Achieving MAC-Layer Fairness in CSMA/CA Networks

Ying Jian, Ming Zhang, *Student Member, IEEE*, and Shigang Chen, *Member, IEEE*

***Abstract*—We demonstrate that CSMA/CA networks, including IEEE 802.11 networks, exhibit severe fairness problem in many scenarios, where some hosts obtain most of the channel's bandwidth while others starve. Most existing solutions require nodes to overhear transmissions made by contending nodes and, based on the overheard information, adjust local rates to achieve fairness among all contending links. Their underlying assumption is that transmissions made by contending nodes can be overheard. However, this assumption holds only when the transmission range is equal to the interference range, which is not true in reality. As our study reveals, the overhearing-based solutions, as well as several nonoverhearing AIMD solutions, cannot achieve MAC-layer fairness in various settings. We propose a new rate control protocol, called Proportional Increase Synchronized multiplicative Decrease (PISD). Without relying on overhearing, it provides fairness in CSMA/CA networks, particularly IEEE 802.11 networks, by using only local information and performing localized operations. It combines several novel rate control mechanisms, including synchronized multiplicative decrease, proportional increase, and background transmission. We prove that PISD converges and achieves (weighted) fairness. We further introduce Queue Spreading (QS) to achieve MAC-layer fairness when there are multiple contention groups, in which case PISD will fail.**

***Index Terms*—CSMA/CA, MAC-layer fairness, wireless LAN (WLAN).**

I. Introduction

WHEN wireless hosts share the same communication channel, they should be given a fair chance of accessing the wireless medium. *Fairness* is one of the core problems that any MAC protocol must address. It prevents the situation that some hosts obtain most of the channel's bandwidth while others starve. A more general problem is *weighted fairness*, where the channel's bandwidth obtained by a host is proportional to its weight, which is assigned by the user based on application requirements. For example, when a Web server and a client host share the same local channel (e.g., in a WLAN), the server may be given a higher weight because it may have to upload content to multiple users on the Internet simultaneously.

Random backoff in the IEEE 802.11 DCF achieves fairness in a WLAN where all hosts are downloading content from the Internet via the same access point. However, as we demonstrate in this paper, it cannot achieve fairness (let alone weighted fairness) in many other scenarios. For example, when a server that uploads content to the Internet shares the access point of a WLAN with a client host that downloads, the client may obtain most of the channel's bandwidth while the server is slowed to crawl. When the access points at two nearby homes choose the same channel,[1] the hosts in one home may obtain most of the channel's bandwidth at the expense of the hosts in the other home. Furthermore, in any *ad hoc* deployment of 802.11 DCF links, bandwidth distribution is likely to be very skewed among those sharing a channel. We will use simulations in ns-2 to show unfairness in the above cases. IEEE 802.11e provides *qualitative* service differentiation among different categories of traffic. It does not solve the fairness problem for flows within the same category, nor does it provide quantitative service differentiation (such as weighted fairness) for flows in different categories.

The fairness problem in IEEE 802.11 networks is mostly due to the fundamental limitation of CSMA/CA, which gives preference in media access to some links over others, depending on their spatial locations. As this problem is well recognized, many fairness solutions have been proposed in the past decade [1]–[8]. They fall in two categories: overhearing-based solutions and nonoverhearing solutions.

The overhearing-based solutions require each node to monitor the activity of all contending nodes and collect their links' information (such as rate, scheduling tag, or buffer status). Based on the collected information, a node decides its own media contention policy: 1) increase/decrease minimum contention window if the local rate is above/below the average rate of all contending links [1], [2], [5]; 2) serialize transmissions among contending links based on their scheduling tags [3], [4]; or 3) emulate TDMA by computing a contention-free slotted schedule among the links [6]. The key question is how to collect the information of contending links, which may be multiple hops away. One naive approach is for each node to flood the information describing its adjacent links to all nodes within a certain number of hops. This approach is not only costly, but also flawed because, as is observed in [6], [9], hop count is not a reliable means to identify the contending relationship. Hence, in virtually all existing solutions, a node learns the information about others by overhearing. However, the overhearing approach also faces a serious problem: Contention is defined by the carrier-sensing range and the interference range, whereas overhearing is limited to the transmission range, which is much shorter. Consequently, transmissions on many contending links

Manuscript received April 12, 2010; revised October 14, 2010; accepted February 05, 2011; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor S. Lu. Date of publication March 07, 2011; date of current version October 14, 2011. This work was supported in part by the U.S. National Science Foundation under Grants CNS-0644033 and CPS-0931969.

Y. Jian is with Google, Mountain View, CA 94043 USA.

M. Zhang and S. Chen are with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville, FL 32611 USA (e-mail: sgchen@cise.ufl.edu).

Digital Object Identifier 10.1109/TNET.2011.2116124

[1]This can happen when there are more neighboring access points than the number of nonoverlapping channels.

(often the majority of them) cannot be overheard, which severely limits the effectiveness of overhearing-based solutions.

Most nonoverhearing solutions use the classical Additive Increase Multiplicative Decrease (AIMD) for rate control. Even though a node may not overhear the exact information in the transmissions made by contending nodes whose radio signal is strong enough to cause interference but too weak to decode, it can still sense the aggregate impact of interference from those transmissions by monitoring how busy the channel is, how frequently its own transmissions fail [7], [10], or how fast its local buffer is filled up. Based on such information, emulating the behavior of TCP in some sense, each node may set a threshold to decide when the channel is congested such that multiplicative decrease should be performed. This direction looks reasonable. However, our simulations in ns-2 show that AIMD fails to achieve fairness, too, not because the rationale behind AIMD is flawed, but because the interaction between AIMD and CSMA/CA neutralizes the effectiveness of AIMD. AIMD may also be applied to the contention window based on the number of idle slots between two transmissions [8], [11] (which can be measured through carrier sense instead of overhearing). We will show that this approach also has limitations.

Our paper reveals the fundamental reasons for exactly why the existing fairness solutions do not work under realistic contention conditions. We demonstrate that when the channel is saturated, nodes will see different channel occupancy levels, experience different frequencies of transmission failure, and encounter different buffer lengths. Such differences in channel perception causes unsynchronized multiplicative decrease, which is the reason for the failure of AIMD in CSMA/CA networks. We introduce a number of novel rate control mechanisms. The first mechanism relies on localized operations to ensure *synchronized multiplicative decrease*. The second mechanism extends Proportional Increase Synchronized multiplicative Decrease (PISD) for weighted fairness through *proportional increase*. The third mechanism uses *background transmission* to ameliorate throughput degradation due to multiplicative decrease. Efforts are also made to simplify the implementation of the rate control mechanisms, which we believe will benefit practical systems that adopt them. Based on these new mechanisms, we propose two new rate control protocols, PISD and its enhancement, Queue Spreading (QS). PISD provides fairness in CSMA/CA networks, particularly IEEE 802.11 networks, when there is a single contention group. We perform detailed analysis on PISD and prove that it will converge and achieve (weighted) fairness. PISD also has a limitation. When it is applied in a wireless network that consists of multiple contention groups, it causes a problem called *cascaded jamming* that degrades network throughput. Our second protocol, QS, is able to solve this problem.

The rest of the paper is organized as follows. Section II discusses the related work. Section III gives the network model. Section IV describes the fairness problem. Section V proposes our PISD solution. Section VI analyzes the performance of PISD. Section VII proposes our QS solution. Section VIII presents additional simulation results. Section IX draws the conclusion.

II. Related Work

To achieve fair bandwidth distribution among contending links in a multihop wireless network, some solutions [1], [2] require every node to measure the rates of contending links through overhearing and then change its own rate by adjusting either the minimum contention window or the contention window directly. In [5], Chen and Zhang also rely on overhearing among contending nodes for appropriate distribution of channel capacity in order to achieve aggregate fairness.

Luo *et al.*'s approach [12] assigns each MAC flow a basic fair share of bandwidth and then maximizes aggregate channel utilization through spatial channel reuse. The distributed implementation requires each sender to know all contending flows (through piggybacking and overhearing) and also requires topology information to be propagated through a conflict-free spanning tree. A follow-up work [4] proposes Maximize-Local-Minimum Fair Queueing (MLM-FQ), which requires contending nodes to transmit in the order of packet service tags (representing transmission deadlines). It relies on each node keeping track of service tags at other nodes through overhearing. In Vaidya *et al.*'s earlier Distributed Fair Scheduling Protocol (DFS) [3], a node sets a backoff timer based on the finish tag of its next packet to be transmitted. DFS depends on overhearing to correctly update the local virtual clock, based on which the final tag is computed.

Xu *et al.* propose NRED [13] for improving TCP fairness in wireless networks. It relies on explicit message exchange to synchronize the detection of channel congestion among contending neighbors. It does not work in scenarios where contending nodes are outside of each other's transmission range. The paper by Pilosof *et al.* [14] is the first to study TCP fairness in 802.11 networks in the presence of both mobile senders and receivers. It identifies a throughput unfairness problem between upstream and downstream flows and gives a simple yet effective solution.

OML [6] emulates TDMA on top of CSMA/CA to implement distributed weighted fair queueing. For each of its packets, the sender must inform the contending nodes that it will participate in the time-slot competition. This information is piggybacked in the packet header and overheard by other nodes. (Alternatively, one can use control messages to flood this information to contending nodes a few hops away, which, however, causes significant overhead.)

