot x_s \sum_{(i-j)\in P_s} \frac{\sum_{(k-l)\in \mathrm{IS}(i-j)} (x_{k,l} - c_{k,l})^+}{C}. \tag{18}$$

Using $\sum_{(k-l)\in IS(i-j)} x_{k,l} \approx C$ and (12), we have

$$\dot{x}_s = \alpha - \beta \cdot \sum_{(i-j)\in P_s} x_s \frac{\left(\sum_{(k-l)\in \mathrm{IS}(i-j)} x_{k,l} - C\right)^+}{\sum_{(k-l)\in \mathrm{IS}(i-j)} x_{k,l}}. \tag{19}$$

Let

$$\lambda_{i,j} = \frac{\left(\sum_{(k-l)\in \mathrm{IS}(i-j)} x_{k,l} - C\right)^+}{\sum_{(k-l)\in \mathrm{IS}(i-j)} x_{k,l}}. \tag{20}$$

Equation (19) can then be rewritten as

$$\dot{x}_s = \alpha - \beta \cdot x_s \cdot \sum_{(i-j)\in s} \lambda_{i,j}. \tag{21}$$

Note that (21) converges to the unique solution to problem defined in (15), where $\lambda_{i,j}(y)$ is taken the form in (20), and $U(x_s) = \log(x_s)$. The proof simply follows the lines in [3]. We know that *the proportional fairness is represented by the utility function* $\log(.)$, and therefore, EWCCP control law achieves *proportional fairness* in multihop wireless networks.

IV. Performance Evaluation

We have implemented EWCCP in an NS2 simulator [29] and conducted extensive packet-level simulations. We demonstrated that EWCCP achieved efficient and fair resource allocation in a multihop wireless environment. In the following tests, we use 802.11a DCF with RTS/CTS enabled. The channel speed is set to 54 Mb/s, and the communication range is tuned to be 250 m. In comparison with TCP, we did not consider any *error model* in the wireless channel. We use TCP selective acknowledgment and dynamic source routing [17]

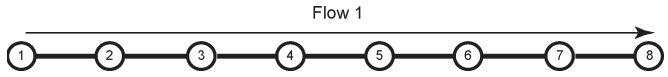


Fig. 4. Chain of seven hops.

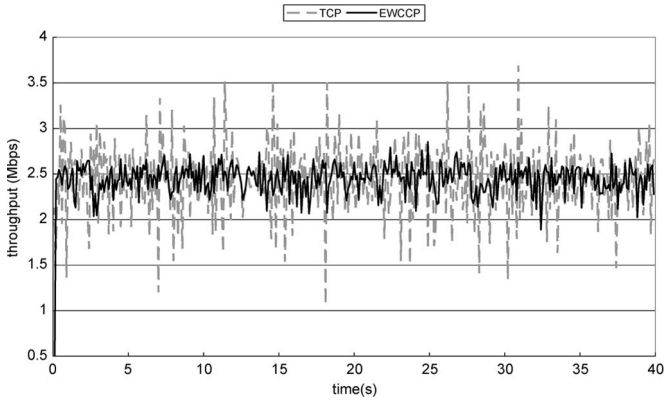


Fig. 5. Instantaneous throughputs of EWCCP and TCP over a chain topology.

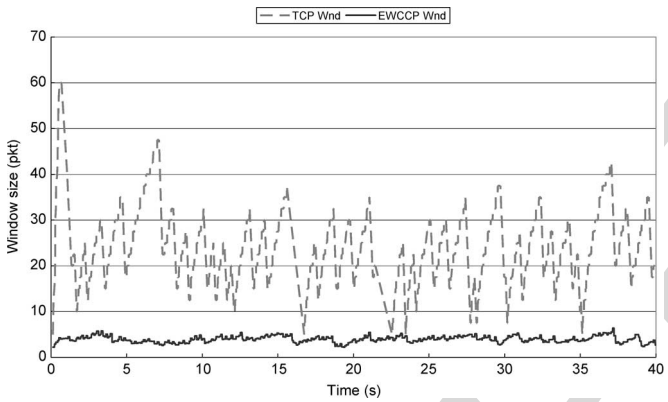


Fig. 6. Window evolution of TCP and EWCCP over a chain topology.

with robust link failure detection.[6] The parameter setting for EWCCP was $\alpha = \beta = 1$. We note that the performance of EWCCP is not sensitive with this parameter setting. We set both the signaling interval and the control interval of EWCCP to 100 ms, and all packet sizes are set to 1000 B.

