IMPROVING FAIRNESS AND THROUGHPUT IN WIRELESS SYSTEMS

By

MING ZHANG

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2010

To my father

ACKNOWLEDGMENTS

I would like to thank my advisor, Prof. Shigang Chen, for his constant guidance, support, and encouragement. I am privileged to have such a wonderful advisor, who is at all times enthusiastic, optimistic, patient, helpful, and encouraging. He gave me extensive advice and insight during the course of my research work.

My special thanks go to Prof. Sartaj Sahni, Prof. Jih-Kwon Peir, Prof. Ahmed Helmy, and Prof. Tan Wong, for their instructive comments and support during my work. I would also like to thank all my colleagues in Prof. Chen's research group, including Zhan Zhang, Liang Zhang, MyungKeun Yoon, Ying Jian, Tao Li, Wen Luo, Yan Qiao, Zhen Mo and Shuang Lin, for providing a high level of research support.

I want to express my deepest love to my darling wife, Xuelian Xiao, my mother Xingguo Li, my father Xianchang Zhang, and my angel Olivia Luoqiao Zhang. Their love, understanding, and encouragement have always been the strongest support to me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

IMPROVING FAIRNESS AND THROUGHPUT IN WIRELESS SYSTEMS

By

Ming Zhang

December 2010

Chair: Shigang Chen
Major: Computer Engineering

With the advancement of wireless technologies, wireless systems have been widely used in today's world. Especially, the IEEE 802.11 Wireless LANs (WLANs) have covered a large portion of the urban areas to provide anytime, anywhere Internet service. In addition, the Radio-Frequency Identification system (RFID) is another important wireless network which promises to revolutionize the inventory management in large warehouses, retail stores, hospitals, transportation systems, etc. In this dissertation, we first propose novel solutions for improving fairness and throughput in WLANs. We then introduce a new method to improve reading throughput in large RFID systems.

Our first work focuses on achieving MAC-layer time fairness among contending WLANs. The WLANs may overlap and contend with each other. We show that the contention among nearby WLANs is location-sensitive, which makes some hosts much more capable than others to obtain the channel for their transmissions. Another reality is that wireless hosts use different transmission rates to communicate with the access points due to attenuation of their signals. We show that location-sensitive contention aggravates the throughput anomaly caused by different transmission rates. It can cause throughput degradation and host starvation. Achieving time fairness across multiple WLANs is a very difficult problem because the hosts may perceive very different channel conditions and they may not be able to communicate and coordinate their operations

due to the disparity between the interference range and the transmission range. In this work, we design a MAC-layer time fairness solution based on two novel techniques: channel occupancy adaptation, which applies AIMD on the channel occupancy of each flow, and Queue Spreading (QS), which ensures that all hosts and only those hosts in a saturated channel detect congestion and reduce their channel occupancies in response. The proposed solution is called AIMD/QS+k. We show that AIMD/QS+k approximates the generic adaptation algorithm for proportional fairness.

Our second work focuses on achieving transport-layer fairness among contending WLANs. TCP is the dominating transport-layer protocol used by many applications over WLANs. Contention among multiple nearby WLANs in urban areas may cause severe TCP unfairness, where some TCP flows can achieve very high throughput at the expense of starving others. This unfairness results from the fact that different physical nodes conveying TCP flows at a wireless bottleneck may have different channel observations and consequently they may provide inconsistent feedbacks to the TCP sources. Existing solutions to this problem try to synchronize channel observations of contending nodes by exchanging control messages among them. They rely on the assumption that these nodes are within each other's transmission range, which however may not always hold. In this work, we design a new protocol, called Wireless Probabilistic Drop (WPD), to improve TCP fairness without requiring direct communication among nodes. In WPD, when a node detects congestion, it probabilistically chooses to either drop some packets to resolve the congestion, or aggressively spread the congestion signal to other contending nodes. Each node makes the choice with a probability that is proportional to its flow rate. Henceforth, high-rate flows tend to perform rate reduction more often, and low-rate flows are more likely to increase their flow rates. Eventually, all flows passing the bottleneck are expected to get a fair share of the channel bandwidth.

Our third work focuses on improving the RFID reading throughput. In large RFID systems, periodically reading the IDs of the tags is an important function to guard against administration error, vendor fraud and employee theft. Given the low-speed communication channel in which a RFID system operates, the reading throughput is one of the most important performance metrics. The current protocols have reached the physical throughput limit that can possibly be achieved based on their design methods. To break that limit, we have to apply fundamentally different approaches. In this work, we investigate how much throughput improvement the analog network coding can bring when it is integrated into the RFID protocols. The idea is to extract useful information from collision slots when multiple tags transmit their IDs simultaneously. Traditionally, those slots are discarded. With analog network coding, we show that a collision slot is almost as useful as a non-collision slot in which exactly one tag transmits. We propose the Framed Collision-Aware Tag identification protocol (FCAT) that optimally applies analog network coding to maximize the reading throughput, which is $51.1\% \sim 70.6\%$ higher than the best existing protocols.

CHAPTER 1
INTRODUCTION

IEEE 802.11 Wireless LANs (WLANs) have been densely deployed in many urban areas [2, 60] to satisfy the demand for Internet access from anywhere at anytime. In a typical WLAN, a wireless router serving as an access point is connected to the Internet via a cable modem. Several wireless clients such as laptops, mobile phones and other wireless-enabled devices are connected to the access point wirelessly. So many WLANs are located near one another. It has been observed that severe fairness problems may occur among nearby WLANs sharing the same channel. We will start from the MAC-layer to study the intriguing interplay between location-sensitive contention and time-allocation anomaly in 802.11 DCF networks. Then, we will move to the transport-layer and propose a new protocol for improving TCP fairness among nearby contending WLANs.

Radio-Frequency Identification systems (RFID) have brought revolutionary change to the inventory management in large warehouses, retail stores, hospitals, transportation systems, etc. A RFID system involves a few readers (interrogators) and a large number of tags (labels). The tag contains an integrated circuit for storing and processing data and an antenna for communicating with the reader. A unique ID is assigned to each tag and a tag is attached to an item. By accessing the IDs of the tags, the reader can wirelessly identify and track the items from a distance even without line of sight. In a large warehouse, it is required to periodically read the IDs of the tags to guard against administration error, vendor fraud and employee theft. Because the RFID is operated in a low-speed wireless channel and the number of tags is expected to be large, the reading throughput becomes a critical performance metric. We will investigate a fundamentally different approach to boost the RFID reading throughput.

1.1 MAC-layer Time Fairness across Multiple Wireless LANs

Cities are now crowded with wireless access points. At airport terminals, office buildings, homes and shops, a laptop can often find several to a few dozens of access points in usable range, most of which run the IEEE 802.11b/g protocol and each supports a WLAN. With only three non-overlapping channels in 802.11b/g, nearby WLANs will inevitably interfere with each other, giving rise to two intriguing problems at the MAC-layer.

The first problem is location-sensitive contention [75]. The contention resolution protocol, 802.11 DCF (Distributed Coordination Function), works well in a symmetric environment where all hosts are downloading content via the same access point. But it does not perform well in asymmetric settings that are common when hosts in nearby WLANs contend in the same channel. Depending on their relative spatial locations, some hosts may gain huge advantage in occupying the channel for their transmissions. As they obtain most of the channel bandwidth, hosts in other WLANs are starved.

Location-sensitive contention is very difficult to deal with because of two fundamental wireless properties. The first property is that contention is defined by the interference or carrier-sensing range, whereas communication happens within the transmission range. Consequently, contending hosts in different WLANs may not be able to explicitly exchange or implicitly overhear necessary information to coordinate their operations. They may not even know whom they contend with. The second property is that hosts in different WLANs may sense very different channel conditions (in terms of channel idle time, transmission failure rate, or buffer occupancy) even when they contend in the same channel. When they observe the same channel in different states, if they cannot communicate (due to the first property), their reactions are bound to be different, causing unfairness. An extensive discussion on this issue can be found in [34]. Up to date, the research on WLAN fairness has largely ignored the above two properties. Most existing solutions rely on the assumption that the contending hosts are able to communicate or overhear each other's transmission or that they sense the same channel conditions. The recent work of PISD [34] breaks away from such assumptions. However, it requires that all wireless hosts must be in the same contention group, which is often untrue in reality. Imagine a large number of WLANs are deployed in a city center. They partially overlap one another to provide a full coverage of the whole area. There is no way that they will all mutually contend with each other. PISD can cause severe problems in such a scenario.

The second problem is time-allocation anomaly. IEEE 802.11b allows four different transmission rates, 11Mbps, 5.5Mbps, 2Mbps and 1Mbps. IEEE 802.11g or 802.11a allows eight different rates. The transmission rate of a wireless host is determined through auto rate fallback based on how reliably the host can communicate with the access point at a certain rate. The transmission rate will be lower if the host is further away from the access point or there are obstacles (such as walls) between them. It is well known that, if a single host in a WLAN chooses a low transmission rate, all other hosts in the same WLAN suffer with low throughput [30]. To address the above anomaly, researchers have proposed to replace throughput fairness with time fairness, in which all hosts occupy the channel for the same fraction of time. However, most prior work does not consider the impact of location-sensitive contention that exists among nearby WLANs.

On one hand, location-sensitive contention can push time-allocation anomaly to the extreme, allowing a low-rate host in one WLAN to obtain an excessive amount of channel time and starve the high-rate hosts in a nearby WLAN. On the other hand, location-sensitive contention makes time-allocation anomaly a much harder problem to solve because contending hosts in different WLANs may be outside of each other's transmission range (but within the interference or carrier-sensing range). Unlike a multihop wireless network that has a communication path between any two nodes, in our WLAN setting, there may not exist any node between two contending hosts to relay their information. Most prior solutions for time fairness have largely ignored the impact of location-sensitive contention. They either rely on a central coordinator [30, 66, 67], or assume that each host has certain knowledge about its contending hosts [35], that the contending hosts will sense the same channel condition [31, 52], or that all hosts in all WLANs mutually contend [34]. These assumptions do not hold in general.

In this work, we solve the time fairness problem under location-sensitive contention among multiple WLANs. Loosely speaking, our solution, called AIMD/QS+k, ensures that each wireless host will receive a fair share of the channel time even when it has no way to know exactly whom it contends with. Precisely speaking, AIMD/QS+k approximately achieves proportional fairness in channel time allocation for the hosts across multiple WLANs. The design of AIMD/QS+k is based on two novel techniques, called channel occupancy adaptation and queue spreading. The former applies additive increase multiplicative decrease on each host's channel occupancy. The latter accurately identifies which hosts saturate the channel during the time of congestion. For example, suppose two hosts, x and y, in different WLANs cause congestion. Due to location advantage, x can send out all its packets at the expense of lowered throughput at y. Hence, only y detects congestion. To resolve congestion, both y and x should perform multiplicative decrease. If y reduces its channel occupancy alone, x will simply pick up the extra bandwidth and widen the unfairness. Without any means of communication, how can y make sure that x (but none of its other contending hosts that do not contribute to the current congestion) will join the channel-occupancy reduction? Remarkably, this problem can be solved in a fully distributed way without requiring the nodes to exchange information, overhear, or even know each other's existence. We demonstrate that AIMD/QS+k achieves almost perfect proportional fairness in scenarios where the existing solutions fail.

1.2 TCP Fairness across Multiple Wireless LANs

TCP is the dominating transport layer protocol used by many applications over WLANs [18, 21]. People surf websites, chat by using instant messengers and download files through TCP connections. However, it has been observed that TCP may demonstrate several fairness problems in WLANs. One is the upstream/downstream TCP unfairness within a single WLAN such that upstream flows may gain much more channel bandwidth than downstream flows [4, 5, 54, 57]. Another one is the TCP unfairness among multiple contending WLANs, where the TCP flows in some WLANs may achieve very high throughput at the expense of starving others in nearby WLANs [75, 78]. In this work, we will focus on the latter problem.

Based on network traffic conditions, TCP dynamically adjusts the source sending rate by performing AIMD (Additive Increase Multiplicative Decrease) on its congestion window. AIMD works well in wired networks, where a bottleneck router detects congestion and notifies the TCP sources of the congestion. However, it may not always work well in wireless networks because TCP flows may pass through different physical nodes that contend in a bottleneck channel. These nodes may have different channel observations and different capabilities of obtaining the channel for transmissions [26, 34]. Consequently, when only some nodes (but not all) detect channel congestion, the TCP flows that pass those nodes will receive congestion feedback but the flows passing other nodes will not. For TCP to achieve fairness, congestion feedback must be consistent across all flows that contend in the bottleneck. Inconsistent feedback to the flow sources will render TCP ineffective in its rate control among contending flows.

Fig. 1-1 shows an example of three partially overlapping WLANs, where s_1, s_2 and s_3 are access points. A TCP flow from a server on the Internet passes through an access point to a wireless client in each WLAN. Note that we only draw the wireless part of each TCP flow (from the access point to the wireless client) in all figures throughout the dissertation. Suppose s_1 and s_2 contend and they form a contention group g_1.

Nodes s_2 and s_3 also contend and they form another contention group g_2. But s_1 and s_3 do not contend; they can transmit simultaneously. Because s_2 contends with both s_1 and s_3, it senses a busier channel and is less capable of accessing the channel due to heavier contention. Hence, when the channel is saturated, as its queue builds up and exceeds a threshold, s_2 is likely to detect the congestion first. At the time when s_2 detects congestion, s_1 and s_3 may still have small queues since they are more capable of sending out their packets due to less contention (as we observe consistently in our ns-2 simulations). If s_2 immediately drops packets to inform the source of f_2 to reduce its sending rate, the congestion will be resolved while the flows passing s_1 and s_3 are still performing additive increase. In this case, f_1 and f_3 will seize up the channel bandwidth given up by s_2. Hence, the TCP rate adaptation actually promotes unfairness among the rates of f_1, f_2 and f_3 because it penalizes the flow that experiences heavier contention (and thus performs multiplicative decrease more frequently).

To solve this problem, Xu et al. [75] proposed NRED (Neighborhood Random Early Detection). The basic idea is to make sure that all contending nodes observe the same channel condition. For instance, in Fig. 1-1, s_1, s_2 and s_3 periodically synchronize their channel observations through exchanging control messages. Therefore, they can detect channel congestion simultaneously and then drop packets to notify the TCP sources to properly perform rate reduction, which leads to enhanced fairness. However, due to the disparity among transmission/interference/carrier-sensing ranges [74], nodes that are not able to directly communicate may still contend in the same channel. In Fig. 1-1, if s_1, s_2 and s_3 are outside of each other's transmission range, NRED will fail because these nodes cannot synchronize their channel observations and provide consistent feedbacks to the TCP sources.

In this work, we propose a new solution, WPD (Wireless Probabilistic Drop), for improving TCP fairness without requiring any means of direct communication among contending nodes. It is a fully distributed solution that does not require any modification to the operational protocols of TCP and 802.11 DCF (except for certain MAC parameter changes during congestion). We use the example in Fig. 1-1 to illustrate the basic idea. Suppose s_2 detects congestion first. Instead of dropping packets immediately, our solution requires s_2 to make a probabilistic choice between two states: resolution or signaling. In the resolution state, s_2 will drop packets to resolve the congestion. In the signaling state, s_2 will not drop packets, but instead it will signal other nodes about the congestion by aggressively pushing more packets into the channel. As s_2 grabs more channel bandwidth, s_1 and s_3 will observe that their queues build up and eventually pass the threshold. Once they detect congestion, they will perform similar operations. In our solution, each node that detects congestion makes a choice between resolution and signaling states; the probability for choosing the resolution state is proportional to the node's channel occupancy (i.e., the fraction of time during which the node occupies the channel for its transmission). Hence, if s_2 has a smaller channel occupancy, it will have a higher probability to choose the signaling state, while s_1 and s_3 are more likely to choose the other state and perform packet dropping. Consequently, f_2 will increase its sending rate more often while f_1 and f_3 perform rate reduction, which shrinks the gap between the rate of f_2 and the rates of other flows. Under such dynamics, our simulations demonstrate that all contending flows receive fair shares of the channel bandwidth over the long run in Fig. 1-1 as well as in other more complex scenarios.

1.3 Improving Reading Throughput in Large RFID Systems

The barcode system brings numerous benefits for the retail stores. It speeds up the checkout process, makes the price change easier, and allows quick access for the properties of each merchandize item. It also has serious limitation. A barcode can only be read in close range. Suppose an inventory management policy requires periodical reading of all items in order to guard against administration error, vendor fraud and employee theft. One will have to use a portable laser scanner and manually read the barcodes one after another, which is tedious and error-prone. RFID tags, which can be read wirelessly, provide an ideal solution to this problem [19, 79]. Each tag carries a unique identification number (ID), and a RFID reader can retrieve the ID of a tag even when there are obstacles between them. Although the passive tags are most popular, they are not suitable for automated inventory management in a large area because they can only be read in a few meters. In order to read all tags, we have to either deploy numerous readers, each covering a small area, or manually move a reader around, which is again inefficient and error-prone. This work considers the battery-powered active (or semi-passive) tags that can be read in a long distance and have more software/hardware resources than the passive tags.

The communication between the RFID reader and the tags is operated in a low-speed channel. Yet the number of tags in a large RFID system is expected to be very large. Therefore, one of the most critical performance metrics is the reading throughput, which is the average number of unique tag IDs that the reader can collect in a second. The current protocols have reached the physical throughput limit that can be achieved based on their design methods. In the time-slotted ALOHA-based protocols [16, 42, 56, 63, 68, 71, 82], a tag transmits its ID in each time slot (or some slot in a frame) with a certain probability p until the receipt of its ID is acknowledged by the RFID reader. The reading throughput is fundamentally limited by the probabilistic collision that occurs in ALOHA-based networks. The optimal throughput is $\frac{1}{eT}$, where e is the natural constant and T is the length of a time slot [61]. It is achieved when p is chosen such that the probability for exactly one tag transmitting in each slot is 36.8%. The other major class is the tree-based protocols, which organize the reading process in a binary tree structure [9, 51, 83] and improve the reading throughput by balancing the tree [3, 51, 72]. Analytical and simulation results have shown that the best performance of the tree-based protocols is comparable to the best of the ALOHA-based protocols.

To break the fundamental limit of the ALOHA-based protocols, we have to resort to fundamentally different approaches. In this work, we apply the recently-proposed analog network coding scheme [36] to RFID systems and investigate how significantly it can improve the reading throughput.

What limits the throughput of the ALOHA-based protocols? Radio collision, which happens when more than one tag transmits in a slot. The conventional wisdom is that collision slots do not carry useful information and therefore those slots are wasted. That is however not true. Recent research shows that, by embracing the interference of wireless communication, physical-layer network coding can significantly improve the network throughput [81]. In particular, the analog network coding scheme [36] has been experimentally implemented. However, its usefulness has only been demonstrated under “toy” examples.

The contributions in this work are two-fold: First, we optimally integrate analog network coding into the RFID system to maximize the reading throughput by making some collision slots almost as useful as non-collision slots (in which only one tag transmits). The difference is that the former allow the RFID reader to learn new tag IDs after some time, while the latter let the reader learn new IDs right away. Second, we demonstrate the practical value of the analog network coding research by providing an interesting application scenario.

Technically, we design the first collision-resolution tag identification protocol that establishes the engineering and theoretical foundation for integrating analog network coding into the process of tag reading. We derive the optimal system parameters for improving the reading throughput. We also reduce the protocol overhead through a framed structure and an embedded estimator for the number of tags that are currently participating in the protocol. The proposed protocol is able to efficiently utilize the information carried in collision slots and thus break the fundamental limit of ALOHA-based protocols that do not use analog network coding. Our work answers two important questions: How to optimally apply analog network coding for RFID reading? How much throughput gain can analog network coding bring? The simulation results show that the reading throughput can be improved by $51.1\% \sim 70.6\%$ when using today's analog network coding method and the throughput can be much higher if the coding method is improved in the future.

The rest of this dissertation is organized as follows. Chapter 2 proposes AIMD/QS+k, the protocol for achieving MAC-layer time fairness under location-sensitive contention. Chapter 3 proposes WPD, the protocol for improving TCP fairness among nearby WLANs. Chapter 4 proposes FCAT, the protocol for improving reading throughput in large RFID systems. Chapter 5 concludes our study.

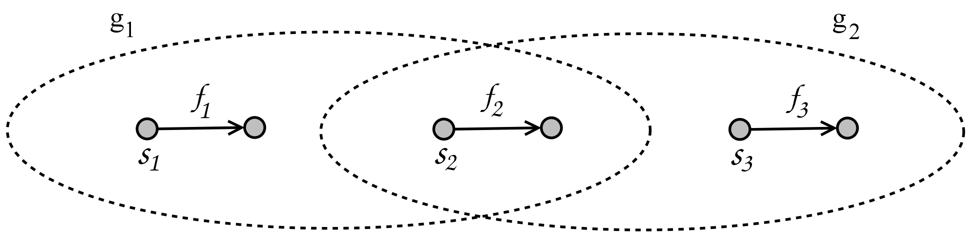


Figure 1-1. Three partially overlapping WLANs form two contention groups g_1 and g_2, where s_1, s_2 and s_3 are three access points. A TCP flow from a server on the Internet passes through an access point to a wireless client in each WLAN. Note that, only the wireless part of a flow (from an access point to a wireless client) is drawn in the figure.

CHAPTER 2
MAC-LAYER TIME FAIRNESS ACROSS MULTIPLE WIRELESS LANS

In this chapter, we study the MAC-layer time-allocation anomaly across multiple WLANs under location-sensitive contention.

With only a limited number of non-overlapping channels, hosts in nearby WLANs inevitably compete for the same channel. When a host transmits at a low transmission rate, all other contending nodes may suffer with low throughput, which is the well-known time-allocation anomaly. In addition, based on their relative spatial locations, contending hosts may have different capabilities of obtaining the channel for their transmissions. Some hosts may grab most of the channel bandwidth while the others are starved. This is called location-sensitive contention. We observe that the location-sensitive contention may push the time-allocation anomaly to the extreme, which causes unfair channel bandwidth allocation and degrades the overall network throughput. We will propose a novel solution to achieve MAC-layer time fairness.

The rest of this chapter is organized as follows. Section 2.1 presents the network model and problem definition. Section 2.2 describes the time-allocation anomaly and the location-sensitive contention. Section 2.3 discusses the related work. Section 2.4 proposes our AIMD/QS+k solution. Section 2.5 shows the simulation results. Section 2.6 gives the summary.

2.1 Network Model and Problem Definition

In this section, we give the network model and the problem definition.

2.1.1 Network Model

Consider a number of wireless access points that are deployed in an area. Each access point connects one or more wireless hosts to the Internet. Each host has a transceiver that can either transmit or receive at a time. An access point and its hosts form a WLAN. The access point selects a channel, i.e., a sub-band of the available frequency range, to communicate with its hosts. IEEE 802.11b/g has 11 channels, among which only three are non-overlapping. Transmissions in nearby WLANs that select the same or overlapping channels may interfere with one another. However, spatial channel reuse happens among distant WLANs that can use the same channel to transmit simultaneously without interference.

Fig. 2-1 shows an example, where each circle represents an access point, each square represents a host, and each solid line represents a wireless connection between a host and an access point. A dashed line between two nodes (which can be either access point or host) means that they are within each other's carrier-sensing range.

We use a node to refer to either an access point or a wireless host. The sequence of data packets sent from a node x to a node y constitute a MAC flow (x, y). The transmission range of node x defines the distance within which another node is able to decode the data transmitted by x. The interference range of node y defines the distance within which another node's transmission will interfere with y's reception of a packet from x. The carrier-sensing range of node x defines the distance within which another node's transmission will cause x to sense a busy channel. To avoid collision, it should be set no less than the maximum interference range, which can be 1.78 times the transmission range as suggested in [74] (also the default value used in ns-2).[1]

We assume a DCF-like MAC protocol. Two MAC flows contend if the following conditions are met: (1) their transmissions are made in the same channel or in channels with overlapping frequency bands that cause interference, and (2) the sender of a flow can carrier-sense the transmission of the other flow or the receiver of a flow is interfered by the transmission of the other flow. If RTS/CTS is turned on, then contention also

[1] Note that the carrier-sensing range can be artificially configured. It can be made to equal the transmission range [12]. 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.

happens if the receiver of a flow can carrier-sense the transmission of the other flow. When the two flows (x, y) and (u, v) contend, we also say that node x has a contending node u.

