Quality of service in wireless local area networks

Imad Aad

THESE

présentée par

Imad AAD

pour obtenir le titre de

Docteur
de l'Université Joseph Fourier de Grenoble

(Arrêté ministériel du 30 mars 1992)

Spécialité: Informatique

Qualité de service dans les réseaux locaux sans-fil (Quality of service in wireless local area networks)

Date de soutenance: 7 octobre 2002
Composition du jury:

M. Andrzej Duda	*Président*
M. Torsten Braun	*Rapporteur*
M. Philippe Jacquet	*Rapporteur*
M. Philippe Rouzet	*Examinateur*
M. Roland Balter	*Directeur de Thèse*
M. Claude Castelluccia	*Directeur de Thèse*

Thèse préparée au sein du projet PLANETE de l'INRIA Rhône-Alpes

Abstract

IEEE 802.11 wireless networks are widely used to connect to the Internet due to their low cost, easy deployment and mobility support. In this thesis we deal with four different aspects of QoS support in these networks:

- *Service differentiation:* The current IEEE 802.11 protocol has no QoS support. Therefore, all terminals equally share the available data rate. We propose mechanisms for service differentiation on the MAC level. We develop several differentiation mechanisms and evaluate their performance through simulation.
- *Noisy environments:* IEEE 802.11 use contention windows to resolve multiple terminals' access to the channel. A terminal doubles its contention window size upon each packet loss. This mechanism reduces collisions on the channel. However, it increases packet overhead, thus reducing the throughput. Packet losses can be due to collisions or to noise on the channel as well. In the latter case, increasing the contention window size decreases the system performance; contention windows should be decreased upon packet collisions only. We propose a mechanism to adapt contention window sizes according to the estimation of the packet loss cause.
- *Congested environments:* After each successful packet transmission, IEEE 802.11 resets its contention window size. However, in congested environments, the channel load may vary so fast, and a terminal takes the risk of observing the same collisions and retransmissions rate for each packet. We propose a mechanism to decrease contention windows slowly, avoiding further collisions and retransmissions. This showed considerable throughput enhancements and reduced energy consumption.
- *Ad-hoc networks:* In ad-hoc networks, packets are routed on multi-hop path routes. Routing is therefore a cooperative work between nodes, and the average available data rate to each node depends on the number of nodes, on interference and on collisions. We propose a mechanism to control data rates at the sources, based on estimations of the throughputs and delays, aiming to optimize the useful throughputs and to reduce battery power consumption.

Keywords: Medium access control, service differentiation, quality of service, wireless networks.

Résumé

Les réseaux IEEE 802.11 sont, entre autres, très souvent utilisés pour se connecter à l'Internet car ils proposent une solution bon marché, facile à déployer et qui supporte la mobilité. Dans cette thèse nous considérons quatre différents aspects de la qualité de service (QoS) dans ces réseaux.

- *Différentiation de service:* Le protocole IEEE 802.11 actuel n'a aucun support de la QoS. Ainsi, tous les terminaux partagent équitablement le débit disponible. Nous proposons des mécanismes de différentiation de services au niveau MAC. Nous développons et simulons plusieurs mécanismes de différentiation pour IEEE 802.11.
- *Environnements bruités:* IEEE 802.11 utilise des fenêtres de contention pour résoudre l'accès multiple des terminaux au canal. Un terminal double la taille de sa fenêtre de contention à chaque perte de paquet. Cette stratégie diminue les collisions au canal, mais augmente le surcoût des paquets, diminuant ainsi le débit. Cependant, les pertes peuvent également être dues à du bruit sur le canal. L'augmentation de la fenêtre de contention peut alors être très néfaste en terme de performance. Il convient d'augmenter la feneêtre de contention uniquement si la perte a été produite par une collision. Nous proposons une stratégie d'adaptation de la fenêtre de contention qui varie selon l'estimation de la cause de perte des paquets.
- *Environnements congestionnés:* Après chaque bonne transmission de paquet, IEEE 802.11 remet la taille de la fenêtre de contention à zero. Cependant, dans un environnement congestionné la charge sur le canal varie lentement, et un terminal risque d'avoir le même taux de collisions et de retransmissions. Nous proposons un mécanisme basé sur une réduction de la taille de la fenêtre de contention plus lente, pouvant mieux éviter les collisions et les retransmissions. Ceci présente un gain considérable en terme de débits et de consommation d'énergie.
- *Réseaux ad-hoc:* Dans un réseau ad-hoc les paquets sont routés suivant des chemins multi-saut. Ainsi le routage est coopératif entre les différents nœuds, et le débit utile moyen disponible à chaque nœud dépend du nombre total des nœuds, des interférences et des collisions. Nous proposons un mécanisme de contrôle de débits aux sources, basé sur l'estimation des débits et des délais, pouvant optimiser les débits utiles ainsi que la consommation d'énergie.

Mots-clés: Contrôle d'accès au medium, différentiation de services, qualité de service, réseaux sans-fil.

Acknowledgment

While I was enjoying my time preparing my Ph.D., many volunteers were offering their humanitarian help somewhere in the “unknown” world, trying to compensate my negligence.

To all of those, thank you.

Contents

Chapter 1

Introduction

Contents

1.1 The evolution of wireless networking

In 1962, the concept of packet switching started, aiming to provide communication networks that can resist nuclear attacks and military could still have control of nuclear arms. The Internet first started in 1968, connecting four hosts between four universities in the USA, and it is growing much faster than what people could imagine at that time: Computers were typically centralized, e.g. within a single large room, high-cost such that companies or institutions owed one or two computers. The idea of owing and carrying multiple computers on a person's body, all communicating between each other and with a fixed network was out of scope. Now, it's becoming reality. This is surely due to the technological advances [5], to the competitive technologies [6, 7, 8], and to the increasing demand for more performant low-cost devices.

Since the early years of the twentieth century, wireless communications started seeing the light. They were all connection-based communications, inspired from classical telephony. In the 1970s Pr. Norman Abramson from the university of Hawaii wanted to radio-connect the university computers located at different islands. The protocol was called Aloha ("Welcome" and "Hello", among several other meanings in local language), the first connectionless random-access system. In 1997, the first wireless LAN (local area network) standard was created: IEEE 802.11. This last saw a great success due to the function it provides: replace the widely deployed Ethernet (for fixed LANs) transparently to higher layers.

Nowadays, more wireless standards are seeing the light, data rates are becoming higher and services are becoming richer. IEEE 802.11b devices are making inroads into public areas such as airports, gaz stations, coffee bars and even in community networks. Bluetooth devices, on their side, have a slow but steady growth in cellular phones, personal digital assistants (PDAs), digital cameras, headphones and notebook computers. The following examples[1] show a futuristic vision, though not very far in time, of tomorrow's communication devices' role:

- In the office: You arrive at the office and put down your briefcase. While in your office, your PDA automatically synchronizes with your desktop PC and transfers files, e-mails and schedule information. While in a meeting, you access your PDA to send your presentation to the electronic white board. You record meeting minutes on your PDA and wirelessly transfer these to the attendees before they leave the meeting.