A. Single Chain Topology

We consider a simple chain topology, as shown in Fig. 4. One flow is running end to end from node 1 to node 8, passing seven hops. The instantaneous throughputs of EWCCP as well as TCP are shown in Fig. 5, and the *window* evolutions of both TCP and EWCCP are shown in Fig. 6. The throughput, as well as the *window*, of TCP had much larger variance compared to that of EWCCP. In Table I, the overall throughput, congestion window, rtt, and link-layer packet losses of the two schemes are summarized. In this simple scenario, although TCP has a much larger window than EWCCP, the total throughput of TCP is even slightly less than that of EWCCP. The reason can be explained by examining the last two columns of each protocol. It can be seen that TCP has much larger RTT value compared to EWCCP. This means that most of the packets in one window were buffered somewhere in the network and, therefore, greatly increased the queuing delay. Pushing too many packets into the network drives the wireless network into saturation, which further increases the link-layer losses due to heavy contention. Note that contention causes not only packet losses but also backoff in MAC, which wastes channel time. On the contrary, with multibit explicit feedback, EWCCP kept the queue in the neighborhood small and stabilized the sending window at a value that can efficiently utilize the channel resource but maintain very small queuing delay.

[6]In our implementation, a link failure is only declared after severe packet loss (i.e., three consecutive packets are lost) is detected on that link. This is unlike the current implementation in NS2 2.27, which declares a route failure on the first loss of a packet. It is reasonable to have such robust link failure detection in a stationary multihop wireless network, as link failure is rather uncommon, and single packet loss may be caused by random interference.

TABLE I
THROUGHPUT, WINDOW, AND RTT FOR EWCCP AND TCP OVER A CHAIN TOPOLOGY

	Ave. throughput	Ave. cwnd	Ave. RTT	Packet loss #
EWCCP	2.35Mbps	3.5pkts	0.01s	1pkt
TCP	2.32Mbps	24.5pkts	0.12s	30pkts

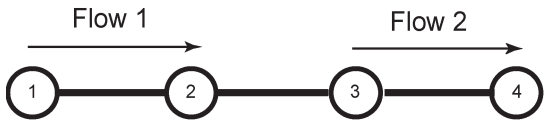


Fig. 7. Exposed terminal topology.

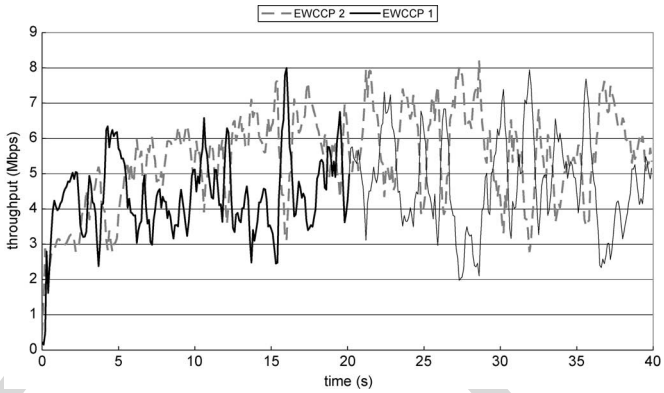


Fig. 8. Instantaneous throughput of EWCCP over an exposed terminal topology.

B. Exposed Terminal Topology

We test a simple yet typical that TCP demonstrated great unfairness because of the lack of coordination in competing flows on different nodes, as shown in Fig. 7. We also evaluate EWCCP in this topology. Figs. 8 and 9 show the instantaneous throughput of both EWCCP and TCP. As expected, TCP 2, which is at a better location to compete the wireless channel, eventually starves TCP 1. However, EWCCP controls the sending window based on the multibit feedback computed from the aggregated queue over the interference set. Therefore, even though EWCCP 2 can easily obtain the channel, it gracefully