The transmission rate of a MAC flow (x, y) is defined as the modulation rate of the sender x, at which x transmits its bits in the channel. The channel occupancy (or occupancy for short) of a flow is defined as the fraction of a unit time during which the flow occupies the channel for transmission. It includes both the time for transmitting data packets and the time for control packets such as ACKs. The delivery rate of (x, y) is defined as the average rate at which y can successfully receive data from x. It is bounded by the transmission rate multiplied by the channel occupancy.

A maximal set of mutually contending flows is called a contention group, also known as a contention clique in the prior work [32, 33]. The flows in a contention group have to take turn to transmit. The sum of the channel occupancies for all flows in a group is called the aggregate occupancy of the group, which is bounded by one. A contention group is said to be saturated or congested if no flow can further increase its delivery rate without decreasing the rate of another flow in the group. Two flows in different contention groups may be able to transmit simultaneously due to channel spatial reuse. An example is given in Fig. 2-2, which has two contention groups: g_1 consists of (w, z) and (x, y); g_2 consists of (x, y) and (u, v). Flows (w, z) and (u, v) can transmit at the same time.

2.1.2 Problem Definition

When there is only one contention group, the time fairness problem is to equalize the channel occupancies of all MAC flows while fully utilizing the channel capacity. However, when there are more than one contention group, it is not always possible to equalize the flows' channel occupancies because each flow experiences different contention. Proportional fairness [37] has been introduced for such cases.

Let F be the set of all MAC flows, G be the set of contention groups, and $q = (q_f, f \in F)$ be a vector of feasible channel occupancies, such that $q_f \geq 0$, $\forall f \in F$, and $\sum_{f \in g} q_f \leq Q_g$, $\forall g \in G$, where Q_g is the maximum aggregate occupancy that g can have before it is congested. Q_g will be smaller than one due to the protocol overhead of DCF. A feasible vector q^* is proportionally fair if for any other feasible vector q, the sum of the proportional changes is non-negative [37], i.e.,

$$\sum_{f \in F} \frac{q_f^* - q_f}{q_f^*} \geq 0. \tag{2–1}$$

It has been shown in [37] that this is equivalent to find a feasible vector q that maximizes the sum of a utility function $U(q_f) = \ln(q_f)$.

$$\begin{aligned} &\mathit{maximize} \sum_{f \in F} U(q_f) \\ &\mathit{subject\ to} \sum_{f \in g} q_f \leq Q_g, \forall g \in G \\ &q_f \geq 0, \forall f \in F. \end{aligned} \tag{2–2}$$

Exactly solving the problem in a run-time environment is extremely hard due to the complex interaction among the contending flows, as we will elaborate in this work. Our goal is to approximate the proportional time-fairness through a fully distributed, DCF-compatible solution. DCF-compatibility means that the solution does not require any modification to the DCF protocol or its random backoff algorithm, but it may retrieve some state information from the MAC layer and modify some MAC parameters (such as the size of the minimum contention window) on the fly.

We stress that the concept of contention group is introduced only to help us describe our solution. The operations in our distributed solution never need to actually identify which flows belong to each contention group or know the value of Q_g. In fact, Q_g does not even have to be a constant. As it may drift over time, the optimal value for q_f will also evolve. Our solution will dynamically converge the channel occupancies of the flows towards their current optimal values. Unlike the previous work that constructs the contention groups explicitly [32, 33], our solution does not do so because this is not always feasible in our general setting where nodes that contend may not know each other's identities. For example, in Fig. 2-2, if the distance between z and x is greater than the transmission range but smaller than the interference range, then x has no way to know if it contends with one link or ten links.

2.2 Time-Allocation Anomaly and Location-Sensitive Contention

2.2.1 Time-Allocation Anomaly

The time-allocation anomaly among hosts in a single WLAN is a well known fact [30]. DCF ensures that the MAC flows in a WLAN have equal chance to send their packets. In other words, each MAC flow will send roughly the same number of packets over long run. Suppose the flows have roughly the same average packet size. It will take more time for a flow with a smaller transmission rate to transmit a packet. Hence, DCF ends up giving more channel time to a MAC flow with a smaller transmission rate, which effectively keeps the channel operate at the smaller rate more often, reducing the WLAN's overall throughput.

We demonstrate this anomaly through a simulation on the WLAN in Fig. 2-3, which has two MAC flows, (x, y) and (x, z), whose transmission rates are 11 Mbps and 1 Mbps, respectively. All simulations in the work are performed in ns-2 [1]. The length of each wireless link is 150m, the interference range is 1.78 times the length of the wireless link [74], the transmission range is 250m, and the carrier-sensing range is 550m. The packet size is 1000 bytes. Without (x, z), when (x, y) is the only active flow, its delivery rate is 622 packets per second. Now with (x, z), one would expect the delivery rate of (x, y) will be cut by half. However, Fig. 2-4A shows that the delivery rate of (x, y) is merely 96 packet per second. Fig. 2-4B shows that its channel occupancy is just 0.12. (The sum of the two flows' channel occupancies is below one because of the protocol overhead at the MAC and physical layers.)

2.2.2 Location-Sensitive Contention

When MAC flows in nearby WLANs contend in the same channel, the relative positions of the WLANs and their hosts can give some hosts much greater capability of occupying the channel than others. This is called location-sensitive contention. Due to the location-sensitive contention, we observe that time-allocation anomaly exists among hosts in nearby WLANs even when the transmission rates of all hosts are the same. We perform simulation on the network in Fig. 2-5, which models the scenario of two neighboring homes whose WLANs operate in the same channel. When the transmission rates of both links are 11 Mbps, the simulation result in Fig. 2-6A demonstrates that the channel occupancies of the two flows are very different and dependent on the distance between y and u. The hidden-terminal problem arising in Fig. 2-5 was first documented and analyzed in [10], which, however, does not consider the fact that the carrier-sensing range and interference range are greater than the transmission range. Readers are referred to [34] for a much more detailed explanation on the cause of location-sensitive contention in this network.

If the hosts that are more capable of obtaining the channel have low transmission rates, their channel occupancies can become so high that other hosts will be totally starved. If we add different transmission rates on top of location-sensitive contention, they drive the time-allocation anomaly problem to the extreme. In Fig. 2-5, when the transmission rate of (x, y) is 11 Mbps but the transmission rate of (u, v) is 1 Mbps, the simulation result in Fig. 2-6B shows that the channel occupancy of (x, y) is almost zero when the distance between y and u is less than 250 m.

2.3 State of the Art

Some existing time-fairness solutions [30, 66, 67] rely on a central coordinator, which is not practically feasible when we deal with multiple WLANs that contend but may not be able to communicate due to the disparity between the carrier-sensing range and the transmission range. Below we survey the distributed solutions, which are classified into several types.

2.3.1 Type-I: Assume Knowledge of Contending Flows

A Type-I solution assumes that each node has certain knowledge about its contending nodes. An example is the recent paper of TFCSMA [35]. It assumes that each node knows the number of contending nodes, which is needed in computing the node's target throughput (equivalent to the delivery rate in this work).

However, in a wireless network, it is difficult to learn the set of contending flows and keep track of which flows are currently active in sending data. Explicit exchange of control packets or implicit overhearing requires the nodes to be within each other's transmission range. However, contention is defined by the carrier-sensing range, which is much greater than the transmission range. Consider an example where the carrier-sensing range is twice of the transmission range. Because the area outside of the transmission range but within the carrier-sensing range is three times the area within the transmission range, the number of contending nodes that cannot be identified can be much greater than the number of flows that can be identified, which seriously affects the performance of TFCSMA. The similar problem exists for the solutions [32, 46, 47, 69, 75], which are designed for throughput fairness instead of time fairness. They require information collected through overhearing, but not all contending nodes can be overheard.

2.3.2 Type-II: Assume Same Channel Perception

A Type-II solution does not require a node to have any knowledge about its contending nodes, but assumes that the contending nodes will sense the same channel condition. An example is Idle Sense [28, 31], which uses a non-DCF protocol to adapt each node's contention window, such that the mean number of idle slots between two transmissions in the channel is maintained at a certain desirable value, e.g., 5.6 for 802.11b. It works well for a single WLAN, but not for multiple WLANs. First, Idle Sense requires all nodes to sense the same idle time in the channel. In Fig. 2-5, suppose node u is within the interference range of y but outside the carrier-sensing range of x. Even when the channel is saturated by the transmission on (u, v), node x will sense an idle channel! Second, even if all nodes sense the same idle time, Idle Sense may still work poorly for multiple WLANs due to location-sensitive contention. We simulate Idle Sense on the network in Fig. 2-5, where x and u are outside each other's transmission range but within the carrier-sensing range. The result is shown in Fig. 2-7. Idle Sense actually starves flow (x, y) in this scenario.

The work by Nandagopal et al. [52] also falls in this category. It uses a non-DCF time-slotted contention resolution protocol and assumes that the senders of all flows in a contention group will have the same transmission failure probability, which is also not true among contending hosts in different WLANs [34]. Consider the network in Fig. 2-5, where u is outside of the carrier-sensing range of x but within the interference range of y. When u transmits a data packet, x will sense an idle channel and transmit, which always result in a transmission failure. When x transmits a data packet, if u also transmits, it will not lead to a transmission failure (unless the ACKs from y and v happen to be transmitted at the same time, which is a rare case).

2.3.3 Type-III: Assume no Knowledge of Contending Nodes and Different Channel Perception

A Type-III solution neither requires a node to have any knowledge about its contending nodes, nor assumes that the contending nodes will sense the same channel condition. One example is to fragment the packets [20]. Flows with smaller transmission rates will have smaller fragment sizes. If the MAC flows transmit for roughly the same number of times and each time transmit one fragment, then their channel occupancies can be equalized. The opposite solution is to aggregate the packets [59, 62]. Flows with larger transmission rates will have larger packet aggregates. Again, if the MAC flows transmit for roughly the same number of times and each time transmit one aggregate, then their occupancies can be equalized. Both solutions use the number of bits in each transmission to compensate the difference in the transmission rate. Their assumption that DCF gives all MAC flows equal chance to transmit is true in the symmetric setting when the flows belong to the same WLAN. However, it is not true in asymmetric settings when the flows belong to different WLANs. Consider the network in Fig. 2-5. Suppose the transmission rates of two flows are both 11 Mbps and the packets have the same size. The above solutions are reduced to DCF. The simulation result is shown in Fig. 2-6B. The channel occupancies of the flows are very different. Flow (u, v) must have transmitted for many more times than flow (x, y) in order to produce its large occupancy.

Another solution, CWSP (Contention Window Scaling Protocol) [38], is to make the size of the minimum contention window inversely proportional to the transmission rate. It decreases the probability for a flow with a small transmission rate to obtain the channel and consequently reduces its channel occupancy. However, this heuristic approach cannot bring quantitative precision and does not always work well. We simulate CWSP on the network in Fig. 2-5, where the transmission rates of (x, y) and (u, v) are 11 Mbps and 1 Mbps, respectively. The results are shown in Fig. 2-8, which exhibits reverse discrimination — the channel occupancy of the 11-Mbps flow (x, y) is about twice the occupancy of the 1-Mbps flow (u, v).

PISD [34] applies proportional increase synchronized multiplicative decrease to control the rate of each flow. Its main objective is to achieve weighted throughput fairness although it may be extended to approximate time fairness. However, it makes an assumption that all flows belong to a single contention group. When the contention group is saturated, any node that detects the congestion will jam the channel to help other nodes also detect the congestion. This ensures that the nodes will perform multiplicative decrease simultaneously. However, because all nodes will participate in jamming, for a large deployment of WLANs that form many partially-overlapping contention groups, jamming will spill out to other contention groups that are not saturated. It leads to cascaded jamming. It begins from one node at a congested hot spot. Its jamming causes the nearby nodes to detect congestion because they can hardly send out packets. When these nodes start their jamming, the nodes further away will feel the impact. As the process repeats, nodes outside of the congested contention group will falsely detect congestion and unnecessarily reduce their rates.

The above observation is confirmed by the simulation on the network in Fig. 2-9, which has two overlapping contention groups: g_1 has two mutually contending flows and g_2 has five flows. The two groups share a common flow (x, y). One would expect the delivery rate of (w, z) will be high because channel spatial reuse occurs between (w, z) and (u, v_i), $1 \leq i \leq 4$, since they do not contend. Let the transmission rates of all flows be 11 Mbps. The simulation result is shown in Table 2-1. Due to cascade jamming, the delivery rate of (w, z) is comparable to the rates of the flows that belong to the bottleneck g_2. The total occupancy of g_2 is high. The total occupancy of g_1 is however very low. The reason is that, when g_2 is congested and x performs jamming, not only u in g_2 will feel it, but w in g_1 will also feel it.

2.3.4 Other Related Work

There is a wealth of theoretical study on utility optimization and proportional fairness [37, 40, 45, 48]. Tremendous progress has been made to apply this theory in wireless networks. However, the existing work assumes different network models and thus cannot be directly applied to solve our problem. Some existing solutions require a centralized, NP-hard scheduling algorithm for all wireless links [23, 43, 53]. Others assume a time-slotted cellular network model [22] or assume a node-exclusive interference model [11, 17, 44] where links can transmit simultaneously as long as they do not share a common node.

There is a large body of work on rate fairness in multihop wireless networks (MWN). It may appear that the WLAN networks studied in this work are special cases of the general MWNs. This is not true. They are different problems and the WLAN networks may pose greater challenges. For one, neighbors in an MWN can exchange information (such as the case in [58, 75]), whereas the contending hosts in nearby WLANs may be too far to communicate and the WLANs can be viewed as components of a partitioned network.

2.4 A Solution for MAC-layer Time Fairness across Multiple WLANs

In this section, we propose our time fairness solution for WLANs, whose design is based on two novel techniques, called channel occupancy adaptation and queue spreading, which together approximately achieve proportional fairness among hosts in contending WLANs.

2.4.1 Overview

The basic idea of our solution is to apply the AIMD (additive increase multiplicative decrease) adaptation on channel occupancy in each contention group. More specifically, all flows in a contention group perform AIMD on their channel occupancies. When the aggregate occupancy of the group does not cause channel congestion, each flow in the group will increase its channel occupancy by a constant amount, which is additive increase. When the aggregate occupancy of the group becomes too large and causes channel congestion, all flows in the group will decrease their channel occupancies by a certain percentage, which is multiplicative decrease.

If there is only one contention group, then AIMD will indeed equalize the channel occupancies of all flows. The gap between the largest occupancy l and the smallest occupancy s among all flows stays the same after each additive increase but shrinks after each multiplicative decrease. That is because, as both l and s are reduced by the same percentage, l is reduced by a larger absolute amount, and hence their gap shrinks. That gap diminishes over time after multiplicative decrease is performed periodically.

When there are multiple contention groups and each node may participate in more than one group, then AIMD — which is performed independently in each group — may not equalize the channel occupancies of all flows. As we will discuss in Section 2.4.5, it approximately achieves proportional fairness. For example, in Fig. 2-9, flow (x, y) participates in two contention groups. In proportional fairness, the channel occupancy of (x, y) will be modestly smaller than those of other flows in g_2 because (x, y) also participates in multiplicative decrease in g_1 (which happens less frequently because g_1 only has two flows and it takes longer time for additive increase to congest the channel). This is generally regarded as a positive feature because it strikes a balance between fairness and throughput improvement. A smaller (but not too small) channel occupancy for (x, y) improves the channel spatial reuse between (w, z) and (u, v_i), $1 \leq i \leq 4$, as they can transmit simultaneously.

Our idea for time fairness faces two major technical problems. The first problem is how to actually perform AIMD on channel occupancy. The second problem is most challenging: When a contention group is congested, we want all flows in the group and only the flows in the group to perform multiplicative decrease. Consider the example in Fig. 2-9. Suppose g_2 is congested. Among all flows in g_2, (x, y) resides at the most disadvantageous location. As (x, y) is least capable of obtaining the channel, it will be the first to detect the channel congestion. If x performs multiplicative decrease to resolve the congestion, other flows in g_2 will continue performing additive increase to pick up the bandwidth it has given up. Hence, node x must perform certain operations that cause u (but not w) to also detect the congestion. Recall that the nodes, x, u and w, may not be able to directly communicate or overhear each other. The rest of the section will solve these problems.

2.4.2 Release Rate and Channel Occupancy Adaptation

The control function of the proposed solution resides at the network layer. It does not require any modification to the DCF protocol, but must be able to query for the queue length at the MAC layer and modify the size of the minimum contention window. It adapts the channel occupancy of a flow by controlling the release rate, at which the network layer releases packets into the MAC layer.

When the combined release rate of all flows in a contention group is small, the channel can forward all packets released to the MAC layer. Hence, the delivery rate is equal to the release rate, and the MAC-layer queue remains empty. The flows can improve their channel occupancies by releasing more packets into the MAC layer. When the combined release rate of the flows in the group becomes too high and the aggregate channel occupancy is too large such that the channel is saturated, not all packets released to the MAC layer can be forwarded. Excess packets have to be buffered at the nodes' MAC-layer queues. Eventually, one node will observe that its queue length persistently grows and exceeds a threshold. The node will perform the operation of queue spreading (in the next subsection), which ensures that the queue lengths at all other nodes in the congested contention group will also exceed the threshold (such that they all detect the congestion). Then, the nodes should reduce their channel occupancies by decreasing the release rates, and fewer packets are made available to the MAC layer in order to relieve the congestion.

Intuitively, the queue-length threshold for congestion detection should be proportional to the transmission rate of the flow. That is because, under time fairness, a flow with a low transmission rate will have a small release rate, which would make its queue harder to pass the threshold (thus harder to detect congestion) if the threshold was a constant. Hence, we define the threshold as $H \times r$, where H is a system wide constant and r is the flow's transmission rate.

The protocol for channel occupancy adaptation is given as follows: Consider an arbitrary MAC flow (x, y) with a transmission rate r. The sender x adapts its channel occupancy after each time period of T. If its queue length at the MAC layer is below the threshold $H \times r$, it will additively increase its channel occupancy by a constant α.

If the queue length reaches the threshold, it will multiplicatively decrease its channel occupancy by a percentage β. To implement multiplicative decrease on the channel occupancy, x simply reduces its release rate by a percentage β. To implement additive increase on the channel occupancy, x increases its release rate by $\alpha \times r$ (because it takes α time to transmit $\alpha \times r$ bits at rate r). Hence, while all flows share the same parameter α, the increments for their release rates, $\alpha \times r$, will be proportional to their transmission rates.

2.4.3 Queue Spreading

We propose a new technique, called queue spreading, which makes sure that the senders of all flows in a congested contention group will detect the congestion and they will perform multiplicative decrease together.

The aggregate release rate $R_g(t)$ of all flows in a contention group g is a function of time t. $R_g(t) = \sum_{f \in g} R_f(t)$, where $R_f(t)$ is the release rate of flow f. Let $C(t)$ be the maximum throughput that the group can possibly obtain from the channel at time t. We stress that $C(t)$ is only needed to describe our idea. The operation of queue spreading does not rely on the knowledge of $C(t)$. When $R_g(t)$ exceeds $C(t)$, 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 $R_g(t) - C(t)$. The problem is that, due to location-sensitive contention, most excess packets may be queued up at one flow that is least capable of accessing the medium. While that flow observes its queue length exceeds the threshold and performs multiplicative decrease, other flows 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.

Our solution to the above problem is to spread the excess packets among the queues of all flows in the group. For an arbitrary flow (x, y) whose transmission rate is r, whenever the packet queue at the sender x exceeds $H \times r$, x will temporarily modify its MAC parameters to increase its ability of obtaining the medium, such that its queue length can be reduced back to $H \times r$. When the queue length becomes $H \times r$, 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 $H \times r$ 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 [34] tries to occupy the channel as much as possible, and consequently it will affect all neighboring nodes, including those outside of the saturated group. On the contrary, a node that performs queue spreading only tries to match its sending rate with its release rate such that the local queue does not grow further. Therefore, it will not affect the neighbors outside of the saturated group. Let r_f be the transmission rate of f. We have the following proposition.

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) $R_g(t) - C(t) > 0,\ \forall t \in [t_0, t_1)$, (2) $\int_{t_0}^{t_1}(R_g(t) - C(t))dt \geq \sum_{f \in g} H \times r_f$, 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 time t_1. On one hand, the flows' packet queues are shorter than their respective thresholds. Thus, the total number of packets in their queues is less than $\sum_{f \in g'} H \times r_f$. On the other hand, by performing the operation of queue spreading, the flows in $g - g'$ are more capable of obtaining the medium than those in g'. Hence, they can control their queue lengths to the threshold values at the expense of the flows in g', whose queues will be growing. The total number of packets queued at the flows in $g - g'$ is $\sum_{f \in g - g'} H \times r_f$. During the time period $[t_0, t_1)$, by Condition (1), the total number of

excess packets that are queued by all flows must be $\int_{t_0}^{t_1}(R_g(t) - C(t))dt$. Therefore, the number of packets queued at the flows in g' must be $\int_{t_0}^{t_1}(R_g(t)-C(t))dt-\sum_{f\in g-g'} H\times r_f$. By Condition (2), this number is no less than $\sum_{f\in g'} H \times r_f$, leading to the contradiction.
□

It is easy to see that, if the congestion is detected by the senders of all flows in a group, then the total number of excess packets in their queues that the channel cannot deliver is at least $\sum_{f\in g} H \times r_f$. We have the following necessary condition for congestion detection.

Proposition 2: Suppose the senders of all flows in a contention group g have empty queues at time t_0 and $R_g(t) - C(t) > 0,\ \forall t \in [t_0, t_1)$. If all flows detect the congestion of g by time t_1, then we must have $\int_{t_0}^{t_1}(R_g(t) - C(t))dt \geq \sum_{f\in g} H \times r_f$.

2.4.4 AIMD/QS+k

We need to integrate queue spreading (QS) into the channel-occupancy adaptation protocol in Section 2.4.2. However, a straightforward combination of the two will not do the trick. The first integrated protocol, called AIMD/QS, is given as follows: Each flow (x, y) performs the AIMD channel occupancy adaptation periodically as described in Section 2.4.2. In addition, when the queue length at the sender x reaches the threshold, x performs queue spreading until the end of the current period T when it does multiplicative decrease. During the operation of queue spreading, if the queue length is above the threshold, the sender x 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 the threshold, x restores the default minimum contention window. By doing so, it keeps the queue length at the threshold.

Proposition 2 states that, in order for the senders of all flows in a congested group g to detect congestion, the number of excess packets buffered by all flows must be at least $\sum_{f\in g} H \times r_f$. However, AIMD/QS has no means to guarantee that. Let (x, y) be the flow in the group that is least capable of obtaining the channel, and r be its transmission rate. In fact, as long as the number of excess packets is $H \times r$, it may happen that all excess packets end up in the queue of x, which detects congestion and performs multiplicative decrease, while other nodes have empty queues.

To solve this problem, we design a generalized protocol, AIMD/QS+k, where k is a non-negative integer. The sender of each flow carries out the operations of AIMD/QS except that, after it finds its queue length reaches $H \times r$, it will continue performing additive increase for k subsequent periods of T before making multiplicative decrease. Suppose a flow's queue reaches $H \times r$ during $[t_0, t_0 + T)$. It will increase the release rate at times $t_0 + T$, $t_0 + 2T$, ..., $t_0 + kT$ by an amount $\alpha \times r$, and then decrease the release 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. The following proposition gives the formula for picking the parameters of AIMD/QS+k, such that when a contention group is congested, all flows in the group will detect the congestion and thus perform multiplicative decrease. (The flows outside of any congested group will not do that because there are no excess packets to push their queues over the threshold.)