- At home: Upon arriving at your home, the door automatically unlocks for you, the entry way lights come on, and the heat is adjusted to your pre-set preferences. Your PDA morphs from business to personal as you enter your home. An electronic bulletin board in the home automatically adds your scheduled activities to the family calendar, and alerts you of any conflicts.

- On the road: You arrive at the airport. A long line is formed for ticketing and seat assignment. You avoid the line, using your PDA to present an electronic ticket and automatically select your seat. The

[1] Courtesy of Motorola corporation: http://www.motorola.com/bluetooth.

airline's on-line system checks identification via the "ID-tag" feature built into your PDA and confirms your reserved seat. You arrive at the hotel. As you enter, you are automatically checked in and your room number and electronic key are transferred to your PDA. As you approach the room, the door automatically opens.

- In the car: As you enter a national park, a map of the park appears on your display. You can view the schedule of activities for the park and your own personal electronic tour guide is downloaded to your vehicle.

- In social settings: At the racetrack, your PDA is used to download information on selected horses and jockies, to perform statistical analysis using historical information, to place bets, to request slow-motion replays, and to order food and beverage.

1.2 Wireless applications and their requirements

As our ability to access sophisticated communication devices grows, the demand for even more sophisticated devices grows faster. Wireless communications are becoming an essential feature of everyday's life in social, scientific, medical, industrial or military field. Wireless communications provide infrastructureless networks, that means faster deployment, mobility and a much wider application field. Figure 1.1 shows some wireless applications and their corresponding requirements in data rate and transmission range. As the transmission range grows, data become more vulnerable to noise and several radio channel aspects, which reduces data rates as indicated by the gray oval in Fig. 1.1.

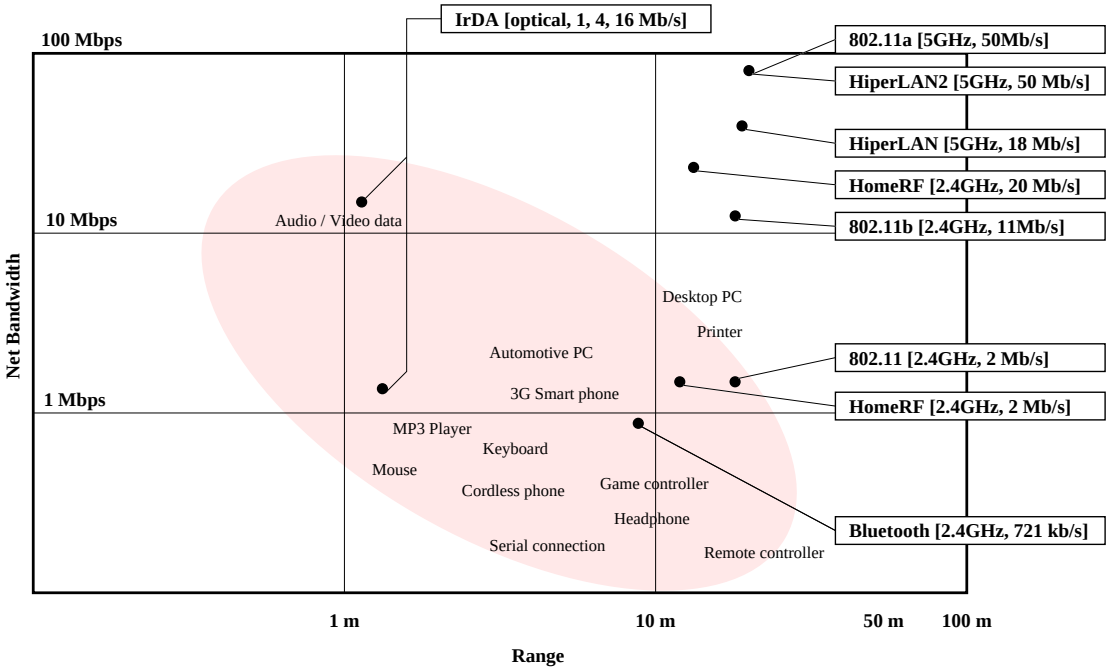


Figure 1.1: Wireless applications and their requirements

Several wireless communications standards are seeing the light to satisfy our needs and demands for more services, e.g. IEEE 802.11, HiperLAN, Bluetooth, HomeRF and infra-red (IR) communications. This growth and diversity is surely based on the fast growth in microelectronics. The use of sophisticated DSPs (digital signal processors) makes theory closer to reality, offering better modulation schemes to better cope with radio media problems. These standards vary in the services they support and in their position in the electromagnetic (EM) spectrum (Fig. 1.2).

Due to their utilization purposes, these standards operate in unlicensed frequency bands like the ISM (Industry Scientific and Medical) and the U-NII (Unlicensed National Information Infrastructure). Therefore they find themselves coexisting with other standards, usually without any coordination to reduce conflicts. Nowadays several working groups (e.g. IEEE 802.15.2) deal with coexistence problems.