Nandagopal *et al.* [15] propose a general analytical fairness model and a MAC protocol to approximate proportional fairness. This work assumes that links in the same contention region will experience the same loss probability, which is however not always true. As we describe in Section IV-A, the loss probabilities of two contending links can be very different, depending on the relative spatial locations of the links. Other solutions require all wireless nodes to sense the same channel condition in order to provide consistent price feedback [16], which is generally not feasible because contending wireless nodes may not be able to carrier-sense each other due to hidden terminals. To simplify the problem, much of the prior work moves away from the general CSMA/CA model adopted by 802.11 to the multichannel CDMA/FDMA model [17], the node-exclusive interference model [18], the centralized scheduling model [19], two-hop interference model [20], or CSMA without hidden terminals [21].

AIMD has been extensively studied in the past, mostly in the context of TCP. Crowcroft and Oechslin [22] modify AIMD to achieve weighted proportional fairness in TCP. As we demonstrate later, the AIMD protocols designed for CSMA/CA networks by Cai *et al.* [7], Xue *et al.* [10], and Heusse *et al.* [8], [11] can only provide fairness under certain situations. AIMD has also been used in wireless networks for congestion control [23], [24].

III. Network Model

We study the fairness problem of CSMA/CA in one or more contending WLANs, or alternatively, among single-hop, *ad hoc* wireless links. Each packet is delivered through RTS-CTS-DATA-ACK exchange. A network is modeled as a set of nodes (access points or hosts) and a set of wireless links. Each node has a transceiver. Each link supports two-way communication (for DATA/ACK exchange) between two nodes that can reliably decode each other's signal when radio interference is not present. All links transmit in the same frequency band. We model a physical network where links transmit at different frequencies as multiple orthogonal networks, each using one frequency band, and then we deal with each network separately. A link whose sender is node a and receiver is b is referred to as (a, b). Link (a, b) has a contending link (c, d) if the transmission made by c (or d) can be carrier-sensed by the sender a or causes interference at the receiver b. When the two links contend, we also say that node a has a contending node c. Generally speaking, the carrier-sensing range is greater than the interference range,[2] which is in turn greater than the transmission range. Two nodes within transmission range may be able to decode (overhear) each other's transmissions. A physical wireless link may carry zero, one, or more MAC flows. If it carries more than one MAC flow, we will model the physical link as multiple logical links, each carrying one flow. The MAC flow carried by (logical) link (a, b) is referred to as flow (a, b). Note that this paper studies single-hop flows in WLANs or *ad hoc* deployment setting. We do not consider multihop flows.

IV. Fairness Problem Remains Open in CSMA/CA Networks

A. Fairness Problem

Fairness is one of the core problems that must be addressed in any MAC design that allows contending nodes to share the same wireless medium. It requires that all wireless links of the same class have an equal right to access the communication channel and no link is starved. The random backoff algorithm in the IEEE 802.11 DCF is designed to give each host a fair chance of obtaining the channel during contention. Random backoff works fine in a symmetric environment where all hosts communicate with the same access point. However, it does not work well in asymmetric settings.

[2]Note that the carrier-sensing range can be *artificially configured*. It can be made to equal the transmission range [25]. This, however, does not alter the fact that contention goes beyond the transmission range because the interference range is not a quantity that can be artificially configured. Reducing the carrier-sensing range to be smaller than the interference range increases the chance of collision.

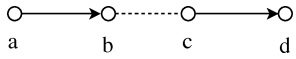

Fig. 1. Network of two flows, (a, b) and (c, d).

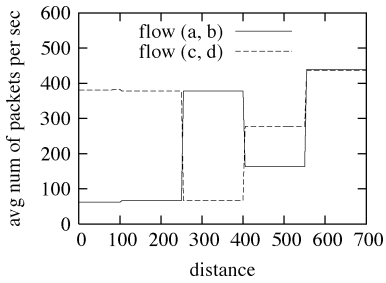

Fig. 2. Rates of two flows with respect to the distance between b and c. The distance from a to b and that from c to d are both 150 m.

An example is shown in Fig. 1, where each of the two 802.11 DCF wireless links carries a MAC-layer flow. The figure shows an ad hoc network or two nearby WLANs whose access points (a and c) each support a wireless host (b and d). When the distance between b and c is zero such that the two merge into one, this example represents one WLAN whose access point b/c supports two hosts: Node a is a server that is uploading to the Internet, and node d is a client that is downloading from the Internet. The fairness problem in the above network topology is first documented and analyzed in [26], which, however, does not consider the situations where the carrier-sensing range and interference range are greater than the transmission range.

We run simulations to study the rates of the two flows. (All simulations in this paper are performed in ns-2 v2.32 [27].) The simulation parameters are given as follows: The transmission range of the nodes is 250 m, the carrier-sensing range is 550 m, and the lengths of both links are 150 m. The transmission rate is 11 Mb/s, and the packet length is 1000 B. The parameters for the IEEE 802.11 DCF are the default values set by ns-2 according to the protocol standards.

Fig. 2 shows the average numbers of packets per second sent over the two links with respect to the distance between nodes b and c. When the distance is below 250 m, flow (c, d) obtains most of the channel's bandwidth. When the distance is between 250 and 400 m, flow (a, b) obtains most of the channel's bandwidth. When the distance is between 400 and 550 m, flow (c, d) regains the upper hand. When the distance is greater 550 m, the two links are out of each other's carrier-sensing range, and they will both obtain high bandwidth. The explanation on why such unfairness happens is given in the Appendix.

Higher-layer rate control such as TCP cannot substitute for a MAC-layer fairness solution. Suppose a TCP connection C_1 traverses link (a, b) while another connection C_2 passes (c, d). These two TCP connections compete for the same resource—the wireless channel shared by (a, b) and (c, d). Consider the scenario where the length of the wireless links is 150 m and the distance between b and c is 100 m. The simulation in ns-2 shows that C_1 is almost starved while the rate of C_2 is around 280 packets per second. The reason is that (c, d) is far more capable of obtaining the channel than (a, b) under CSMA/CA, which makes packets from C_1 prone to more drops and larger delay. For the two TCP connections to receive fair bandwidth, a MAC-layer solution must exist to distribute the channel's bandwidth fairly between (a, b) and (c, d).

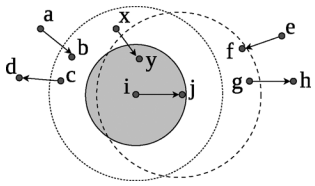

Fig. 3. There are many contending nodes that cannot be overheard by i.

B. Limitation of Overhearing-Based Solutions

Realizing the fairness problem in CSMA/CA networks, researchers have proposed numerous solutions [1]–[6], most relying on traffic information that each node collects by overhearing transmissions made by contending nodes. The most prevalent rate control scheme is to modify the random backoff algorithm such that a MAC flow that has a smaller rate than others will set a smaller backoff window and thus acquire more bandwidth [1], [2], [5]. How does a flow learn that its rate is smaller than the rates of its contending flows? The common approach requires the sender of the flow to estimate the rates of other flows by overhearing. Other rate control schemes, such as OML [6], DFS [3], and EMLM-FQ [4], also depend on overhearing (see Section II).

The problem is that overhearing is limited within the transmission range, but contention is defined by the interference range and the carrier-sensing range. Consider a wireless link (i, j) in Fig. 3, where the transmission range of the sender i is shown by the solid circle, the carrier-sensing range of i is shown by the dotted circle, and the interference range of the receiver j is shown by the dashed circle. When any node in the carrier-sensing range of i makes a transmission, i will sense a busy channel and withhold its own transmission. When any node in the interference range of the receiver j makes a transmission, it will interfere with the signal from i. In the 802.11 DCF, if j senses a busy channel before receiving an RTS, it will not return CTS. In this case, any node in the carrier-sensing range of j will interfere with the communication on (i, j). Clearly, the interference range is determined by the signal strength at the receiver, which is related to the distance between the sender i and the receiver j. The carrier-sensing range is typically set to be no less than the maximum interference range, which can be 1.78 times the transmission range as suggested in [28] (also the default value used in ns-2).

Under CSMA/CA, on one hand, (i, j) contends with any wireless link whose sender or receiver is located within the carrier-sensing range of i or the interference range of j, including (a, b), (c, d), (e, f), (g, h), and (x, y). On the other hand, the sender i can only overhear the CTS/ACK packets sent by y in the figure. Comparing the area in which contending nodes may reside (within the dotted and dashed circles or beyond such as a and e) with the shaded area from which s can overhear, it is clear that the number of contending nodes that cannot be overheard can be greater than the number of nodes that can be overheard. This seriously limits the effectiveness of any solution based on overhearing.

We implement the Huang–Bensaou protocol [2], where fairness is achieved by each node adjusting its contention window based on the overheard information of the contending flows. The simulation result for the network of Fig. 1 is shown in Fig. 4. The Huang–Bensaou protocol achieves almost perfect fairness when b and c are within the transmission range of each other (such that

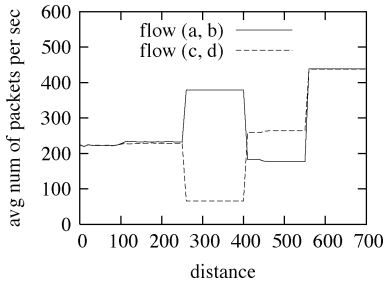

Fig. 4. In general, Huang–Bensaou protocol does not work if any one of the contending links cannot be overheard.

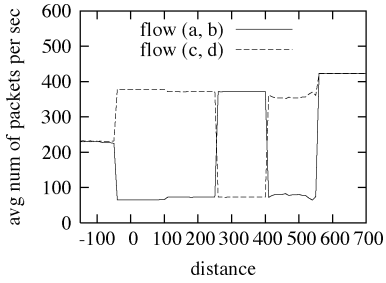

Fig. 5. AIMD: Multiplicative decrease occurs upon transmission failure.

c can overhear b's CTS/ACK). However, when the distance between b and c is beyond 250 m, the Huang–Bensaou protocol is totally ineffective. The same is true for all other schemes relying on overhearing.