Proposition 3: AIMD/QS+k ensures the detection of congestion by all flows in a congested group if

$$H \leq \frac{k(k-1)}{2}\alpha T, \text{ for } k \geq 2. \tag{2–3}$$

Proof: Consider an arbitrary time $t = 0$. Let g be the first contention group that becomes congested after $t = 0$. When g becomes congested, its constituent flows release more packets than the channel can deliver. The excess packets will eventually push the queue length of a flow (u, v) in g over the threshold. Without losing generality, suppose it happens during $(iT, (i + 1)T]$. Let C be the channel capacity that can be maximally obtained by the group g at the moment. Clearly, $R_g((i+1)T) > C$. Otherwise, there would be no excess packets to push the queue length of (u, v) to the threshold.

Flow (u, v) will be the first one to perform multiplicative decrease and that will happen at time $t = (i + k + 1)T$. Hence, $R_g(.)$ is a non-decreasing function before $t = (i + k + 1)T$. Because the release rate of each flow f in g increases by $\alpha \times r_f$ after each period T, we must have $R_g((i+1+j)T) = R_g((i+1)T) + j\sum_{f\in g} r_f\alpha$. Hence,

$$\begin{aligned}
&\int_{(i+1)T}^{(i+k+1)T}(R_g(t) - C)dt \geq \sum_{j=0}^{k-1}(R_g((i+1+j)T) - C)T \\
&= \sum_{j=0}^{k-1}(R_g((i+1)T) - C)T + \sum_{j=0}^{k-1} j \sum_{f\in g} r_f\alpha T \\
&> \sum_{j=0}^{k-1} j \sum_{f\in g} r_f\alpha T \;=\; \frac{k(k-1)}{2}\sum_{f\in g} r_f\alpha T.
\end{aligned} \tag{2–4}$$

If (2–3) is met, we will have $\int_{(i+1)T}^{(i+k+1)T}(R_g(t) - C)dt > \sum_{f\in g} r_f \cdot H$. Hence, by Proposition 1, AIMD/QS+k makes sure that all flows in g detect the congestion. $\square$

2.4.5 Proportional Fairness and Interpretation of AIMD/QS+k

In their seminal paper [37], Kelly, Maulloo and Tan show that there exist fully distributed algorithms that achieve global optimization of the system utility. Below we rewrite their primal algorithm in our notations (with simplification of removing the weight). Let F be the set of all flows and G_f be the set of contention groups to which flow f belongs.

$$\frac{d}{dt}q_f(t) = \alpha - q_f(t) \times \frac{1}{\varepsilon^2}\sum_{g\in G_f}(\sum_{f'\in g} q_{f'}(t) - Q_g + \varepsilon)^+. \tag{2–5}$$

The price functions $(\sum_{f'\in g} q_{f'}(t) - Q_g + \varepsilon)^+$ for $g \in G_f$ take the form suggested in [37], such that when $\varepsilon \to 0$, the above adaptation, when performed independently by the senders of all flows, will maximize the system utility, $\sum_{f\in F} \ln q_f$.

AIMD/QS+k can be interpreted as a discrete approximation of (2–5). The first item on the right side of (2–5) suggests additive increase. AIMD/QS+k increases q_f by a constant amount α after each time period of T. The second item on the right side suggests multiplicative decrease. $\sum_{g\in G_f}(\sum_{f'\in g} q_{f'}(t) - Q_g + \varepsilon)^+$ is the penalty factor. AIMD/QS+k decreases q_f by a percentage β when the accumulated penalty $\int_{t_0}^{t_1}\sum_{g\in G_f}(\sum_{f'\in g} q_{f'}(t) - Q_g + \varepsilon)^+ dt$ reaches a threshold, where t_0 is the time when the previous multiplicative decrease is performed and t_1 is the current time.

The key is to measure the penalty for each contention group g, $\int_{t_0}^{t}(\sum_{f'\in g} q_{f'}(t) - Q_g + \varepsilon)^+ dt$, which becomes positive only when g is congested. It is likely that, for any flow f, only one group in G_f is congested at a time if α is chosen small such that a contention group will spend much more time without congestion than time with congestion.

Interestingly, the price can be indirectly measured through the local queue length. Let $R_f(t)$ be the release rate of flow f and $r_f(t)$ be the transmission rate. In AIMD/QS+k, the channel occupancy q_f is controlled through the release rate R_f. In (2–5), q_f represents the resource demand of flow f. It is the channel occupancy that the flow demands in order to transmit all released packets. Hence, q_f is proportional to R_f (because each bit takes the same amount of time to transmit). After g is congested, further additive increase made by AIMD/QS+k will linearly increase the release rate (or the channel occupancy) of each flow. In the mean time, the number of excess packets that have to be buffered will increase proportionally. This implies that, approximately, the combined queue length for all flows in g grows at a speed proportional to the amount of excess channel occupancy, $(\sum_{f'\in g} q_{f'}(t) - Q_g)^+$. Furthermore, the technique of queue spreading ensures that the senders of all flows will see the same price as the excess packets are spread among the queues to push all of them over the threshold. Here, the price takes a discrete form, 0 if the threshold is not reached and 1 if the threshold is reached.

2.5 Simulation

We perform extensive simulations on various scenarios to evaluate the performance of AIMD/QS+k in achieving MAC-layer time fairness. We compare AIMD/QS+k with some existing work, including 802.11 DCF, CWSP [38], and Idle Sense [31].

All simulations are performed on ns-2. AIMD/QS+k works on top of DCF. The parameters of DCF use the default values set by ns-2 according to the protocol standards. For the channel occupancy adaptation, $\alpha = 0.03$, $\beta = 0.5$, and $T = 1$ second. The parameters for queue spreading are determined based on Proposition 3. In particular, we choose $k = 2$ and $H = 0.03$ seconds, which satisfy (2–3) in the proposition. The parameters of other solutions are chosen based on the original papers.

We use three simulation scenarios with increasing complexity. The results from a simple scenario are easier to interpret, while the results from a complex scenario are closer to what will be seen in practice.

2.5.1 One Contention Group

The first simulation scenario is based on the network of Fig. 2-5, which contains only one contention group of two MAC flows, (x, y) and (u, v). The length of each wireless link is 150m, the interference range is 1.78 times the length of the wireless link [74], the transmission range is 250m, and the carrier-sensing range is 550m. The packet size is 1000 bytes. These parameters will also be used in other scenarios.

Fig. 2-10 compares the channel occupancies of the flows achieved under DCF, CWSP, Idle Sense, and AIMD/QS+k, respectively, with the transmission rate of (x, y) being 11 Mbps and the rate of (u, v) being 1 Mbps. Under DCF, the channel occupancy of (u, v) is much higher than the occupancy of (x, y). Under CWSP, the situation is opposite. The occupancy of (x, y) is better. Idle Sense performs very well until the distance is beyond 250m, where (u, v) is totally starved.

In real deployment of WLANs, RTS/CTS is often turned off. Our simulations find that the network throughput is consistently higher without RTS/CTS. The reason is that, according to the 802.11 standard, RTS/CTS are sent at the lowest transmission rate. Hence, they will take 656 μs at 1 Mbps, which represent significant overhead, considering that it only takes 940 μs to transmit a data packet of 1,000 bytes at 11 Mbps. We turn off RTS/CTS and re-run the simulation. The result is shown in Fig. 2-11.

Indeed, the aggregate channel occupancy is improved in each case. AIMD/QS+k can maintain time fairness. The performance of Idle Sense improves when the distance is greater than 250m, but degrades when the distance is smaller than 250m as the 11-Mbps flow (x, y) is depressed by the 1-Mbps flow (u, v).

We change the transmission rate of (u, v) from 1 Mbps to 5.5 Mbps and re-run the above simulation (with RTS/CTS turned off). The result is shown in Fig. 2-12. Again, only AIMD/QS+k achieves time fairness under location-dependent contention for all distance values.

2.5.2 Two Contention Groups

The second simulation scenario uses the network of Fig. 2-9, which has two contention groups. Group g_1 contains two flows, (x, y) and (w, z). Group g_2 contains five flows, (x, y), (u, v_1), (u, v_2), (u, v_3), and (u, v_4). The length of each wireless link is 150m. The distance between z and x is 200m. The distance between y and u is also 200m. The transmission rate of (w, z) is 2 Mbps. The transmission rate of (u, v_3) is 1 Mbps. The transmission rates of other flows are 11 Mbps.

The simulation result is shown in Table 2-2. Under DCF, (x, y) is starved and the occupancies of (u, v_1), (u, v_2) and (u, v_4) are very low. Under either CWSP or Idle Sense, (x, y) is starved. Under AIMD/QS+k, (x, y) has a decent channel occupancy, even though it is smaller than others due to the nature of proportional fairness. Comparing with the result of PISD in Table 2-1, AIMD/QS+k improves the channel occupancy of (w, z) from 0.161 to 0.546. We shall not compare the delivery rates because the transmission rate of (w, z) is 11 Mbps there and 2 Mbps here. The total throughput of g_1 under AIMD/QS+k is comparable to the throughput under DCF or Idle Sense, but much better than the throughput under CWSP. The total throughput of g_2 under AIMD/QS+k is much better than the throughput under DCF or Idle Sense, but smaller than the throughput under CWSP.

2.5.3 Multiple Contention Groups

The third simulation scenario is designed based on the network of Fig. 2-13, where WLANs are deployed along two crossing streets. The relative positions of the nodes are drawn in the figure. The length of each wireless link is 150m. The distance between the closest nodes in two adjacent WLANs is 200m. The contention relationship among the flows is automatically determined in ns-2 based these parameters and those in Section 2.5.1. The transmission rates of some flows are specified in the figure, and the rates of others are 11 Mbps by default.

The simulation result is shown in Table 2-3. Under DCF, flows 5, 11 and 15 have very low channel occupancies. A few others have low occupancies. Under CWSP, flows 8 and 14 have very low channel occupancies. Under Idle Sense, flows 1, 2, 3, 4, and 9 have very low channel occupancies. This is not a surprising outcome because Idle Sense was designed to work among hosts in a single WLAN. Under AIMD/QS+k, all flows have reasonable channel occupancies. Its overall distribution of channel occupancies is much fairer than those of others. We believe this simulation result demonstrates the strong performance of AIMD/QS+k under a complex scenario.

2.6 Summary

This chapter proposes a new time fairness solution that approximates the generic adaptation algorithm for proportional fairness among multiple WLANs. The new solution addresses the problem of location-sensitive contention, and considerably outperforms the existing solutions. It is fully distributed. Each node only performs localized operations. It is DCF-compatible and only needs to modify the size of the minimum contention window.

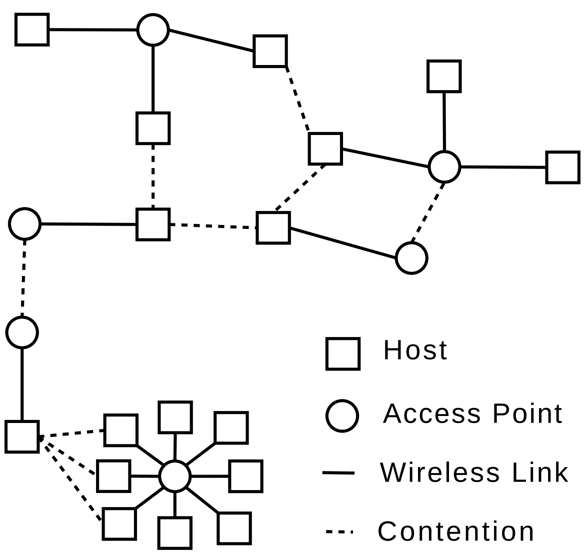


Figure 2-1. Network Model.

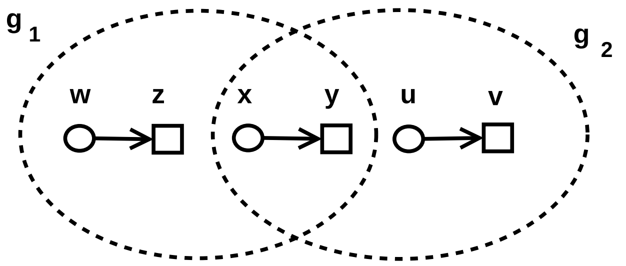


Figure 2-2. Two contention groups. Flows (w, z) and (u, v) can transmit simultaneously.

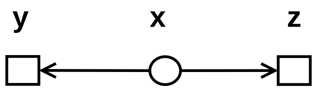


Figure 2-3. Two MAC flows in the same WLAN.

Table 2-1. Delivery rate (in packets per second) and channel occupancy under PISD on the Network of Fig. 2-9

flow	rate	occupancy	flow	rate	occupancy
(w, z)	123.26	0.161	(u, v_2)	123.27	0.161
(x, y)	112.35	0.147	(u, v_3)	123.16	0.161
(u, v_1)	123.17	0.161	(u, v_4)	123.23	0.161

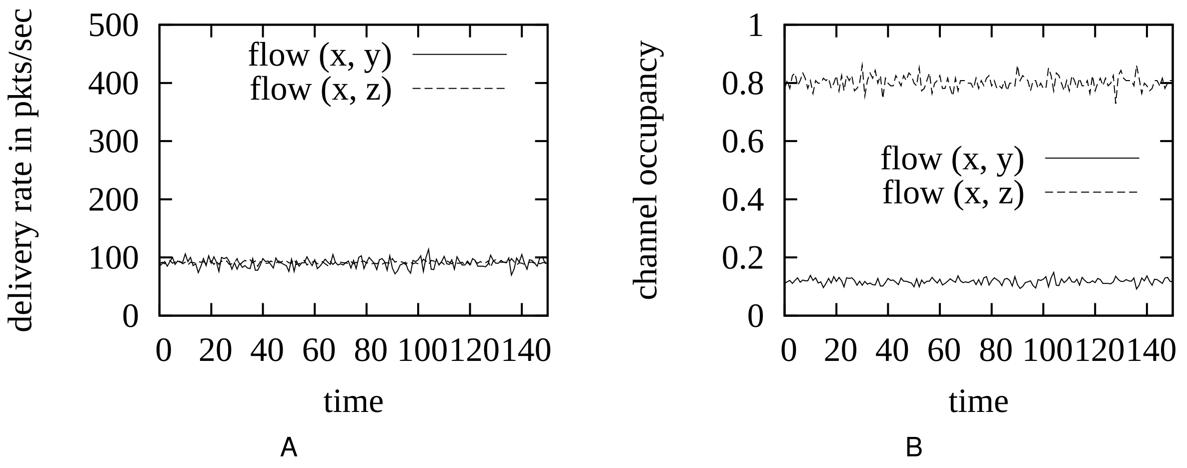


Figure 2-4. DCF on the network in Fig. 2-3 with flow (x, y) at 11 Mbps and flow (x, z) at 1 Mbps. A) The delivery rates. B) The channel occupancies.

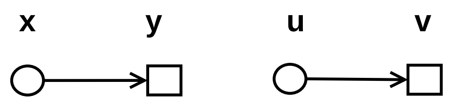


Figure 2-5. Two MAC flows in different WLANs contend.

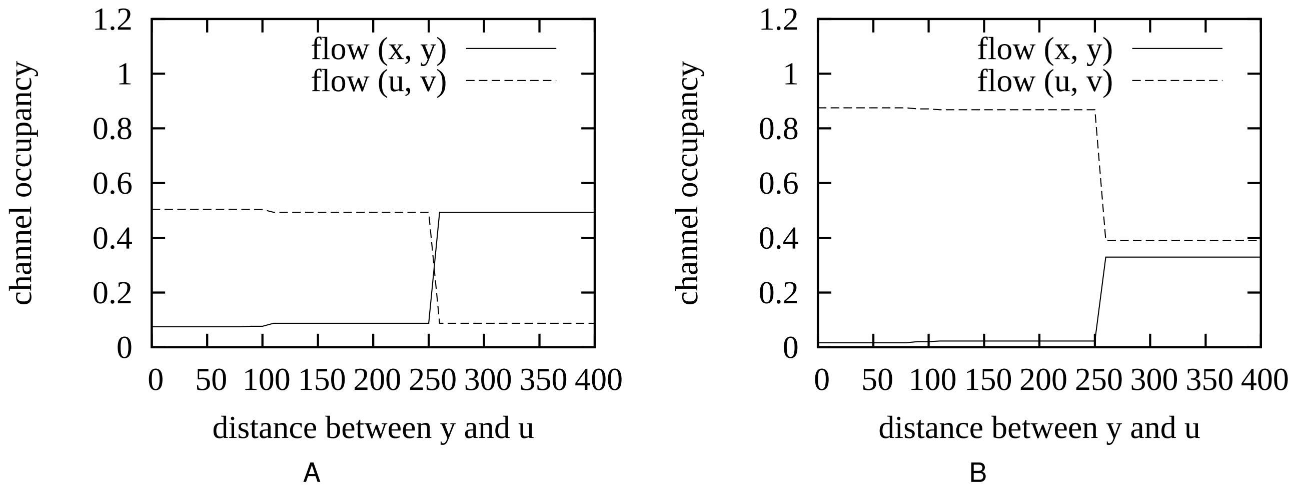
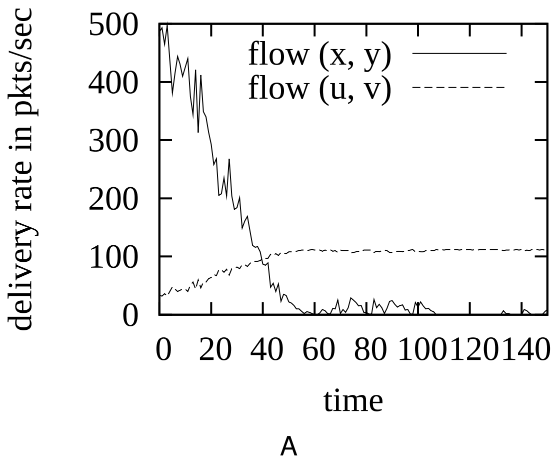


Figure 2-6. Channel occupancies of the flows in Fig. 2-5 under DCF. A) Both flows transmit at 11 Mbps. B) Flow (x, y) transmits at 11 Mbps and flow (u, v) at 1 Mbps.



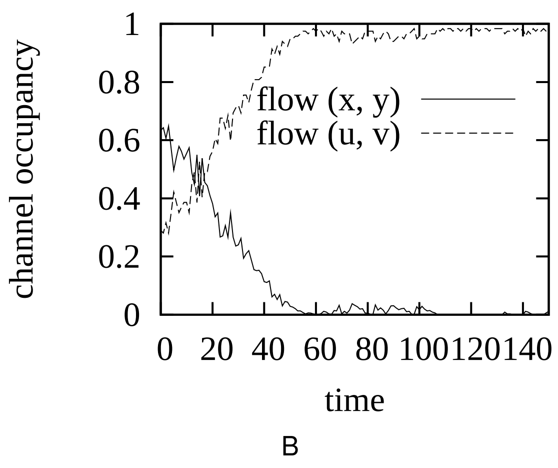
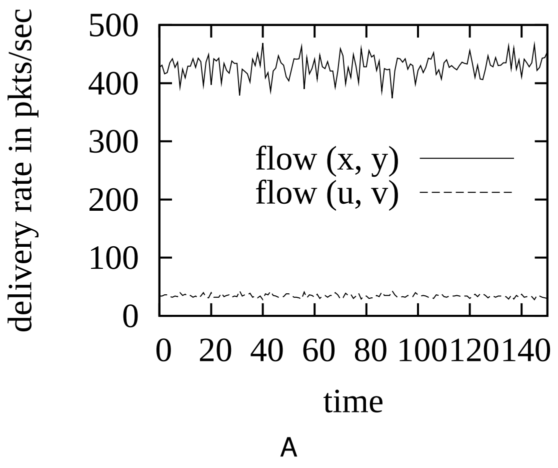


Figure 2-7. Idle Sense on the network in Fig. 2-5 with flow (x, y) at 11 Mbps and flow (u, v) at 1 Mbps. A) the delivery rates. B) the channel occupancies.



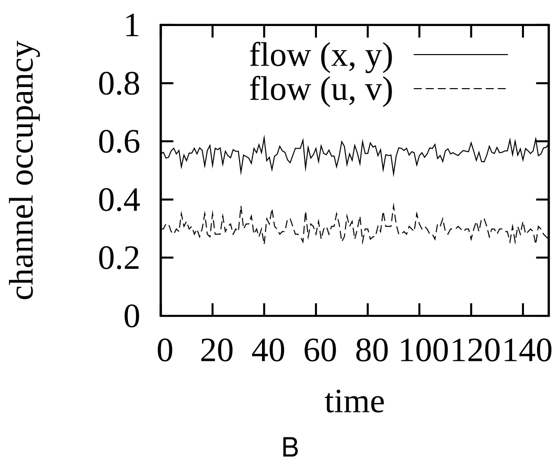


Figure 2-8. CWSP on the network in Fig. 2-5 with flow (x, y) at 11 Mbps and flow (u, v) at 1 Mbps. A) the delivery rates. B) the channel occupancies.

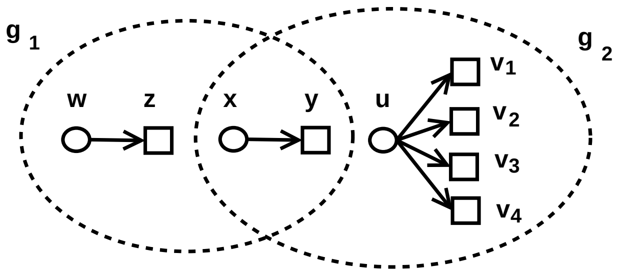


Figure 2-9. Three WLANs form two contention groups g_1 and g_2.