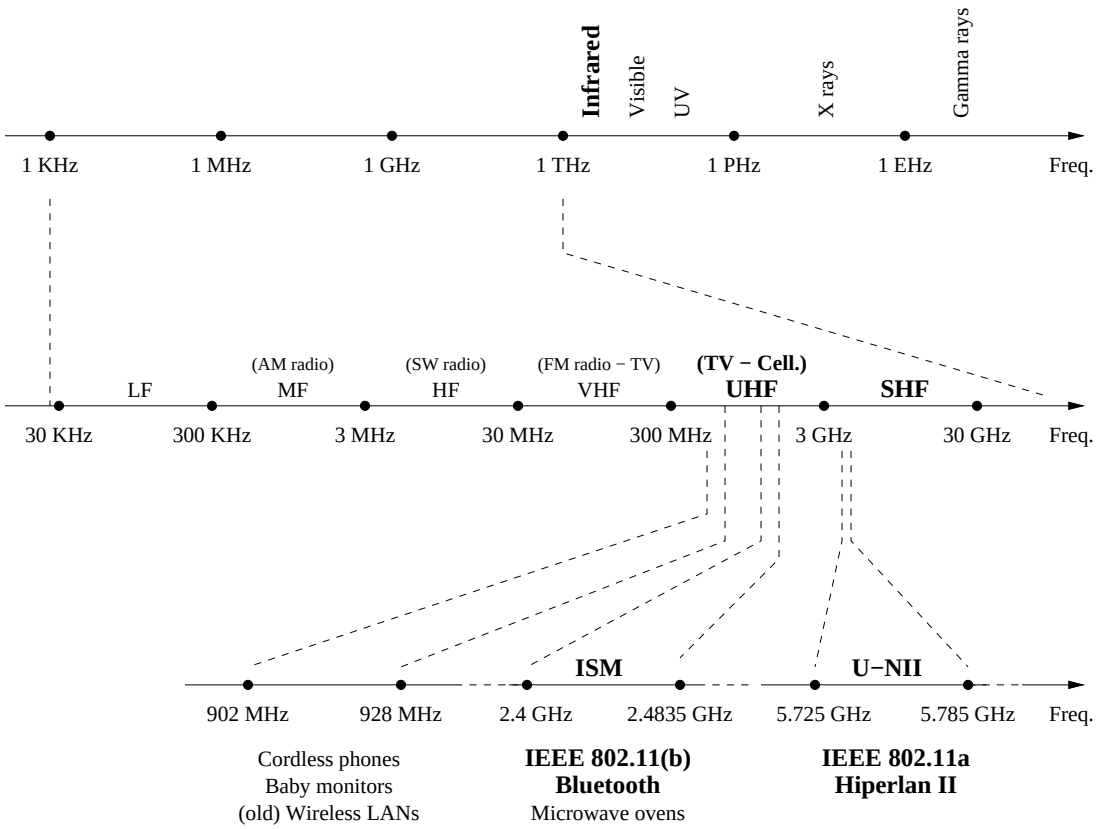


Figure 1.2: The EM spectrum utilization

1.3 Radio environments

Wireless communications differ from wired communications by the fact that electromagnetic waves will propagate in free air instead of inside cables. Therefore many issues emerge from this fact such as multipath, pathloss attenuation, shadowing, noise and interference on the channel, making the radio channel a hostile medium which behavior is difficult to predict.

Just like light, electromagnetic waves propagate in free air in a diffuse way. Between a transmitter and a receiver, not only a direct beam (case if a line of site exists) propagates. Other surrounding obstacles may also reflect the transmission, leading to multiples copies of the same signal at the receiver, delayed in time. This is called multipath (Fig. 1.3). This delay is proportional to the path length, therefore the transmitted symbol may overlap with its own copies or with other symbols (what is called inter-symbol interference, ISI) depending on the delay and the symbol period. Multipath can be exploited in a constructive manner, to reconstruct received symbol sequences. It may also make the received data incomprehensible, depending on the ISI.

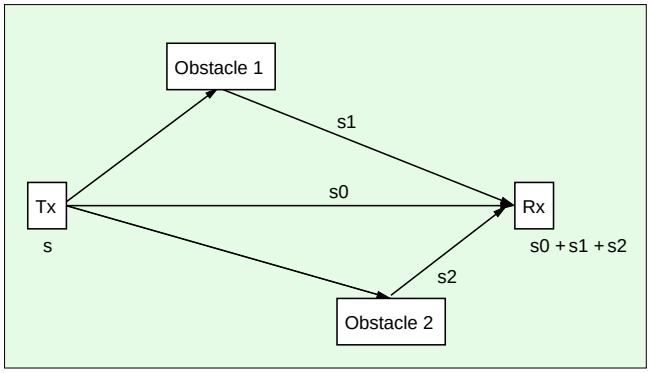


Figure 1.3: Multipath

Radio waves are severely attenuated in the air. The attenuation is proportional to the exponent of the distance. Figure 1.4 shows how the signal is attenuated with distance, in presence of line of sight (LOS) and with no LOS (turning around a corner for example).

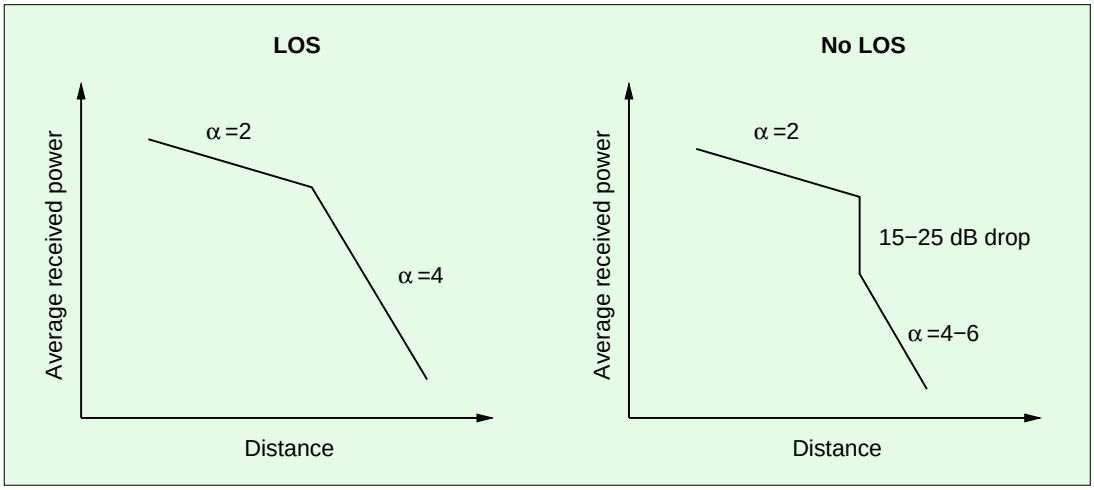


Figure 1.4: Microcell propagation in LOS and NLOS

If we combine multipath and pathloss attenuation shown above, the resulting signal will vary in space and in time. Figure 1.5 shows a signal strength map for a simple square room with a standard metal desk and an open door-way [6]. Figure 1.5 is a static snapshot; the propagation patterns change dynamically as stations and objects in the environment move. The dark (solid) blocks in the lower left are a metal desk and there is a door-way at the top right of the figure. The figure indicates relative differences in field strength with different intensities and indicates the variability of field strength even in a static environment.

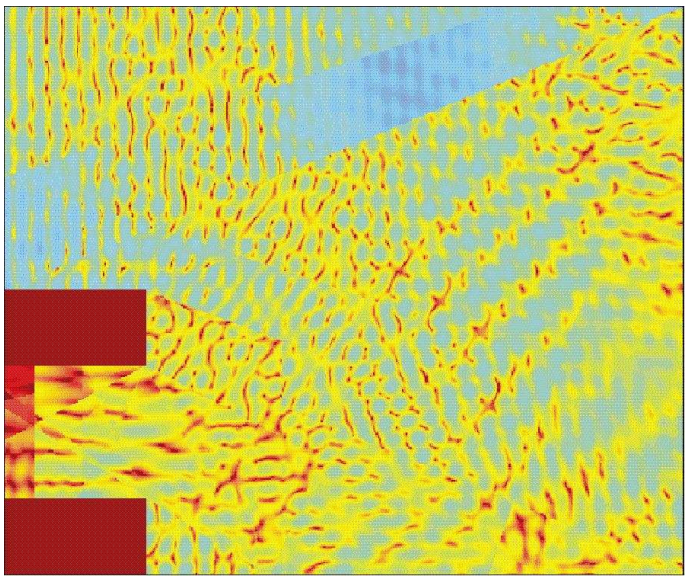


Figure 1.5: Ray tracing

Traditionally, radio channels are modeled in a statistical way using real propagation measurement data. The signal fading model is decomposed into three components:

- A large-scale path loss component, used to describe the area mean power at the receiver (hundreds to thousands of meters large area).
- A medium-scale slow varying component having a log-normal distribution, describing the local mean power within the receiver's area (tens to hundreds of meters). This medium-scale variation is called *shadowing*, caused by obstruction by trees and foliage.