C. AIMD Does not Work Either

Is there a fairness solution that allows a wireless link to adapt its rate without knowing the rates of its contending links? The classical fairness control scheme of AIMD may be first to come into mind. TCP uses AIMD (together with slow-start) to achieve approximately proportional fairness among end-to-end flows without requiring each flow to know the rates of its contending flows. AIMD has been used in multihop wireless networks for congestion control [23], [24], but there is very limited research on applying AIMD to achieve fairness among MAC flows. In the following, we will show that AIMD is ill-fitted to this purpose.

For AIMD to work, the sender of a flow must be able to detect when the channel is saturated (congested), which is the time for multiplicative decrease. There are a number of possible approaches. First, the sender may measure how busy the channel is. The channel is considered to be saturated if the fraction of time for which it is busy exceeds a certain threshold. This straightforward approach does not work. Consider the network in Fig. 1 and assume that node c is within the interference range of b, but outside the carrier-sensing range of a. In this case, even when the channel is saturated by transmissions on (c, d), node a will sense an idle channel.

Second, the sender may treat every failed transmission as a signal of channel saturation and perform multiplicative decrease [7], [10]. We simulate the AIMD protocol in [7] (the one in [10] is similar) on the network in Fig. 1, and the result is shown in Fig. 5. The protocol works fine in a WLAN environment where the links are all from a common access point to different hosts, which corresponds to the data points for distance being -150 m (such that a and c overlap to serve as the access

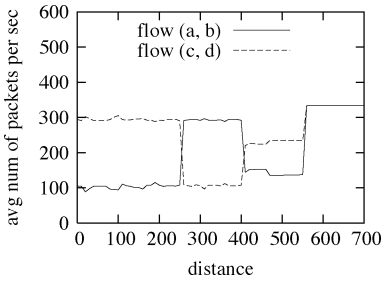

Fig. 6. AIMD: Multiplicative decrease occurs when buffer occupancy passes a certain threshold.

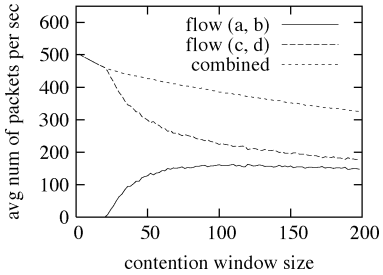

Fig. 7. Idle Sense: The same contention window size does not ensure fairness.

point while b and d are hosts.) However, it performs poorly in asymmetric settings when the distance between b and c is greater than zero.

Third, the sender may monitor its buffer occupancy. Each sender generates packets for transmission at a certain rate, which is controlled by AIMD. It signals congested channel when the buffer length exceeds a threshold. The simulation result on the network of Fig. 1 is shown in Fig. 6. Again, fairness is not achieved.

AIMD may also be used to indirectly control the flow rates. Idle Sense [8], [11] replaces DCF's random backoff by adaptively setting the same optimal size for the contention window at all hosts. It was shown in [8] that if the mean number of idle slots between two transmissions in the channel is controlled to a certain desirable value, e.g., 5.6 for 802.11b, the contention window size will be near-optimal for traffic throughput and fairness. Idle Sense makes the assumption that if the contention windows at all hosts reach the same optimal size, the hosts will send at the same rate. This assumption is true in a WLAN where all hosts can symmetrically sense one another's carriers. It is, however, not true in general for multiple contending WLANs. Consider the network of Fig. 1, where the distance between b and c is 150 m. Suppose the contention windows at the senders are set to the same size. Fig. 7 shows that the rate of flow (c, d) is far greater than that of (a, b) because the former's spatial location gives it a better chance to obtain the channel even when its contention window is the same.

V. Proportional Increase Synchronized Multiplicative Decrease

In this section, we analyze why AIMD does not work in CSMA/CA networks and propose our solution, PISD, which consists of three rate control mechanisms: synchronized multiplicative decrease, proportional increase, and background transmission. We have extensively explored alternative ways for realizing the objectives of these mechanisms and used simplicity and effectiveness as guiding selection criteria.

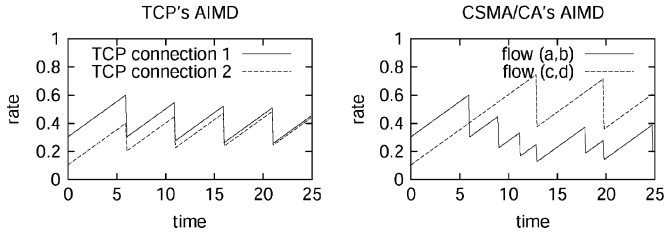

Fig. 8. *(left)* Synchronized multiplicative decrease in TCP achieves fairness. *(right)* Unsynchronized multiplicative decrease in CSMA/CA cannot achieve fairness.

A. Synchronized Multiplicative Decrease

Why does AIMD work for TCP, but not for CSMA/CA? The reason is that AIMD achieves fairness with synchronized multiplicative decrease. Contending TCP connections perform multiplicative decrease simultaneously, but that is not true for MAC flows in CSMA/CA networks.

When a router becomes congested, packet loss is felt by all TCP connections that pass the router. Hence, synchronized multiplicative decrease will be performed at the senders. We illustrate the rates of two TCP connections over time in the left plot of Fig. 8. The rates are normalized such that the congestion happens when their sum is equal to 1. Initially, the rates are different. At each multiplicative decrease, the two rates are reduced by the same percentage, and consequently the larger rate will be reduced by a larger amount, closing the gap between the two, which will eventually converge to the same value.

In a CSMA/CA network such as Fig. 1, the wireless links have different opportunities to obtain the wireless medium for transmission, depending on their spatial locations. From Fig. 2, we know that (c, d) is more capable of obtaining the medium than (a, b) when the distance between b and c is shorter than 250 m. At time 6 in the right plot of Fig. 8, when the combined rate of flows (a, b) and (c, d) reaches the channel capacity, *because* (c, d) *is able to obtain the bandwidth it needs*, node c sends all its packets out, but node a observes buffer buildup and transmission failure (with a much larger likelihood). Consequently, a detects channel congestion and performs multiplicative decrease, while c does not. Since the rate of (a, b) experiences multiplicative decrease more frequently, it will be smaller than the rate of (c, d).

B. AISD: Additive Increase Synchronized Multiplicative Decrease

In order to achieve fairness in CSMA/CA networks, we must ensure that multiplicative decrease is performed at contending senders simultaneously. We design a new protocol, called Additive Increase Synchronized multiplicative Decrease (AISD), for this purpose. There are two major problems to be solved.

The first problem is how to detect channel congestion. For each flow (i, j), the sender i stores all arrival packets in a *repository buffer* above the MAC layer. It locally maintains a time-dependent *target rate* $r_{i,j}(t)$ at which packets from the repository buffer are released to the MAC layer for transmission to the receiver j. The flow is *backlogged* if the packet arrival rate is greater than the target rate such that the repository is not empty. The target rate of a backlogged flow is additively increased over time. The actual rate at which the MAC layer sends out packets is called the *sending rate*, which is bounded by the target rate.

When the sum of the target rates of all contending flows in the channel is smaller than the capacity of the channel, all (or most) packets released by the senders to the MAC layer can be transmitted. Consequently the senders will not observe persistently growing packet queues at their MAC layer. However, additive increase will eventually improve the target rates such that their sum exceeds the channel capacity. When this happens, the flow that is least capable of competing for media access will see its packet queue growing. When the queue length passes a threshold, the sender claims that the channel is congested. Other flows that are more capable of obtaining the wireless medium may still find their queues empty. When we refer to *packet queues*, we always mean *the queues storing packets released to the MAC layer*, not the repository above the MAC layer.

The second problem is how the sender that detects channel congestion informs the contending nodes such that they can perform synchronized multiplicative decrease. One solution is for the sender and its receiver to jam the channel with a radio signal for an extended period of time. Before jamming, the contending nodes are able to transmit at decent rates (because the sum of all target rates has just passed the channel capacity for a small amount after the most recent additive increase). During jamming, they can hardly send out any packets, which gives them a clear indication that someone is jamming, and the only reason for jamming is that channel congestion has been detected. As their queue lengths exceed the threshold, they will join jamming, which provides additional assurance that all contending nodes in the channel will learn that the channel is congested. Although the jamming approach works, it wastes bandwidth. Instead of using a dedicated radio signal, a node can jam the channel with its own packets. During jamming, to ensure that the node is able to occupy the channel, we reduce its minimum congestion window to a small fraction of the default size. Besides window reduction, the jamming packets are expected to follow the same collision avoidance/resolution protocol (such as DCF) as other packets do.

The AISD protocol is summarized as follows. After each unit of time, the sender of a backlogged flow (i, j) increases its target rate by

$$r_{i,j}(t) = r_{i,j}(t-1) + \alpha. \tag{1}$$

At this rate, the sender releases packets to a queue, from which the MAC layer picks up packets for transmission. In one time unit, packets of total size $r_{i,j}(t)$ will be released. We define the *quota* as the number of bytes that remain available for transmission in the current time unit, which is equal to $r_{i,j}(t)$ minus the number of bytes that have been transmitted during the current time unit.