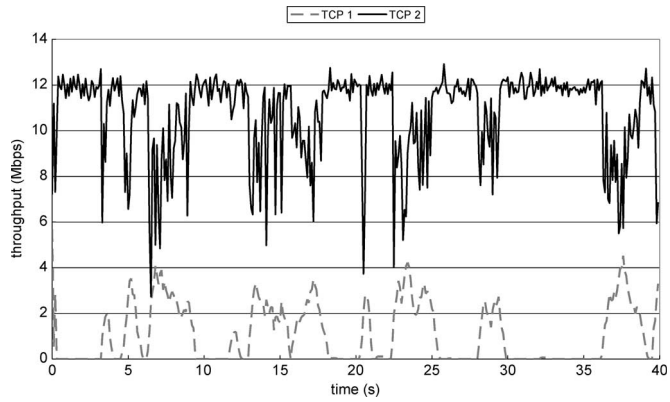


Fig. 9. Instantaneous throughput of TCP over an exposed terminal topology.

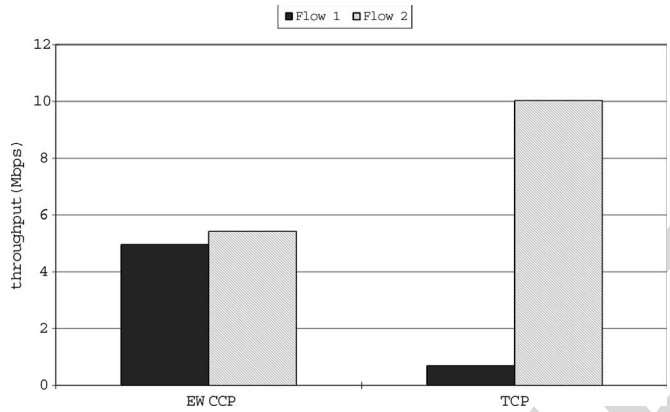


Fig. 10. Overall throughput of EWCCP and TCP in an exposed terminal scenario.

retreats when the queue is building at node 1, which yields the channel time for EWCCP 1. As a consequence, these two flows share the channel fairly. Fig. 10 shows the overall throughput of each flow and the aggregated throughput of TCP and EWCCP. Note that with multibit feedback, EWCCP still provides rather stable control in this heavy contention situation. On the other hand, although the distributed RED/ECN proposed in [12] helps to improve fairness in the average throughput, the instantaneous throughputs of two flows alternatively oscillate from zero to the maximum, which occupies the whole channel resource.

C. Bar Topology

In the preceding scenarios, we test EWCCP over symmetric topology. Next, we test EWCCP with a topology where two flows experience different congestions when competing for the same channel resource, as shown in Fig. 11. The simulation results of both TCP and EWCCP are shown in Fig. 12. In this figure, the aggregated throughput of the two flows is also plotted. It can be seen that TCP has higher aggregated throughput compared to EWCCP. However, this reflexes the fundamental tradeoff between efficiency and fairness. Since EWCCP 1 consumes twice the channel resource to deliver packets, it only gets about half the throughput of EWCCP 2, according to the proportional fairness achieved by EWCCP. In this experiment, the throughput of EWCCP 1 is 2.87 Mb/s,

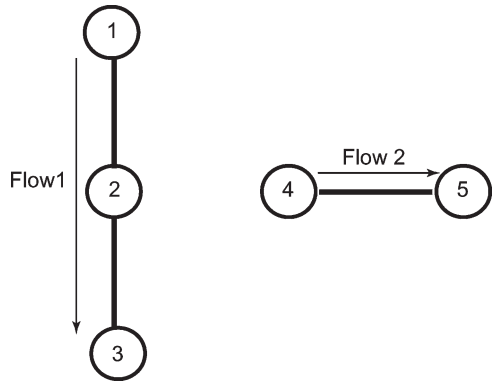


Fig. 11. Bar topology. Nodes 2 and 4 can interfere with each other.

