

Can CSMA/CA networks be made fair?

Ying Jian Shigang Chen
Department of Computer and Information Science and Engineering
University of Florida, Gainesville, Florida, USA
{yjian, sgchen}@cise.ufl.edu

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 carrier sensing range, which is not true in reality. As our study reveals, the overhearing-based solutions, as well as several non-overhearing AIMD solutions, cannot achieve MAC-layer fairness in various settings. We propose a new rate control protocol, called PISD (Proportional Increase Synchronized multiplicative Decrease). 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.

Categories and Subject Descriptors

C.2.1 [Network Architecture and Design]: Wireless communication

General Terms

Design, Performance

Keywords

CSMA/CA, Fair Bandwidth Allocation

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

MobiCom'08, September 14–19, 2008, San Francisco, California, USA.
Copyright 2008 ACM 978-1-60558-096-8/08/09 ...\$5.00.

1. 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 that of *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 the 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. In this paper, we focus on wireless networks consisting of only single-hop flows. We assume an endpoint of one single-hop flow is within the carrier sensing range of another single-hop flow, but some nodes may not be in the carrier sensing range of one another.

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 [2, 12, 21, 15, 5, 18, 4, 11]. They fall in two categories: overhearing-based solutions and non-overhearing solutions.

¹This can happen when there are more neighboring access points than the number of non-overlapping channels.

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: i) increase/decrease minimum contention window if the local rate is above/below the average rate of all contending links [2, 12, 5], ii) serialize transmissions among contending links based on their scheduling tags [21, 15], or iii) emulate TDMA by computing a contention-free slotted schedule among the links [18]. The key question is how to collect information for the 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 [18, 25], hop count is not a reliable means to identify a contending relationship. Hence, in virtually all existing solutions, a node learns the information about others by overhearing. However, the overhearing approach faces another 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 (often the majority of them) cannot be overheard for information gathering, which severely limits the effectiveness of overhearing-based solutions.

Most non-overhearing solutions use the classical AIMD (Additive Increase Multiplicative Decrease) for rate control. On one hand, a node cannot overhear the exact information in transmissions made on contending links whose radio signal is strong enough to cause interference but too weak to decode. On the other hand, without overhearing, the node can still sense the aggregate impact of interference from those links by monitoring how busy the channel is, how frequently its own transmissions fail [4, 24], 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 [11, 8] (which can be measured through carrier sense instead of overhearing). We will show later that this approach also has limitations.

We propose a new rate control protocol, PISD (Proportional Increase Synchronized multiplicative Decrease), that provides fairness in CSMA/CA networks, particularly in IEEE 802.11 networks. We make three contributions in this paper. First, our study reveals the fundamental reasons exactly why the existing fairness solutions, as well as AIMD, do not work under realistic contention conditions. Particularly, for AIMD, 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. Unsynchronized multiplicative decrease is the reason for the failure of AIMD in CSMA/CA networks. Second, we introduce a number of novel rate control mechanisms, on which PISD is designed. The first mechanism relies on localized operations to ensure *synchronized multiplicative decrease*. The second mechanism

extends PISD for weighted fairness through *proportional increase*. The third mechanism uses *background transmission* to ameliorate throughput degradation due to multiplicative decrease. Efforts are made to simplify the implementation of the rate control mechanisms, which we believe will benefit practical systems that adopt them. Third, we perform detailed analysis on PISD and prove that it will converge and achieve (weighted) fairness.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 gives the network model. Section 4 describes the fairness problem. Section 5 proposes our PISD solution. Section 6 analyzes the performance of PISD. Section 7 presents additional simulation results. Section 8 concludes.

2. RELATED WORK

In [2, 12], to achieve fair bandwidth distribution among contending wireless links in a multihop wireless network, every node is required 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 [16] 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. In follow-up work [15] they propose MLM-FQ (Maximize-Local-Minimum Fair Queueing), 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 DFS (Distributed Fair Scheduling Protocol) [21], 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.

OML [18] 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 timeslot 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. [17] 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 4.1, the loss probabilities of two links can be very different — the actual values depend on the relative spatial locations of the links. In [20], Tassiulas and Sarkar address the max-min fairness problem using a multi-channel MAC model that is different from the CSMA/CA model used in this paper. Their later work [19] for end-to-end bandwidth guarantees is also based on the multi-channel model.

AIMD has been extensively studied in the past [6, 26, 13, 14], mostly in the context of TCP. Crowcroft and Oechslin

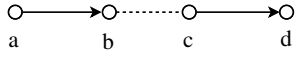


Figure 1: A network of two flows, (a, b) and (c, d) .

[7] 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. [4], by Xue et al. [24] and by Heusse et al. [11, 8] can only provide fairness under certain situations. AIMD has also been used in wireless networks for congestion control [22, 9].