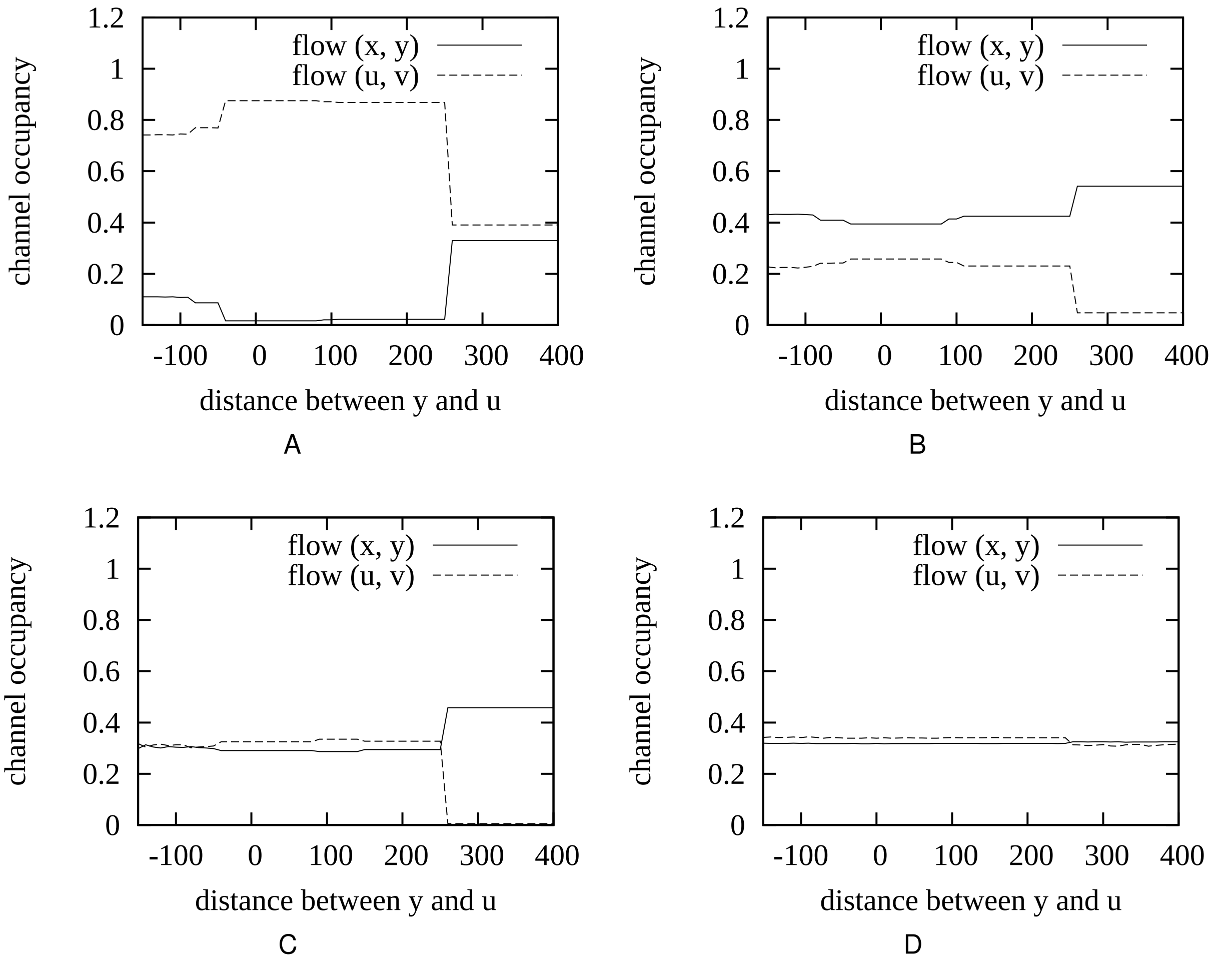
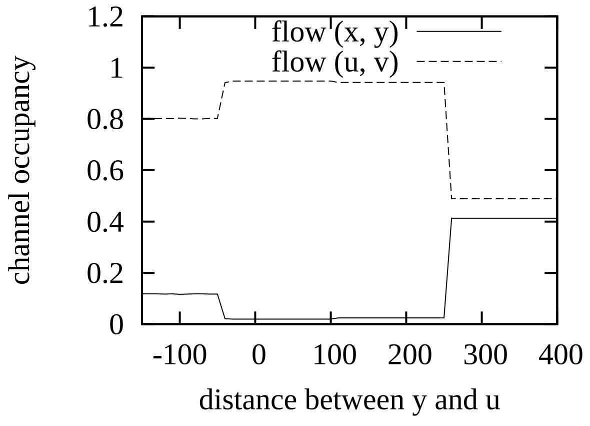


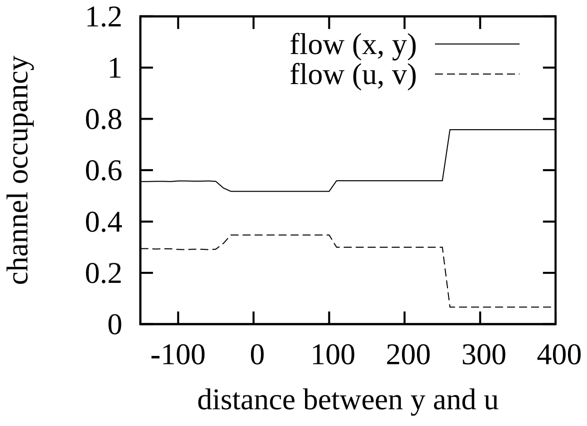
Figure 2-10. Comparing the channel occupancies of flows (x, y) and (u, v) in the network of Fig. 2-5. The transmission rates of (x, y) and (u, v) are 11 Mbps and 1 Mbps, respectively. A) Under 802.11 DCF. B) Under CWSP. C) Under Idle Sense. D) Under AIMD/QS+k.

Table 2-2. Comparing the delivery rates (in packets/sec) and the channel occupancies of the flows in the network of Fig. 2-9 under 802.11 DCF, CWSP, Idle Sense, and AIMD/QS+k.

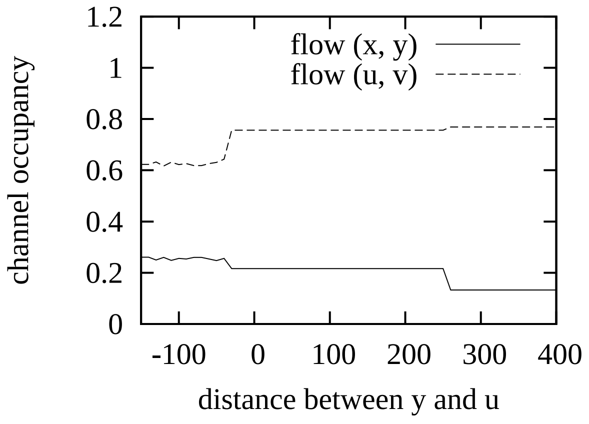
	802.11 DCF		CWSP		Idle Sense		AIMD/QS+k	
flow	rate	occupancy	rate	occupancy	rate	occupancy	rate	occupancy
(w, z)	201.08	0.865	155.64	0.670	205.83	0.886	126.83	0.546
(x, y)	0.347	0.001	5.96	0.008	2.07	0.003	75.84	0.099
(u, v_1)	66.18	0.086	183.47	0.239	107.87	0.141	138.01	0.180
(u, v_2)	64.48	0.084	182.53	0.238	112.69	0.147	138.59	0.181
(u, v_3)	67.67	0.594	14.63	0.128	42.29	0.371	22.62	0.199
(u, v_4)	67.85	0.088	183.67	0.240	109.37	0.143	138.63	0.181



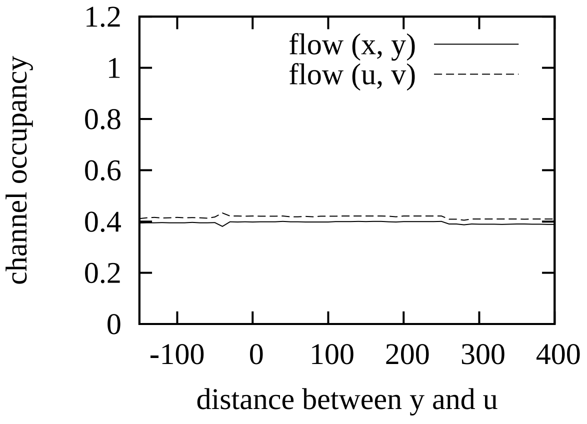
A



B



C



D

Figure 2-11. Same as the caption of Fig. 2-10, but this time RTS/CTS is turned off. A) Under 802.11 DCF. B) Under CWSP. C) Under Idle Sense. D) Under AIMD/QS+k.

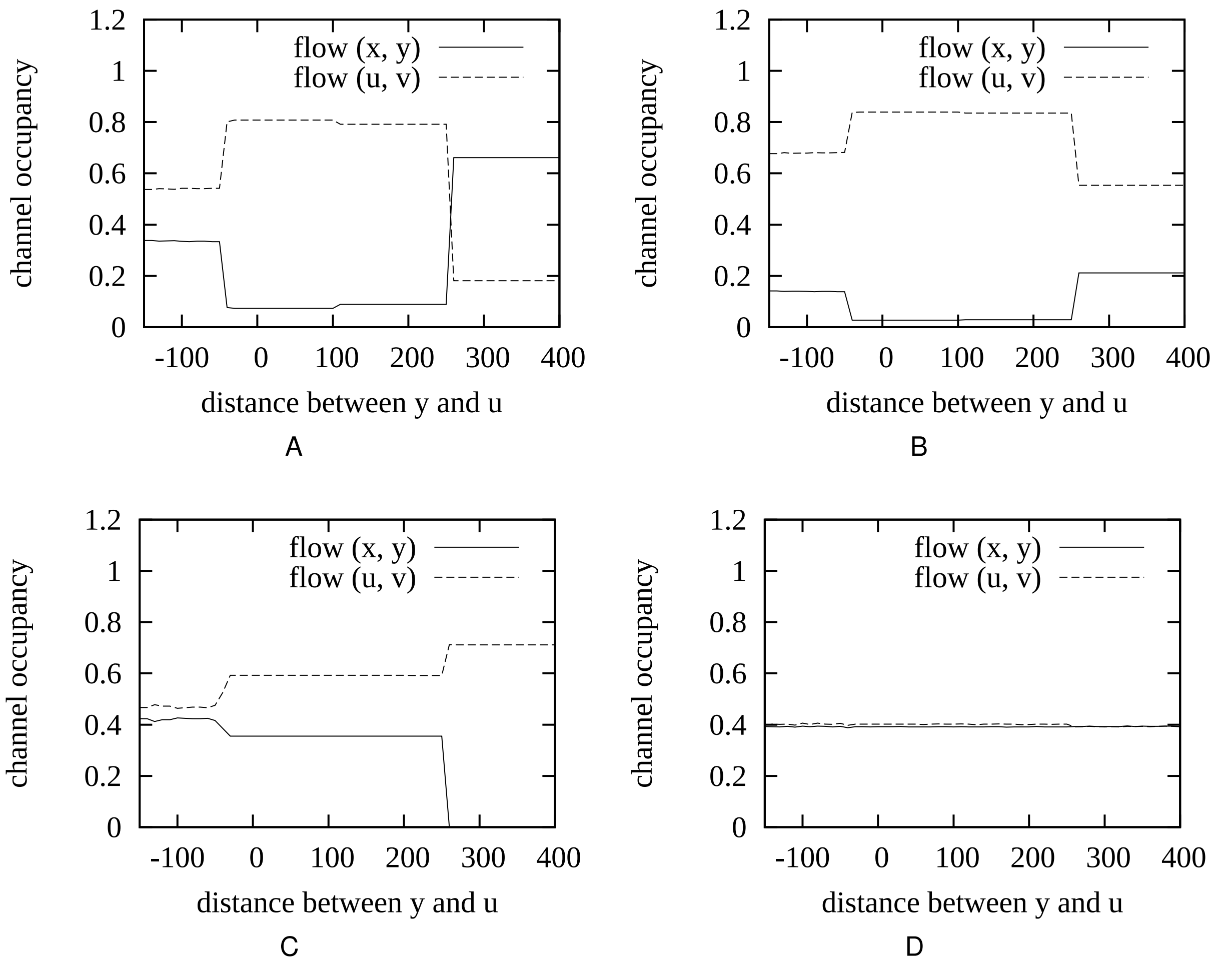


Figure 2-12. Comparing the channel occupancies of flows (x, y) and (u, v) in the network of Fig. 2-5. The transmission rates of (x, y) and (u, v) are are 11 Mbps and 5.5 Mbps, respectively. RTS/CTS is turned off. A) Under 802.11 DCF. B) Under CWSP. C) Under Idle Sense. D) Under AIMD/QS+k.

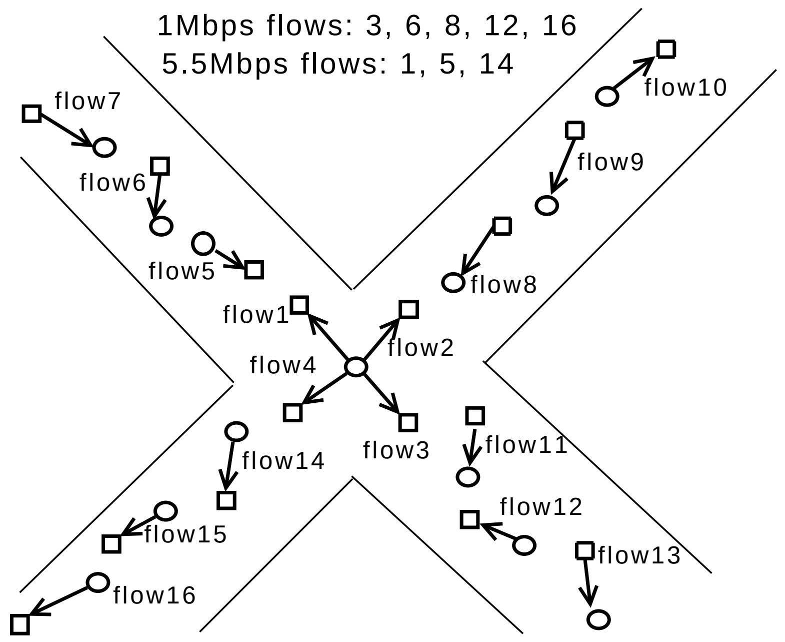


Figure 2-13. Sixteen WLANs are deployed along two streets. Unless specified in the figure, the default transmission rate of a flow is 11 Mbps.

Table 2-3. Comparing the delivery rates (in packets/sec) and the channel occupancies in the network of Fig. 2-13 under 802.11 DCF, CWSP, and AIMD/QS+k.

	802.11 DCF		CWSP		Idle Sense		AIMD/QS+k	
flow	rate	occupancy	rate	occupancy	rate	occupancy	rate	occupancy
flow1	52.88	0.108	79.79	0.164	3.29	0.007	61.12	0.125
flow2	51.18	0.067	169.37	0.221	4.54	0.006	81.74	0.107
flow3	54.45	0.452	13.23	0.116	0.193	0.002	20.59	0.181
flow4	52.27	0.068	139.29	0.182	6.25	0.008	77.97	0.102
flow5	21.41	0.044	48.55	0.099	303.07	0.622	78.03	0.160
flow6	97.18	0.853	36.03	0.316	6.85	0.060	42.51	0.373
flow7	37.29	0.076	226.47	0.464	367.19	0.753	169.03	0.347
flow8	15.11	0.133	1.92	0.017	61.01	0.536	17.31	0.152
flow9	73.08	0.095	72.82	0.095	23.01	0.031	169.95	0.222
flow10	469.73	0.613	538.69	0.702	132.73	0.173	257.31	0.336
flow11	10.35	0.013	76.78	0.100	120.99	0.158	86.21	0.112
flow12	89.71	0.788	7.51	0.066	45.31	0.398	43.30	0.380
flow13	107.07	0.140	549.74	0.717	349.47	0.456	264.33	0.345
flow14	31.41	0.064	1.87	0.004	195.93	0.402	43,48	0.089
flow15	16.82	0.022	362.29	0.472	63.43	0.083	203.11	0.265
flow16	105.77	0.929	40.15	0.352	95.09	0.835	60.01	0.527

CHAPTER 3
TCP FAIRNESS ACROSS MULTIPLE WIRELESS LANS

In this chapter, we study the transport-layer TCP fairness problem among multiple contending WLANs.

It has been observed that TCP flows in some WLANs may achieve very high throughput at the expense of starving the flows in other nearby WLANs. This unfairness may barely be noticeable if users access the network intermittently. However, as the Internet has become increasingly video-rich, unfairness at its wireless perimeter will be noticeable to the users engaging in long video-streaming sessions, such as wireless IPTV and remote camera monitoring. As we know, to ensure fair channel bandwidth allocation in wired networks, we have CSMA/CD at the MAC-layer for one-hop links and TCP at the transport-layer for end-to-end flows. Hence, after we propose AIMD/QS+k in Chapter 2 for achieving MAC-layer time fairness in WLANs, we will continue to propose a novel solution for improving fairness among contending TCP flows.

The rest of this chapter is organized as follows. Section 3.1 presents the network model. Section 3.2 describes the TCP fairness problem among contending WLANs. Section 3.3 discusses the related work. Section 3.4 proposes our new protocol for improving TCP fairness. Section 3.5 shows the simulation results. Section 3.6 gives the summary.

3.1 Network Model

In this section, we present the network model used in this chapter.

We consider a common scenario where multiple WLANs coexist in an area. A typical WLAN consists of an access point and multiple wireless clients. We assume IEEE 802.11 a/b/g DCF or other DCF-like protocols at the MAC layer. Each WLAN selects one channel (a sub-band of frequency range) for data transmissions between the access point and clients. With only a limited number of non-overlapping channels in 802.11 wireless networks, a WLAN may contend in the same channel with nearby WLANs.

We consider TCP flows (or TCP connections) between wireless clients and Internet servers. Each flow passes an Internet wired path and a final wireless link across a WLAN. TCP performs well for congestion on the wired path, but not for the wireless link, which is the focus in this work. It is well known that packet drops due to wireless transmission failures can interfere with TCP congestion signaling. This problem has been studied extensively and can be largely solved by increasing the number of retransmission attempts. This work investigates a less-studied issue, i.e., how inconsistent channel observations by different contending nodes will cause the TCP rate adaptation to fail in achieving its fairness objective (see the introduction) and how to solve this problem.

The flow rate of a TCP connection is defined as the number of packets that the wireless client successfully receives per second. The sending rate is defined as the number of packets that the source of a TCP flow (e.g., a server on the Internet) sends out per second. When a TCP flow crosses a wireless link, the channel occupancy of the wireless node that forwards the packets is defined as the fraction of time that the node occupies the channel for its data delivery (i.e., DATA/ACK exchanges).

Both access points and wireless clients are referred to as nodes. Two nodes contend if one’s data transmission can make the other sense a busy channel or corrupt the other’s data reception. A group of mutually contending nodes forms a contention group. A node may participate in multiple contention groups. We use the concept of contention group only to simplify the presentation of the work; the operations in our solution do not need to know the actual contention relationship among wireless nodes. In the discussion of a wireless node carrying a TCP connection, we often use the flow rate of the wireless node to refer to the flow rate of the TCP connection that the node carries.

3.2 TCP Unfairness among Contending WLANs

TCP performs AIMD (Additive Increase Multiplicative Decrease) on congestion window size to dynamically adjust source sending rate for achieving high throughput and fairness. Specifically, the sender of a TCP flow additively increases its sending rate to exploit residual channel bandwidth. And it multiplicatively decreases its sending rate when congestion is detected.

On the Internet, when a router is congested and begins to randomly drop packets, packet loss is felt by all TCP flows that pass through the router. Hence, multiplicative decrease will be performed at all senders. We illustrate the rates of two TCP flows over time in Fig.3-1A. The rates are normalized such that congestion occurs 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. Eventually, the rates converge to the same value after a series of multiplicative decreases. Now, consider the wireless network in Fig.3-2, where two wireless nodes are downloading from the Internet via access points s_1 and s_2, respectively. At time 6 in Fig.3-1B, when the combined rate of flow f_1 and f_2 reaches the channel capacity, because node s_2 is more capable of obtaining the channel bandwidth than node s_1 (due to the hidden-terminal problem), it can successfully forward most of its packets, while node s_1 observes buffer buildup and packet drop. Consequently, the sender of flow f_1 detects congestion and performs multiplicative decrease, while the sender of flow f_2 does not. Since flow f_1 experiences multiplicative decrease more frequently, its average rate will be smaller than that of flow f_2. The complexity of the problem will become extraordinary when we move to multiple partially-overlapping WLANs.

3.3 State of the Art

In this section, we discuss the existing solutions for improving TCP fairness in wireless networks. We first discuss why RED [25], NRED [75] and PISD [34] cannot solve the TCP fairness problem in WLANs. We then present other related work.

3.3.1 Random Early Detection (RED)

RED [25] has been widely deployed in wired networks to relieve channel congestion and enhance TCP fairness. The basic idea of RED is that, when a router detects congestion, it will probabilistically drop each new arrival packet to inform the senders of all passing TCP flows to reduce their sending rates. The flows experience packet dropping proportional to their flow rates, which improves fairness among contending flows. However, among a group of contending WLANs, there is no such a central router acting as a traffic coordinator. The TCP flows contending at the bottleneck may pass different physical nodes. These nodes may have different channel observations in terms of buffer occupancy, transmission failure rate or channel idle time. Consequently, they will provide inconsistent feedbacks to the senders of TCP flows, causing unfairness.

3.3.2 Neighborhood RED (NRED)

To solve RED's problem, Xu et al. proposed NRED [75] for wireless networks. The basic idea of NRED is to make sure all contending nodes observe the same channel condition. Specifically, all nodes monitor the channel busy/idle status and periodically synchronize their observations by explicit message exchanges. Consequently, all nodes can detect congestion simultaneously. They will then drop packets with probabilities proportional to their flow rates, which help enhance TCP fairness. However, NRED relies on the assumption that all contending nodes can overhear each other, which may not always be true. Multiple contending WLANs may be deployed in a large area such that they may contend for the same channel yet not be able to directly communicate due to the limited transmission range.

We show an example in Fig.3-3. There are three overlapping WLANs forming two contention groups. The nodes in different WLANs are outside of each other's transmission range and thus they cannot directly communicate. Fig.3-4 shows that the nodes s_1, s_2 and s_3 have quite different channel observations in terms of channel idle time. Nodes s_1 and s_3 always sense much more channel idle time than node s_2 does. Since they cannot synchronize their perceptions of the channel conditions by exchanging messages, they will experience unfair packet dropping. This unfair packet dropping will lead to different rates of the TCP flows passing s_1, s_2 and s_3. Fig.3-5 shows that f_1 and f_3 achieve very high throughput while f_2 is almost starved.

3.3.3 Proportional Increase Synchronized Multiplicative Decrease (PISD)

As we can see, the TCP unfairness stems from the MAC-layer unfairness. Can we solve the TCP fairness problem by achieving the MAC-layer fairness?

In [34], Ying et al. proposed PISD to achieve MAC-layer rate fairness in asymmetric wireless network settings. Without requiring direct communications, PISD uses a channel jamming technique to force all contending nodes to detect congestion. Then, all nodes perform synchronized rate reduction to resolve the congestion. Eventually, all flows can achieve the same flow rate. However, PISD only focuses on the MAC-layer and it lacks of interaction with TCP congestion control mechanism. For example, PISD needs to observe a certain number of queued packets to detect congestion. And it has to push out enough packets within a short period of time to jam the channel. But the sender of a TCP flow may have extremely low sending rate when the flow experiences heavy channel contention. Consequently, the node carrying this TCP flow at the bottleneck may not be able to accumulate enough packets to trigger and perform PISD's rate control function.

We perform a simulation on the network of Fig.3-6, where five TCP flows form two contention groups. The heavy channel contention may push the rate of flow f_2 to extremely low. Table 3-1 shows that the flow f_2 is still starved under PISD. This example demonstrates that MAC-layer fairness does not directly lead to TCP fairness.

3.3.4 Other Related Work

Yang et al. [78] proposed a non-work-conserving scheduling for improving TCP fairness among flows crossing wireless ad hoc networks and wired networks. The basic idea is to set a timer to control the speed of sending packets to the MAC layer. The length of the timer is determined by the flow rate. It tries to enhance fairness among contending flows by punishing high-rate flows. This approach is simple; however, it significantly downgrades the overall throughput.

Ergin et al. [21] verified the TCP performance degradation by testbed traces in an unplanned deployment of WLANs. Some works has been proposed to mitigate the interference between two nearby WLANs by transceiver parameter optimization [6, 70, 84], channel assignment [49, 50] and association control [8, 49]. However, they may cause channel underutilization or incur long delays. Moreover, they cannot solve the TCP fairness problem in a dense deployment.

There is a large body of work on TCP upstream/downstream fairness in WLANs [4, 5, 54, 57]. Pilosof et al. [57] illustrated the TCP upstream/downstream throughput anomaly within WLANs through analysis, simulation and experiment. They proposed to control the receiver window of all TCP flows at the access point side to provide more channel bandwidth for downstream flows. Park et al. [54] introduced channel access cost that is used to inform the TCP sender to adjust its sending rate to achieve per-station fairness. Aguilera et al. [5] used Idle Sense [31] method to assign sufficient bandwidth to the access point. Abeysekera [4] dynamically adjusted the minimum contention window at access points according to the ratio of the total rate of downstream flows to the rate of an upstream flow. All the above works focus on the TCP unfairness within a single WLAN and they cannot be applied to the scenarios of multiple overlapping WLANs.

Another line of research is to address fairness issues over multihop wireless networks [58, 73, 76, 77, 80]. However, the TCP unfairness among nearby WLANs studied in this work has different emphasis. In a multihop network, a flow can carry certain information for the nodes along a routing path [58, 80]. Contrarily, two WLANs may contend for the same channel but they have no means of direct communication.

3.4 Wireless Probabilistic Drop (WPD)

In this section, we first explain the basic idea behind our solution WPD (Wireless Probabilistic Drop), then introduce several rate control techniques, and finally describe how they work together to enhance TCP fairness.