When the packet queue at node i for link (i, j) exceeds a threshold length, i claims that the channel is congested and jams the channel immediately. If the quota for the current time unit is sufficiently large, it jams for the rest of the time unit. Otherwise, it jams for one more time unit. The jamming is performed by releasing all packets within the quota to the queue and reducing the minimum contention window to a small value. Multiplicative decrease is performed at the end of the time unit during which jamming is performed, namely

$$r_{i,j}(t) = r_{i,j}(t-1) \times (1-\beta). \tag{2}$$

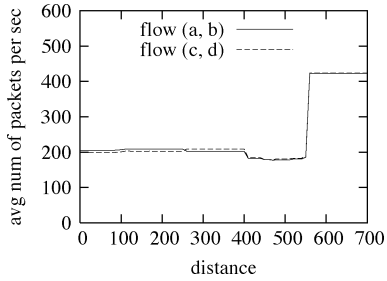

Fig. 9. Synchronized multiplicative decrease equalizes the flow rates for the network in Fig. 1.

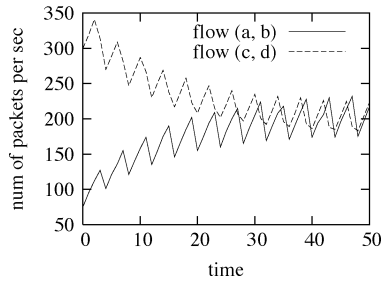

Fig. 10. Rates of two contending flows under AISD with respect to time.

As a safeguard, multiplicative decrease should not be performed for two consecutive time units.

The protocol does not require the clocks of the nodes to be synchronized. If a node is the sender for multiple flows, it performs media access and random backoff independently for each flow. Packets for different flows are queued separately.

We simulate AISD on the network of Fig. 1 with the following additional parameters: α is 5 kB/s, β is 25%, the time unit is 1 s, the queue-length threshold that triggers jamming is 10 packets, and the minimum contention window for jamming is one tenth of the default size. In the simulation, each packet is 1 kB long. We find AISD can robustly ensure synchronized multiplicative decrease. Fig. 9 shows that, using AISD, the rates of the two flows are about the same for any distance between b and c. Fig. 10 shows AISD in action over time when the distance between b and c is 100 m. At time 0, the rate of flow (c, d) is much larger. Then, AISD kicks in to equalize the two rates.

C. PISD: Proportional Increase Synchronized Multiplicative Decrease

Next, we extend AISD for weighted fairness by replacing additive increase with proportional increase. The resulting protocol is called PISD. Suppose the network administrator assigns a weight $w_{i,j}$ to each MAC flow (i, j) based on application requirements. For example, a MAC flow serving an important server should be given a higher weight than a MAC flow serving a regular client host. The problem is for the MAC layer to allocate the channel's bandwidth among contending MAC flows in proportion to their weights. This is called *weighted fairness*.

The PISD protocol is similar to AISD except for how the target rates are increased: After each unit of time, the sender of flow (i, j) increases its target rate by

$$r_{i,j}(t) = r_{i,j}(t-1) + \alpha w_{i,j}. \tag{3}$$

The rest of the protocol is the same as AISD. We prove in Section VI that PISD achieves weighted fairness.

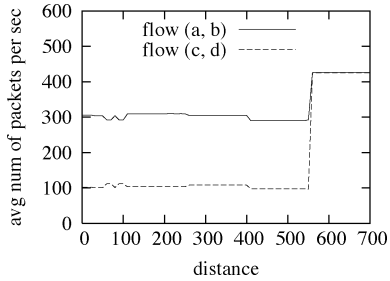

Fig. 11. Given $w_{a,b} = 3$ and $w_{c,d} = 1$, under PISD, the rate of flow (a, b) is about three times that of flow (c, d).

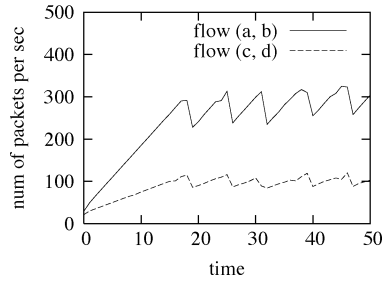

Fig. 12. Rates of two contending flows under PISD with respect to time. $w_{a,b} = 3$, $w_{c,d} = 1$, and the distance from b to c is 100 m.

We again simulate PISD on the network in Fig. 1. We assign $w_{a,b} = 3$ and $w_{c,d} = 1$, and the result is shown in Fig. 11. Weighted fairness is achieved. Fig. 12 shows the rates of the two flows with respect to time when the distance between b and c is 100 m. Clearly, flow (a, b) achieves three times the rate of flow (c, d) because it increases the rate at three times the speed of the latter.

D. PISD With Background Transmission

Using PISD, CSMA/CA will not fully utilize the channel capacity right after multiplicative decrease. It can be easily shown that, in theory, the average rate of a flow is smaller than the optimal value by a fraction no more than $\frac{\beta}{2}$ (see Section VI). If $\beta = 25\%$, then the fraction is 12.5%. However, in our simulations, the degradation is mostly around 5% and sometimes up to 10%. No matter what the degradation may be, we augment PISD with a new technique, *background transmission*, which will utilize the unused channel bandwidth for packet transmission.

Right before each multiplicative decrease, the sum of the target rates at all contending nodes exceeds the channel capacity by a small amount. After the target rates are multiplicatively decreased, their sum is below the channel capacity by a fraction of β at most. For each flow (i, j), the sender i remembers its target rate right before the most recent multiplicative decrease. This rate is called the *background rate*, which stays the same until the next multiplicative decrease. The idea is that we want to ensure that all senders are able to transmit at their target rates and, if there is extra channel bandwidth, we allow the senders to compete for additional transmissions up to their background rates. When a node's sending rate is above its target rate, its transmission is called a *background transmission*. When the sending rate is below the target rate, its transmission is called a *regular transmission*. When a regular transmission of one node contends with a background transmission of another, the former should be given priority. To achieve such differentiation, we increase the minimum contention window for background transmission.

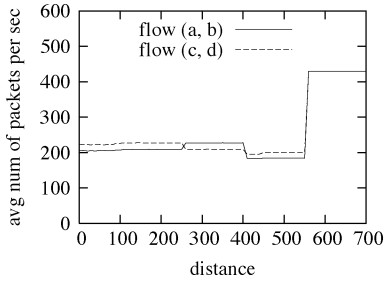

Fig. 13. Background transmission will utilize some unused channel bandwidth for packet transmission.

The PISD protocol with background transmission is described as follows. Proportional increase synchronized multiplicative decrease is performed on the target rate as usual. However, a sender i releases packets to the MAC layer at the background rate (the rate before the last multiplicative decrease). The node also keeps track of the number n_{t} of bytes that would have been released at the target rate. Let δ be the time that has elapsed in the current time unit. $n_{\mathrm{t}} = r_{i,j}(t) \times \delta$. Let n_{s} be the number of bytes that has been delivered to the receiver in the current time unit. When n_{s} is equal to or greater than n_{t}, the sender knows that it is now making background transmissions, and therefore it increases the minimum contention window. When n_{s} becomes smaller than n_{t}, the sender changes its minimum contention window back.

We simulate PISD with background transmission on the network in Fig. 1 with a minimum contention window for background transmission twice the size of the default minimum contention window for regular transmission. (The default minimum contention window is set based on the standard of 802.11 DCF.) Fig. 13 shows that the flows pick up the extra bandwidth left by PISD for additional packet transmission. This extra bandwidth, representing only a small fraction of channel capacity, is not regulated by proportional increase multiplicative decrease, and consequently it is unevenly distributed between the flows based on the IEEE 802.11 DCF.

VI. Analysis

Much work about AIMD has been performed in the context of TCP. In this section, we analyze PISD and show that it achieves weighted fairness after convergence. More importantly, we derive the convergence time and the channel coverage with respect to α and β and reveal the performance tradeoff that can be made by changing these two parameters.

A. Weighted Fairness and Convergence Time

Consider a set L of MAC flows that contends in the same wireless channel whose effective capacity is C. When the sum of the target rates of all flows is below the channel capacity, the channel will be able to deliver the packets of the flows, and the senders will proportionally increase their target rates. Once the sum exceeds the channel capacity, the senders will immediately decrease their target rates multiplicatively. PISD performs the following rate control:

$$r_{i,j}(t+1) = \begin{cases} r_{i,j}(t) + \alpha w_{i,j}, & \text{if } \sum_{(i,j)\in L} r_{i,j}(t) \le C \\ r_{i,j}(t)(1-\beta), & \text{if } \sum_{(i,j)\in L} r_{i,j}(t) > C. \end{cases}$$

We derive how much time it takes PISD to converge such that the rates of the flows are stabilized and proportional to their weights. Our results show that the convergence time is a decreasing function of both α and β.

When multiplicative decrease happens, even if the combined target rate of all flows may be greater than the channel capacity, it will be greater only by a small amount due to the nature of additive increase. To simplify the analysis, we treat them as equal. A *PISD period*, denoted as P, is defined as the time between two consecutive multiplicative decreases. We derive the value of P as follows. Consider an arbitrary multiplicative decrease, which is triggered when $\sum_{(i,j)\in L} r(i,j)(t) = C$. It reduces all target rates by a fraction of β, and hence leaves βC of channel capacity unused. The proportional increase improves the combined rate of all flows at a speed of αW, where $W = \sum_{(i,j)\in L} w_{i,j}$. After a period P, the combined rate should be increased by βC in order to make the channel saturated again and cause the next multiplicative decrease. Since $P \times \alpha W = \beta C$, we have

$$P = \frac{\beta C}{\alpha W}.$$

Without loss of generality, for $l = 0, 1, 2, \ldots$, let $t = lP$ be the time units right before multiplicative decrease, and $t = lP + 1$ be the time units after multiplicative decrease. Multiplicative decrease occurs at the time instant between lP and $lP+1$. Given arbitrary values for $r_{i,j}(0), \forall (i,j) \in L$, we show that $r_{i,j}(t)$ will converge toward a value that is proportional to $w_{i,j}$ as t increases.