3. 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. Throughout the entire paper, we consider CSMA/CA to mean the full 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 also 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 . In this case, we also say that node a has a contending node c . Generally speaking, the carrier sensing range is greater than the interference range, 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 a WLAN or ad hoc deployment setting. We do not consider multi-hop flows.

4. FAIRNESS PROBLEM REMAINS OPEN IN CSMA/CA NETWORKS

In this section, we use a simple example to illustrate the fairness problem in CSMA/CA networks and explain why this problem remains unsolved. The existing solutions based on overhearing would work if the transmission range, the interference range and the carrier sensing range were all identical. However, in reality, the transmission range is much shorter than the other two. Consequently a node will not be able to gather, through overhearing, the necessary information from all contending flows. We will show that the classic fairness approach of AIMD (which is widely used on wired and wireless networks) cannot solve the problem in CSMA/CA networks, either.

4.1 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

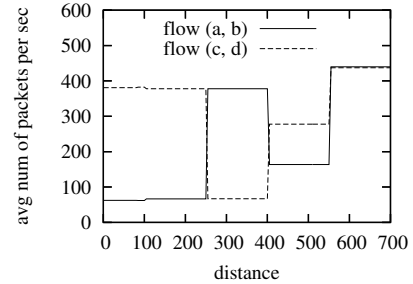


Figure 2: Rates of two flows with respect to the distance between b and c . The distance from a to b and that from c to d are both $150m$.

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.

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 [3], 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 [1].) The simulation parameters are given as follows: The transmission range of the nodes is $250m$, the carrier sensing range is $550m$, and the lengths of both links are $150m$. The transmission rate is 11 Mbps, and the packet length is 1,000 bytes. 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 node b and node c . When the distance is below $250m$, flow (c, d) obtains most of the channel's bandwidth. When the distance is between $250m$ and $400m$, flow (a, b) obtains most of the channel's bandwidth. When the distance is between $400m$ and $550m$, flow (c, d) regains the upper hand. When the distance is greater $550m$, 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 Appendix A.

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 $150m$ and the distance between b and c is $100m$. The simulation in ns-2 shows that C_1 is almost starved while the rate of

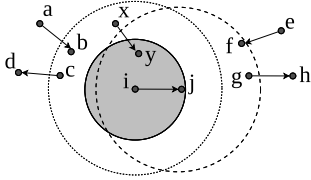


Figure 3: There are many contending nodes that cannot be overheard by i .

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) .

4.2 Limitation of Overhearing-based Solutions

Realizing the fairness problem in CSMA/CA networks, researchers have proposed numerous solutions [2, 12, 21, 15, 5, 18], 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 [2, 12, 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 [18], DFS [21] and EMLM-FQ [15], also depend on overhearing (see Section 2).

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 [23] (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.

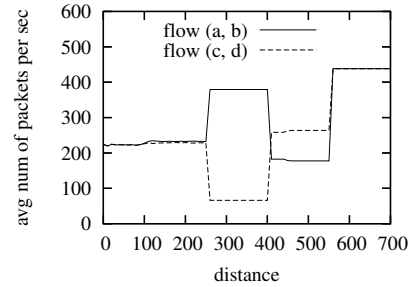


Figure 4: In general, Huang-Bensaou protocol does not work if any one of the contending links cannot be overheard.

We implement the Huang-Bensaou protocol [12], 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 c can overhear b ’s CTS/ACK). However, when the distance between b and c is beyond $250m$, the Huang-Bensaou protocol is totally ineffective. The same is true for all other schemes relying on overhearing.

4.3 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 (Additive Increase Multiplicative Decrease) 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 [22, 9], 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 however 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 [4, 24]. We simulate the AIMD protocol in [4] (the one in [24] 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 $-150m$ (such that a and c overlap to serve as the access point while b and d are hosts.) However, it performs

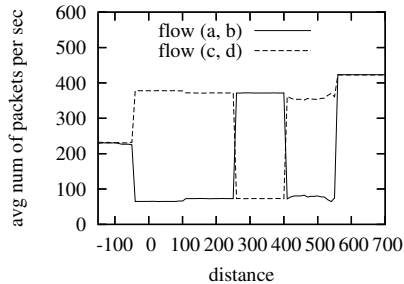


Figure 5: AIMD: multiplicative decrease occurs upon transmission failure.

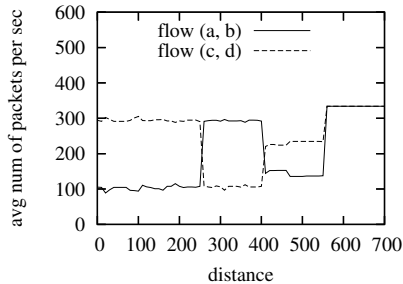


Figure 6: AIMD: multiplicative decrease occurs when buffer occupancy passes a certain threshold.