- A small scale fast varying Rician or Rayleigh distribution, depending on the presence or absence of line of sight between the transmitter and the receiver. It characterizes the fast variation of the signal power over short distances (few wavelengths), or over short time durations (seconds). It is usually called small-scale fading, short-term fading or multipath fading, caused by multipath reflections over houses, buildings, forests etc.

Wired communication mediums are usually protected against external noise sources (cable shields, cable twisting etc.). However, such protection do not exist in wireless communications. Therefore radio channel errors can be due to background noise or to radio transmissions by other devices using/interfering with the same frequency band. This results in a wireless communication medium which is much less deterministic and more erroneous than its wired counterpart. In wired networks, typical bit error rates (BERs) are relatively very small, in the order of 10^{-6}. In contrast, BERs in wireless channels are in the order of 10^{-3}, and errors usually occur in bursts. Several methods are used to cope with noise on wireless channels, such as using short frames, using forward error correction (FEC) or retransmission methods. This enhances wireless packet error rates so it becomes comparable to its wired counterpart.

1.4 Quality of service (QoS)

The growth of the number of hosts in the Internet lead researchers and industrials to the question of how to support QoS, or how to support simple service differentiation. Several working groups like *DiffServ* and *IntServ* deal with these questions for wired media, on the network layer. Furthermore, wireless networks show a more critical medium which also needs QoS support for real-time applications, to cope with the increasing number of terminals and standards and with the nature of the wireless channel described in the previous section. This QoS support can be done at the network layer and/or at the medium access control (MAC) sub-layer.

Wireless networks, like their wired counterpart, can follow one of the two philosophies: Circuit switched or packet switched networks. The former is inspired from circuit switched telephony/voice networks. In these networks, QoS support is relatively easy due to the centralized nature of the approach, the signaling used and the simple admission control.

On the other hand, packet switched networks, typically the Internet, offer only best-effort services eliminating the need for signaling. No QoS guarantees can be given, and no central control is needed neither. This allows the network to scale better, and to support a very wide range of applications, which lead the Internet to its current success. However, new applications like real-time voice or video require a minimum level of service guarantees and separation between traffic classes which is not supported yet in the Internet, nevertheless it started seeing the light.

The existing wireless networking standards follow the previous two philosophies. HiperLAN emerges from the circuit switched networks but also aims to support a wide range of protocols, including Ethernet-like ones. IEEE 802.11, however, is inspired from Ethernet, underlying IP (Internet protocol) packet switched networks.

Our work in this thesis aims to support QoS in packet switched wireless networks, mainly IEEE 802.11. It is divided into four parts:

- Part I, *Wireless LAN technologies:* Chapter 2 describes wireless medium access control protocols. Next, chapter 3 shows several aspects of IEEE 802.11 MAC and physical protocols, emphasizing details on the MAC sub-layer which is of interest to our work. Other wireless networking standards, namely HiperLAN-2 and Bluetooth are described in Chapter 4.

- Part II, *Service differentiation in IEEE 802.11:* Chapter 5 contains several differentiation mechanisms we propose for IEEE 802.11, describing their behavior in noisy environments and showing the need for per-flow differentiation that we also explored. QoS support in wireless networks is currently the aim of several research groups. In Chapter 6 we cite some of them and detail four of them, of more interest to our work.

- Part III, *Enhancing IEEE 802.11 in noisy and in congested environments:* This part is composed of two chapters, 7 and 8, in which we describe the behavior of IEEE 802.11 in noisy environments and in congested environments respectively. In each of the two chapters, we propose and analyze possible enhancements to the protocol.

- Part IV, *IEEE 802.11-based ad-hoc multi-hop networks:* In the previous chapters, several aspects of wireless medium access protocols helped us to investigate some ways to estimate throughputs and delays in ad-hoc networks. We introduce our approach in Chapter 9 and validate it with simulation results.

Last, Chapter 10 concludes this thesis. We should note that some parts of the thesis were published and are known to the community, while others are still being developed and opened the way for many future research topics.

Part I

Wireless LAN technologies

Chapter 2

Wireless medium access control protocols

Contents

2.1 Introduction

Nowadays, several types of communication mediums are used by terminals to exchange voice and data: Light can be used to communicate through fibers, sound can be used to communicate through the water and electromagnetic (EM) waves can be used to communicate via cables, each with different propagation properties. Light, sound or EM waves[1] can also be used to carry voice and data through the air.

Regardless of the type of the medium, a protocol is needed by the communicating parties to orderly access the shared resource fairly, and in an efficient way. Just like when people communicate/talk in real life, these protocols have different conveniences in different scenarios: one may ask for a permission before talking, or just listen if someone is talking before speaking, etc.

This metaphor can describe the medium access control (MAC) to further extents: In a given area, only one speaker is allowed to talk at a time, else, the listener would hear noise, unless one is speaking much higher than the others. However, speaking loud prevents (more) further people from talking at the same time. Furthermore, it exhausts the speaker. Talking low interferes with less people, enabling the ones far enough to communicate with each other at the same time. However, talking low is vulnerable to noise. Two or more persons in the same area speaking simultaneously result in incomprehensible noise and they will have to repeat what they said. This wastes time, the speakers' and the listener's energy too. To avoid further conflicts, they would either wait different times before talking again, or wait for a coordinator to ask them to speak, etc. depending on the situation.

MAC protocols moderate access to the shared medium by defining rules that allow parties to communicate in an orderly manner, ensuring efficiency and fairness. This chapter surveys different types or wireless radio MAC protocols and discusses their characteristics.

Section 2.2 shows background concepts in wireless MAC protocols. Section 2.3 shows the wireless MAC protocols evolution since the 1970s, the various MAC protocol types are described in Section 2.4. Last, Section 2.5 concludes this chapter. Several parts of this chapter are mainly inspired from [9].

2.2 MAC basics

In the previous chapter we cited several aspects of the radio channel which have destructive impact on EM signals. A radio signal transmitted by a station propagates through the air while getting attenuated. Below

[1] Physically, light is an EM wave, but we use them separately to distinguish fiber communications from cable communications.

a given *receive threshold* (RX_Threshold), the information carried by the signal becomes unreadable, and the corresponding distance from the transmitter is called the range of the sender. Two other thresholds, namely *interference threshold* and *carrier sense threshold* (CS_Threshold) define signal levels above which the signal still can interfere with other transmissions and above which the signal can still be sensed, respectively. Obviously, the receive threshold is greater than the interference threshold, which is greater than the sensing threshold. In this chapter we just consider the receive threshold and its corresponding range only.