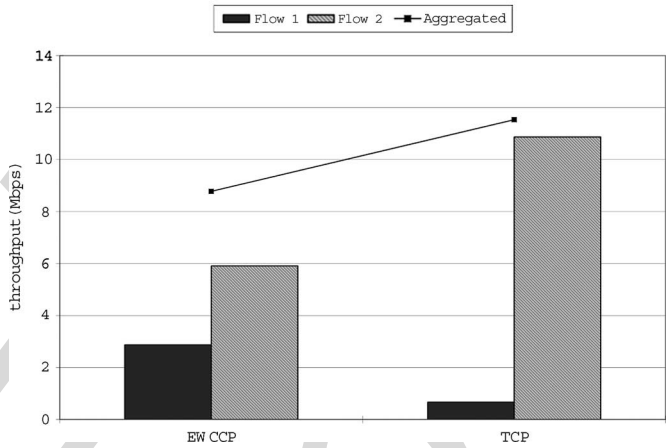


Fig. 12. Overall throughput of EWCCP and TCP in a bar scenario.

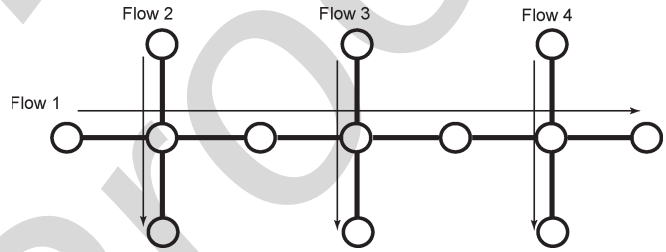


Fig. 13. Parking-lot topology.

while that of EWCCP 2 is about 5.92 Mb/s. The aggregated throughput is 8.79 Mb/s, which should be three fourth of the throughput if all channel resources are allocated only to EWCCP 2, which is the case of TCP. Without coordination, TCP allocates all channel resources to the flow with better condition, e.g., shorter hops, and the flow with longer hops is almost starved. TCP 1 has a very high throughput of 10.87 Mb/s, while TCP 2 only has merely 660 kb/s.

D. Parking-Lot Scenario

In this section, we examine a parking-lot topology where a long flow travels through multiple bottlenecks. The topology is shown in Fig. 13. In addition, the overall throughput of each flow using EWCCP and TCP is shown in Fig. 14. It can be seen that EWCCP preserves fairness for all flows pretty well. Note that flow 1 receives nearly one third of the throughput of other

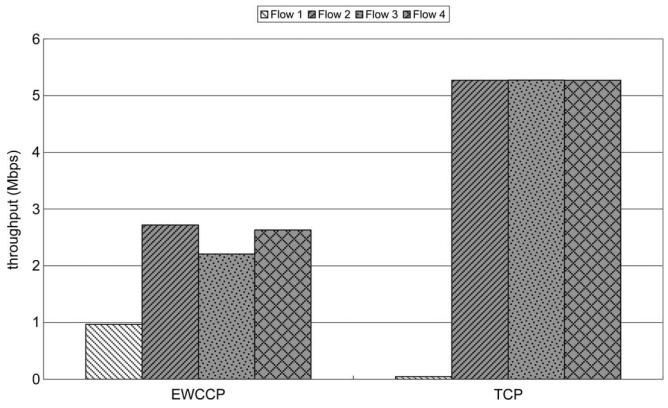


Fig. 14. Overall throughput of EWCCP and TCP in a parking-lot topology.

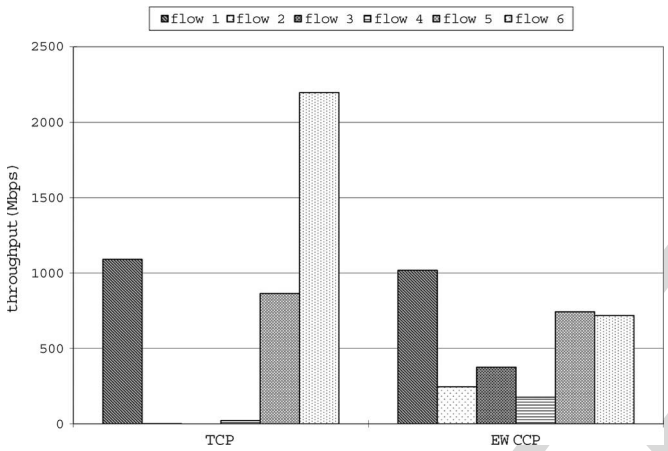


Fig. 15. Overall throughput of six TCP and EWCCP flows under a large grid topology.

flows. This is because the proportional fairness implemented by EWCCP and long flow 1 traverses three bottlenecks, while others only pass one. In this case, TCP fails to fairly allocate resources to the long flow. Short flows occupy all channel resources and achieve high throughput, while the long flow is eventually starved.