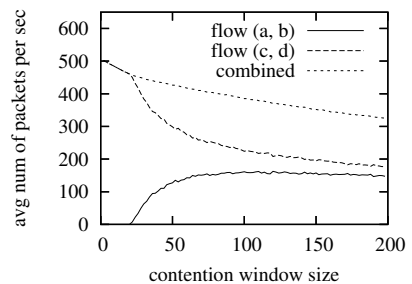


Figure 7: Idle Sense: The same contention window size does not ensure fairness.

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 [11, 8] replaces DCF’s random backoff by adaptively setting the same optimal size for the contention window at all hosts. It was shown in [11] 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. The algorithm of Idle Sense is for each host to measure the mean number \hat{n}_i of idle slots between two transmissions in the channel and to gradually increase its contention window when \hat{n}_i is below the desirable value or multiplicatively decrease its window when \hat{n}_i is above the desirable value. Idle Sense makes the assumption that, when AIMD converges, the contention windows at all hosts will reach the same size, and thus, 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 150m. 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.

5. 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.

5.1 Synchronized Multiplicative Decrease

Why does AIMD work for TCP but not for CSMA/CA? The reason is that AIMD achieves fairness only with synchronized multiplicative decrease. Contending TCP connec-

tions always 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.

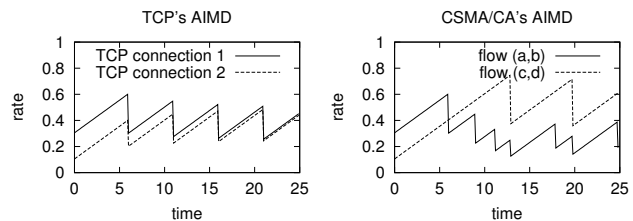


Figure 8: Left: Synchronized multiplicative decrease in TCP achieves fairness. Right: Unsynchronized multiplicative decrease in CSMA/CA cannot achieve fairness.

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 250m. At time 6 in the right plot of Fig. 8, when the combined rate of flow (a, b) and flow (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) .

5.2 AISD: Additive Increase Synchronized Multiplicative Decrease

In order to achieve fairness in CSMA/CA networks, we must ensure that multiplicative decrease is performed at con-

tending senders simultaneously. We design a new protocol, called 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. (This is what we do in all our simulations.)

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

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

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. The *quota* is defined as the number of bytes that remain available for transmission in the current time unit, which is equal to

$r_{i,j}(t)$ less the number of bytes that have been transmitted during the current time unit. (Note that it includes both packets to be released and packets already released to the queue but not transmitted yet.)

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 is sufficiently large, it jams for the rest of the current 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.

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

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. Consider flows (a, b) and (a, c) . Suppose (a, b) contends with (i, j) while (a, c) does not. When (i, j) is transmitting, node a performs independent media access for its two flows. For example, it may send an RTS to b and then set the backoff timer for (a, b) due to an RTS collision at b . While waiting on the timer for (a, b) , it sends an RTS to c and subsequently delivers a packet on (a, c) .

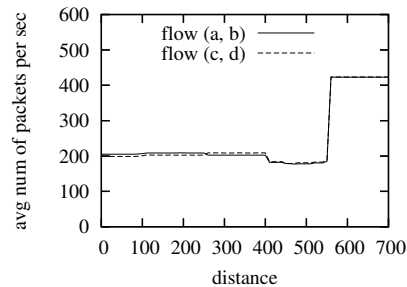


Figure 9: Synchronized multiplicative decrease equalizes the flow rates for the network in Fig. 1.

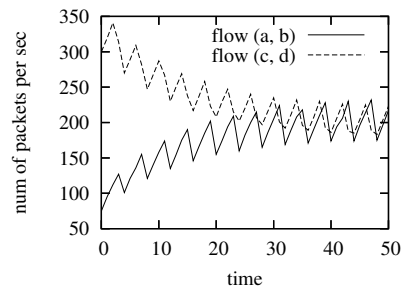


Figure 10: Rates of two contending flows under AISD with respect to time.

We simulate AISD on the network of Fig. 1 with the following additional parameters: α is 5 kbps, β is 25%, the time unit is one second, 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,

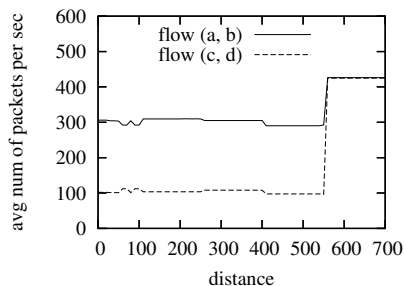


Figure 11: Given $w_{a,b} = 3$ and $w_{c,d} = 1$, under PISD, the rate of flow (a, b) is about three times that of flow (c, d) .

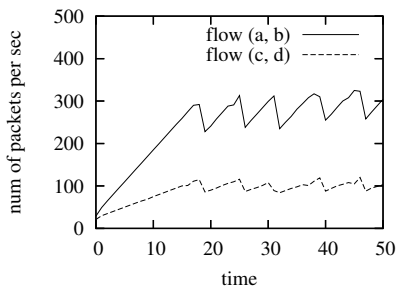


Figure 12: Rates of two contending flows under PISD with respect to time. $w_{a,b} = 3$, $w_{c,d} = 1$, and the distance from b to c is 100m.