Due to this signal attenuation, data transmission and reception becomes location dependent, function of the position of the receiver relative to the transmitter. Note that a node senses the channel for ongoing transmissions before starting its own transmission to avoid collisions at the receiver. This will be detailed in the next section. We distinguish two possible situations (refer to Fig. 2.1, taken from [9]):

- *Hidden nodes*: A hidden node is one that is within the range of the receiver, but out of range of the sender [10, 11]. e.g. in Fig. 2.1 node A is transmitting to node B. Meanwhile, node C has a packet to transmit to node B. C senses an idle channel as it is out of range of A. Therefore it starts its transmission which causes collision at B, which is in range of both A and C. Node A is *hidden* to node C. Hidden nodes increase collisions, therefore reducing efficiency.
- /textitExposed nodes: Exposed nodes are complementary to hidden nodes. An exposed node is one that is in the range of the transmitter, but out of range of the receiver [11]. e.g. in Fig. 2.1, consider that B is transmitting a packet to A. C senses the channel busy, and therefore defers the transmissions of any packet it has, to avoid collisions. However, C could start its transmissions without causing collisions since A is out of range of C. Node C is *exposed* to node B. Exposure reduces throughput efficiency.

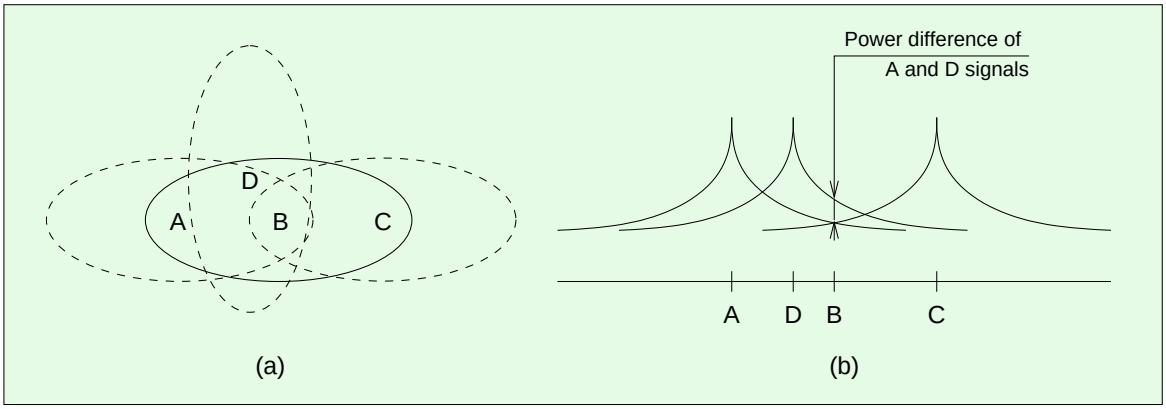


Figure 2.1: Hidden nodes, exposed nodes and the capture effect.

Collisions may occur when a node, hidden to another, starts a transmission while the other node is transmitting. It could also be the case of two nodes within range of each other, starting transmissions at the same time. However, the reception still can be successful if the power signal of one node is considerably higher than the signal of the other node (namely, the power ratio is higher than a given *capture threshold*). In Fig. 2.1, if A and D transmit simultaneously to node B. The signal from D may be considerably higher than that of node A at node B, therefore B can cleanly receive the information of node D. Note that the power curves in Fig. 2.1 (b) are drawn for presentation convenience only. The slopes must be much steeper due to attenuations inversely proportional to powers (≥ 2) of the distance, making the curves look like thin pulses.

The fact that the signal is highly attenuated with distance makes collision detection impossible in wireless networks. In fact, when a node is transmitting, a large fraction of the signal leaks into the receive path. Since the self-transmitted signal differs from signals transmitted by distant stations by orders of magnitude, the receiver would be "blinded" by the self-transmission. Therefore, while transmitting data a node cannot listen to the channel, for instance to detect collisions as in Ethernet, making it half-duplex. A feedback channel can be used to inform the stations of eventual collisions, as detailed in the following sections. Since collisions cannot be detected immediately, collision avoidance protocols should be considered to increase efficiency.

Communications can be multiplexed in time (TDD, time division duplex) since they nodes can only operate in half-duplex mode. Using TDD, nodes transmit during given time slots, and receive during other time slots, in the same frequency band. When using very high data rates, the overhead of switching time between the transmitter and the receiver becomes considerable. FDD (frequency division duplex) refers to multiplexing transmission and reception on different frequency bands, which allows nodes to transmit and receive at the

same time, which is not possible with TDD. However, this technique requires more complex transceivers.

Wireless networks can either be distributed or centralized. Distributed wireless networks, also called ad-hoc wireless networks, have no centralized coordinators/administrators, making it more robust than centralized architectures. Ad-hoc networks only operate in TDD. On the other hand, centralized wireless networks are usually connected to wired infrastructure, extending it to the wireless terminals at the last hop. Centralized wireless networks have base stations (BSs), also called access points (APs), which act as interfaces between the wired and the wireless parts of the network. The centralized nature of these networks make them able to support extra services easily, such as QoS support. However, they are less robust and more complex to deploy than ad-hoc networks. FDD and TDD can be used in centralized wireless networks to multiplex communications on the up-link and the down-link[2].

For each of these two architectures, several MAC protocols were proposed in the literature. Each of these protocols has its own characteristics and may be more convenient for some scenarios than for others. We will describe some of these protocols in the following sections. The common metrics used to evaluate these protocols, either in this chapter or in this thesis in general are:

- *Delay:* Real-time traffic flows are sensitive to packet delays. Delay is the amount of time spent by a packet to successfully reach its destination, taking into account queuing delays, retransmission delays etc.
- *Throughput:* MAC protocols are mainly compared by their efficiency using the channel resources. Throughput is the fraction of the channel capacity used to transmit data. To maintain this fraction high enough, overhead should be reduced, collisions (and retransmissions) should be avoided etc.
- *Fairness:* This describes the MAC protocol capability to distribute the available resources equally among communicating terminals [12]. This definition can be biased when we intend to support QoS and service differentiation in wireless networks. In this case, fairness is the capability to distribute bandwidth in proportion to their intended allocation.
- *Stability:* Due to the overhead in a MAC protocol, the system may be able to handle sustained source loads that are much smaller than the maximum transmission capacity of the channel. A stable system can handle instantaneous loads that are greater than the maximum sustained load when the long-term offered load is less than the maximum.
- *Power consumption:* Power consumption is critical to all wireless devices since it is decreases battery life. Power consumption is usually composed of two factors. Processing energy and radio transmission energy. This makes MAC protocols responsible of optimizing power consumption. This can done by reducing transmission overhead, collision (and subsequent retransmission) avoidance, and the capability to support low power modes etc.

2.3 Evolution of MAC protocols

As introduced in the previous chapter, research in medium access control for wireless networks first started in the 1970s. The initial protocols were studied for data and satellite communications. Aloha [13, 14] is the first wireless MAC protocol to see the light, in 1970, used by Pr. Abramson to connect the university computers on different islands using radio transceivers. The protocol simply works as follows: When a node has a packet to send, it transmits it. If the packet collides with another transmission, the node retransmits it after a random period. Obviously, when the number of nodes increases, each with a given packet data rate, the probability of having a collision increases too, and packets will be retransmitted. When the number of packets to send becomes considerably high, packets would suffer many collisions and retransmissions before being successfully received. Therefore the rate of packets successfully reaching the receiver (i.e. the throughput) is considerably reduced, as shown in Fig. 2.2.