E. More Complex Topology

In this section, we evaluate EWCCP under a more complex yet realistic scenario. We construct a stationary multihop wireless network that contains 60 nodes. These nodes form a 10×6 grid topology. The distance between two adjacent nodes is set to 240 m. We randomly generate six flows in the networks. The six flows are started randomly within 100 ms after the simulation is started. Each simulation lasts for 100 s. Fig. 15 shows a typical result of the overall throughput of the six TCP and EWCCP flows. TCP fails to allocate the bandwidth fairly among the competing flows. In general, flows with longer hops are easily starved by shorter flows. For example, in the simulation run shown in Fig. 15, flow 6 has only two hops, and therefore, it quickly starves the nearby flows, i.e., flows 2, 3, and 4, which have longer hops. However, EWCCP achieves reasonable fairness among all flows. We repeat our simulations for several times with different flow patterns, and we get similar results, in which EWCCP allocates bandwidth rather fairly among all flows. Note that EWCCP allocates less bandwidth to longer flows than shorter flows. This is because EWCCP is designed to achieve the proportional fairness where flows consuming more resources (i.e., channel time) to deliver a packet should get less throughput.

V. Related Work

The degradation of TCP's performance in a multihop wireless network has been well recognized in the last decade. Many of the performance issues are related to mobility; they are mainly addressed by cross-layer design, which incorporates event notifications from the network or even MAC layer [7], [13], [14]. In this paper, we focus on static multihop wireless networks.

Fu *et al.* [15] reports that the greedy behavior of TCP drives the wireless network into saturation, which in turn reduces the throughput. They provide a link-layer solution, i.e., link RED (LRED) and adaptive pacing, to mitigate the problem. With our experiments, the degradation due to link-layer contention is around 1%. Therefore, in our setup, adaptive pacing, which deliberately adds extra backoff between successive transmissions, may only add additional delay in delivering packets. Moreover, LRED calculates dropping based only on its own perception. Without coordinating with other interference links, it still has the same fairness problem of TCP if different links in the same congestion region observe different congestions. Chen *et al.* [30] studied the TCP congestion window limit (CWL) by considering the bandwidth delay production of a path in multihop wireless networks. They propose a heuristic algorithm that sets the CWL to one fifth of the round-trip hop count (RTHC). Although their scheme can effectively reduce the self-introduced interface of TCP, it does not consider the fairness among different flows since it only sets an upper bound of the window based on the RTHC of each flow. When two flows competing for the same congestion region observe different packet losses, the one with less packet losses may reach its CWL and dominate the channel, while the other one may be starved eventually.

Xu *et al.* [12] develops a neighborhood RED (NRED) to address TCP unfairness issues in *ad hoc* networks. The idea is to coordinate a common packet dropping rate based on a virtual distributed queue size in the neighborhood of each node. The virtual queue contains queues on the node and its two-hop neighbors. This is different from our neighborhood queue definition. The neighborhood queue in this paper is defined over a wireless link and its interference set, which captures the interference between transmissions more precisely. In [12], it is assumed that nodes can monitor channel utilization and uses the measured utilization as a congestion indicator. Based on this, NRED tries to maintain the channel utilization at some lower level. Although this helps improving the fairness, it reduces the aggregated throughput. Furthermore, accurately monitoring channel utilization is still difficult in practice. In EWCCP, we directly use the size of the neighborhood queue as congestion indicator. The control law properly distributes the neighborhood queue over links that mutually interfere. In this way, EWCCP improves the fairness but still keeps the channel

at high utilization. Moreover, NRED still relies on 1-bit congestion feedback, which may make the control protocol unstable. EWCCP provides a systematic way for resource allocation in multihop wireless networks using multibit explicit rate control. EWCCP also removes RTT unfairness. EWCCP provides stable control and implements proportional fairness among flows. Motivated by lacking information by only 1 bit, recent research reveals that adding only limited bits (e.g., 2 bits) in explicit feedback may be good enough to improve the performance of transport protocols in the Internet [31]. Using only 2 bits for explicit feedback has benefits in retaining the format of TCP/IP format and does not need the addition of new congest headers. However, it is mainly designed and evaluated for high-speed Internet connections. Therefore, it is an interesting future work for us to investigate the proper bit length in explicit feedbacks in multihop wireless networks.