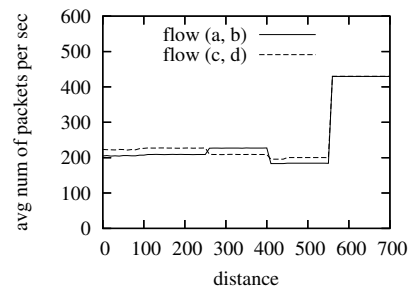


Figure 13: Background transmission will utilize some unused channel bandwidth for packet transmission.

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 100m. At time 0, the rate of flow (c, d) is much larger. Then, AISD kicks in to equalize the two rates.

5.3 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*. Fairness as discussed previously is a special case in which all weights are equal.

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

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

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

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 100m. 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.

5.4 PISD with Background Transmission

Using AIMD, TCP will not utilize the bottleneck router's full capacity at all times due to multiplicative decrease, which is evident from Fig. 8. Similarly, 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 the next section). If $\beta = 25\%$, then the fraction is 12.5%. However, in our simulations, the degradation is mostly around 5% and sometimes up to 10%.

We believe this is due to the interaction of PISD with the protocol details of CSMA/CA, particularly the IEEE 802.11 DCF, many of whose details concerning RTS, CTS, DIFS, EIFS, minimum/maximum contention windows, and backoff algorithm can impact flow rate. 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. Our basic 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.

The PISD protocol with background transmission is as follows. Proportional increase synchronized multiplicative decrease is performed on the target rate as usual. But 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. $n_t = r_{i,j}(t) \times \delta$. 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. (Note

that the default value for regular transmission is set by ns-2 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.

5.5 Discussion

The fairness problem becomes tricky when wireless links have different transmission (modulation) rates. The operations of PISD are independent of transmission rate. It achieves fairness regardless of whether the transmission rates of the links are the same or different. However, under non-uniform transmission rates, it is well known that ensuring each flow a fair share of bandwidth may cause significant reduction in a WLAN's overall throughput [10]. One solution to this problem is to change the definition of fairness. Instead of ensuring a fair bandwidth share, we allocate each flow a fair share of channel occupation time. PISD can be adapted to serve this purpose. Consider three contending wireless links whose transmission rates are 11 Mbps, 5 Mbps and 2 Mbps, respectively. If we let the weight of the 11 Mbps flow be one, we shall assign the weights of other two flows to be $\frac{5}{11}$ and $\frac{2}{11}$, respectively. While the two flows will send at lower rates, their transmissions take inversely proportionally longer time, resulting in the same channel occupation time for the three flows.

For implementation, PISD performs all rate adaptation operations on top of the MAC layer. It requires the MAC to support multiple queues (one for each adjacent link) and provide an API that allows MAC parameters to be changed. For example, jamming is implemented by reducing the minimum contention window.

6. ANALYSIS

It is well known that AIMD converges [6]. Much work about AIMD has been performed in the context of TCP [26, 13, 14]. In this section, we analyze PISD and show that it achieves weighted fairness after convergence. More importantly, we derive the convergence time, the channel coverage, and the convergence accuracy with respect to α and β , and reveal the performance tradeoff that can be made by changing these two parameters.

6.1 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.

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

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 chan-

nel 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 $l = 0, 1, 2, \dots$, 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 towards 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 $l > 0$,

$$\begin{aligned} r_{i,j}(lP) &= r_{i,j}((l-1)P) \times (1-\beta) + \alpha w_{i,j} \times P \\ &= r_{i,j}((l-1)P) \times (1-\beta) + \beta C \frac{w_{i,j}}{W}. \end{aligned}$$

By induction over the above iterative formula, we have

$$r_{i,j}(lP) = C \frac{w_{i,j}}{W} + (r_{i,j}(0) - C \frac{w_{i,j}}{W})(1-\beta)^l \quad (4)$$

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 $r_{i,j}(t)$, $t < lP$, for $l = 0, 1, 2, \dots$. The rate is multiplicatively decreased immediately after time $\lfloor t/P \rfloor P$ and then proportionally increased. We have

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

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 below

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

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

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

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:

$$\frac{|A_{i,j}(l) - A_{i,j}^*|}{A_{i,j}^*} \leq \varepsilon$$

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

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

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

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

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

6.2 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.

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

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.

6.3 Convergence Accuracy

If every flow is able to deliver all packets released to the MAC layer in a timely fashion, then the sending rate will be equal to the target rate and therefore weighted fairness is accurately achieved once all target rates are fully converged. That is not the case if not all packets released based on the target rate can be delivered.