We can see that the best throughput Aloha can achieve is 18% , when the probability of a packet transmission at a given time is 0.5. This relatively low throughput can further be doubled ($1/e$) if we consider slotted time, and that stations can only transmit at the beginning of a time slot. The vulnerable period of a transmission in therefore halved, doubling the efficiency of the system [15], to the cost of more complexity of synchronizing all stations together. The protocol is called slotted Aloha (S-Aloha).

Carrier sense multiple access (CSMA) proposed in [16] showed considerable enhancements over Aloha and S-Aloha (see Fig. 2.3). The basic idea in CSMA is that a node listens to the channel before starting a new transmission. CSMA is location dependent, due to the considerable radio signal attenuation in wireless mediums. Hidden nodes transmitting packets still cannot be sensed, limiting the performance of CSMA in

[2]This terminology is inherited from satellite communications.

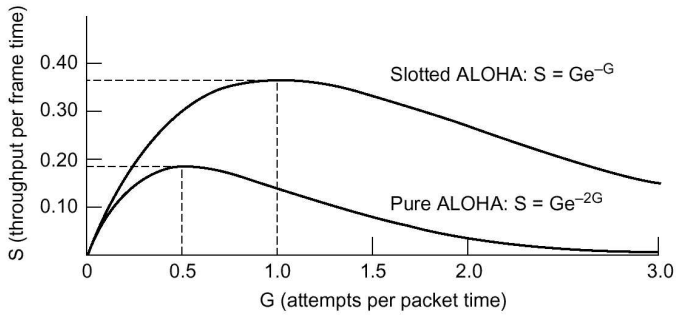


Figure 2.2: Pure Aloha and Slotted Aloha throughputs.

wireless networks. CSMA also stops exposed nodes from starting a new transmission, limiting the overall network throughput as well. However, CSMA's advantages overrun its drawbacks, therefore it is used in all random access protocols considered later on.

CSMA has persistence versions: 1-persistent, non-persistent and p-persistent.

In 1-persistent CSMA, whenever a node has a packet to transmit, it senses the channel. If the channel is idle, it transmits the packet with probability 1, i.e. immediately. If we consider radio propagation delays to be negligible, the probability of having two stations starting transmission at the same time is also negligible, due to the fact that one node would most probably detect the transmission of the other and defer its own transmission in time. However, this probability is increased when two nodes sense a third node's transmission. Both will wait the channel to become idle and start their transmissions immediately, causing collisions. This can be avoided by using non-persistent CSMA: When a node senses a busy channel, it defers its transmission to a random time in the future, instead of sensing the channel continuously. If the channel is sensed busy again, the same procedure is repeated, until the channel becomes idle, then the node transmits its packet. This considerably reduces collisions, increasing the throughput (Fig. 2.3) to the cost of higher packet delays.

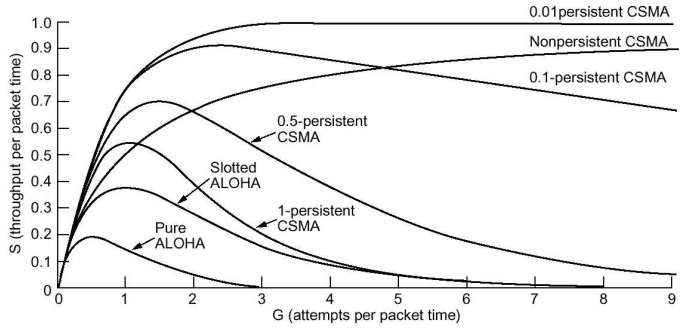


Figure 2.3: Multiple access throughputs.

An intermediate scheme shows better throughputs than 1-persistent and non-persistent CSMA: p-persistent CSMA. This scheme senses the channel for ongoing transmissions, if the channel is idle, a node transmits its packet with probability p and differs until the next slot with probability $1 - p$. If the channel remains idle during the next slot, the same procedure is applied again. The less the persistence factor p is, the more we avoid collisions, therefore the throughput becomes higher, to the cost of higher packet delays.

During a collision, the concerned nodes keep transmitting their frames unaware of the collision taking place. This wastes time and bandwidth since none of the transmitting nodes succeeds to deliver its packet (if we don't consider the capture effect). The waste becomes more considerable when packets are large.

To cope with this, CSMA is enhanced using collision detection (CSMA/CD). Whenever a transmitting node senses the signal on the channel to be different from the one of its transmitter, it aborts transmission, saving time and bandwidth. CSMA/CD has been widely used in local are networking (IEEE 802.3 [17], Ethernet [18]). However, collision detection is only possible in wired networks, since a node cannot listen to the channel while transmitting on a radio channel. Collision detection is therefore replaced by other mechanisms aiming to avoid collisions in wireless networks. Collision avoidance can be out-of-band based or handshaking based, as detailed in the following paragraphs.

Busy tone multiple access (BTMA) [10] is an example of collision avoidance protocols that use out-of-band signaling. Any node that hears an ongoing transmission transmits a busy tone. Any node that hears a busy

tone does not initiate a transmission. If we consider the range of a node to be R, BTMA ensures that no other node within $2R$ from the transmitter starts a new transmission. For different frequencies, transmission ranges are different considering the same transmission power. We assume they are equal ranges for convenience. This solution eliminates the hidden node problem, however, it increases the number of exposed nodes, reducing the overall network throughput.

An enhanced version of BTMA is the receiver initiated BTMA (RI-BTMA) [19]. In this approach a node transmits a busy tone only after it decodes the transmission and identifies itself as the intended receiver. Therefore, the number of exposed nodes is reduced since only the neighbors of the receiver are inhibited from starting a new transmission. This scheme is relatively more complex than BTMA, and it also needs more time for the receiver to decode the transmission before starting the busy tone which increases the collision probability.

Using out-of-band signaling such as in BTMA has two major drawbacks:

- Different frequencies for data and busy tone have different propagation characteristics, therefore different ranges.
- Hardware becomes more complex.

Another alternative for collision avoidance, using a single band, is multiple access with collision avoidance, MACA [20, 21]. The idea is based on three-way handshaking to deal with the hidden node problem. A node with a packet to transmit transmits a short request to send (RTS) and waits for the corresponding clear to send (CTS) packet from the destination. All stations within the ranges of the sender and the receiver defer their transmission upon hearing the RTS and CTS respectively. Collisions are not completely avoided when using RTS/CTS, however their collision probability is considerably reduced when hidden terminals exist, due to the small packet sizes, and the bandwidth waste is also reduced in case of collisions. MACA shows and in-band alternative for collision avoidance which enhances the network performance considerably. It does not, however, show a general solution to all possible scenarios like exposed terminals etc. Further enhancements to MACA can be found in [11, 22, 23, 24], which have other handshaking overhead drawbacks as well.