3.4.1 Basic Idea

Each wireless node monitors the size of the local MAC-layer queue and measures its channel occupancy. A node detects channel congestion when its queue size exceeds a certain threshold[1] . If the node that detects congestion drops packets immediately, it may cause unfairness because the contending nodes in the same saturated channel may not yet detect congestion and hence they may not drop packets. To solve this problem, instead of dropping packets immediately, we make the node to transit from its normal state to either resolution or signaling state for a period of time T (such as 1 second in our simulations).

In the resolution state, the node emulates RED (Random Early Detection) [25] by probabilistically dropping packets to resolve congestion. It causes the TCP source to observe triple duplicate ACKs and consequently reduce the sending rate by shrinking the congestion window. In the signaling state, instead of dropping packets, the node aggressively competes for channel access by modifying its MAC parameters, such as reducing the minimum contention window. This is called the aggressive mode. Our

[1] The same method is used in [34], whose focus is however on MAC flows, instead of TCP flows.

previous research has shown that reducing the minimum contention window is very effective for enhancing a node's ability to occupy the channel. As this node enters the aggressive mode for a period of time and consumes more channel bandwidth, other contending nodes send out fewer packets. They will observe their queues build up and exceed the threshold. In this way, the congestion signal is spread to these nodes without explicit communication, causing them to transit to either resolution or signaling state.

The probability for a node to choose the resolution state is proportional to the node's current channel occupancy. Hence, nodes with low channel occupancies tend to select the signaling state and grab more channel bandwidth, while nodes with high channel occupancies are more likely to select the resolution state that causes the traversing TCP flows to reduce rates, which helps achieving fairness. It may happen that the contending nodes all choose the signaling state. In this case, some nodes will observe queue overflows, and the standard taildrop is performed to resolve congestion.

Below we give more detailed description for various techniques that are employed by WPD.

3.4.2 Periodical Measurement of State Information

Each wireless node periodically measures its average queue size $\bar{q}$ and average channel occupancy $\bar{u}$ to learn the channel condition. It also measures the average rate $\bar{r}$ of each TCP flow that it carries. At the end of every measurement period m (such as 0.1 second), the new value of $\bar{q}$ is computed as $\bar{q} := (1 - w) \times \bar{q} + w \times q$, where $:=$ is the assignment operator, q is the current queue size, and $0 < w < 1$, which is the parameter for weighted moving average. Similarly, $\bar{u} := (1 - w) \times \bar{u} + w \times u$, where u is the current channel occupancy, and $\bar{r} := (1 - w) \times \bar{r} + w \times (pks/m)$, where pks is the number of packets successfully delivered by the wireless node during the measurement period.

3.4.3 Aggressive Mode

In the 802.11 DCF, after a node successfully transmits a packet, it randomly picks a number from [0, CW_{min}] as the value of the backoff timer for its next packet. CW_{min}

is known as the minimum contention window, which is the initial window size for the exponential backoff algorithm. The default value of CW_{min} is 31. In WPD, a node in the signaling state will enter the aggressive mode by temporarily reducing CW_{min} to a small value (such as 3 in our simulations) for a period of time. With a reduced value of CW_{min}, the node is more capable of obtaining the channel than other contending nodes. As this node aggressively occupies the channel, less bandwidth is left for others, whose queue lengths are then forced up. Hence, the aggressive mode is used by a node that has detected congestion to spread the congestion signal to other contending nodes.

3.4.4 Probabilistic Dropping

In the resolution state, a node resolves congestion by dropping packets, which informs the TCP sources to reduce their sending rates. We adopt a probabilistic dropping algorithm that emulates the RED [25] in wired networks: A base probability p_b is first computed as $p_{max} \times \bar{u}$, where p_{max} is a predefined maximum packet dropping probability. Then the node drops each arrival packet with a probability $p_a := p_b/(1 - count \times p_b)$, where $count$ is the number of packets that have been forwarded since the previous packet drop. Hence, the dropping probability is an increasing function of the node's channel occupancy. A node with higher channel occupancy will drop more packets in its resolution state, which helps improve fairness among the contending nodes.

3.4.5 Minimum Rate Assurance

To avoid starvation, it is desirable to ensure that each TCP connection has a minimum rate r_{min}, even for one that is carried by a wireless node under the heaviest contention. As an optimization, when a wireless node detects that the average flow rate $\bar{r}$ is below r_{min}, it enters the aggressive mode until $\bar{r}$ reaches r_{min}. Ensuring the minimum flow rate has a positive impact in WPD because it assures that a node in the signaling state will have enough arrival packets to occupy the channel in order to spread the congestion signal. Without enough arrival packets, the node would have to inject fake packets to occupy the channel, which wastes bandwidth.

3.4.6 Adaptive Intermittent Release

When multiple wireless nodes contend in the same channel, if all of them continuously compete for channel access, collision can happen frequently. We design a method called adaptive intermittent release to interrupt the pattern of continuous channel access. A node stores the received packets in a network-layer queue and releases the packets to a MAC-layer queue for transmission. Based on the channel condition, we control the time t_{ips} (called Inter-Packet Spacing) between two consecutive releases. After the MAC layer transmits all its packets, it may have to wait for the network layer to release the next one. We observe that, at the congestion time, such wait helps reduce collisions.

The adaptation of t_{ips} is described as follows: When a node performs probabilistic dropping in its resolution state, it performs multiplicative increase on t_{ips} such that $t_{ips} := (1 + \alpha) \times t_{ips}$ after each measurement period m until the congestion is resolved (i.e., the average queue length falls below the threshold), where α is the factor of multiplicative increase. In all other cases, the node performs additive decrease on t_{ips} such that $t_{ips} := t_{ips} - \beta$, where β is a constant value. If t_{ips} becomes less than 50μs, we set it to be 50μs. The rationale behind the above adaptation is to keep t_{ips} small unless congestion occurs. When that happens, we increase the inter-packet spacing for nodes in the resolution state. As we will see in the next subsection, nodes with higher channel occupancies are more likely to be in this state. Their larger inter-packet spacings make it easier for other nodes with lower channel occupancies to access the channel during congestion.

When adaptive intermittent release is used, the queue length $\bar{q}$ is measured based on the combined length of the network-layer queue and the MAC-layer queue.

3.4.7 WPD Protocol

We now assemble the above rate control techniques to form WPD. Each node can be in one of three states: normal, resolution and signaling.

In the normal state, a node releases packets from the network layer to the MAC layer through adaptive intermittent release. After each measurement period m, the node computes $\bar{q}$, $\bar{u}$ and $\bar{r}$. When $\bar{q}$ is below a threshold H, it additively decreases t_{ips}. Once $\bar{q}$ exceeds H, it draws a random number rd from $[0, 1]$. If $rd \leq \bar{u}$, the node transits to the resolution state to perform probabilistic dropping; otherwise, if $rd > \bar{u}$, it transits to the signaling state to enter the aggressive mode. It is clear that the probability for the resolution state is proportional to the channel occupancy.

In the resolution state, a node performs probabilistic dropping to inform the TCP source to reduce its sending rate. In the mean time, it performs multiplicative increase on t_{ips} to give other contending nodes more chance of accessing the channel. After a period of time T, the node transits to the normal state.

In the signaling state, a node uses the aggressive mode to consume more channel bandwidth in order to spread the congestion signal to other nodes. After a period of time T, the node transits to the normal state. When the minimum rate assurance is implemented, a node may also temporarily enter the aggressive mode to bring up its flow rate if it is too low.

The pseudo code for the operations of WPD is given below.

Wireless Probabilistic Drop

```
1.  at the end of each measurement period $m$
2.      $\bar{q} := (1 - w) \times \bar{q} + w \times q$
3.      $\bar{u} := (1 - w) \times \bar{u} + w \times u$
4.      $\bar{r} := (1 - w) \times \bar{r} + w \times (pks/m)$
5.  if $\bar{q} > H$ then
6.      $rd_1 := random(0, 1)$
7.      if $rd_1 > \bar{u}$ then
```

```
8.          during a period $T$ do
9.              $CW_{min} := 3$ for aggressive mode
10.         end during
11.         restore $CW_{min}$ to the original value
12.     else
13.         $t_{ips} := (1+\alpha) \times t_{ips}$
14.         during a period $T$ do
15.             $p_b := p_{max} \times \bar{u}$
16.             $count := 0$
17.             for each arrival packet do
18.                 $count := count + 1$
19.                 $p_a := p_b/(1 - count \times p_b)$
20.                 if $p_a < 0$ then $p_a = 1$
21.                 $rd_2 := random(0, 1)$
22.                 if $rd_2 < p_a$ then
23.                     drop the arrival packet
24.                     $count := 0$
25.             end for
26.         end during
27. else
28.     $t_{ips} := t_{ips} - \beta$
29.     if $t_{ips} < 50\mu s$ then $t_{ips} := 50\mu s$
30.     if $\bar{r} < r_{min}$ then
31.         $CW_{min} := 3$ for aggressive mode
32.     else restore $CW_{min}$ to the original value
```

3.5 Simulation

We evaluate the proposed solution WPD by simulations.

3.5.1 Simulation Setup

All simulations are performed in ns-2 [1]. We use TCP NewReno at the transport layer. The size of a TCP data packet is 1,000 bytes. FTP is used to generate each TCP flow. We use IEEE 802.11 DCF at the MAC layer. RTS/CTS is turned off by default due to its high overhead, which is today's common practice. For the transmission range and the carrier-sensing range, we use ns-2 default values, which are 250m and 550m, respectively. The interference range is equal to the length of a wireless link times 1.78, which is also the default setting in ns-2. The transmission rate is 11Mbps. We compare WPD with DropTail (which drops packets when the queue is overflowed) and NRED [75], whose parameters are chosen based on the original paper. We set the parameters for WPD as follows: the rate measurement period m is 0.1 second, the threshold H is 5 packets, the state period T is 1 second, the minimum rate $rate_{min}$ is 25 packets per second, the maximum drop probability p_{max} is 0.03, the weighting factor w is 0.20, the α is 2 and the β is 50μs. Each simulation is executed for 150 seconds, and the average TCP rates are reported.

3.5.2 Fairness Index

We use the theoretically-computed rates under the proportional fairness model [37] as the benchmarks for comparison. Proportional fairness strives to balance between the fairness requirement and the overall network throughput. Besides these benchmark flow rates, we also compute an overall fairness index, which is the summation of a utility function, $\sum_{f \in F} \ln r_f$, where r_f is the rate of a TCP flow f and F is the set of flows. Our computed benchmark rates maximize this index. Among the fairness solutions under comparison, one that achieves a higher value of the fairness index has better performance in terms of proportional fairness.

3.5.3 Case Study: A Base Scenario

We first perform a case study on a base scenario in Fig. 3-3, where three access points s_1, s_2 and s_3 form two contention groups even though they are placed outside of each other's transmission range. We observe that unfairness arises among the three TCP flows, f_1, f_2 and f_3. As shown in Table 3-2, when DropTail is used as the queue management scheme, f_1 and f_3 can both obtain flow rates above 430 packets per second, whereas f_2 is almost starved. In this case, without the ability for the access points to explicitly synchronize their channel observations (due to being outside of each other's transmission range), NRED makes little improvement. For WPD, we incrementally deploy the techniques proposed in Section 3.4 to demonstrate their individual impact on the performance. We first adopt the aggressive mode (Section 3.4.3) and probabilistic dropping (Section 3.4.4). Table 3-2 shows that the flow rate of f_2 is increased to 35.83 packets per second. After the minimum rate assurance (Section 3.4.5) is incorporated, the flow rate of f_2 is increased to 74.75 packets per second, which demonstrates its positive impact on the performance of WPD. Finally, we apply the adaptive intermittent release (Section 3.4.6), which further improves fairness. The flow rate of f_2 is increased to 126.46 packets per second. The final results of WPD are close to the theoretical flow rates under the proportional fairness model that are shown in the last row of the table.

We compare WPD with DropTail and NRED in terms of their fairness indices in Table 3-3, which shows that WPD has the highest index value, close to the optimal index value that is achievable under the proportional fairness model.

3.5.4 Scalability Study: Three Contention Groups

Next we evaluate the performance of WPD on a scenario with more than two contention groups. Fig.3-7 has six TCP flows belonging to three contention groups. The simulation results in Table 3-4 show that f_3 has very low throughput values under DropTail and NRED. WPD improves it to 75.30 packets per second. It is worth noting that comparing with DropTail and NRED, WPD also achieves better fairness among the other five flows.

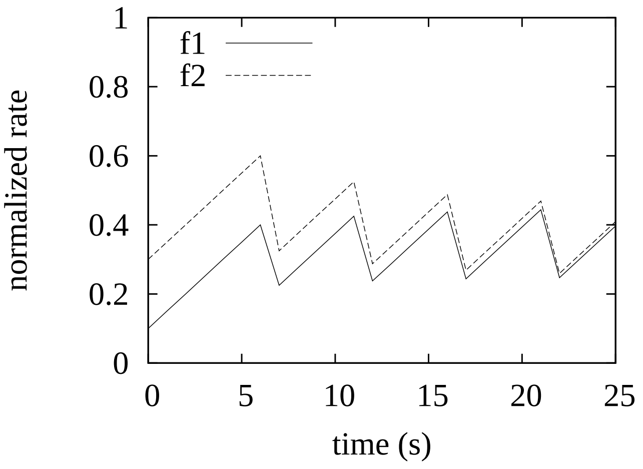
3.5.5 A Street Scenario

Fig.3-8 shows a more complicated scenario, where twenty-four TCP flows are carried by WLANs randomly deployed along two crossing streets. The relative positions of the nodes are drawn in the figure. The length of each wireless link is 150m. The contention relationship among the flows, which is much more complicated than those in the previous scenarios, is automatically determined by ns-2.

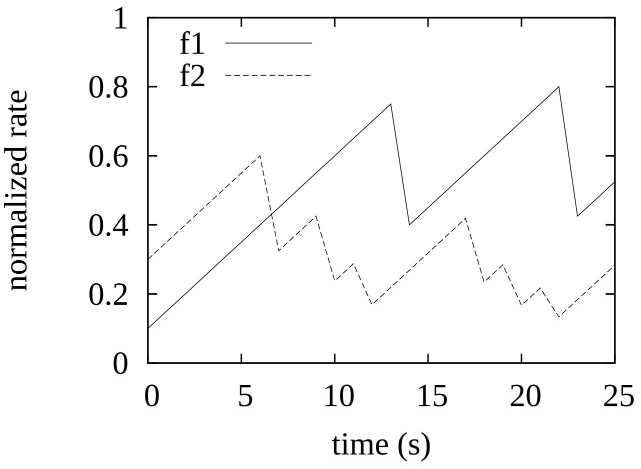
Table 3-5 shows that, under DropTail, six TCP flows are starved (less than 10 packets per second) and five TCP flows have low flow rates (less than 40 packets per second). NRED performs better than DropTail. But it still has three starved TCP flows and three low-rate flows. Under WPD, all flows can achieve decent throughput values. We believe that this simulation demonstrates the effectiveness of WPD in complex settings.

3.6 Summary

This chapter proposes WPD for achieving TCP fairness among multiple contending WLANs. It is a fully distributed solution only based on local information. It can work seamlessly with current TCP and MAC standards. It implicitly spreads congestion information to all contending flows in a bottleneck without requiring direct communications. Extensive simulations show that it can significantly improve TCP fairness in various scenarios when other existing solutions fail.



A



B

Figure 3-1. Why TCP AIMD doesn't work in wireless networks? A) TCP synchronized AIMD in wired networks. B) TCP unfair AIMD in wireless networks.

Figure 3-2. Two contending WLANs, where s_1 and s_2 are two access points. A TCP flow from a server on the Internet passes through an access point to a wireless node in each WLAN. Note that, only the wireless part of each flow (from an access point to a wireless client) is drawn in all figures throughout this chapter.

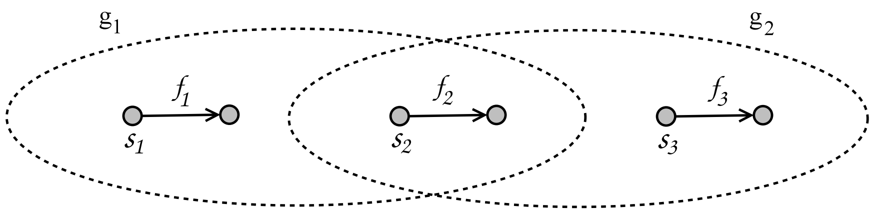


Figure 3-3. Three partially overlapping WLANs form two contention groups g_1 and g_2, where s_1, s_2 and s_3 are three access points. A TCP flow from a server on the Internet passes through an access point to a wireless client in each WLAN.

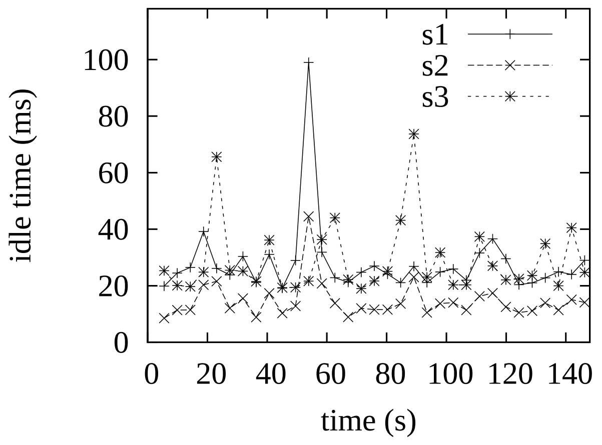


Figure 3-4. The channel idle time sensed by the three senders s_1, s_2 and s_3 in Fig. 3-3 under NRED.

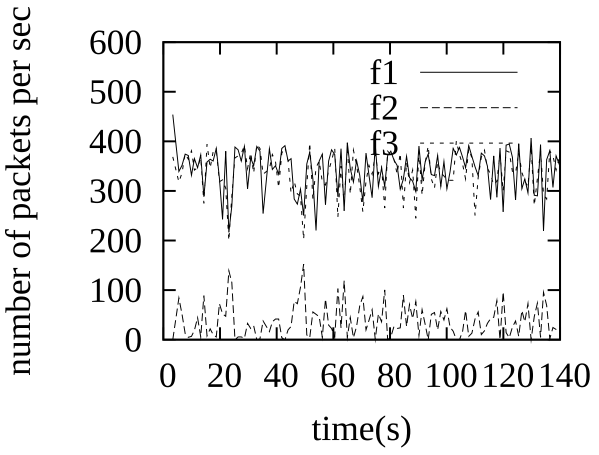


Figure 3-5. The flow rates of the three TCP flows, f_1, f_2 and f_3 in Fig. 3-3 under NRED.

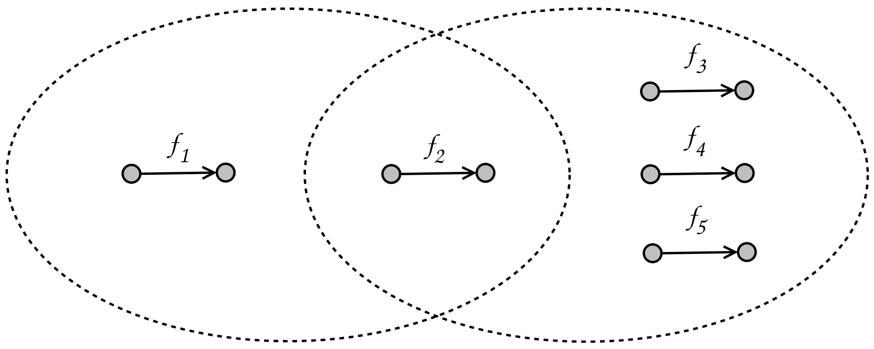


Figure 3-6. Five TCP flows form two contention groups.

Table 3-1. Comparing the flow rates (in packets/sec) of the flows in the network of Fig. 3-6 under PISD.

	f_1	f_2	f_3	f_4	f_5
flow rate	356.83	5.34	141.45	89.43	166.75

Table 3-2. Comparing the flow rates (in packets/sec) of the flows in the network of Fig. 3-3 under DropTail, NRED and WPD.

	f_1	f_2	f_3
DropTail	433.34	1.60	430.58
NRED	379.97	17.48	377.00
aggressive mode + probabilistic dropping	379.76	35.83	378.75
aggressive mode + probabilistic dropping + minimum rate assurance	339.43	74.75	343.20
aggressive mode + probabilistic dropping + minimum rate assurance + adaptive intermittent release (WPD)	252.34	126.46	260.54
Optimal Proportional Fairness	288.67	144.33	288.67

Table 3-3. Comparing the Fairness Index in Terms of Proportional Fairness in the network of Fig. 3-3 under DropTail, NRED and WPD.

DropTail	NRED	WPD	Optimal
12.61	14.73	15.93	16.30

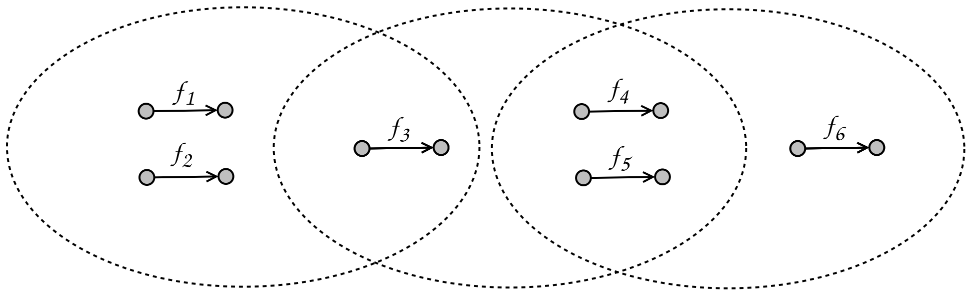


Figure 3-7. Six WLANs form three contention groups and each WLAN contains one TCP flow.

Table 3-4. Comparing the flow rates (in packets/sec) of the flows in the network of Fig. 3-7 under DropTail, NRED and WPD.

flow	DropTail	NRED	WPD
f_1	221.65	203.17	153.03
f_2	234.10	196.89	189.98
f_3	7.50	22.29	75.30
f_4	205.91	154.11	136.07
f_5	205.28	154.71	135.42
f_6	55.58	114.38	145.17

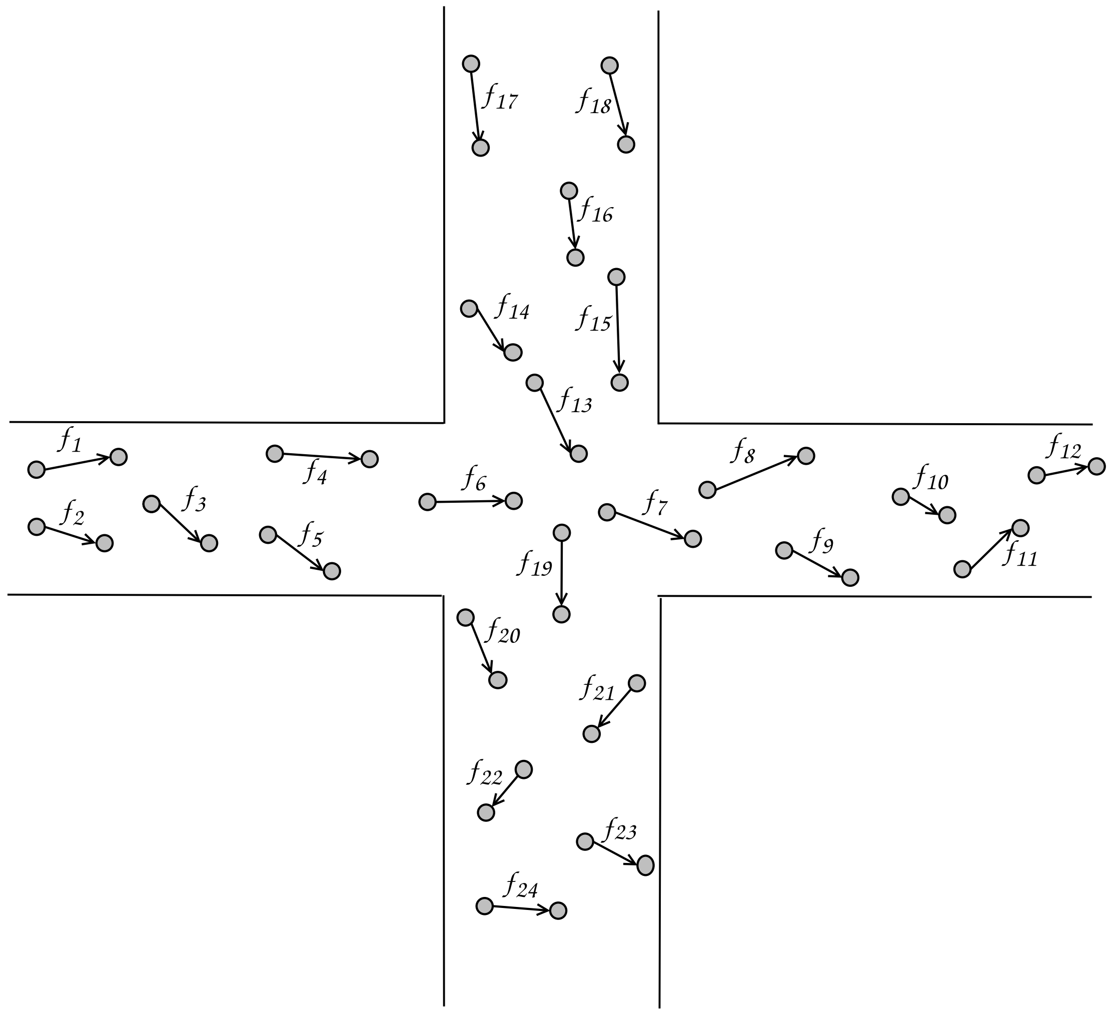


Figure 3-8. Twenty four WLANs are randomly deployed along two crossing streets and each WLAN contains one TCP flow.