Beside the enhancements to TCP, several new protocols have also been proposed for *ad hoc* networks [10], [11]. Sundaresan *et al.* develops an *ad hoc* transport protocol (ATP) that relies on routers to give explicit rate control. The control information is the bottleneck service delay (including transmission and queuing delay). However, like TCP, ATP provides feedback only based on its own observation. Without coordination with other competing nodes, ATP suffers the same unfairness issue as TCP due to asymmetric information. Unlike ATP, which uses rate-based control, EWCCP adopts a window-based scheme and enjoys self-clocking for more stable control. In EXACT [11], explicit rate calculation is used to implement max–min fairness. However, like ATP, lack of coordination exposes EXACT to the same unfairness. Moreover, EXACT assumes that the node has the full states of all flows passing by. On the contrary, EWCCP puts congestion information into packets, and routers do not need to remember per-flow information. This makes EWCCP more scalable for potential high user volume in a high-speed multihop wireless network.

Note that there are also a number of theoretic works on the rate control in multihop wireless networks [26]–[28]. They usually assume either a slotted system with perfect scheduling [26], [27] or a simple interference model [28]. These studies do give some insights for congestion control protocol design but may have a wide gap from a practical scheme for multihop wireless networks.

VI. Discussion

EWCCP is a new congestion control protocol specifically designed for static multihop wireless networks. It has no means to replace the widely deployed TCP stack. EWCCP can be implemented as a thin layer between IP and TCP, which provides enhanced congestion control functionality in multihop wireless networks (similar to the idea of *ad hoc TCP* [13]). When packets are sent from the TCP layer to the EWCCP layer, they are also regulated by EWCCP. Packets are only sent out when they are allowed by the EWCCP window. EWCCP can also be implemented to be invisible to TCP. Therefore, the standard TCP stack does not need to be modified, but applications can enjoy the efficiency and fairness brought by EWCCP. Another benefit of this design is that EWCCP will not adversely interact with TCPs functioning in cases where the communication is between a node in the multihop wireless network and another on the Internet. The gateway nodes that bridge the two networks can automatically remove/add the EWCCP congestion header according to the direction in which the packet goes.

In previous discussions and simulations, we have assumed that the transmission range equals the interference range. However, in some cases (e.g., outdoor open environment), the interference range may be much larger than the transmission range. As shown in [1] and [6], the distance and shape of the interference range heavily depend on the physical environment. The general ways to precisely identify the interference range are out of the scope of this paper. We assume that the interference information is available when the multihop wireless network is planned. We note that EWCCP can still be effective when the interference range is much larger than the transmission range. The only change will be made to the signaling protocol. Instead of broadcasting within one-hop neighbors, rebroadcasting within k-hop neighbors may be allowed. For example, if the interference range is twice the transmission range, the local queue information may be propagated within two-hop neighbors. We regard this as another advantage of using an explicit signaling protocol than passive channel monitoring in [12], which might not be effective if the interference range is much larger than the transmission range. Note that *separated control* can still help in reducing the signaling overhead. Evaluating the signaling overhead in this situation is the subject of future work.

VII. Conclusion

In this paper, we have proposed a novel EWCCP for stationary multihop wireless networks. EWCCP can exploit explicit coordination with wireless links that mutually interfere and provide multibit explicit feedbacks for fine-grained control. With explicit feedback, the EWCCP sending window stabilizes at a smaller but more optimal value compared to TCP and therefore, achieves low buffer occupation and delay. With explicit coordination among wireless links that compete for the shared channel, EWCCP allocates channel resource fairly. With our analysis and packet-level simulation, it is concluded that EWCCP is a viable congestion control scheme for a stationary multihop wireless network. Further analytical results on congestion control behavior in multihop wireless networks are forthcoming.

References

[1] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris, "Link-level measurements from an 802.11b mesh network," in *Proc. SIGCOMM*, 2004, pp. 121–132.

[2] S. H. Low, F. Paganini, J. Wang, S. Adlakha, and J. C. Doyle, "Dynamics of TCP/RED and a scalable control," in *Proc. IEEE INFOCOM*, 2002, pp. 239–248.