First, we determine the value of $r_{i,j}(lP)$. During each PISD period, the target rate is first multiplicatively decreased and then proportionally increased. Hence, for $l > 0$

$$r_{i,j}(lP) = r_{i,j}((l-1)P) \times (1-\beta) + \alpha w_{i,j} \times P.$$

By induction over the above iterative formula, we have

$$r_{i,j}(lP) = C\frac{w_{i,j}}{W} + \left(r_{i,j}(0) - C\frac{w_{i,j}}{W}\right)(1-\beta)^l.$$

The second term on the right side diminishes to zero when l becomes large. Hence, $r_{i,j}(lP)$ converges to

$$r^*_{i,j} = C\frac{w_{i,j}}{W}.$$

Next, we determine the value of $r_{i,j}(t)$, $t \le lP$, for $l = 0, 1, 2, \ldots$. The rate is multiplicatively decreased immediately after time $\lfloor t/P \rfloor P$ and then proportionally increased. We have

$$\begin{aligned} r_{i,j}(t) &= r_{i,j}(\lfloor t/P \rfloor P) \times (1-\beta) + \alpha w_{i,j} \times (t \bmod P) \\ &= C\frac{w_{i,j}}{W}(1-\beta) + \left(r_{i,j}(0) - C\frac{w_{i,j}}{W}\right)(1-\beta)^{\lfloor t/P \rfloor + 1} \\ &\quad + \alpha w_{i,j} \times (t \bmod P). \end{aligned}$$

As t increases, the second term on the right side diminishes to zero. Hence, $r_{i,j}(t)$ converges to the following curve:

$$r^*_{i,j}(t) = C\frac{w_{i,j}}{W}(1-\beta) + \alpha w_{i,j} \times (t \bmod P)$$

which is independent of the initial value $r_{i,j}(0)$. The average of $r_{i,j}(t)$ over the l period, denoted as $A_{i,j}(l)$, is given as

$$\begin{aligned} A_{i,j}(l) &= \frac{\left(\sum_{(l-1)P<t<lP} r_{i,j}(t)\right) + r(lP)}{P} \\ &= C\frac{w_{i,j}}{W}\left(1 - \frac{\beta}{2}\right) + \frac{\alpha w_{i,j}}{2} \\ &\quad + \left(r_{i,j}(0) - C\frac{w_{i,j}}{W}\right)(1-\beta)^l. \end{aligned}$$

The third term on the right side diminishes to zero when l increases. Hence, the average rate converges to

$$A^*_{i,j} = C\frac{w_{i,j}}{W}\left(1 - \frac{\beta}{2}\right) + \frac{\alpha w_{i,j}}{2}$$

which is proportional to the weight $w_{i,j}$.

We define the *convergence time* as the time it takes for $A_{i,j}(l)$ to be ε-*close* to its target $A^*_{i,j}$. The ε-*closeness* is defined as follows:

$$\frac{\left|A_{i,j}(l) - A^*_{i,j}\right|}{A^*_{i,j}} \le \varepsilon.$$

We derive the lower bound of l that can satisfy the above inequality. Note that $r_{i,j}(0) \le C$

$$\begin{aligned} l &\ge \log_{1-\beta} \frac{\varepsilon\left(C\frac{w_{i,j}}{W}\left(1-\frac{\beta}{2}\right) + \frac{\alpha w_{i,j}}{2}\right)}{\left|r_{i,j}(0) - C\frac{w_{i,j}}{W}\right|} \\ &\ge \log_{1-\beta} \frac{\varepsilon\left(C\frac{w_{i,j}}{W}\left(1-\frac{\beta}{2}\right) + \frac{\alpha w_{i,j}}{2}\right)}{\left|C - C\frac{w_{i,j}}{W}\right|}. \end{aligned}$$

The time for l periods is $t = lP$. Hence, the convergence time is

$$t \ge \frac{\beta C}{\alpha W} \log_{1-\beta} \frac{\varepsilon\left(C\frac{w_{i,j}}{W}\left(1-\frac{\beta}{2}\right) + \frac{\alpha w_{i,j}}{2}\right)}{\left|C - C\frac{w_{i,j}}{W}\right|}.$$

From the above formula we see that, the convergence time is a decreasing function for both $\alpha (\alpha > 0)$ and $\beta (\beta \in (0,1))$. In other words, the larger the value of α (or β) is, the faster the convergence is.

B. Channel Coverage

We study how much bandwidth is regulated (or *covered*) by PISD. The channel bandwidth covered by PISD is distributed to the flows in proportion to their weights. The channel bandwidth not covered by PISD is arbitrarily distributed to flows through background transmission. Formally, the *channel coverage*, denoted as Cov, is defined as the sum of the average target rates of the flows after PISD fully converges divided by the channel capacity.

$$\text{Cov} = \frac{\sum_{(i,j)\in L} A^*_{i,j}}{C} = 1 - \frac{\beta}{2} + \frac{\alpha W}{2C}.$$

We know that $\frac{\alpha W}{2C} = \frac{\beta}{2P}$, where P is the PISD period that is greater than 1. Hence, the channel coverage is mainly controlled by β. The smaller the value of β is, the more the channel bandwidth PISD controls. Simulation results on the performance tradeoff made via α and β can be found in our earlier work [29].

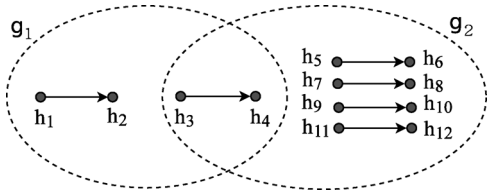

Fig. 14. Network with two contention groups.

VII. Queue Spreading

Although PISD works well in most scenarios, its performance can be significantly degraded when applied in networks consisting of multiple contention groups. We propose another protocol, called QS, to remedy the problem.

A. Cascaded Jamming

A contention group is defined as a maximal group of mutually contending flows. In Fig. 14, the flows form two contention groups, g_1 and g_2. In each contention group, the constituent flows mutually contend, and thus only one flow can transmit at a time. However, flows in different contention groups may be able to transmit simultaneously. For instance, (h_1, h_2) can transmit while (h_5, h_6) is transmitting because they do not contend. This is called *spatial channel reuse*.

Suppose g_2 is congested, but g_1 is not, meaning that the aggregate rate on flows (h_3, h_4) through (h_{11}, h_{12}) exceeds the channel capacity, while the aggregate rate on (h_1, h_2) and (h_3, h_4) is below the channel capacity. Ideally, multiplicative decrease should be performed on flows (h_3, h_4) through (h_{11}, h_{12}), but not on (h_1, h_2) because its local channel still has residual bandwidth. However, under PISD, when h_3 detects congestion (due to the channel saturation in g_2), it will perform jamming. During jamming, the queue at h_1 will build up and exceed the threshold. Hence, h_1 will falsely detect congestion. Worse yet, h_1 will perform jamming,[3] which in turn causes the nodes in its neighborhood (not drawn in the figure) to falsely detect congestion and join jamming. As this process repeats, all nodes that are transitively reachable from g_2 through the contention relationship will falsely detect congestion and perform multiplicative decrease. We name this problem as *cascaded jamming*.

Imagine a number of WLANs are deployed in a certain region (e.g., a city), and they partially overlap one another to provide a full coverage. Most areas have light communication traffic, but there is a hotspot (e.g., a public library) where many wireless users cause frequent channel congestion. The problem is that any jamming—which is supposed to be performed only by the nodes at the hotspot during congestion for synchronized multiplicative decrease—may propagate out to the entire region. All nodes in the region will end up performing multiplicative decrease. This phenomenon is captured by the simulation results in Table I. Flow (h_1, h_2), which is outside of the hotspot g_2, should have a higher rate. However, PISD not only equalizes the rates of the flows in g_2, but also reduces the rates of other flows to the same level.

[3]Node h_1 does not know that h_3 is jamming. It cannot make a query to h_3 either because h_3 may reside outside its transmission range or even outside the transmission range of h_2. Node h_1 only knows that the channel is saturated (falsely in this case), but does not know whether it is the first one to detect so. It has to dutifully perform jamming to ensure the correctness of PISD in synchronized multiplicative decrease.

TABLE I
PISD's Performance Under Spatial Channel Reuse

Flow	Rate	Flow	Rate
(h_1, h_2)	74.0	(h_3, h_4)	73.9
(h_5, h_6)	73.5	(h_7, h_8)	73.6
(h_9, h_{10})	73.9	(h_{11}, h_{12})	74.0

B. QS

PISD performs channel jamming, which solves the fairness problem for an isolated contention group. However, it causes the cascaded jamming problem. To solve this problem, we have to abandon channel jamming and invent new techniques to guarantee that when a contention group's channel is saturated, all nodes in the group will detect congestion, while the nodes outside of the group will not falsely do so. Our solution is called *queue spreading*, which is described as follows.

Each node additively increases its target rate (at which packets are released to the MAC layer) until its MAC queue length exceeds a threshold H. When that happens, the node multiplicatively decreases the target rate. The key issue is to make sure that all those and only those in a congested contention group perform multiplicative decrease.