Table 3-5. Comparing the flow rates (in packets/sec) of the flows in the network of Fig. 3-8 under DropTail, NRED and WPD.

flow	DropTail	NRED	WPD	flow	DropTail	NRED	WPD
f_1	216.4	235.4	163.9	f_{13}	0.8	3.5	45.6
f_2	215.9	174.4	84.2	f_{14}	191.5	181.4	162.6
f_3	1.1	6.8	163.9	f_{15}	242.3	226.7	181.1
f_4	205.2	189.5	165.0	f_{16}	0.4	0.8	52.9
f_5	218.6	205.6	99.8	f_{17}	217.4	207.3	194.5
f_6	9.8	25.5	106.4	f_{18}	214.2	209.5	167.4
f_7	10.2	91.9	89.4	f_{19}	388.9	198.9	125.9
f_8	23.5	95.4	124.4	f_{20}	17.0	117.0	168.0
f_9	382.3	183.0	144.6	f_{21}	402.9	237.4	155.1
f_{10}	7.1	38.2	88.2	f_{22}	4.0	26.1	84.9
f_{11}	30.6	104.0	147.7	f_{23}	26.2	93.6	146.7
f_{12}	397.7	286.3	218.6	f_{24}	403.2	301.5	217.4

CHAPTER 4
IMPROVING READING THROUGHPUT IN LARGE RFID SYSTEMS

In this chapter, we focus on improving reading throughput in large RFID systems.

In a RFID-equipped large warehouse, the RFID reader needs to periodically collect information from the tags for the purpose of inventory management. Hence, the reading throughput becomes a very important performance metric. During the reading process, multiple tags may report to the reader simultaneously, causing collision. Existing reading protocols strive to increase the reading throughput by minimizing the chance of collision. It has been proved that the traditional methods have reached their physical limit. We will demonstrate a fundamentally different approach to improve the reading throughput by extracting useful information from the collision.

The rest of this chapter is organized as follows. Section 4.1 presents the motivation of our work. Section 4.2 gives the problem definition. Section 4.3 discusses the related work. Section 4.4 proposes a collision-resolution tag identification protocol and derives the optimal system parameters. Section 4.5 improves the protocol for less overhead. Section 4.6 shows the simulation results. Section 4.7 gives the summary.

4.1 Background

4.1.1 Motivation

Consider a RFID system with a large number of active (or semi-active) tags deployed in a region. We assume that the RFID reader and the tags transmit with sufficient power such that they can communicate over a long distance. The problem is for the reader to collect the IDs of all tags within the communication range. If the communication range cannot cover the whole deployment region, the reader may have to perform the reading process at several locations and remove the duplicate IDs when some tags are covered by multiple readings. In this chapter, we focus on the reading operation at a single location, Our goal is to optimize the reading throughput, which is the average number of tag IDs that the reader is able to collect in each second.

During the reading process, multiple tags may transmit their IDs simultaneously, causing collision. Some collision-avoidance methods such as FDMA or CDMA require sophisticated scheduling methods to minimize bandwidth waste due to idle sub-channels or unused codes [65]. The overhead for sophisticated scheduling can be too costly for a RFID system where each tag only needs to deliver one piece of information (i.e., its ID) to the reader. Hence, contention-based time-slotted protocols have become the industrial standards [7].

In a contention-based protocol, each tag transmits its ID in a time slot with a report probability p that is tuned to reduce collision. A tag stops when it receives the acknowledgement from the reader that its ID has been successfully received. It can be shown that the optimal reading throughput is theoretically bounded by $\frac{1}{eT}$, where e is the natural constant and T is the length of a time slot [61]. In such a protocol, 36.8% of the time slots will be idle and 26.4% of the slots will have collision.

Can we do better than $\frac{1}{eT}$? We observe that the reading throughput can be improved if we make good use of the collision slots. Suppose the reader receives a mixed signal in a collision slot when both tag t_1 and tag t_2 transmit their IDs. In a later slot, if the reader receives the individual signal for the ID of tag t_1, it can remove this signal from the mixed signal and recover the individual signal for the ID of tag t_2.

Consider the example in Fig. 4-1, where four tags transmit their IDs to the reader. In Fig. 4-1A, when a contention-based protocol is used, it takes 11 slots for the reader to collect all four IDs. In Fig. 4-1B, when a collision-resolution protocol is used to resolve collision, only 6 slots are necessary. In particular, when the reader receives the signal from t_1 in the third slot, it removes this signal from the mixed signal received in the first slot and recovers the ID of t_4. Similarly, when it receives the signal from t_3 in the sixth slot, it also learns the ID of t_2 from the fourth slot.

4.1.2 Analog Network Coding (ANC)

Can we remove an individual signal from a mixed signal to recover the other constituent signal? This question has recently been brought up in the context of physical-layer networking code. Significant progress has been made in both theory and implementation [36, 81].

Katti et al. implemented the Analog Network Coding (ANC) and demonstrated its effectiveness in the Alice-Bob network shown in Fig. 4-2. Traditionally, four timeslots are needed for Alice and Bob to exchange a pair of messages: Alice sends a message to the router and the router forwards it to Bob, and vice versa. However, by using ANC, only two timeslots are necessary: Alice and Bob transmit simultaneously to the router. Instead of dropping the mixed signal, the router simply amplifies and broadcasts it to both Alice and Bob. Alice subtracts her own signal from the mixed signal and decodes Bob's message. Similarly, Bob can extract Alice's message.

We briefly describe the method used by Katti et al. Readers are referred to [36] for more details. The ANC protocol is designed based on MSK (Minimum Shift Keying) [55] and has been implemented using software defined radios. The signal transmitted by Alice can be represented as

$$s[n] = A_s e^{i\theta_s[n]}, \tag{4–1}$$

where A_s is the amplitude of the n^{th} sample and $\theta_s[n]$ is its phase. Similarly, Bob's signal can be represented as

$$s[n] = B_s e^{i\phi_s[n]}. \tag{4–2}$$

If Alice and Bob transmit simultaneously, the router will receive the sum of the two signals, which can be represented as

$$y[n] = h\prime A_s e^{i(\theta_s[n]+\gamma\prime)} + h\prime\prime B_s e^{i(\phi_s[n]+\gamma\prime\prime)}, \tag{4–3}$$

where $h\prime$ and $h\prime\prime$ are the channel attenuation and $\gamma\prime$ and $\gamma\prime\prime$ are the phase shift. We rewrite it for simplicity as

$$y[n] = Ae^{i\theta[n]} + Be^{i\phi[n]}, \tag{4–4}$$

where $A = h\prime A_s$, $B = h\prime\prime B_s$, $\theta[n] = \theta_s[n] + \gamma\prime$, and $\phi[n] = \phi_s[n] + \gamma\prime\prime$. Upon receiving the mixed signal from the router, Alice can resolve A and B from the following two energy equations [29, 36],

$$\mu = E[|y[n]|^2] = A^2 + B^2, \tag{4–5}$$

$$\sigma = \frac{2}{W} \sum_{|y[n]|^2 > \mu} |y[n]|^2 = A^2 + B^2 + 4AB/\pi, \tag{4–6}$$

where $E[.]$ is the expectation and W is a sampling window size. In MSK, a bit '1' is represented as a phase difference of $\pi/2$ over a time interval t, whereas a bit '0' is represented as a phase difference of $-\pi/2$ over t. For example, if the phase difference between the $(n+1)^{th}$ sample and the n^{th} sample, $\theta[n+1] - \theta[n]$, is $\pi/2$, then a bit "1" is transmitted. Since Alice knows her own signal, from (4–4), she can estimate the phase differences of Bob's signal, $\phi[n+1] - \phi[n]$, which can be translated into the bit stream sent by Bob [36].

The task of resolving the mixed signal in a collision slot in a RFID system is simpler than the same task in the wireless network shown in Fig. 4-2. First, Alice knows the amplitude of her signal when it is transmitted out, but she does not know the amplitude of her signal when it reaches the router and mixed with Bob's signal. When Alice received the amplified mixed signal from the router, it becomes difficult for her to remove her own signal from the mixed one. In the RFID system, suppose the reader receives the mixed signal from t_1 and t_2 in one slot and the individual signal of t_1 in another slot. Because the same signal of t_1 appears in the two slots, it becomes easier to remove it from the mixed signal.

Second, it is very difficult to synchronize transmissions between wireless nodes, and thus the proposed ANC protocol has to introduce a complicated mechanism to relieve this problem, whereas transmissions in a RFID system can be synchronized by the reader's signal.

Given that the technology for collision resolution exists, the next question is how to optimally use it to maximize the performance of a wireless system. We will answer the question in the context of RFID systems.

4.2 Terminology and Problem Definition

4.2.1 Terminology

During the execution of a time-slotted contention-based protocol, if no tag transmits in a time slot, we call it an empty slot. If one tag transmits, it is called a singleton slot. If more than one tag transmits, it is a collision slot. In particular, if k tags transmit simultaneously, the slot is called a k-collision slot, where $k \geq 2$. In order to guard against channel error, each ID carries a CRC code. In a singleton slot, the RFID reader receives the ID signal from a single tag. It will verify the correctness of the received ID by checking the CRC code.

4.2.2 Resolvable Collision Slots

An empty slot is easy to identify because no signal is received. The reader can distinguish a singleton slot from a collision slot by first converting the signal into an ID and then verifying the correctness of the CRC code. For a collision slot, the reader records a mixed signal that combines the individual signals of the tags that transmit simultaneously. In later singleton slots, the reader will receive the individual ID signals from some of those tags. When the reader eventually receives the ID signals from all but one of those tags, we say the k-collision slot is resolvable if we can derive the ID signal of the last tag by removing the $(k-1)$ ID signals from the mixed signal. The experimental study of Analogy Network Coding by Katti et al. in [36] has shown that 2-collision slots are resolvable.

4.2.3 Problem Definition

The main problem we want to solve in this work is how to optimally apply analog network coding to maximize the RFID reading throughput. We design a collision-aware tag identification protocol and derive the optimal report probability (at which a tag transmits its ID in each slot) that maximizes the number of singleton and 2-collision slots (from which IDs can be extracted by ANC).

In [36], the authors state that ANC can be applied to resolve collision involving more than two signals. On one hand, as we will demonstrate in Section 4.6, resolving 2-collision slots based on today's technology will already provide a practically significant boost to the reading throughout. On the other hand, instead of restricting our work to 2-collision slots, we decide to generalize our protocol so that it can work with any future ANC method that resolves λ-collision slots, where λ (≥ 2) is an input parameter. Such generalization sheds light on the amount of throughput gain that can possibly be obtained through analog network coding. In particular, the results in Section 4.6 show that the reading throughput will be higher when λ is larger (because more collision slots become useful). However, the rate of throughput improvement diminishes quickly as λ increases. Hence, it is not necessary to make λ too large. In practice, we expect λ to be a small number (such as 2, 3 or 4).

Clearly, ANC and other physical-layer network coding methods can be applied in various different communication contexts, each of which has its unique technical challenges. For example, collision resolution has been used in satellite access networks, where each terminal transmits a single packet twice at two randomly-selected time slots in each MAC frame [15]. The throughput upper bound can be predicted if the number of packets per slot is known (which requires the knowledge of the number of transmitting terminals). In our context, we do not derive a throughput upper bound for a given set of system parameters. Instead, we determine the best system parameter that optimizes the throughput. We do not assume the knowledge for the number of tags that is participating in the protocol. In fact, this number changes over time because after a tag successfully delivers its ID to the reader, it will stop transmitting. A tag may transmit for one, two or more times at any time slots during the reading process. Moreover, because the number of participating tags changes, the optimal system parameter also changes over time.

4.3 State of the Art

All existing contention-based tag reading protocols are called anti-collision protocols because they treat collision as waste and try to avoid it [24]. Most of these protocols fall into two classes: the ALOHA-based protocols [16, 42, 56, 63, 68, 71, 82] and the tree-based protocols [3, 9, 51, 72, 83].

In the ALOHA-based protocols, the reader broadcasts a request and each tag randomly selects a time slot to report its ID. If exact one tag reports, the reader retrieves its tag ID and this tag will remain silent for the rest of reading process. Simultaneous reports in a slot will lead to collision. Therefore, the ALOHA-based protocols try to maximize the probability that exact one tag reports in a slot. The ALOHA-based protocols differ in how the reader sends the request and how the tag selects a slot to report. In the slotted ALOHA [68], the reader sends out a contention probability at the beginning of each slot and each unread tag with this probability to reply with its ID. In the basic framed slotted ALOHA [42], slots are grouped into frames with the same fixed frame size. Each unread tag picks up a random slot within each frame to report. It is possible that the number of tags far exceeds the number of slots in a frame so that the frame is full of collision. To overcome this problem, the dynamic framed slotted ALOHA (DFSA) [16] introduces frames with dynamic frame size. It is proved that the maximal reading throughput is achieved when the frame size is equal to the number of unread tags [16]. DFSA determines the size of the next frame by estimating the number of unread tags after each frame. However, in practice, it may be impractical to set the frame size indefinitely high considering there exist a large number of tags [42]. The enhanced dynamic framed slotted ALOHA [42] uses frames with limited frame size by restricting the number of responding tags in a frame. The maximal reading throughput of the ALOHA-based protocols is bounded by $\frac{1}{eT}$ [61]. In other words, for each slot, the probability of successfully reading a new tag is 36.8%.

In the tree-based protocols, the tag reading procedure can be interpreted as a recursive splitting procedure. The general schema works as follows: In a slot, the reader sends a query with a certain condition and each tag that meets the condition will respond. If a set of tags respond concurrently, the reader split them into smaller subsets. The procedure repeats until every subset only contains a single tag which can be identified by the reader. Different splitting criteria lead to different protocols. The binary-tree protocols [3, 51, 72] split a set of tags using a random binary number. Specifically, each tag has a counter initialized to 0. Upon receiving a query, each tag that has a counter value 0 will respond. Once collision happens, the reader sends a new query with an indication of the collision. Each colliding tag draws a random binary number (i.e. 0 or 1) and adds it to its counter. The set of colliding tags is thus divided into two subsets: one is the set of tags whose counters remain 0 and the other one is the set of tags whose counters increase to 1. When collision happens, all other tags that do not transmit also increase their counters by one; otherwise, they decrease their counters by one. An analysis shows that the maximal reading throughput of the binary-tree protocols is $\frac{1}{2.88T}$ [13]. The query-tree protocols [9, 51, 83] use the tag ID for splitting. A tag ID is a unique bit string. Each query contains a prefix $p_1..p_i$ where i is the length of the prefix. Each tag, whose ID contains this prefix, transmits its ID as a response. If multiple responses collide, the reader will generate two new prefixes $p_1..p_i0$ and $p_1..p_i1$ by attaching a bit 0 and 1, respectively. The set of colliding tags is divided into two subsets: one subset is the group of tags whose IDs contain the prefix $p_1..p_i0$ and the other subset is the group of tags whose IDs contain the prefix $p_1..p_i1$. A query-tree protocol can have quite different reading throughputs determined by the tag ID distribution. It is shown that the maximal reading throughput is bounded by $\frac{1}{2.88T}$ for a set of uniformly distributed tag IDs [41].

4.4 Slotted Collision-Aware Tag Identification Protocol (SCAT)

In this section, we propose the Slotted Collision-Aware Tag identification protocol (SCAT). In the next section, we will optimize the protocol for less overhead.

4.4.1 Protocol Description

SCAT is a time-slotted protocol. The time slots are synchronized by the reader's signal. Each time slot consists of three segments: the advertisement segment, the report segment, and the acknowledgement segment.

In the advertisement segment, the RFID reader sends out a slot index i and a report probability p_i, where i begins from zero and increases by one after each slot.

In the report segment, each tag decides to transmit its ID with probability p_i. To actually broadcast the report probability, the reader may send out an l-bit integer $\lfloor p_i \times 2^l \rfloor$ instead of a real number p_i. A tag computes a hash function $H(\mathrm{ID}|i)$ whose range is $[0..2^l)$. If $H(\mathrm{ID}|i) \leq \lfloor p_i \times 2^l \rfloor$, the tag transmits its ID.

For an empty slot, the reader transmits a negative acknowledgement. For a collision slot, the reader will not be able to tell how many tags have transmitted simultaneously in the report segment. It will record the mixed signal and transmit a negative acknowledgement. The mixed signal and the slot index form a collision record. Over time the reader will collect a group of such records. The operation for a singleton slot is more complicated. The reader learns the ID of a tag in the report segment. Using this ID, it tries to resolve some collision records to learn more tag IDs (see the next subsection). It then transmits a positive acknowledgement, together with the IDs that are learned from the resolution of the previous collision records.

When the tag that transmits in the report segment receives the positive acknowledgement, it will stop participating in the SCAT protocol as its ID has been successfully delivered to the reader. Similarly, when a tag that transmitted its ID at an earlier slot but has not received a positive acknowledgement yet receives its own ID in the acknowledgement segment, it will stop participating in SCAT.

The SCAT protocol stops when no tag transmits any more. When the reader finds a certain number of consecutive empty slots, it sets $p_i = 1$ for one slot and if it still finds an empty slot, it knows that the IDs of all tags have been collected.

4.4.2 Collision Resolution

When the CRC received in the report segment is verified to be correct, the reader learns the ID of a tag from the current slot i. Knowing the ID, the RFID reader can easily figure out the previous slots in which this tag has transmitted. For an arbitrary collision record with slot index j, the tag must have transmitted if $H(\text{ID}|j) \leq \lfloor p_i \times 2^l \rfloor$. If that is the case, the reader removes the signal received in the current slot from the mixed signal in the collision record, treats the result as if it was the ID signal of a single tag, and extracts the CRC code. If the CRC code is verified to be correct, the collision record is resolved and the reader learns an additional tag ID. The signal for that ID can be used to resolve other collision records in a similar process as described above.

Resolving the collision slots incurs computation overhead. Hence, we expect the reader to be computationally capable or connected to a powerful computing device. It is worth noting that the RFID system works in a low speed channel (53 Kbps for the Philips I-Code system), while the original ANC [36] and the follow-up work [27] are designed for 11 Mbps or higher throughput channels, which is far more demanding, yet experimentally-demonstrated feasible.

4.4.3 Determining the Optimal Value for p_i

We want to determine the optimal report probability p_i for each slot such that the number of slots for collecting the IDs of all tags is minimized. Consider an arbitrary time slot with index i. When there is only one tag transmitting, the RFID reader will learn the ID of the tag. If there are two tags transmitting, the reader will not learn any ID now but will learn one ID later when the other ID is known (such that the collision record of this slot can be resolved). Similarly, when k tags transmit in this slot for $k \leq \lambda$, the reader will learn one ID from the collision record when the other $(k-1)$ IDs are known. Essentially, a singleton slot or a k-collision slot will allow the reader to learn one ID now or later. Hence, we shall choose the value of p_i that maximizes the probability for one, two, ..., or λ tags to transmit in the current slot.

Let N be the number of tags in the system. Its value can be estimated to an arbitrary accuracy [39] in a pre-step of SCAT. This pre-step will be removed in the next section. Before slot i, the reader may have successfully collected and acknowledged a number n_i of tag IDs, and those tags will no longer participate in the protocol of SCAT. Let N_i be the number of tags that the reader has not identified yet before slot i. Since n_i is known to the reader, N_i is also known.

As each tag decides to transmit with probability p_i, the number of tags that transmit will be a random variable X_i that follows the binomial distribution. The probability for $X_i = k, \forall k \in [0..N_i]$ is $\binom{N_i}{k} \cdot p_i^k(1-p_i)^{N_i-k}$. Our objective is to maximize the probability of $X_i \in [0..\lambda]$, which is

$$\sum_{k=1}^{\lambda} Prob\{X_i = k\} = \sum_{k=1}^{\lambda} \binom{N_i}{k} \cdot p_i^k(1-p_i)^{N_i-k}. \qquad (4–7)$$

We expect λ to be small. In the following, we consider λ = 2, 3, or 4. When $\lambda = 2$, (4–7) becomes

$$\begin{aligned}&\sum_{k=1}^{2} Prob\{X_i = k\}\\ &= N_i p_i (1-p_i)^{N_i-1} + \frac{N_i(N_i-1)}{2} p_i^2 (1-p_i)^{N_i-2}\\ &\simeq N_i p_i e^{-N_i p_i} + \frac{N_i^2 p_i^2}{2} e^{-N_i p_i}. \end{aligned} \qquad (4–8)$$

Let $\omega = N_i p_i$. Substituting $N_i p_i$ by ω in (4–8), we have

$$\sum_{k=1}^{2} Prob\{X_i = k\} \simeq (\omega + \frac{\omega^2}{2})e^{-\omega}. \qquad (4–9)$$

To find the value of ω that maximizes the above formula, we differentiate the right side and let it be zero.

$$\frac{d(\omega + \frac{\omega^2}{2})e^{-\omega}}{d\omega} = (1 - \frac{\omega^2}{2})e^{-\omega} = 0. \tag{4–10}$$

Solving the above equation, we have $\omega = 1.414$. Hence, the optimal report probability is $p_i = 1.414/N_i$.

Following the same process, we derive that, when $\lambda = 3$, the optimal report probability is $p_i = 1.817/N_i$, and when $\lambda = 4$, it is $p_i = 2.213/N_i$.

Resolving the collision slots requires a sufficient number of singleton slots. Otherwise, if all slots have collision, none of them will be resolved. Fortunately, when λ is small (which should be the case as we have discussed in Section 4.2.3 and will further elaborate in Section 4.6.1), there are sufficient singleton slots to resolve most collision slots. Our simulation results in Section 4.6.3 show that the optimal report probabilities obtained by exhaustive search match closely with the above computed values.