[3] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: Utility functions, random losses and ECN marks," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 689–702, Oct. 2003.

[4] F. P. Kelly, "Charging and rate control for elastic traffic," *Eur. Trans. Telecommun.*, vol. 8, no. 1, pp. 33–37, Jan. 1997.

[5] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, Aug. 1993.

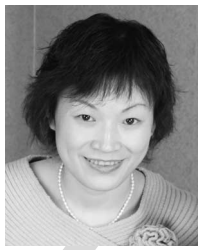
[6] D. Kotz, C. Newport, and C. Elliott, "The mistaken axioms of wireless-network research," Dartmouth College, Hanover, NH, Dartmouth College Tech. Rep. TR2003-467, Jul. 2003.
[7] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash, "A feedback-based scheme for improving TCP performance in ad hoc wireless networks," *IEEE Pers. Commun.*, vol. 8, no. 1, pp. 34–39, Feb. 2001.
[8] S. Xu and T. Saadawi, "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, Jun. 2001.
[9] K. Tang and M. Gerla, "Fair sharing of MAC under TCP in wireless ad hoc networks," in *Proc. IEEE MMT*, Venice, Italy, Oct. 1999, pp. 127–133.
[10] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh, and R. Sivakumar, "ATP: A reliable transport protocol for ad-hoc networks," in *Proc. ACM Mobihoc*, 2003, pp. 64–75.
[11] K. Chen, K. Nahrstedt, and N. Vaidya, "The utility of explicit rate-based flow control in mobile ad hoc networks," in *Proc. IEEE WCNC*, 2004, pp. 1921–1926.
[12] 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, pp. 16–28.
[13] J. Liu and S. Singh, "ATCP: TCP for ad hoc network," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 6, pp. 1300–1315, Jul. 2001.
[14] G. Holland and N. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *Proc. ACM/IEEE Mobicom*, 1999, pp. 219–230.
[15] 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, pp. 1744–1753.
[16] K. Ramakrishnan and S. Floyd, *Proposal to Add Explicit Congestion Notification (ECN) to IP*, Jan. 1999. RFC 2481.
[17] D. B. Johnson, D. A. Maltz, and Y. Hu, *The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)*, Jul. 2004. Internet Draft.<draft-ietf-manet-dsr-10.txt>.
[18] R. Draves, J. Padhye, and B. Zill, "Routing in multi-radio, multi-hop wireless mesh networks," in *Proc. ACM Mobicom*, 2004, pp. 114–128.
[19] W. R. Stevens, *TCP/IP Illustrated*, vol. 1. Reading, MA: Addison-Wesley, 1994.
[20] *MIT Roofnet*. [Online]. Available: http://www.pdos.lcs.mit.edu/roofnet
[21] "Microsoft research," *Self-Organizing Neighborhood Wireless Mesh Networks*. [Online]. Available: http://research.microsoft.com/mesh/
[22] A. Aggarwal, S. Savage, and T. Anderson, "Understanding the performance of TCP pacing," in *Proc. IEEE INFOCOM*, 2000, pp. 1157–1165.
[23] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 1992.
[24] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422–426, Jul. 1970.
[25] J. Li, C. Blake, D. De Couto, H. I. Lee, and R. Morris, "Capacity of ad hoc wireless networks," in *Proc. ACM Mobicom*, 2001, pp. 61–69.
[26] Y. Xue, B. Li, and K. Nahrstedt, "Price-based resource allocation in wireless ad hoc networks," in *Proc. IWQoS*, 2003, pp. 79–96.
[27] Z. Fang and B. Bensaou, "Fair bandwidth sharing algorithms based on game theory frameworks for wireless ad-hoc networks," in *Proc. IEEE INFOCOM*, 2004, pp. 1284–1295.
[28] Y. Yi and S. Shakkottai, "Hop-by-hop congestion control over a wireless multi-hop network," in *Proc. IEEE INFOCOM*, 2004, pp. 2548–2558.
[29] *The Network Simulator ns-2*. [Online]. Available: http://www.isi.edu/nsname/ns
[30] K. Chen, Y. Xue, and K. Nahrstedt, "On setting TCP s congestion window limit in mobile ad hoc networks," in *Proc. IEEE ICC*, May 11–15, 2003, vol. 2, pp. 1080–1084.
[31] Y. Xia, L. Subramanian, and I. Stocia, "One more bit is enough," in *Proc. ACM SIGCOMM*, 2005, pp. 37–48.