The aggregate target rate $R_g(t)$ of all flows in a contention group g is a function of time. Let C be the maximum throughput that the group can possibly obtain from the channel at time t. We stress that C is only needed to describe our idea. The operation of queue spreading does not rely on the knowledge of C. When $R_g(t)$ exceeds C, we say the group is congested. In this case, if we look at the flows as a whole, there are more packets released to the MAC layer than it can send out. The excess packets increase the queue lengths of the flows at a combined rate of $R_g(t) - C$. The problem is that, depending on the flows' relative locations, most excess packets may be queued up at one flow that is least capable of accessing the medium. While that flow observes that its queue length exceeds the threshold H and performs multiplicative decrease, other flows that are more capable of obtaining medium may still find their queues empty and thus continue with additive increase, which will enlarge the gap among the flow rates and result in worse unfairness.

One solution to the above problem is to make sure that excess packets are *spread* among the *queues* of all flows in the group. For an arbitrary flow (u, v), whenever the packet queue at the sender u exceeds H, u will temporarily modify its MAC parameters to increase its ability of obtaining the medium, such that its queue length can be reduced back to H. When the queue length becomes H, the node will restore the original MAC parameters. The idea behind queue spreading is very intuitive: After a node detects congestion, the node will keep its queue length at H by dynamically adjusting its MAC parameters. Because its queue no longer grows, the excess packets in the channel will have to be buffered elsewhere, pushing the queues at other nodes up. Once their queues reach the threshold, they will do the same thing. Excess packets will always be pushed to the nodes that have not detected congestion yet.

A node that performs channel jamming in PISD tries to *occupy the channel as much as possible* (which is why it is

called "jamming"), and consequently it will affect all neighboring nodes, including those outside of the congested group. On the contrary, a node that performs queue spreading only tries to *match its sending rate with its target rate* such that the local queue does not grow further. Therefore, it will not affect the neighbors outside of the congested group. For example, in Fig. 14, if h_3 jams the channel due to the congestion in g_2, h_1 will feel the jamming and cannot send out packets. However, if h_3 simply tries to send out all packets currently released to its MAC layer, h_1 will not be affected if g_1 is not congested because the channel shared by (h_1, h_2) and (h_3, h_4) has enough bandwidth for their combined target rate. The following proposition provides guidance for our protocol design.

Proposition 1: The senders of all flows in a contention group g will detect congestion by the end of a time period $[t_0, t_1)$ if the following conditions are satisfied: 1) $R_g(t) - C > 0, \forall t \in [t_0, t_1)$; 2) $\int_{t_0}^{t_1} (R_g(t) - C)dt \geq |g| \cdot H$; and 3) queue spreading is performed.

Proof: To prove by contradiction, we assume that a subset of flows, $g' \subset g$, does not detect congestion by the end of the period. On one hand, their packet queues are shorter than H. Thus, the total number of packets in their queues is less than $|g'| \cdot H$. On the other hand, by performing the queue spreading, the flows in $g - g'$ are more capable of obtaining media access than those in g'. Hence, they can control their queue lengths to be H at the expense of the flows in g', whose queues are forced to grow. The total number of packets queued at the flows in $g - g'$ is $|g - g'| \cdot H$. During the time period $[t_0, t_1)$, by Condition 1), the total number of excess packets that are queued by all flows is $\int_{t_0}^{t_1} (R_g(t) - C)dt$. Therefore, the number of excess packets that are queued at the flows in g' must be $\int_{t_0}^{t_1} (R_g(t) - C)dt - |g - g'| \cdot H$. By Condition 2), this number is no less than $|g'| \cdot H$, leading to the contradiction. □

C. Implementation of Queue Spreading

Without channel jamming and the problem it brings, we show that a protocol based on QS can also ensure synchronized multiplicative decrease. However, a straightforward combination of AIMD and QS will not do the trick. The first protocol, called AIMD/QS, is described as follows. Consider an arbitrary flow (u, v). To implement AIMD, the sender u adapts its target rate after each time period of T.[4] If its queue length at the MAC layer does not reach the threshold H during the last period, it increases the target rate for a constant amount α. Otherwise, it decreases the rate for a constant percentage β. To implement QS, whenever the queue length exceeds the threshold H, the sender aggressively reduces its minimum contention window to a small fraction of the default size in order to ensure that it has the priority to occupy the channel. Once the queue length is reduced to H, the sender restores the default minimum contention window. By doing so, it keeps the queue length at H.

We use an example to show that AIMD/QS cannot always ensure synchronized multiplicative decrease. Consider a contention group g. Suppose additive increase happens at time t_0 such that $R_g(t_0) > C$. Before time t_0, $R(t) < C$ and all flows have empty queues. If $R_g(t_0) - C$ is small, the number of excess packets in the next period, $(R_g(t_0) - C)T$, will also be small. Suppose it is greater than H, but smaller than $|g| \cdot H$. Namely, it is enough to push the queue at one node over the threshold, but not enough to do so for all. In this case, the node that is least capable of obtaining the channel will detect congestion and perform multiplicative decrease. However, some other nodes will not do so. Instead, they will take advantage of the bandwidth released by multiplicative decrease and continue performing additive increase.

To solve this problem and make sure that synchronized multiplicative decrease is always achieved, we generalize the above base protocol to AIMD/QS+k, where k is a nonnegative integer. The sender of each flow performs the operations of AIMD/QS, except that after it finds its queue length reaches H, it will continue performing additive increase for k subsequent periods of T before making multiplicative decrease. Suppose a flow's queue reaches H during $[t_0, t_0 + T)$. It will increase the target rate at times $t_0 + T$, $t_0 + 2T, \ldots, t_0 + kT$ by a constant amount α, and then decrease the target rate at time $t_0 + (k+1)T$ by a percentage β. The idea is to make sure that there will be enough excess packets to allow all nodes to detect congestion. We have the following proposition.

Proposition 2: Suppose the senders of all flows in a contention group always adapt their target rates simultaneously. AIMD/QS+k ensures the detection of congestion by all flows in the group if

$$\frac{k(k+1)}{2}\alpha T \geq H.$$

Proof: Consider an arbitrary time at which an update on the flows' target rates is performed. Without losing generality, let this time be $t = 0$. Let g be the first congested group after $t = 0$. Once g is congested, excess packets will eventually push the queue length of a flow in g over the threshold H. Let (u, v) be the first flow whose length reaches H, and without losing generality, suppose it happens during $(iT, (i+1)T]$. We have $R_g(iT) > C$. Otherwise, there would be no excess packets to push the queue length of (u, v) to the threshold.

Because flow (u, v) is the first one to perform multiplicative decrease and that happens at time $t = (i + k + 1)T$, all flows perform additive increase at times $t = (i+j)T$ for $j \in [1 \ldots k]$. Because the rate of each flow in g increases by α after each period T, we must have $R_g((i+j)T) = R_g(iT) + j|g|\alpha$. Hence

$$\begin{aligned}
&\int_{iT}^{(i+k+1)T} (R_g(t) - C)\, dt \\
&\quad = \sum_{j=0}^{k} (R_g((i+j)T) - C)\, T \\
&\quad = \sum_{j=0}^{k} (R_g(iT) - C)\, T + \sum_{j=0}^{k} j|g|\alpha T \\
&\quad > \sum_{j=0}^{k} j|g|\alpha T = \frac{k(k+1)}{2}|g|\alpha T \geq |g| \cdot H. \qquad (4)
\end{aligned}$$

Hence, by Proposition 1, AIMD/QS+k makes sure that all flows in g detect the congestion before time $(i + k + 1)T$ and will perform multiplicative decrease before time $(i + 2k + 1)T$ in case that some flows first see their queues reach H during $((i + k)T, (i + k + 1)T]$. □

[4]Alternatively, we may simply refer to T as one unit of time as we did in PISD.

TABLE II
AIMD/QS+2'S PERFORMANCE UNDER SPATIAL CHANNEL REUSE

Flow	Rate	Flow	Rate
(h_1, h_2)	280.3	(h_3, h_4)	59.1
(h_5, h_6)	80.7	(h_7, h_8)	79.6
(h_9, h_{10})	80.2	(h_{11}, h_{12})	80.3

This result requires the clocks of all nodes to be synchronized. If we remove this requirement and allow the nodes to adapt their rates at different times, we have the following proposition.

Proposition 3: Suppose the senders of the flows in a contention group are allowed to update their target rates at different times. AIMD/QS+k ensures the detection of congestion by all flows in the group if

$$\frac{k(k-1)}{2}\alpha T \geq H.$$

Proof: The proof is similar to that for Proposition 2 except that, because the flows in g may perform additive increase at different times, $R_g(t)$ is not a constant for $t \in (iT, (i+1)T]$ during which the flow (u, v)'s queue length reaches H. Hence, $R_g(iT) > C$ may not be true. However, $R_g((i+1)T) > C$ must be true because by then the congestion has been detected. We also have $R_g((i+1+j)T) = R_g((i+1)T) + j|g|\alpha$ because the flow rates are each increased by α after each period of T even though the increase may happen at different times during the period. Through a process similar to (4), we can derive $\int_{(i+1)T}^{(i+k+1)T}(R_g(t) - C)dt \geq |g| \cdot H$. Hence, by Proposition 1, AIMD/QS+k makes sure that all flows in g detect the congestion. □

The above proposition provides theoretical guidance when we select system parameters. They must meet the condition of $\frac{k(k-1)}{2}\alpha T \geq H$ in order to ensure synchronized multiplicative decrease. In particular, when $k = 2$, the condition becomes $\alpha T \geq H$. We simulate AIMD/QS+2 on the network of Fig. 14 with the same parameters that we have used on PISD: $\alpha = 5$ kB/s, $\beta = 25\%$, $T = 1$ s, $H = 10$, and each packet is 1 kB long. They satisfy the condition. The simulation results are presented in Table II. When compared to Table I, the rate of (h_1, h_2) is significantly improved.