As the flows' target rates are proportionally increased in a PISD period, it is possible that, right before multiplicative decrease, the sum of all target rates is slightly greater than the channel capacity, in which case not all packets can be delivered. We study the impact of this case below.

In the worst case, a flow (i, j) cannot transmit any packet in the last time unit of a PISD period. Suppose the target rates of all flows are fully converged. The amount of data that flow (i, j) cannot deliver in the last time unit is bounded by $r_{i,j}^*$. The amount of data that is supposed to be delivered in the whole period is $A_{i,j}^*P$. The *convergence accuracy* of PISD, denoted as Acc , is defined as follows.

$$Acc = 1 - \frac{r_{i,j}^*}{A_{i,j}^*P} = 1 - \frac{1}{(1 - \frac{\beta}{2}) \frac{\beta C}{\alpha W} + \frac{\beta}{2}}$$

Hence, the convergence accuracy decreases as α increases.

Putting all of the above analysis together, we can see that choosing the values of α and β is actually making a tradeoff among three system properties: convergence time, channel coverage, and convergence accuracy.

7. ADDITIONAL SIMULATIONS

We perform additional simulations under two scenarios to evaluate the effectiveness of the proposed PISD protocol. 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 Section 4.1 and Section 5.2.* The parameters for 802.11 DCF use the default values set by ns-2 according to the protocol standards. By default, $\alpha = 2$ kbps, and $\beta = 25\%$. We will study how different values for α and β affect PISD's performance. By default, background transmission is turned off.

Shown in the left plot of Fig. 14, our first simulation scenario consists of two access points, a and b , located at two nearby buildings. Node a sends data to three client hosts, $h1$, $h2$ and $h3$. Node b also sends data to three client hosts, $h4$, $h5$ and $h6$. The clients are evenly spread around the access points. The length of each wireless link is $80m$, and the distance between a and b is $480m$. The middle plot of Fig. 14 shows the flow rates under the IEEE 802.11 DCF. When the rate curves of several flows overlap, we will explain which flows each curve represents in both text and figure caption. Under the 802.11 DCF, the rates of flows $(a, h1)$, $(a, h2)$ and $(a, h3)$ are the same (the lower curve in the middle plot) because node a schedules packets in round robin order among local flows. The rates of flows $(b, h4)$, $(b, h5)$ and $(b, h6)$ are also the same (the upper curve in the middle plot). However, the rates of flows from a are much smaller than the rates of flows from b . Because the distances from a and its clients to b and its clients are greater than the transmission range $250m$ (see simulation parameters in Section 4.1), no overhearing-based solutions work here. The simulation result for the Huang-Bensaou protocol [12] is shown in the right plot of Fig. 14, which is comparable to what the 802.11 DCF produces.

PISD is able to achieve fairness among all flows, as shown in Fig. 15. Starting from different initial rates, all flows converge to the same fair rate. The total throughput is 428.1 packets per second, comparing with 444.6 under the 802.11 DCF with or without the Huang-Bensaou protocol. Next, we turn on background transmission, and the result is shown in Fig. 16. Some flows achieve higher average rates, and the total throughput becomes 455.8 packets per second. It is higher than the throughput under the 802.11 DCF because of reduced radio collisions, thanks to intermittent release of packets to the MAC layer at the background rate. Since the additional rate acquired through background transmission obscures the rate curve produced by PISD, for presentation clarity, we will turn it off in other simulations.

We now study how β and α affect the performance of PISD. The simulations confirm the analytical results in Section 6.1. First, we double the value of β while keeping α the same, and the simulation result in Fig. 17 shows that the flow rates converge quicker, when comparing with Fig. 15. It also shows that the average flow rate is smaller, indicating a smaller channel coverage by PISD (as predicted in Section 6.2), and therefore more bandwidth is allocated for background trans-

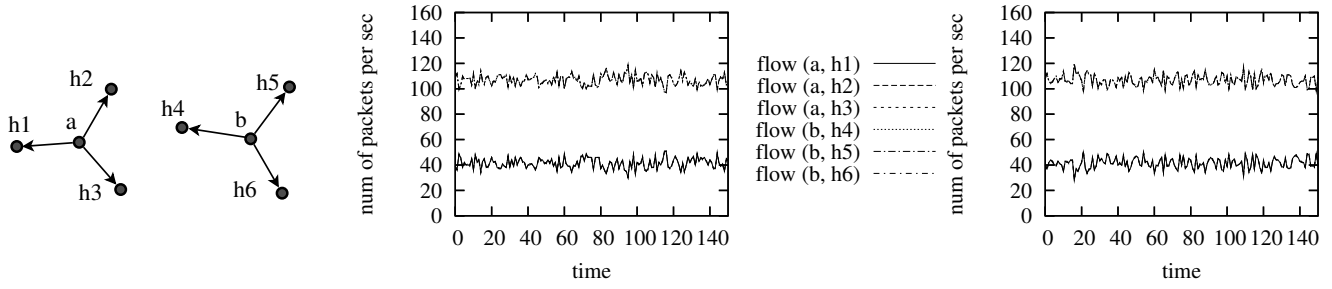


Figure 14: *Left: Network topology. Middle: Flow rates under 802.11 DCF. The lower curve shows the rates of flows (a, h1), (a, h2) and (a, h3), which are the same. The upper curve shows the rates of flows (b, h4), (b, h5) and (b, h6), which are the same. Right: Flow rates under Huang-Bensaou protocol. Similarly, the lower curve shows the rates of flows (a, h1), (a, h2) and (a, h3). The upper curve shows the rates of the other three flows.*