Kun Tan (M'04) received the B.E., M.E., and Ph.D. degrees from Tsinghua University, Beijing, China, in 1997, 1999, and 2002, respectively, all in computer science and engineering.

He joined Microsoft Research Asia, Beijing, as an Associate Researcher in April 2002 and is currently a Researcher with the Wireless and Networking Group. He filed six pending patents after he joined Microsoft Research Asia. His research interests include transport protocols, congestion control, delay-tolerant networking, and wireless networks and systems.

Feng Jiang received the B.S. and M.S. degrees from Huazhong University of Science and Technology, Wuhan, China, in 2002 and 2005, respectively, both in electronics and information engineering.

From summer of 2004 to spring of 2005, he was a Visiting Student at the Wireless and Networking Group, Microsoft Research Asia, Beijing, China. He is currently with Corporate Technology, Siemens Ltd., Beijing, focusing on system evolution of future mobile networks. His research interests include wireless *ad hoc* networks, multimedia streaming, and WCDMA 3G systems.



Qian Zhang (M'00–SM'04) received the B.S., M.S., and Ph.D. degrees from Wuhan University, Wuhan, China, in 1994, 1996, and 1999, respectively, all in computer science.

She joined the Department of Computer Science, Hong Kong University of Science and Technology, Hong Kong, in September 2005 as an Associate Professor. Before that, as of July 1999, she was with Microsoft Research Asia, Beijing, China, where she was the Research Manager of the Wireless and Networking Group. She has published more than 150 refereed papers in international leading journals and key conferences in the areas of wireless/Internet multimedia networking, wireless communications and networking, and overlay networking. She is the inventor of about 30 pending patents. Her current research interests include wireless communications, IP networking, multimedia, P2P overlay, and wireless security.

Dr. Zhang is the Associate Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, the IEEE TRANSACTIONS ON MULTIMEDIA, *Computer Networks*, and *Computer Communications*. She has also served as a Guest Editor of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the IEEE WIRELESS COMMUNICATIONS, *ACM/Springer Journal of Mobile Networks and Applications* (MONET), and *Computer Networks*. She is the Vice Chair of the Multimedia Communication Technical Committee of the IEEE Communications Society. She was a recipient of the TR 100 (MIT Technology Review) World's Top Young Innovator Award in 2004. She also received the Best Asia Pacific Young Researcher Award from the IEEE Communication Society in 2005, the Best Paper Award from the Multimedia Technical Committee of the IEEE Communications Society, the Best Paper Award in QShine 2006, and the Overseas Young Investigator Award from the National Natural Science Foundation of China in 2006.

Xuemin (Sherman) Shen (M'97–SM'02) received the B.Sc. degree from Dalian Maritime University, Dalian, China, in 1982 and the M.Sc. and Ph.D. degrees from Rutgers University, New Brunswick, NJ, in 1987 and 1990, respectively, all in electrical engineering.

He is currently with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada, where he is a Professor and the Associate Chair for Graduate Studies. His research focuses on mobility and resource management in interconnected wireless/wireline networks, UWB wireless communications systems, wireless security, and *ad hoc* and sensor networks. He is a coauthor of three books and has published more than 200 papers and book chapters on wireless communications and networks, control, and filtering.

Prof. Shen was a Technical Cochair for the IEEE GLOBECOM'03, ISPAN'04, QShine'05, IEEE Broadnets'05, and WirelessCom'05, and is the Special Track Chair of the 2005 IFIP Networking Conference. He serves as Associate Editor for the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, *Computer Networks*, *ACM/Wireless Networks*, *Wireless Communications and Mobile Computing (Wiley)*, and the *International Journal of Computer and Applications*. He has also served as Guest Editor for the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, the *IEEE Wireless Communications*, and the *IEEE Communications Magazine*. He received the Premier's Research Excellence Award in 2003 from the Province of Ontario, Canada, for demonstrated excellence of scientific and academic contributions and the Distinguished Performance Award in 2002 and 2004 from the University of Waterloo for outstanding contributions in teaching, scholarship, and service.