4.4.4 Pseudo Code

The pseudo code for the operation of the RFID reader during the ith slot is given below. Let S be the set of newly known IDs (together with their signals) that can be used to resolve some of the collision records. Let I be the set of IDs that are learned by resolving the collision records. Let R_j be the collision record for slot j.

```
Reader's Operation at Slot $i$
1.  broadcast an advertisement $\langle i, p_i \rangle$
2.  record the signal $s_i$ in the report segment
3.  extract $ID_i$ from $s_i$
4.  **if** the channel is idle during the report segment **then**
5.      broadcast a negative acknowledgement
6.  **else if** CRC in $ID_i$ is verified to be correct **then**
7.      $S := \{\langle ID_i, s_i \rangle\}$
```

```
8.       $I := \emptyset$
9.       **while** $S \neq \emptyset$ **do**
10.          remove an element $\langle ID, s\rangle$ from $S$
11.          **for** each collision record $R_j$ **do**
12.              **if** $H(ID|j) \leq p_j$ **then**
13.                  add $s$ to the set of known individual signals in $R_j$
14.                  remove known signals from the mixed signal in $R_j$
15.                  extract $ID'$ from the resulting signal $s'$
16.                  **if** CRC in $ID'$ is verified to be correct **then**
17.                      $S := S + \{\langle ID', s'\rangle\}$
18.                      $I := I + \{ID'\}$
19.                      remove the collision record $R_j$
20.          **end for**
21.      **end while**
22.      broadcast a positive acknowledgement and the IDs in $I$
23.  **else**
24.      add $\langle i, s_i\rangle$ as a collision record
25.      broadcast a negative acknowledgement
```

4.4.5 Unresolvable Collision Slots and Channel Error

The reading process normally takes a short period of time (minutes for tens of thousands of tags). During this time, we expect the tags to be statically located. The MSK employed by ANC can tolerate a certain level of noise and channel variation. However, if the spontaneous noise is too large, a collision slot may not be resolvable. The only impact is that the slot is not useful, and the reader can still learn the IDs from other slots. A tag will stop transmitting only after it receives positive confirmation from the reader. As long as most 2-collision slots can be resolved, the proposed protocol still achieves much higher reading throughput.

Channel error may corrupt the signal transmitted by a tag or the acknowledgement transmitted by the reader. This problem is common to all RFID reading protocols. The solution is also common: The tag will keep transmitting its ID until it receives the positive confirmation from the reader. In this case, the reader may receive an ID more than once and the duplicates will be discarded.

The proposed protocol is not suitable for an environment where the channel noise is so severe or the tags move so much and so fast during the reading operation that most collision slots are not resolvable. In this case, we should use a contention-based protocol without collision resolution. It is beyond the scope of this work to investigate the noise level of each specific environment. Instead, we are more interested in knowing what is the upper limit of throughput improvement that ANC can bring (in an environment where most 2-collision slots are resolvable).

4.5 Framed Collision-Aware Tag Identification Protocol (FCAT)

In this section, we propose a framed version of the previous protocol to improve its efficiency.

4.5.1 Inefficiencies of SCAT

SCAT utilizes the information carried in the collision slots. However, it is not practically efficient due to a number of reasons.

First, to calculate p_i, the RFID reader has to know N_i, which in turn requires it to know N. It incurs considerable overhead to accurately estimate the number of tags in the system as a pre-step to SCAT. We want to remove such a pre-step and estimate N as a byproduct during the protocol execution.

Second, the advertisement segment of each slot represents significant overhead which is not always necessary. For consecutive slots, the slot index changes from i to $i + 1$ and the report probability changes from ω/N_i to ω/N_{i+1}, where N_i and N_{i+1} at most differ by one. As the report probability changes little when N_i is reasonably large, the reader does not have to make advertisement in each slot. It may advertise once every certain number of slots, and the tags will use the same report probability in those slots.

Third, after resolving a collision record, the reader learns an extra ID and it broadcasts the ID in order to inform the corresponding tag to stop participating in the protocol. However, instead of transmitting the whole ID (which is 96 bits for GEN2 tags), the reader may transmit the slot index of the collision record. A tag stores the indices of the slots in which it has transmitted. If the tag receives a slot index that matches a stored one, it knows that the reader must have collected its ID. A slot index can be much smaller than 96 bits. If we use 23-bit slot indices, more than 8 million slots are allowed. In our simulations, the number of slots required never exceeds $2N$.

4.5.2 Using Frames

We propose the Framed Collision-Aware Tag identification protocol (FCAT), which improves SCAT by eliminating the inefficiency described in Section 4.5.1. FCAT shares much of the protocol details with SCAT. Below we will focus on describing their differences.

In FCAT, time is divided into frames of size f. That is, each frame consists of f time slots. Each frame has an index, starting from zero. The index of the jth slot in the ith frame is $i \times f + j$. Before a frame begins, the RFID reader broadcasts a pre-frame advertisement, including the frame index i and the report probability p_i. Each slot of the frame consists of a report segment, during which the tags transmit their IDs, and an acknowledgement segment, during which the reader transmits either a positive acknowledgement or a negative acknowledgement.

In any slot of the ith frame, each tag transmits its ID with probability p_i. After receiving the signal in the report segment, the reader performs the same operations as in SCAT, except that it does not transmit the IDs learned from resolving the collision records in the acknowledgement segment. Instead, it transmits the slot indices of the resolved collision records, which are shorter than the IDs themselves. If a tag receives a slot index that matches a slot in which it has transmitted its ID, it stops participating in FCAT. Certainly, if a tag receives a positive acknowledgement for its ID just transmitted in the report segment, it will also stop participating in FCAT.

4.5.3 Estimating the Number of Tags within FCAT

Finally, we address the problem of how to learn the value of N. There exist efficient methods for estimating the number of tags. However, using them as a pre-step of FCAT incurs considerable overhead. In the following, we embed an estimation method within FCAT.

Consider an arbitrary frame with index i. Let n_0, n_1 and n_c be the random variables for the numbers of empty, singleton and collision slots, respectively. We can estimate the statistical relationship between these random variables and the number N_i of tags that are currently participating in the protocol. Based on that relationship, we can estimate N_i from the measured values of n_0 and n_c. Our approach shares some similarity with [39]. However, in [39], each tag transmits at most once in the frame. In FCAT, each tag participates probabilistically in every slot of the frame.

Let X_j be the indicator random variable for the event that the jth slot in the frame is empty, i.e., $X_j = 1$ means the jth slot is empty and $X_j = 0$ means it is not empty. Similarly, let Y_j be the indicator random variable for the event that the jth slot is a singleton slot. Because each tag decides to transmit with probability p_i in every slot in the frame, we have

$$Prob\{X_j = 1\} = (1 - p_i)^{N_i}, \ \ \forall j \in [1..f]. \tag{4–11}$$

The expected value of n_0 is

$$E(n_0) = \sum_{j=1}^{f} (1 - p_i)^{N_i} = f(1 - p_i)^{N_i}. \tag{4–12}$$

The probability for the jth slot in the frame to be a singleton is

$$Prob\{Y_j = 1\} = \binom{N_i}{1} p_i(1-p_i)^{N_i-1} = N_i p_i (1-p_i)^{N_i-1}. \tag{4–13}$$

The expected value of n_1 is

$$E(n_1) = \sum_{i=1}^{f} N_i p_i (1-p_i)^{N_i-1} = fN_i p_i (1-p_i)^{N_i-1}. \tag{4–14}$$

Obviously, $E(n_0) + E(n_1) + E(n_c) = f$. Hence

$$\begin{aligned} E(n_c) &= f - E(n_0) - E(n_1) \\ &= f(1-(1-p_i)^{N_i} - N_i p_i (1-p_i)^{N_i-1}) \\ &= f(1-(1-p_i)^{N_i-1}(1-p_i+\omega)). \end{aligned} \tag{4–15}$$

The above equation can be rewritten as

$$N_i = \frac{\ln(1-\frac{E(n_c)}{f}) - \ln(1-p_i+\omega)}{\ln(1-p_i)} + 1. \tag{4–16}$$

At the end of the ith frame, the reader counts the value of n_c. Substituting $E(n_c)$ by the instance value n_c (obtained in the ith frame), the reader obtains an estimation of N_i by the following formula:

$$\hat{N}_i = \frac{\ln(1-\frac{n_c}{f}) - \ln(1-p_i+\omega)}{\ln(1-p_i)} + 1. \tag{4–17}$$

Next, we derive $E(\hat{N}_i)$. To simplify the equations, let $C_1 = \frac{1}{\ln(1-p_i)}$, $C_2 = -\frac{\ln(1-p_i+\omega)}{\ln(1-p_i)} + 1$, and function $g(n_c) = \ln(1-\frac{n_c}{f})$. We expand the right hand side of (4–17) by its Taylor series about $q = E(n_c)$.

$$\begin{aligned} \hat{N}_i = C_1\Big[& g(q) + (n_c-q)g\prime(q) + \frac{1}{2}(n_c-q)^2 g\prime\prime(q) \\ & + \frac{1}{6}(n_c-q)^3 g\prime\prime\prime(q) + \ldots\Big] + C_2. \end{aligned} \tag{4–18}$$

Since $q = E(n_c)$, the mean of the second term in (4–18) is 0. Therefore, we keep the first three terms when computing the approximated value of $E(\hat{N}_i)$.

$$E(\hat{N}_i) \simeq C_1\left[g(q) + \frac{1}{2}E((n_c - q)^2)g\prime\prime(q)\right] + C_2. \tag{4–19}$$

We have $E((n_c - q)^2) = V(n_c)$ by definition and $g\prime\prime(q) = -\frac{1}{(q-f)^2}$ since $g(q) = \ln(1 - \frac{q}{f})$. The variance $V(n_c)$ is derived in Section 4.5.4. Applying (4–24) in Section 4.5.4, we have

$$E(\hat{N}_i) \simeq N_i - \frac{e^{\omega} - 1 - \omega}{2f\ln(1 - p_i)(1 + \omega)}. \tag{4–20}$$

Therefore,

$$Bias(\frac{\hat{N}_i}{N_i}) = E(\frac{\hat{N}_i}{N_i}) - 1 = \frac{1 + \omega - e^{\omega}}{2fN_i\ln(1 - p_i)(1 + \omega)}. \tag{4–21}$$

Fig. 4-3 shows the absolute value of $Bias(\frac{\hat{N}_i}{N_i})$ with respect to the number of tags N_i. The three lines show that the absolute values of $Bias(\frac{\hat{N}_i}{N_i})$ are $0.0082, 0.011$ and 0.014, for $\omega = 1.414, 1.817$ and 2.213, respectively. They are all very small.

Adding the number of tags whose IDs are already known, the reader has an estimation for the total number of tags in the system, denoted as $\hat{N}_i^*$. The variance of $\hat{N}_i^*$ is the same as the variance of $\hat{N}_i$, i.e., $V(\hat{N}_i^*) = V(\hat{N}_i)$. Because $N_i < N$, $V(\frac{\hat{N}_i^*}{N}) < V(\frac{\hat{N}_i}{N_i})$. The value of $V(\frac{\hat{N}_i}{N_i})$ is derived in Section 4.5.4. It is approximately 0.0342, 0.0287 or 0.0265, for $\omega = 1.414, 1.817$ and 2.213, respectively (i.e., when 2-collision slots, 3-collision slots or 4-collision slots are resolvable). This is the variance when only one instance of n_c is used. It is small though not negligible. The RFID reader obtains one estimation after each frame. If it uses the average $\frac{\sum_{j=0}^{i}\hat{N}_j^*}{i}$ as the estimation for N, then the variance will decrease in the square root of i and therefore diminish as the protocol executes frame after frame.

We can also design a similar estimator by using the number of empty slots, n_0, based on (4–12). However, we find in our simulations that the variance of such an estimator is larger. As shown in Fig. 4-4, N_i is not a monotonic function with respect to the number of singleton slots. Hence, we cannot use n_1 to estimate the value of N_i.

4.5.4 Estimation Variance, $V(\frac{\hat{N}_i}{N_i})$

Consider an arbitrary frame with index i. Let Z_j be the indicator random variable for the event that the jth slot in the frame is a collision slot. Since no slot is special, $Z_j, \forall j \in [1..f]$, follows the same distribution. They are independent random variables. Because $n_c = \sum_{j=1}^{f} Z_j$, we have

$$V(n_c) = \sum_{j=1}^{f} V(Z_j) = fV(Z_1). \tag{4–22}$$

$E(Z_1) = 1 - (1 - p_i)^{N_i} - N_i p_i (1 - p_i)^{N_i - 1} \approx 1 - e^{-N_i p_i} - N_i p_i \cdot e^{-N_i p_i}$. $E(Z_1^2) = E(Z_1)$ because Z_1 is an indicator random variable. Hence, we have

$$\begin{aligned} V(Z_1) &= E(Z_1^2) - (E(Z_1))^2 \\ &= (1 + N_i p_i)e^{-N_i p_i}(1 - (1 + N_i p_i)e^{-N_i p_i}). \end{aligned} \tag{4–23}$$

Therefore,

$$V(n_c) = f(1 + N_i p_i)e^{-N_i p_i}(1 - (1 + N_i p_i)e^{-N_i p_i}). \tag{4–24}$$

According to the central limit theorem, if f is large, n_c is approximately normally distributed. When $f \to \infty$, n_c converges to the normal distribution, $n_c \xrightarrow{D} Norm(\theta, \delta^2)$, where θ is $E(n_c)$ as given in (4–15), δ^2 is $V(n_c)$ as given in (4–24), and $\xrightarrow{D}$ means convergence in distribution.

According to the δ-method [14], we have

$$h(n_c) \xrightarrow{D} Norm(h(\theta), \delta^2 [h\prime(\theta)]^2) \tag{4–25}$$

for any function $h(.)$ such that $h\prime(\theta)$ exists and takes a non-zero value.

In Section 4.5.3, the estimation formula is designed based on (4–15), which is copied here $E(n_c) = f(1 - (1 - p_i)^{N_i} - N_i p_i (1 - p_i)^{N_i - 1})$. Let $g(.)$ be the mapping function from N_i to n_c. The above equation can be rewritten as $E(n_c) = g(N_i)$. Fig. 4-4 shows that $g(.)$ is a monotonic function, and hence it has a unique inverse function, denoted as $h(.)$.

According to Section 4.5.3, $\hat{N}_i$ is computed from (4–15) by substituting $E(n_c)$ with the instance value of n_c (obtained after the ith frame).

$$\begin{aligned} n_c &= f(1-(1-p_i)^{\hat{N}_i} - \hat{N}_i p_i (1-p_i)^{\hat{N}_i - 1}) \\ &\approx f(1-e^{-\hat{N}_i p_i} - \hat{N}_i p_i e^{-\hat{N}_i p_i}). \end{aligned} \tag{4–26}$$

Clearly, $n_c = g(\hat{N}_i)$ and $\hat{N}_i = h(n_c)$. Applying $\hat{N}_i = h(n_c)$ to (4–25), we have

$$\hat{N}_i \xrightarrow{D} Norm(h(\theta), \delta^2 [h\prime(\theta)]^2). \tag{4–27}$$

We know that $h(g(N_i)) = N_i$. Differentiating both sides, we have $h\prime(g(N_i))g\prime(N_i) = 1$. Hence,

$$h\prime(\theta) = h\prime(E(n_c)) = h\prime(g(N_i)) = \frac{1}{g\prime(N_i)}. \tag{4–28}$$

Therefore, from (4–27), the variance of $\hat{N}_i$ is

$$\begin{aligned} V(\hat{N}_i) &= \delta^2 [h\prime(\theta)]^2 = \frac{V(n_c)}{[g\prime(N_i)]^2} \\ &= \frac{(1+N_i p_i)e^{N_i p_i} - (1 + 2N_i p_i + N_i^2 p_i^2)}{f N_i^2 p_i^4}, \end{aligned} \tag{4–29}$$

$$V(\frac{\hat{N}_i}{N_i}) = \frac{(1+N_i p_i)e^{N_i p_i} - (1 + 2N_i p_i + N_i^2 p_i^2)}{f N_i^4 p_i^4}. \tag{4–30}$$

Below we perform approximate computation to give a rough idea on how big this variance is. In SCAT or FCAT, $\hat{N}_i p_i = \omega$, where ω is 1.414, 1.817 or 2.213 for $\lambda = 2$, 3 or 4, respectively. Our simulations show that $\hat{N}_i$ reliably converges to N_i when i is large. Hence, we substitute $N_i p_i$ with ω in (4–30), and the variance $V(\frac{\hat{N}_i}{N_i})$ is 0.0342, 0.0287 or 0.0265 respectively for different ω values.

4.6 Simulation Results

In this section, we present simulation results to evaluate the performance of our main protocol FCAT. We compare FCAT with the existing work, including the Dynamic Framed Slotted ALOHA (DFSA) [16], Enhanced Dynamic Framed Slotted ALOHA (EDFSA) [42], Adaptive Binary Splitting (ABS) [51] and Adaptive Query Splitting (AQS) [51]. The first two are ALOHA-based and the next two are tree-based.

We use FCAT-λ to denote the FCAT protocol in which k-collision slots with $k \leq \lambda$ are resolvable, where $\lambda = 2, 3, 4$. The report probability p_i is determined based on the formula given in Section 4.4.3. Specifically, p_i is set to be $1.414/N_i$, $1.817/N_i$ and $2.213/N_i$ in FCAT-2, FCAT-3 and FCAT-4, respectively. Other values of p_i are also investigated in Section 4.6.3. The frame size f is set to 30 time slots; the performance of FCAT under different f values will also be studied. The parameters used in other protocols are selected based on their original papers whenever possible.

In the simulations, we set the time slot length based on the Philips I-Code specification [64]. The transmission rate is 53 kbit/sec. Hence, it takes 18.88 μs to transmit each bit. We set the ID length to be 96 bits (including the 16 bits CRC code), which takes 1812 μs. The reader's acknowledgement consists of 20 bits, (including the CRC code), which takes 378 μs. The waiting time before the report segment or the acknowledgement segment is 302 μs to separate transmissions. Therefore, each slot is about 2.8 ms. The simulation results are the average outcome of 100 runs.

4.6.1 Reading Throughput Comparison

We first compare the protocols in terms of the reading throughput, which is the average number of tag IDs that the RFID reader can collect in each second during the protocol execution time before all IDs are read. Table 4-1 shows the reading throughputs of the protocols when the number of tags varies from 1,000 to 20,000. Due to collision resolution, FCAT-2 achieves $51.1\% \sim 55.6\%$ throughput improvement over DFSA,

$54.8\% \sim 70.6\%$ improvement over EDFSA, $59.6\% \sim 62.9\%$ improvement over ABS, $64.1\% \sim 67.7\%$ improvement over AQS.

As expected, FCAT-3 performs better than FCAT-2, and FCAT-4 performs better than FCAT-3. However, the improvement of FCAT-4 over FCAT-3 is much smaller than that of FCAT-3 over FCAT-2. FCAT-5 (whose results are not shown in the table) performs only slightly better FCAT-4. For example, when $N = 10,000$, its reading throughput is 270.9 tag IDs per second, which is slightly better than 265.1 of FCAT-4. This indicates a quickly shrinking margin of improvement as λ increases and suggests that a large value of λ is practically unnecessary.

We also evaluate the reading time in terms of time slots. Table 4-2 shows the numbers of empty, singleton and collision slots used to read 10,000 tags. We can see that fewer empty slots are wasted in FCAT than in all other compared protocols. FCAT also uses much fewer singleton slots to collect all tag IDs because FCAT can extract tag information from the collision slots, while other protocols have to read tags solely in the singleton slots. FCAT-4 has more collision slots than FCAT-2. The reason is that FCAT-4 can utilize a collision slot in which up to four tags collide, and hence FCAT-4 encourages more tags to transmit simultaneously.

4.6.2 Effectiveness of Collision Resolution

In Table 4-3, we show the number of tag IDs that are resolved from the collision slots. FCAT-2 obtains about 40% of tag IDs from the collision slots. The percentage is above 57% for FCAT-3 and above 68% for FCAT-4. For example, when there are 10,000 tags in the system, FCAT-2 will read more than 4,000 of them from the collision slots, which are ignored by the previous protocols.

4.6.3 Report Probability

The report probability p_i is calculated as ω/N_i. N_i is the number of tags participating in slot i and the method in Section 4.5.3 is used to estimate N_i after each frame. The optimal value of ω is set in Section 4.4.3. We use simulation to confirm our analytical result and demonstrate how the value of ω affects the performance of FCAT. Fig. 4-5 shows the reading throughput with respect to ω when there are 10000 tags. If ω is set too small, the reading throughput decreases because many slots are empty and thus wasted. If ω is set too large, it also hurts the performance because there are too many collision slots and too many tags collide in those slots, making them unresolvable.

By trying all possible values of ω, we can use simulation to find the true optimal ω (and the corresponding optimal report probability) that maximizes the reading throughput. As shown in Table 4-4, the optimal value of ω observed in the simulation matches closely with the value computed in Section 4.4.3, i.e., 1.414 when $\lambda = 2$, 1.817 when $\lambda = 3$, and 2.213 when $\lambda = 4$. Also shown in the same table, the reading throughput achieved by FCAT using the computed reporting probability is almost the same as the maximum-achievable throughput under the optimal reporting probability obtained by simulation through exhaustive search.

4.6.4 Impact of Frame Size

Fig. 4-6 shows the impact of the frame size f in a system with 10,000 tags. We can see that the reading throughput is stabilized when $f \geq 10$.

4.7 Summary

This chapter concludes that the physical-layer network coding can indeed significantly improve the speed at which a RFID reader collects information of the tags. The reason is that the information carried in many collision slots, which was previously discarded, can be utilized almost as effectively as the information carried in the singleton slots. The current analog network coding method can improve the reading throughput of a RFID system by $51.1\% \sim 70.6\%$. As the technologies of physical-layer network coding are improved, the reading throughput can potentially be doubled.

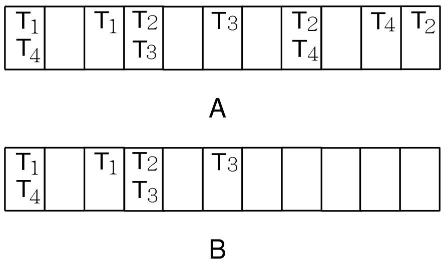


Figure 4-1. This example shows that a collision-resolution protocol may reduce the number of time slots used to identify four tags from 11 time slots to 6 time slots. A) Contention-based protocol. B) Collision-resolution protocol.

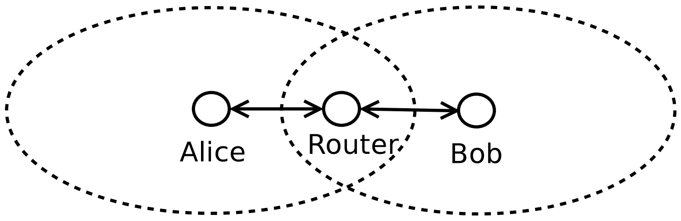


Figure 4-2. Alice-Bob example for Analog Network Coding.

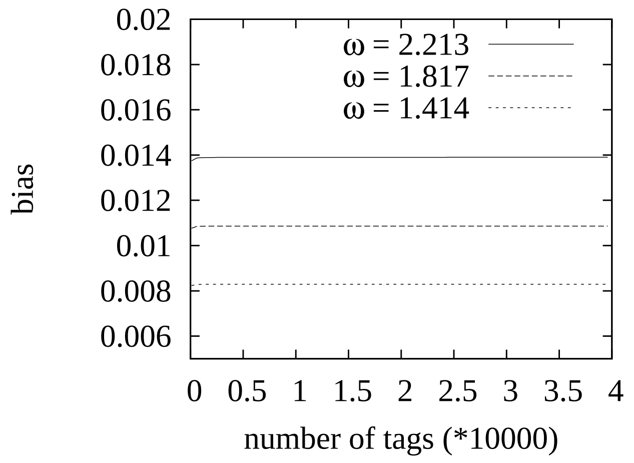


Figure 4-3. The relative bias of $\hat{N}_i$ with respect to the number of tags.