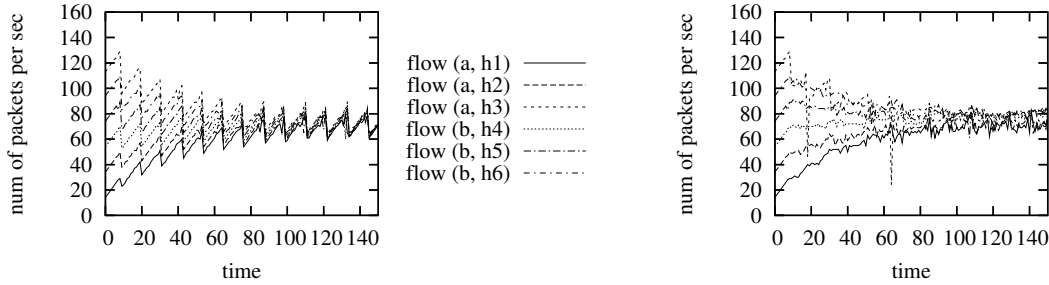


Figure 15: PISD achieves fairness among all flows.

Figure 16: Flow rates are slightly improved with background transmission.

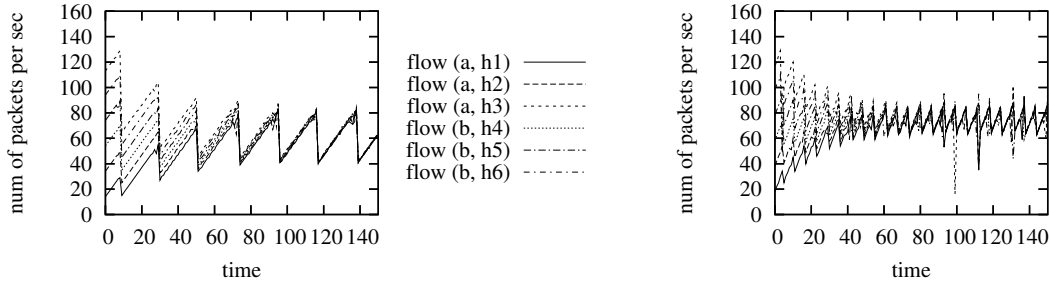


Figure 17: Increasing the value of β reduces both convergence time and channel coverage.

Figure 18: Increasing the value of α reduces both convergence time and convergence accuracy.

mission. Second, we double the value of α while keeping β the same, and the simulation result in Fig. 18 shows that the flow rates converge quicker, when comparing with Fig. 15. It also shows that convergence accuracy decreases due to the sudden drop in the rates of some flows right before multiplicative decrease, as predicted in Section 6.3.

So far we have set all flow weights to 1. As shown in the left plot of Fig. 19, we modify the topology by turning $h2$ and $h6$ into servers that upload data to access points a and b , respectively. We set the servers' weights to 3, and the simulation result in the middle plot of Fig. 19 shows that the rate of a server is about twice the rate of a client. One may notice the spikes in the rate adaptation curves in the figure. Such spikes are expected according to our analysis in Section 6.3. They only happen in the last time unit of a PISD period when the aggregated target rate of all contending flows exceed the channel capacity. When nodes that detect channel congestion jam the channel with their packets, the node

that detects congestion last may send at a low rate, causing a downward spike. Since the spikes may only happen at the end of a PISD period, its impact on the average flow rate is limited, which is confirmed by the analytical result in Section 6.3 and the simulation result in the middle plot, where the average rates of the server flows are 124.0 and 125.1 packets per second respectively, and the average rates of the client flows are 41.8, 42.3, 42.4 and 42.7 packets per second respectively. Moreover, Section 6.3 shows that decreasing α will improve convergence accuracy (i.e., reduce spikes), which is confirmed by the simulation result in the right plot of Fig. 19, where α is reduced by two thirds.