VIII. ADDITIONAL SIMULATIONS

We perform additional simulations under several different scenarios to evaluate the effectiveness of the proposed PISD and QS protocols. All simulations in this paper are performed using ns-2. PISD is implemented on top of the IEEE 802.11 DCF. *If not specified otherwise, the simulation parameters are the same as those in Sections IV-A, V-B, and VII-C*. The parameters for 802.11 DCF use the default values set by ns-2 according to the protocol standards. We let $\alpha = 2$ kB/s and $\beta = 25\%$. By default, background transmission is turned off.

Our first simulation scenario consists of four access points and 10 hosts. The access points are located at the corners of a 380×380 m^2 square. The distance from the hosts to their access points varies from 70 to 150 m. Their relative positions are shown in Fig. 15. The rates of the flows under PISD, PISD with background transmission (PISD-b), AIMD/QS+2, DCF, the Huang–Bensaou protocol (H.-B.) are shown in Table III. PISD is able to achieve fairness, while the DCF and the Huang–Bensaou protocol cannot in this scenario. AIMD/QS+k achieves comparable performance as PISD. In a later scenario, we will show that QS can perform much better than PISD when PISD causes cascaded jamming. The average flow rate of 802.11 DCF is slightly higher due to a well-known fact of throughput/fairness tradeoff in wireless networks [12]. By starving heavily contended flows, it creates more opportunity of channel spatial reuse for other flows.

TABLE III
FLOW RATES ACHIEVED BY DIFFERENT PROTOCOLS

flow	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	avg
PISD	42.7	43.2	43.7	43.7	43.4	42.8	43.7	43.7	43.7	43.6	43.4
PISD-b	43.0	45.6	48.1	43.0	48.8	46.2	45.9	43.4	46.1	46.0	45.6
AIMD/QS+2	39.1	46.7	44.5	46.7	47.3	41.7	42.4	41.5	45.2	47.2	44.2
DCF	74.7	55.1	99.7	51.4	51.4	15.3	15.3	15.3	40.8	40.8	46.0
H.-B.	18.6	17.2	36.8	35.5	35.5	47.5	47.5	47.5	84.0	84.0	45.4

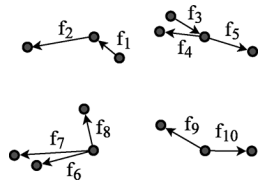

Fig. 15. 4-WLAN network topology.

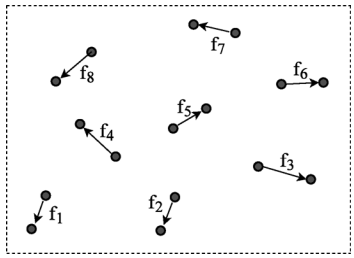

Fig. 16. Ad hoc network topology.

Our next simulation scenario is an ad hoc network shown in Fig. 16, where visitors to a commercial conference download information from exhibit booths to their laptops via direct wireless links that share the same channel. The size of the area is 400×600 m^2, and the nodes are plotted in the area based on their assigned coordinates. The topology is randomly generated by first placing the senders at random locations, and then placing the receivers at random locations near their senders. The simulation results are shown in Table IV. Each row except for the last one contains one weight assignment and the corresponding flow rate achieved by PISD. The results demonstrate the great flexibility and *quantitative precision* that PISD is able to bring into CSMA/CA networks. The last row shows that 802.11 DCF causes unfairness, where f_2 is almost starved.

The final simulation is performed on a network of four neighboring WLANs in Fig. 17. The WLANs form three contention groups: Home 1 and Home 2, Home 2 and Home 3, and Home 3 and Coffee house. We show that, under PISD, the network's throughput suffers from the cascaded jamming problem, while QS is able to overcome this problem. The three homes in the figure each have an access point (A_1, A_2, or A_3) and a host (h_1, h_2, or h_3). The coffee house provides wireless access (A_4) to its customers (h_4–h_9). Assume that interference exists and

TABLE IV
UNDER DIFFERENT WEIGHT ASSIGNMENTS, FLOW RATES ARE ALWAYS PROPORTIONAL TO FLOW WEIGHTS

flow	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8
weight	1	1	1	1	1	1	1	1
rate	53.3	51.9	53.2	53.3	53.1	53.4	53.3	53.0
weight	1	2	1	1	1	1	2	1
rate	43.4	82.1	41.6	43.4	41.7	41.7	82.5	43.4
weight	1	2	1	1	2	1	2	1
rate	38.9	75.2	38.6	38.9	77.2	38.9	77.31	38.7
weight	1	1	1	2	1	2	1	1
rate	43.6	41.2	42.9	86.7	43.4	86.6	43.1	43.3
weight	2	1	3	1	1	1	1	2
rate	72.4	35.4	105.5	35.8	36.2	36.3	36.0	72.3
weight	1	2	1	4	1	4	1	1
rate	28.4	55.5	30.9	112.8	30.9	112.6	30.8	27.8
DCF	97.9	5.1	42.1	92.7	55.2	72.1	48.3	54.7

TABLE V
FLOW RATES IN FOUR NEIGHBORING WLANS

Flow	DCF	P-fair	PISD	PISD-b	AIMD/QS+2
(A_1, h_1)	79.7	225.0	85.9	139.5	188.3
(A_2, h_2)	365.4	225.0	81.1	127.7	188.9
(A_3, h_3)	14.9	64.0	54.7	54.0	59.0
(A_4, h_4)	72.3	64.0	57.8	64.0	59.0
(A_4, h_5)	73.1	64.0	57.1	62.6	60.5
(A_4, h_6)	75.5	64.0	57.3	63.0	60.3
(A_4, h_7)	75.7	64.0	56.7	63.4	61.7
(A_4, h_8)	74.4	64.0	57.9	62.2	61.2
(A_4, h_9)	73.0	64.0	56.5	63.3	62.5
avg	100.4	99.8	62.8	77.7	89.0

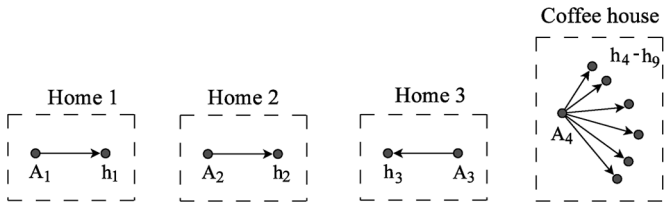

Fig. 17. Four neighboring WLANs.

only exists between any adjacent WLANs. All hosts are downloading UDP video from their access points. The bottleneck is the contention group formed by Home 3 and the coffee house. Ideally, being far away from the bottleneck, flows in the left two homes should have higher rates. Simulation results are shown in Table V. Again, IEEE 802.11 DCF has the fairness problem, where (A_3, h_3) gets a very low rate. The P-fair column shows the theoretical proportional-fairness rates that are computed numerically by solving the nonlinear programming problem in [30] for a channel capacity of 450 packets per second. In the PISD column, due to cascaded jamming, the rates of both (A_1, h_1) and (A_2, h_2) are both depressed and close to the rates of the flows in the bottleneck. When background transmission is turned on (in the PISD-b column), their rates are improved. When AIMD/QS+2 is used, their rates are further improved. Indeed, we observe that the rates under Queue Spreading nicely approximate proportional fairness. The average flow rate of 802.11 DCF is higher because when (A_3, h_3) sends fewer packets, the flows on its two sides have more chance of sending simultaneously. This increases the overall network throughput at the expense of being unfair to Home 3. The theoretical average rate of P-fair is also high because much of the protocol overhead is ignored.

IX. CONCLUSION

In this paper, we have investigated the unfairness problem in CSMA/CA networks. We show that existing solutions based on overhearing are not effective when contending nodes are outside each other's transmission range. We also show that the existing nonoverhearing AIMD solutions do not work either. We propose a new fairness protocol, PISD, which performs proportional increase synchronized multiplicative decrease with background transmission to support not only fairness, but also weighted fairness in CSMA/CA networks, including IEEE 802.11 networks. We then propose another protocol, called Queue Spreading, to achieve fairness when there are multiple contention groups. To the best of our knowledge, this is the first work that is able to achieve provable fairness in CSMA/CA networks under realistic conditions where the carrier-sensing range and the interference range can be much larger than the transmission range.

APPENDIX

We explain Fig. 1 segment by segment.

Segment 1: For distance from 0 to 100, (c, d) has a higher rate. On one hand, after flow (c, d) transmits a packet through RTS/CTS/DATA/ACK exchange, the sender c performs random backoff after DIFS, while node a counts down its backoff timer after waiting for EIFS (equal to DIFS+SIFS+ACK time). That is because it can sense d's ACK, but not understand it (out of the transmission range). On the other hand, after flow (a, b) transmits a packet, node c will only wait for DIFS instead of EIFS because it can overhear b's ACK. In the above analysis, only node a may wait for EIFS. This introduces asymmetry in the waiting time of the two links, and consequently node a has a smaller probability of obtaining the channel than node c.

Segment 2: For distance from 100 to 250, (c, d) has a higher rate. First, by the same token as explained above, after flow (c, d) transmits a packet, node a will wait for EIFS, which is caused by the ACK transmission from d that it can sense but not understand. Second, after flow (a, b) transmits a packet, node c will also wait for EIFS, which is caused by the DATA transmission from a that it cannot understand. However, the subsequent ACK transmission from b, which node c can understand, will interrupt EIFS according to the DCF standard [31]. Therefore, it remains that node a has a smaller probability of obtaining the channel than node c.