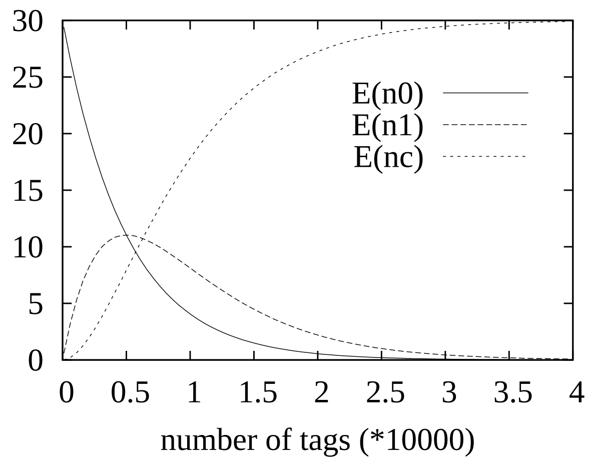


Figure 4-4. The number of tags, N_i, is not a monotonic function in $E(n_1)$. Parameters: $p_i = 1.414/N_i$ and $f = 30$.

Table 4-1. Reading throughput comparison when N varies from 1,000 to 20,000

N	FCAT-2	FCAT-3	FCAT-4	DFSA	EDFSA	ABS	AQS
1000	197.7	234.8	238.8	130.8	115.9	123.9	117.9
2000	199.5	237.2	257.5	131.8	121.5	123.7	119.4
3000	200.2	239.7	261.4	132.1	122.9	123.8	120.4
4000	201.0	240.1	262.1	132.8	124.8	123.9	120.5
5000	201.3	240.4	262.3	130.1	126.1	123.8	120.8
6000	201.3	241.5	263.7	132.4	126.3	123.6	120.9
7000	201.3	241.2	264.9	131.1	126.4	123.8	121.1
8000	201.4	241.8	265.1	131.9	127.1	123.6	121.1
9000	201.2	241.5	265.4	131.0	127.8	123.7	121.1
10000	201.3	241.8	265.1	131.4	127.8	123.9	121.2
11000	201.7	241.5	266.0	130.0	127.6	123.9	121.1
12000	200.8	241.8	265.9	130.3	126.8	123.8	121.2
13000	201.0	241.7	265.9	129.2	127.3	123.8	121.2
14000	200.4	241.3	266.2	130.9	127.6	123.5	121.3
15000	200.8	241.2	266.0	131.7	127.7	124.2	121.3
16000	200.9	241.8	265.9	131.3	128.2	123.8	121.3
17000	200.2	241.3	265.5	130.5	128.1	124.1	121.3
18000	199.7	240.7	265.9	130.0	128.2	123.6	121.3
19000	199.1	240.9	266.4	129.2	128.2	123.7	121.3
20000	199.1	241.3	266.1	129.1	128.6	123.9	121.3

Table 4-2. Empty, Singleton and Collision Time Slots when N = 10000

	FCAT-2	FCAT-3	FCAT-4	DFSA	EDFSA	ABS	AQS
empty	4189	2257	1345	10076	10705	4410	4737
singleton	5861	4055	2935	10000	10000	10000	10000
collision	7016	7497	8050	7208	7234	14409	14735
total	17066	13809	12330	27284	27939	28819	29472

Table 4-3. Tag IDs Resolved from Collision Slots

N	FCAT-2	FCAT-3	FCAT-4
1000	423	600	707
5000	2102	3008	3561
10000	4139	5945	7065
15000	6062	8819	10482
20000	7905	11507	13656

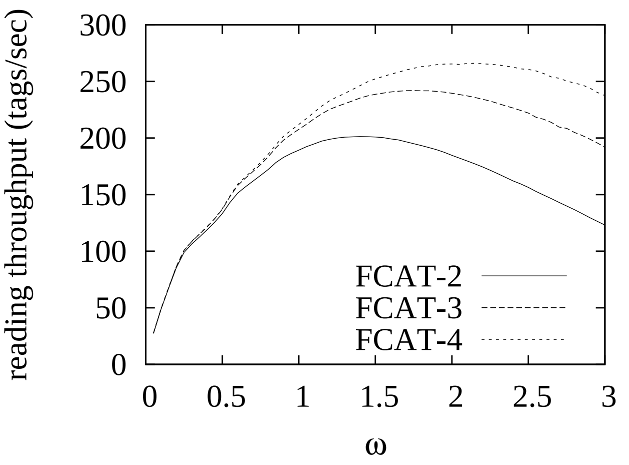


Figure 4-5. FCAT reading throughput with respect to ω.

Table 4-4. The computed value of ω matches closely with the optimal value of ω obtained by simulation.

λ	Optimal ω	Maximum Throughput	computed ω	FCAT Throughput
2	1.42	202.1	1.41	201.3
3	1.90	241.9	1.82	241.8
4	2.12	266.2	2.21	265.1

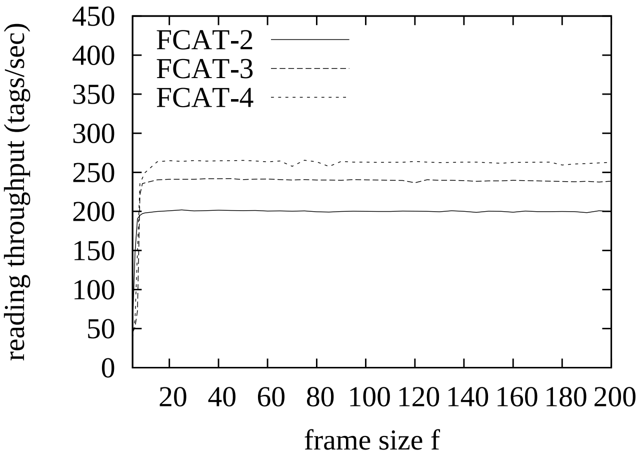


Figure 4-6. The reading throughput of FCAT is stabilized when $f \geq 10$.

CHAPTER 5
CONCLUSION

In this dissertation, we focus on fairness and throughput issues in wireless systems.

We propose novel solutions for improving fairness and throughput in contending WLANs. We start from the MAC-layer and reveal that the location-sensitive contention can exaggerate the time-allocation anomaly in 802.11 DCF networks. We propose a new protocol AIMD/QS+k which achieves proportional MAC-layer time fairness in multiple contention groups under location-sensitive contention. We then move to the transport-layer and show that a severe TCP fairness problem may occur among nearby WLANs. Most existing solutions rely on the assumption that all contending flows can explicitly exchange messages, which however may not always hold. We present our solution WPD which implicitly spreads the congestion information among contending flows. This solution can significantly improve TCP fairness when other existing solutions fail.

We introduce a new method to boost the RFID reading throughput. We demonstrate that the physical-layer network coding can be effectively integrated into RFID reading protocols. We believe this is the first work that applies physical-layer network coding to help improve the reading throughput of large RFID systems. The proposed protocol FCAT can increase the reading throughput by over 50% using the current analog network coding technology.

REFERENCES

[1] "The Network Simulator - ns-2." http://www.isi.edu/nsnam/ns/ (2008).

[2] "IEEE 802.11 WG, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification." *IEEE* (1999).

[3] "Information technology automatic identification and data capture techniques C radio frequency identification for item management air interface - part 6: parameters for air interface communications at 860-960 MHz." (2003).

[4] Abeysekera, B. A. H. S., Matsuda, T., and Takine, T. "Dynamic Contention Window Control Mechanism to Achieve Fairness between Uplink and Downlink Flows in IEEE 802.11 Wireless LANs." *IEEE Transactions on Wireless Communications* 7 (2008).9: 3517–3525.

[5] Aguilera, E. L., Heusse, M., Grunenberger, Y., Rousseau, F., Duda, A., and Casademont, J. "An Asymmetric Access Point for Solving the Unfairness Problem in WLANs." *IEEE Transactions on Mobile Computing* 7 (2008).10: 1213–1227.

[6] Akella, A., Judd, G., Seshan, S., and Steenkiste, P. "Self-management in Chaotic Wireless Deployments." In *Proc. of ACM MobiCom* (2005).

[7] Azambuja, M. C., Marcon, C. A. M., and Hessel, F. P. "Survey of Standardized ISO 18000-6 RFID Anti-collision Protocols." In *Proc. of IEEE International Conference on Sensor Technologies and Applications (SENSORCOMM)* (2008).

[8] Bejerano, Y., Han, S.-J., and Li, L. E. "Fairness and Load Balancing in Wireless LANs Using Association Control." In *Proc. of ACM MobiCom* (2004).

[9] Bhandari, N., Sahoo, A., and Iyer, S. "Intelligent Query Tree (IQT) Protocol to Improve RFID Tag Read Efficiency." *IEEE International Conference on Information Technology (ICIT)* (2006).

[10] Bharghavan, V., Demers, A., Shenker, S., and Zhang, L. "MACAW: A Media Access Protocol for Wireless LANs." In *Proc. of ACM SIGCOMM* (1994).

[11] Bui, L., Eryilmaz, A., Srikant, R., and Wu, X. "Joint Asynchronous Congestion Control and Distributed Scheduling for Multi-hop Wireless Networks." In *Proc. of IEEE INFOCOM* (2006).

[12] Camp, J., Mancuso, V., Gurewitz, O., and Knightly, E. W. "A Measurement Study of Multiplicative Overhead Effects in Wireless Networks." In *Proc. of IEEE INFOCOM* (2008).

[13] Capetenakis, J. I. "Tree Algorithms for Packet Broadcast Channels." *IEEE Transactions on Information Theory* 25 (1979).5: 505–515.

[14] Casella, G. and Berger, R. L. “Statistical Inference.” *2nd edition, Duxbury Press* (2002).

[15] Casini, E., Gaudenzi, R. D., and Herrero, O. R. “Contention Resolution Diversity Slotted ALOHA (CRDSA): An Enhanced Random Access Schemefor Satellite Access Packet Networks.” *IEEE Transactions on Wireless Communications* 6 (2007).4: 1408–1419.

[16] Cha, J. R. and Kim, J. H. “Dynamic Framed Slotted ALOHA Algorithms Using Fast Tag Estimation Method for RFID Systems.” *IEEE Consumer Communications and Networking Conference (CCNC)* (2006).

[17] Chen, L., Low, S. H., Chiang, M., and Doyle, J. C. “Cross-layer Congestion Control, Routing, and Scheduling Design in Ad Hoc Wireless Networks.” In *Proc. of IEEE INFOCOM* (2006).

[18] Chen, X., Zhai, H., Wang, J., and Fang, Y. “A Survey on Improving TCP Performance over Wireless Networks.” *Resource Management in Wireless Networking* 16 (2005): 657–695.

[19] Das, R. “Global RFID Market Tops $5.5 Billion.” http://www.convertingmagazine.com/article/CA6653688.html (2009).

[20] Dunn, J., Neufeld, M., Sheth, A., Grunwald, D., and Bennett, J. “A Practical Cross-Layer Mechanism for Fairness in 802.11 Networks.” In *Proc. of International Conference on Broadband Networks (BROADNETS)* (2004).

[21] Ergin, M. A., Ramachandran, K., and Gruteser, M. “An Experimental Study of Inter-cell Interference Effects on System Performance in Unplanned Wireless LAN Deployments.” *ACM Computer Networks* 52 (2008).14: 2728–2744.

[22] Eryilmaz, A. and Srikant, R. “Fair Resource Allocation in Wireless Networks Using Queue-length based Scheduling and Congestion Control.” In *Proc. of IEEE INFOCOM* (2005).

[23] ———. “Joint Congestion Control, Routing and MAC for Stability and Fairness in Wireless Networks.” *IEEE Journal on Selected Areas in Communications, special issue on Nonlinear Optimization of Communication Systems* 24 (2006).8: 1514–1524.

[24] Finkenzeller, K. “RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification.” (2003).

[25] Floyd, S. and Jacobson, V. “Random Early Detection Gateways for Congestion Avoidance.” *IEEE/ACM Transactions on Networking* 1 (1993).4.

[26] Garetto, M., Salonidis, T., and Knightly, E. W. “Modeling Per-flow Throughput and Capturing Starvation in CSMA Multi-hop Wireless Networks.” In *Proc. of IEEE INFOCOM* (2006).

[27] Gollakota, S. and (MIT), D. Katabi. “ZigZag Decoding: Combating Hidden Terminals in Wireless Networks.” In *Proc. of ACM SIGCOMM* (2008).

[28] Grunenberger, Y., Heusse, M., Rousseau, F., and Duda, A. “Experience with an Implementation of the Idle Sense Wireless Access Method.” In *Proc. of CoNEXT* (2007).

[29] Hamkins, J. “An Analytic Technique to Separate Cochannel FM Signals.” *IEEE Transactions on Communications* 48 (2000).4: 543–546.

[30] Heusse, M., Rousseau, F., Berger-Sabbatel, G., and Duda, A. “Performance Anomaly of 802.11b.” In *Proc. of IEEE INFOCOM* (2003).

[31] Heusse, M., Rousseau, F., Guillier, R., and Duda, A. “Idle Sense: An Optimal Access Method for High Throughput and Fairness in Rate Diverse Wireless LANs.” In *Proc. of ACM SIGCOMM* (2005).

[32] Huang, X. and Bensaou, B. “On Max-Min Fairness and Scheduling in Wireless Ad Hoc Networks: Analytical Framework and Implementation.” In *Proc. of ACM MOBIHOC* (2001).

[33] Jain, K., Padhye, J., Padmanabhan, V. N., and Qiu, L. “Impact of Interference on Multi-hop Wireless Network Performance.” In *Proc. of ACM MobiCom* (2003).

[34] Jian, Y. and Chen, S. “Can CSMA/CA Networks Be Made Fair?” In *Proc. of ACM MobiCom* (2008).

[35] Joshi, T., Mukherjee, A., Yoo, Y., and Agrawal, D. “Air Time Fairness for IEEE 802.11 Multirate Networks.” *IEEE Transactions on Mobile Computing* 7 (2008).4: 513–527.

[36] Katti, S., Gollakota, S., and Katabi, D. “Embracing Wireless Interference: Analog Network Coding.” In *Proc. of ACM SIGCOMM* (2007).

[37] Kelly, F., Maulloo, A., and Tan, D. “Rate Control in Communication Networks: Shadow Prices, Proportional Fairness and Stability.” *Journal of the Operational Research* 49 (1998).3: 237–252.

[38] Kim, H., Yun, S., Kang, I., and Bahk, S. “Resolving 802.11 Performance Anomalies through QoS Differentiation.” *IEEE Communications Letters* 9 (2005).7: 655–657.

[39] Kodialam, M. and Nandagopal, T. “Fast and Reliable Estimation Schemes in RFID Systems.” In *Proc. of ACM MobiCom* (2006).

[40] Kunniyur, S. and Srikant, R. “End-to-end Congestion Control: Utility Functions, Random Losses and ECN Marks.” *IEEE/ACM Transactions on Networking* 11 (2003).5: 689 – 702.

[41] Law, C., Lee, K., and Siu, K. Y. “Efficient Memoryless Protocol for Tag Identification.” In *Prof. of International Workshop on Discrete Algaorithms and Methods for Mobile Computing and Communications* (2000): 75–84.

[42] Lee, S. R., Joo, S. D., and Lee, C. W. “An Enhanced Dynamic Framed Slotted ALOHA Algorithm for RFID Tag Identification.” *IEEE International Conference on Mobile and Ubiquitous Systems: Networks and Services (MOBIQUITOUS)* (2005).

[43] Lin, X. and Shroff, N. “Joint Rate Control and Scheduling in Multihop Wireless Networks.” In *Proc. of IEEE CDC* (2004).

[44] Lin, X. and Shroff, N. B. “The Impact of Imperfect Scheduling on Cross-Layer Congestion Control in Wireless Networks.” *IEEE/ACM Transactions on Networking* 14 (2006).2: 302–315.

[45] Low, S. H. and Lapsley, D. E. “Optimization Flow Control, I: Basic Algorithms and Convergence.” *IEEE/ACM Transactions on Networking* 7 (1999).6: 861 – 874.

[46] Luo, H., Cheng, J., and Lu, S. “Self-Coordinating Localized Fair Queueing in Wireless Ad Hoc Networks.” *IEEE Transactions on Mobile Computing* 3 (2004).1: 86–98.

[47] Luo, H., Lu, S., and Bharghavan, V. “A New Model for Packet Scheduling in Multihop Wireless Networks.” In *Proc. of ACM MobiCom* (2000).

[48] Massoulie, L. and Roberts, J. “Bandwidth Sharing: Objectives and Algorithms.” In *Proc. of IEEE INFOCOM* (1999).

[49] Mishra, A., Brik, V., Banerjee, S., Srinivasan, A., and Arbaugh, W. “A Client-driven Approach for Channel Management in Wireless LANs.” In *Proc. of IEEE INFOCOM* (2006).

[50] Mishra, A., Shrivastava, V., Agarwal, D., Banerjee, S., and Ganguly, S. “Distributed Channel Management in Uncoordinated Wireless Environments.” In *Proc. of ACM MobiCom* (2006).

[51] Myung, J. and Lee, W. “Adaptive Splitting Protocols for RFID Tag Collision Arbitration.” In *Proc. of ACM MobiHoc* (2006).

[52] Nandagopal, T., Kim, T., Gao, X., and Bharghavan, V. “Achieving MAC Layer Fairness in Wireless Packet Networks.” In *Proc. of ACM MobiCom* (2000).

[53] Neely, M., Modiano, E., and Li, C. “Fairness and Optimal Stochastic Control for Heterogeneous Networks.” *IEEE/ACM Transactions on Networking* 16 (2008).2: 396–409.

[54] Park, E. C., Kim, D. Y., Kim, H., and Choi, C. H. “A Cross-Layer Approach for Per-Station Fairness in TCP over WLANs.” *IEEE Transactions on Mobile Computing* 7 (2008).7: 898–911.

[55] Pasupathy, S. “Minimum Shift Keying: A Spectrally Efficient Modulation.” *IEEE Communications Magazine* (1979).

[56] Peng, Q., Zhang, M., and Wu, W. “Variant Enhanced Dynamic Frame Slotted ALOHA Algorithm for Fast Object Identification in RFID System.” *IEEE International Workshop on Anti-counterfeiting, Security, Identification* (2007).

[57] Pilosof, S., Ramjee, R., Raz, D., Shavitt, Y., and Sinha, P. “Understanding TCP Fairness over Wireless LAN.” In *Proc. of IEEE INFOCOM* (2003).

[58] Rangwala, S., Jindal, A., Jang, K. Y., Psounis, K., and GovindanL, R. “Understanding Congestion Control in Multi-hop Wireless Mesh Networks.” In *Proc. of ACM MobiCom* (2008).

[59] Razafindralambo, T., Guerin-lassous, I., Iannone, L., and Fdida, S. “Dynamic and Distributed Packet Aggregation to Solve the Performance Anomaly in 802.11 Wireless Networks.” *Computer Networks* 52 (2008).1: 77–95.

[60] Release, RSA Press. “Annual RSA Wireless Security Survey: With Encrypted Wi-Fi Vulnerable, How Companies and Individuals are Risking their Assets and Reputations.” http://www.rsa.com/press_release.aspx?id=9725 (2008).

[61] Roberts, L. G. “ALOHA Packet System with and without Slots and Capture.” *ACM SIGCOMM Computer Communication Review* 5 (1975).2: 28–42.

[62] Sadeghi, B., Kanodia, V., Sabharwal, A., and Knightly, E. “Opportunistic Media Access for Multirate Ad Hoc Networks.” In *Proc. of ACM MobiCom* (2002).

[63] Sarangan, V., Devarapalli, M. R., and Radhakrishnan, S. “A Framework for Fast RFID Tag Reading in Static and Mobile Environments.” *The International Journal of Computer and Telecommunications Networking* 52 (2008).5: 1058–1073.

[64] Semiconductors, Philips. “I-CODE Smart Label RFID Tags.” http://www.nxp.com/acrobat_download/other/identification/SL092030.pdf (2004).

[65] Shih, D., Sun, P. L., Yen, D. C., and Huang, S. M. “Taxonomy and Survey of RFID Anti-collision Protocols.” *Computer and Communications* 29 (2006).11: 2150–2166.

[66] Tan, G. and Guttag, J. “Time-based Fairness Improves Performance in Multi-rate Wireless LANs.” In *Proc. of USENIX Annual Technical Conference* (2004).

[67] ———. “The 802.11 MAC Protocol Lead to Inefficient Equilibria.” In *Proc. of IEEE INFOCOM* (2005).

[68] Telatar, I. E. and Gallager, R. G. “Combining Queuing Theory and Information Theory for Multiaccess.” *IEEE Journal on Selected Areas Communication* 13 (1995).6: 963–969.

[69] Vaidya, N. H., Bahl, P., and Gupta, S. “Distributed fair scheduling in a wireless LAN.” In *Proc. of ACM MobiCom* (2000).

[70] Vasan, A., Ramjee, R., and Woo, T. Y. C. “ECHOS - Enhanced Capacity 802.11 Hotspots.” In *Proc. of IEEE INFOCOM* (2005).

[71] Vogt, H. “Efficient Object Identification with Passive RFID Tags.” In *Proc. of International Conference on Pervasive Computing* (2002).

[72] Weis, S. A., Sarma, S. E., Rivest, R. L., and Engels, D. W. “Security and Privacy Aspects of Low-cost Radio Frequency Identification Systems.” In *Proc. of International Conference on Security in Pervasive Computing (SPC)* (2003).

[73] Xu, K., Bae, S., LEE, S., and Gerla, M. “TCP Behavior across Multihop Wireless Networks and the Wired Internet.” In *Proc. of ACM WoWMoM* (2002).

[74] Xu, K., Gerla, M., and Bae, S. “How Effective is the IEEE 802.11 RTS/CTS Handshake in Ad Hoc Networks?” In *Proc. of IEEE GLOBECOM* (2002).

[75] Xu, K., Gerla, M., Qi, L., and Shu, Y. “Enhancing TCP Fairness in Ad Hoc Wireless Networks using Neighborhood RED.” In *Proc. of ACM MobiCom* (2003).

[76] Xu, S. and Saadawi, M. “Does the IEEE 802.11 MAC Protocol Work Well in Multihop Wireless Ad Hoc Networks?” *IEEE Communications Magazine* 39 (2001).6: 130–137.

[77] ———. “Revealing TCP Unfairness Behavior in 802.11 Based Wireless Multi-hop Networks.” In *Proc. of IEEE PIMRC* (2001).

[78] Yang, L., Seah, W., and Yin, Q. “Improving Fairness among TCP Flows Crossing Wireless Ad Hoc and Wired Networks.” In *Proc. of ACM MOBIHOC* (2003).

[79] Zecca, G., Couderc, P., Banatre, M., and Beraldi, R. “Swarm Robot Synchronization Using RFID Tags.” In *Proc. of IEEE PerCom* (2009).

[80] Zhai, H., Chen, X., and Fang, Y. “Improving Transport Layer Performance in Multihop Ad Hoc Networks by Exploiting MAC Layer Information.” *IEEE Transactions on Wireless Communications* 6 (2007).5: 1692–1701.

[81] Zhang, S., Liew, S., and Lam, P. P. “Physical Layer Network Coding.” In *Proc. of ACM MobiCom* (2006).

[82] Zhen, B., Kobayashi, M., and Shimizu, M. “Framed ALOHA for Multiple RFID Object Identification.” *IEICE Transactions on Communications* (2005).

[83] Zhou, F., Chen, C., Jin, D., Huang, C., and Min, H. “Evaluating and Optimizing Power Consumption of Anti-collision Protocols for Applications in RFID Systems.” In *Proc. of ACM International Symposium on Low Power Electronics and Design (ISLPED)* (2004).

[84] Zhu, J., Metzler, B., Guo, X., and Liu, Y. "Adaptive CSMA for Scalable Network Capacity in High-Density WLAN: A Hardware Prototyping Approach." In *Proc. of IEEE INFOCOM* (2006).

BIOGRAPHICAL SKETCH

Ming Zhang was born in Changsha, China, in 1977. He received the B.S. degree from Sun Yat-sen University, Guangzhou, China, in 2000, and the M.S. degree from Peking University, Beijing, China, in 2004, both in computer science. In 2004, he joined the Department of Computer and Information Science and Engineering at the University of Florida, to pursue his Ph.D. degree. His advisor was Dr. Shigang Chen. His research interests included QoS and security in wireless networks.