Next we expand the network to have four access points and ten hosts. The access points are located at the corners of a $380\text{m} \times 380\text{m}$ square. The distance from the hosts to their access points varies from 70m to 150m. Their relative positions are shown in Fig. 20. The rates of the flows under PISD, PISD with background transmission (PISD-b), DCF,

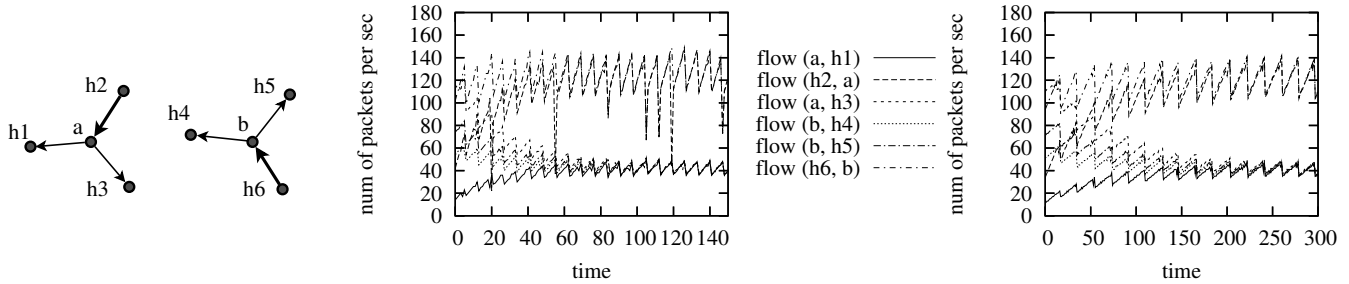


Figure 19: *Left:* Hosts $h2$ and $h6$ are changed to servers. *Middle:* When the servers each have weight 3 and the clients each have weight 1, the rate of a server is three times that of a client. *Right:* Downward spikes are reduced when α is decreased.

and the Huang-Bensaou protocol (H.-B.) are shown in Table 1. PISD is able to achieve fairness while the DCF and the Huang-Bensaou protocol cannot in this scenario.

Our second simulation scenario is an ad hoc network shown in Fig. 21, 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 $400m$ by $600m$, and the nodes are plotted in the area based on their assigned coordinates. The simulation results are shown in Table 2. Each row 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.

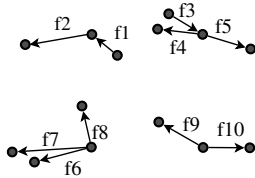


Figure 20: 4-WLAN network topology.

Table 1: Flow rates achieved by different protocols.

flow	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10
PISD	42.7	43.2	43.7	43.7	43.4	42.8	43.7	43.7	43.7	43.6
PISD-b	43.0	45.6	48.1	43.0	48.8	46.2	45.9	43.4	46.1	46.0
DCF	74.7	55.1	99.7	51.4	51.4	15.3	15.3	15.3	40.8	40.8
H.-B.	18.6	17.2	36.8	35.5	35.5	47.5	47.5	47.5	84.0	84.0

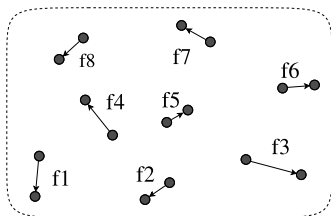


Figure 21: Ad hoc network topology.

8. CONCLUSION

In this paper, we have investigated the unfairness problem in CSMA/CA networks. We have shown that existing solutions based on overhearing are not effective when contending

Table 2: Under different weight assignments, flow rates are always proportional to flow weights.

flow	f1	f2	f3	f4	f5	f6	f7	f8
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

nodes are outside each other's transmission range. We have also shown that the existing non-overhearing AIMD solutions do not work either. We then propose our new fairness solution, 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. 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.

9. ACKNOWLEDGMENTS

This work was supported in part by the US National Science Foundation under grant CNS-0644033 and grant CNS-0721731. We would also like to thank our shepherd, Dr. Brad Karp, and the anonymous reviewers for their constructive comments.

10. REFERENCES

- [1] The Network Simulator - ns-2. <http://www.isi.edu/nsnam/ns/>.
- [2] B. Bensaou, Y. Wang, and C. Ko. Fair Medium Access in 802.11 Based Wireless Ad Hoc Networks. *Proc. ACM MOBIHOC*, 2000.
- [3] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang. MACAW: A Media Access Protocol for Wireless LANs. *Proc. ACM SIGCOMM*, 1994.