Segment 3: For distance from 250 to 400, (a, b) has a higher rate. On one hand, after flow (a, b) transmits a packet, node c will wait for EIFS, which is caused by ACK from b. This makes c less likely to obtain the channel. On the other hand, after flow (c, d) transmits a packet, node a will wait for EIFS, which is caused by DATA from c. However, this does not make a less likely to obtain the channel because the EIFS overlaps with the transmission of ACK from d. Note that $\mathrm{EIFS} = \mathrm{DIFS}+\mathrm{SIFS}+\mathrm{ACK}$. Essentially, after we subtract the time for transmitting ACK (which includes SIFS), node a waits for the regular DIFS.

Segment 4: For distance from 400 to 550, (c, d) has a higher rate. Both c and d are outside the carrier-sensing range of a, but they are in the carrier-sensing range of b. On one hand, when c is transmitting, a will sense an idle channel and attempt a

transmission to b, which is deemed to fail, causing exponential backoff and slowing down the rate of flow (a, b). On the other hand, after a and b perform RTS/CTS exchange, c will only wait for EIFS because it senses but does not understand CTS from b. Hence, when a transmits DATA, c may complete EIFS as well as its backoff timer and begin transmitting RTS, which will corrupt the reception of DATA at b.

Segment 5: For distance greater than 550, (a, b) and (c, d) have equal rates. Two flows move out of each other's carrier-sensing range. They can both send packets at the highest rate supported by the channel capacity.

REFERENCES

[1] B. Bensaou, Y. Wang, and C. Ko, "Fair medium access in 802.11 based wireless ad hoc networks," in *Proc. ACM MobiHoc*, 2000, pp. 99–106.

[2] X. Huang and B. Bensaou, "On max-min fairness and scheduling in wireless ad hoc networks: Analytical framework and implementation," in *Proc. ACM MobiHoc*, 2001, pp. 221–231.

[3] N. H. Vaidya, P. Bahl, and S. Gupta, "Distributed fair scheduling in a wireless LAN," in *Proc. ACM MobiCom*, 2000, pp. 167–178.

[4] H. Luo, J. Cheng, and S. Lu, "Self-coordinating localized fair queueing in wireless ad hoc networks," *IEEE Trans. Mobile Comp.*, vol. 3, no. 1, pp. 86–98, Jan.–Feb. 2004.

[5] S. Chen and Z. Zhang, "Localized algorithm for aggregate fairness in wireless sensor networks," in *Proc. ACM MobiCom*, 2006, pp. 274–285.

[6] A. Rao and I. Stoica, "An overlay MAC layer for 802.11 networks," in *Proc. ACM MobiSys*, 2005, pp. 135–148.

[7] S. Cai, Y. Liu, and W. Gong, "Analysis of an AIMD based collision avoidance protocol in wireless data networks," in *Proc. IEEE CDC*, 2003, vol. 1, pp. 104–109.

[8] M. Heusse, F. Rousseau, R. Guillier, and A. Duda, "Idle sense: An optimal access method for high throughput and fairness in rate diverse wireless LANs," in *Proc. ACM SIGCOMM*, 2005, pp. 121–132.

[9] Y. Yang and R. Kravets, "Throughput guarantees for multi-priority traffic in ad hoc networks," *Ad Hoc Netw. J.*, vol. 5, no. 2, pp. 228–253, 2007.

[10] Q. Xue, W. Gong, and A. Ganz, "Proportional service differentiation in wireless LANs using spacing-based channel occupancy regulation," *Mobile Netw. Appl.*, vol. 11, no. 2, pp. 229–240, 2006.

[11] Y. Grunenberger, M. Heusse, F. Rousseau, and A. Duda, "Experience with an implementation of the Idle Sense wireless access method," in *Proc. ACM CoNEXT*, 2007, Article no. 24.

[12] H. Luo, S. Lu, and V. Bhurghawn, "A new model for packet scheduling in multihop wireless networks," in *Proc. ACM MobiCom*, 2000, pp. 76–86.

[13] K. Xu, M. Gerla, L. Qi, and Y. Shu, "Enhancing TCP fairness in ad hoc wireless networks using neighborhood RED," in *Proc. ACM MobiCom*, Sep. 2003, pp. 16–28.

[14] S. Pilosof, R. Ramjee, D. Raz, R. Ramjee, Y. Shavitt, and P. Sinha, "Understanding TCP fairness over wireless LAN," in *Proc. IEEE INFOCOM*, 2003, vol. 2, pp. 863–872.

[15] T. Nandagopal, T. Kim, X. Gao, and V. Bharghavan, "Achieving MAC layer fairness in wireless packet networks," in *Proc. ACM MobiCom*, 2000, pp. 87–98.

[16] B. Badunovic, C. Gkantsidis, D. Gunawardena, and P. Key, "Horizon: Balancing TCP over multiple path in wireless mesh network," in *Proc. ACM MobiCom*, Sep. 2008, pp. 247–258.

[17] Y. Yi and S. Shakkottai, "Hop-by-hop congestion control over a wireless multi-hop network," in *Proc. IEEE INFOCOM*, Mar. 2004, vol. 4, pp. 2548–2558.

[18] S. Sanghavi, L. Bui, and R. Srikant, "Distributed link scheduling with constant overhead," in *Proc. ACM SIGMETRICS*, Jun. 2007, pp. 313–324.

[19] M. Neely, E. Modiano, and C. Li, "Fairness and optimal stochastic control for heterogeneous networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 2, pp. 396–409, Apr. 2008.

[20] A. Eryilmaz, A. Ozdaglar, and E. Modiano, "Polynomial complexity algorithms for full utilization of multi-hop wireless networks," in *Proc. IEEE INFOCOM*, May 2007, pp. 499–507.

[21] L. Jiang and J. Walrand, "A distributed CSMA algorithm for throughput and utility maximization in wireless networks," in *Proc. Allerton Conf. Commun., Control, Comput.*, Sep. 2008, pp. 1511–1519.

[22] J. Crowcroft and P. Oechslin, "Differentiated end-to-end internet services using a weighted proportional fair sharing TCP," *Comput. Commun. Rev.*, vol. 28, no. 3, pp. 53–69, 1998.

[23] C.-Y. Wan, S. B. Eisenman, and A. T. Campbell, "CODA: Congestion detection and avoidance in sensor networks," in *Proc. ACM SenSys*, Nov. 2003, pp. 266–279.

[24] T. He, J. A. Stankovic, C. Lu, and T. F. Abdelzaher, "SPEED: A stateless protocol for real-time communication in sensor networks," in *Proc. ICDCS*, May 2003, pp. 46–55.

[25] J. Camp, V. Mancuso, O. Gurewitz, and E. W. Knightly, "A measurement study of multiplicative overhead effects in wireless networks," in *Proc. IEEE INFOCOM*, Mar. 2008, pp. 76–80.

[26] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A media access protocol for wireless LANs," in *Proc. ACM SIGCOMM*, 1994, pp. 212–225.

[27] "The network simulator—ns-2," [Online]. Available: http://www.isi.edu/nsnam/ns/

[28] K. Xu, M. Gerla, and S. Bae, "How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks?," in *Proc. IEEE GLOBECOM*, 2002, vol. 1, pp. 72–76.

[29] Y. Jian and S. Chen, "Can CSMA/CA networks be made fair?," in *Proc. ACM MobiCom*, 2008, pp. 235–246.

[30] F. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: Shadow prices, proportional fairness and stability," *J. Oper. Res.*, vol. 49, pp. 237–252, 1998.

[31] *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11, 1997.

Ying Jian received the B.E. and M.E. degrees in computer science from Tsinghua University, Beijing, China, in 2001 and 2004, respectively, and the Ph.D. degree in computer engineering from the University of Florida, Gainesville, in 2008.

After graduation, he was with Microsoft, Redmond, WA, for two years, and then joined Google, Mountain View, CA, in 2010. His research interests include QoS and security in multihop wireless networks and sensor networks.

Ming Zhang (S'10) received the B.S. degree in computer science from Sun Yat-sen University, Guangzhou, China, in 2000, the M.S. degree in computer science from Peking University, Beijing, China, in 2004, and the Ph.D. degree in computer engineering from the University of Florida, Gainesville, in 2010.

His research interests include real-time communication and security in wireless sensor networks, fairness in wireless LANs, and throughput in large RFID systems.

Shigang Chen (M'10) received the B.S. degree from the University of Science and Technology of China, Hefei, China, in 1993, and the M.S. and Ph.D. degrees from the University of Illinois at Urbana–Champaign in 1996 and 1999, respectively, all in computer science.

He is an Associate Professor with the Department of Computer and Information Science and Engineering, University of Florida, Gainesville. After graduation, he had worked with Cisco Systems, San Jose, CA, for three years before joining the University of Florida in 2002. His research interests include network security and wireless networks.

Dr. Chen was a Guest Editor for *Wireless Networks* and the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGIES. He served as a Technical Program Committee (TPC) Co-Chair for the Computer and Network Security Symposium of IEEE IWCCC 2006, a Vice TPC Chair for IEEE MASS 2005, a Vice General Chair for QShine 2005, a TPC Co-Chair for QShine 2004, and a TPC member for many conferences including IEEE ICNP, IEEE INFOCOM, IEEE ICC, IEEE GLOBECOM, etc. He received IEEE Communications Society Best Tutorial Paper Award in 1999 and the NSF CAREER Award in 2007.