2.4 MAC protocol types

MAC protocols can be classifies as in Fig. 2.4 [9]. On the first level, MAC protocols can be divided into two groups: Distributed or centralized protocols. Distributed protocols can be used in any network architecture, while centralized protocols can be used in centralized networks only. Typically, distributed MAC protocols use random access techniques, while centralized MAC protocols offer a wider variety of access techniques: Random, guaranteed and hybrid. They differ by their complexity, their efficiency and the overhead they add on the communication channel. Some are more convenient for delay sensitive packets, others are convenient for data packets. We will describe these protocol classes and give example protocols in the following subsections. In the following chapters, we describe several wireless networking standards most of which are based on these protocols.

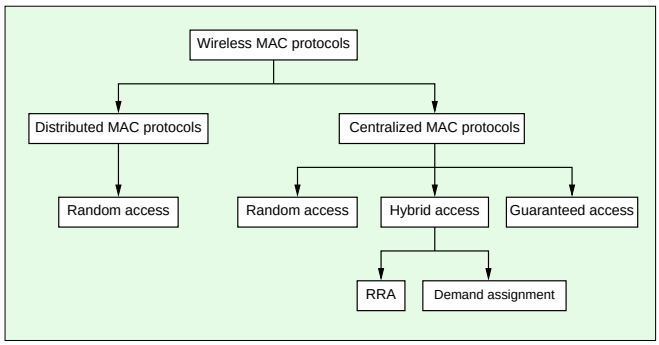


Figure 2.4: Wireless MAC protocols.

2.4.1 Distributed MAC protocols

Distributed MAC protocols are typically based on random access mechanisms: Nodes contend for access to the medium. Furthermore, all but Aloha use CSMA and collision avoidance mechanisms, as described in the previous section.

Distributed foundation wireless medium access control (DFWMAC)[25] is an enhancement of the MACA protocol previously described, and it is the basic protocol for the IEEE 802.11 standard. DFWMAC is a four-way handshake protocol, which has an ACK (acknowledgment) packet sent directly after successful data packet reception in addition to the MACA three-wake handshake. Therefore the handshake in DFWMAC is a RTS-CTS-DATA-ACK. The description of the protocol can be found in Chapter 3. A short description follows for completeness. When a node has a packet to transmit, it waits the channel to become idle for a DIFS (distributed interframe spacing) period (see Fig. 3.5), then it chooses a random backoff time. The backoff time is decreased as long as the channel is sensed idle, and is frozen when the channel is busy. When the backoff expires, the node transmits an RTS to the destination. If the destination is ready to receive data, it responds with a CTS to the sender after a short interframe spacing (SIFS). When the sender receives the CTS, it sends the data packet after a SIFS and waits for an ACK from the destination. If the sender receives no ACK, it retransmits the packet. If it receives no CTS, the sender assumes the channel is high loaded, and it doubles the range of its backoff for future retransmissions, to reduce the collision probability. This is called binary exponential backoff (BEB). The current transmission duration is included in the header of the data packet, in the RTS and in the CTS, so when other nodes hear one of these, they update their network allocation vectors (NAV) and wait them to expire before starting new transmissions. To keep the four-way handshake in order, without possible interruptions from new data transmissions, SIFS is lower than DIFS, therefore new RTS transmissions has no chance to grab the channel before the current transmissions. The performance of this protocol (in the context of IEEE 802.11) has been heavily discussed in the literature [26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38]. [22] discusses further possible enhancements to DFWMAC.

Elimination yield - Non-preemptive priority multiple access (EY-NPMA) is the second distributed random access protocol we are going to describe here. EY-NPMA is the channel access protocol used in HiperLAN-1 [39] illustrated in Fig. 2.5.

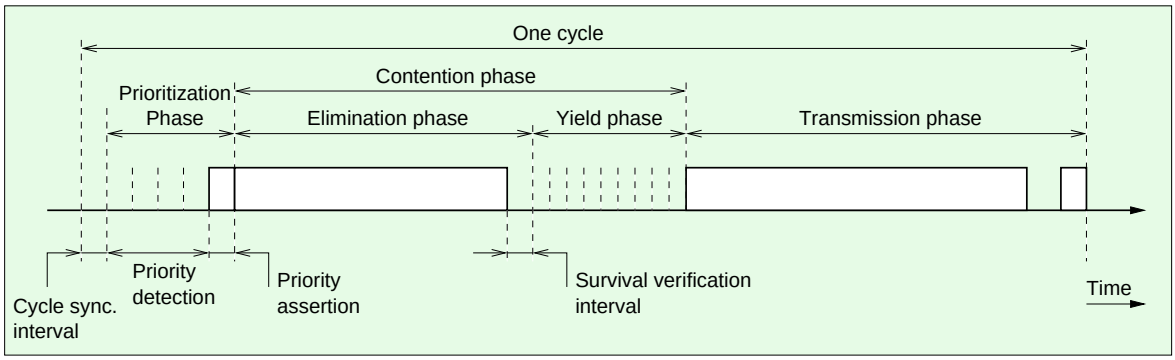


Figure 2.5: Access method in EY-NPMA.

A node that has a packet to transmit waits the channel to become idle for a given period of time. If the channel stays idle, the node can transmit its packet immediately. Else, the node waits the end of the transmission and synchronizes itself with other possible waiting nodes. After the synchronization period, a node enters the prioritisation phase where its priority over other nodes is resolved, based on the time that the packet has been in the queue waiting for transmission. The more the packet is delayed, the higher is its priority. Prioritisation phase is followed by the contention phase, composed of two sub-phases:

- *Elimination phase:* In this phase each node occupies the channel for a random number of slots. At the end of its transmission, a node listens to the channel. If the channel is busy, transmission is aborted. Else, the node goes to the yield phase.
- *Yield phase:* In this phase a node listens to the channel for a random number of slots. If the channel remains idle, the node transmits its packet (in the following transmission phase).

More details concerning EY-NPMA can be found in [40, 41, 42].

2.4.2 Centralized MAC protocols

Centralized MAC protocols move the complexity to the base station which acts as a coordinator between the nodes. Therefore the BS is assumed to be able to communicate with all nodes associated with it. All communications must go through the BS, therefore the bandwidth is not used in an efficient manner. This is

one major drawback of centralized protocols. On the other hand, hidden and exposed node problems do not exist, and contention between nodes is considerably reduced.

Centralized random access protocols

In the following, we are going to describe three centralized random access protocols. The first is ISMA, (idle sense multiple access) [43]. In this protocol the BS senses the channel and announces if it is idle by sending an *idle signal*. Upon hearing this signal, each node that has a packet to send starts transmission with a probability p, so collisions between two or more waiting nodes are reduced. If one transmission is received, the BS sends an *ACK+idle signal* packet. However, in case of collisions, the BS does not receive any of the transmitted packets properly, and it transmits an *idle signal* again. A collision between two or more data packets wastes time and bandwidth, which led to the same motivations for using MACA in distributed systems: Transmit short reservation packets prior to actual data packets, thus reducing collision overhead. This is the idea behind R-ISMA (reservation idle sense multiple access) [44], where a node transmits a reservation packet a random time after hearing the idle signal from the BS. The BS then sends a polling signal, after which the concerned source transmits its data packet. Performance evaluation of this protocol can be found in [45, 46, 47].