- [4] S. Cai, Y. Liu, and W. Gong. Analysis of an AIMD based Collision Avoidance Protocol in Wireless Data Networks. *Proc. IEEE CDC*, 2003.
- [5] S. Chen and Z. Zhang. Localized Algorithm for Aggregate Fairness in Wireless Sensor Networks. *Proc. ACM Mobicom*, 2006.
- [6] D. M. Chiu and R. Jain. Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks. *Computer Networks and ISDN Systems*, 17:1–14, 1989.
- [7] J. Crowcroft and P. Oechslin. Differentiated End-to-End Internet Services Using a Weighted Proportional Fair Sharing TCP. *ACM SIGCOMM Computer Communication Review*, 28(3):53–69, 1998.
- [8] Y. Grunenberger, M. Heusse, F. Rousseau, and A. Duda. Experience with an Implementation of the Idle Sense Wireless Access Method. *Proc. CoNEXT*, 2007.
- [9] T. He, J. A. Stankovic, C. Lu, and T. F. Abdelzaher. SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks. *Proc. International Conference on Distributed Computing Systems (ICDCS'03)*, May 2003.
- [10] M. Heusse, F. Rousseau, G. Berger-Sabbatel, and A. Duda. Performance Anomaly of 802.11b. *Proc. IEEE INFOCOM*, 2003.
- [11] 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. *Proc. ACM SIGCOMM*, 2005.
- [12] X. Huang and B. Bensaou. On Max-Min Fairness and Scheduling in Wireless Ad Hoc Networks: Analytical Framework and Implementation. *Proc. ACM MOBIHOC*, 2001.
- [13] S. Jin, L. Guo, I. Matta, and A. Bestavros. TCP-friendly SIMD Congestion Control and Its Convergence Behavior. *Proc. IEEE ICNP*, 2001.
- [14] F. Kelly, A. Maulloo, and D. Tan. Rate control in communication networks: shadow prices, proportional fairness and stability. *Journal of the Operational Research*, 49, 1998.
- [15] H. Luo, J. Cheng, and S. Lu. Self-Coordinating Localized Fair Queueing in Wireless Ad Hoc Networks. *IEEE Trans. on M. Comp.*, 3(1), 2004.
- [16] H. Luo, S. Lu, and V. Bhurghawn. A New Model for Packet Scheduling in Multihop Wireless Networks. *Proc. ACM MOBICOM*, 2000.
- [17] T. Nandagopal, T. Kim, X. Gao, and V. Bharghavan. Achieving MAC Layer Fairness in Wireless Packet Networks. *Proc. ACM MOBICOM*, 2000.
- [18] A. Rao and I. Stoica. An Overlay Mac Layer for 802.11 Networks. *Proc. ACM MOBISYS*, 2005.
- [19] S. Sarkar and L. Tassiulas. End-to-end Bandwidth Guarantees Through Fair Local Spectrum Share in Wireless Adhoc Networks. *IEEE Transactions on Automatic Control*, 50(9), September 2005.
- [20] L. Tassiulas and S. Sarkar. Maxmin Fair Scheduling in Wireless Networks. *Proc. IEEE INFOCOM*, 2002.
- [21] N. H. Vaidya, P. Bahl, and S. Gupta. Distributed fair scheduling in a wireless LAN. *Proc. ACM MOBICOM*, 2000.
- [22] C.-Y. Wan, S. B. Eisenman, and A. T. Campbell. CODA: Congestion Detection and Avoidance in Sensor Networks. *Proc. ACM SenSys'03*, November 2003.
- [23] K. Xu, M. Gerla, and S. Bae. How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks? *Proc. IEEE GLOBECOM*, 2002.
- [24] Q. Xue, W. Gong, and A. Ganz. Proportional Service Differentiation in Wireless LANs Using Spacing-based Channel Occupancy Regulation. *Mobile Networks and Applications*, 11(2), 2006.
- [25] Y. Yang and R. Kravets. Throughput Guarantees for Multi-priority Traffic in Ad Hoc Networks. *Elsevier Ad Hoc Networks Journal*, 5(2), 2007.
- [26] Y. R. Yang and S. S. Lam. General AIMD Congestion Control. *Proc. IEEE ICNP*, 2000.

Appendix A.

We explain why unfairness happens in IEEE 802.11 DCF.

- *Segment 1: for distance from 0 to 100, (c, d) has a higher rate.* First, after flow (c, d) transmits a packet, node c will have a better chance to obtain the channel for the next transmission than node a. The reason is that, after d transmits ACK, the sender c performs random backoff, while node a waits for EIFS because it can sense d's ACK but not understand it (out of the transmission range). EIFS is greater than the average random backoff performed by c. Hence, node c has a greater probability of obtaining the channel. Second, after flow (a, b) transmits a packet, node c again has a greater chance to obtain the channel because node a will start a random backoff timer while node c only needs to count down the existing one.
- *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 c will have a better chance to obtain the channel. Second, after flow (a, b) transmits a packet, node a performs random backoff while node c waits for EIFS when it senses (but cannot understand) DATA transmitted by a. Because EIFS subtracted by the transmission time of ACK (sent by b) is smaller than the average random backoff performed by a, node c again has a greater chance of obtaining the channel for the next transmission.
- *Segment 3: for distance from 250 to 400, (a, b) has a higher rate.* This is the reversal of the case in segment 2. On one hand, node c's wait for EIFS caused by ACK from b makes it less likely to obtain the channel after flow (a, b) transmits a packet. On the other hand, node a's wait for EIFS caused by DATA from c does not give node c a higher probability of obtaining the channel after flow (c, d) transmits a packet, by the same token as explained in Segment 2.
- *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. 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).
- *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.