Randomly addressed polling (RAP) [48] is a contention based multiple access protocol. Nodes transmit pseudo-random orthogonal codes during a contention phase. Orthogonality allows the BS to receive and decode all the transmitted codes simultaneously, using CDMA (code division multiple access) technologies, required for this type of contention. The base station then polls each of the nodes consequently, which transmit their data after being polled. The BS then acknowledges data reception. Possible collisions occur when two or more nodes use the same pseudo-random code during the contention period. The code is clearly received by the base station which would use it for polling. More than a node will respond to the call by sending their data packet, resulting in a collision, to which the BS responds by a negative ACK (NACK). Reservation RAP (R-RAP) [49] extends RAP to support stream traffic. Using R-RAP, a reservation pseudo-random code used by a node is reserved for that node during the call duration, if the node has stream traffic to send. Therefore this code is removed from the set of available random numbers. GRAP (Group RAP) [50] shows some improvements over RAP by grouping reserved transmissions into super-frames, and allowing new transmissions at the last frame. GRAPO (GRAP optimized) [51] further enhances GRAP efficiency by allowing dynamic changes in the number of groups in a super-frame.

In resource auction multiple access (RAMA) [52, 53], contention resolution is based on an ID each node has. During the contention phase, each node that has a packet to send transmits its ID symbol by symbol. What the BS hears is the bitwise sum (OR) of the symbols. After each symbol, the BS broadcasts what it heard, which is considered as an ACK for the nodes with the matching-symbol. The procedure continues until the end of the ID the BS heard. The node with the highest ID always wins the contention, and transmit its data packet. The main advantage of this approach is that a slot is never wasted, since there is always a single winning node. However, the protocol is highly unfair, always in favor nodes with the highest IDs. [54] tries to deal with the fairness problem of RAMA, by randomly choosing a node among contending ones. However, it is not clear how symbols can be distinguished on the channel.

Guaranteed access protocols

In a guaranteed access protocol, nodes access the medium in an orderly manner, e.g. round-robin. This can be achieved in two ways: using a coordinator which polls each of the nodes for transmission or by using tokens. This last mechanism consists of passing a single token between the nodes. The node having the token is the only one allowed to transmit its packet. Note that token-based mechanisms can be used as distributed MAC protocols. However, token loss is common, and recovering it consumes a lot of time. Therefore token-passing is not commonly used in distributed protocols which remain typically random access protocols.

Therefore polling is the only mechanism capable of providing guaranteed access, and this requires a centralized architecture with a coordinator/BS at the "center" of the wireless network.

[55] proposes that a BS polls all the network nodes in a round-robin manner. Nodes with packets ready to transmit reply with a *request*. Nodes with no packets ready to transmit reply with a *keep alive* message. The BS then polls again the nodes that sent *request* messages, one after the other, so they can transmit their packets. All nodes must be periodically polled for requests in a way that the BS keeps knowledge of existing nodes after possible channel changes. [56] modifies the previous protocol by eliminating the poll-request from the poll-request-poll-data. The protocol is called disposable token MAC protocol (DTMP). The basic idea is that a BS sends a poll with an indication whether the BS has data packet to send to the polled node. If neither the BS nor the polled node has packets to transmit, the polled node remains silent. If the BS informed the polled node it has data to send, the node responds with a short message if it has no data, or with a data packet

the node might have. Thereafter, the BS transmits its data packet and the node acknowledges it.

In [57], the BS starts a polling phase where first the BS polls each node using a unique code, and the node replies by an echo code if it has packets to transmit. The BS then broadcasts the code so every node is aware of all other active nodes. Follows the request phase where each node, in an ordered manner, sends a request to send its data. The BS station then polls each of the nodes that sent requests, so they can transmit their data.

These three protocols show similar performance properties. Moreover, we should note that they do not offer any QoS support.

Hybrid access protocols

Most hybrid access protocols are based on request-grant mechanisms. Each node sends a request to the base station, indicating how much time or bandwidth is required to transmit the data in its buffer. The request may be sent using a random access protocol, the base station may then reserve a time slot (periodically) in the upstream for that request, based on admission control algorithms. In this case it is called random reservation access.

It can also be the case where the base station collects all the requests from the nodes with all their QoS requirements then it makes bandwidth allocations on the upstream based on scheduling algorithms, similar to those in wired networks. These protocols are called demand assignment protocols.

Random reservation access (RRA) protocols: RRA protocols try to achieve stochastic multiplexing of data on TDMA systems. Nodes that have packets to send use random access schemes, such as p-persistence or S-ALOHA to transmit their reservation packets. These reservation packets are used by the BS to reserve time slots at the uplink. The first RRA protocol we are going to cite is packet reservation multiple access (PRMA) [58]. This protocol is proposed to multiplex data and speech on cellular networks. A frame is composed of several slots. A node with a voice packet to send waits an idle slot and transmits its packet with probability p, which implicitly reserves subsequent periodic slots. A data packet contends similarly, however it does not make subsequent reservations. [59] enhanced the scheme by allowing data to make slot reservations, decreasing collision probabilities. However, a maximum reservation threshold is used to avoid nodes from sending long data bursts, starving other nodes. [60] separates voice and data contentions to avoid data from decreasing voice system performance. The ratio of voice/data shares is dynamically adjusted from frame to frame. [61] makes another separation between request slots and data slots. This makes the protocol more stable in high contention periods, where many slots are reserved and many nodes are still contending to access other slots.

Random reservation access - independent stations algorithm (RRA-ISA) [62, 63] is a protocol where the BS uses an algorithm aiming to maximize the throughput from slot to slot. Based on the history of previous slots, the BS computes the set of nodes to poll so the probability of a single transmission in a slot is maximized. This reduces collisions (throughput waste) and increases efficiency.

Demand assignment protocols: These protocols aim to allocate bandwidth to nodes according to their QoS requirements. Obviously, strict QoS guarantees cannot be satisfied using random access protocols, for instance, under heavy contention loads. Typically, strict QoS guarantees can be offered using centralized architectures and protocols, where the BS gathers different requirements from different nodes and then schedules their transmissions accordingly.

Centralized PRMA (C-PRMA)[64] uses scheduling in the BS in addition to PRMA previously described to support QoS. Nodes send their QoS requirements to the BS along with the reservation packets in the random access slots. The BS schedules different uplink transmissions, granting the next transmission to the node with the closest deadline. This is called EDD (earliest due date) scheduling. DQRUMA (distributed-queuing request update multiple access) [65] multiplexes uplink and downlink using FDD. The uplink is sub-divided into request channel where nodes send their contention requests (using random access protocol), and the data channel to send data packet, with eventual requests piggybacked to data. On the downlink, the BS sends data packets, ACK packets and transmit-permissions for nodes to send their data.

MASCARA (mobile access scheme based on contention and reservation for ATM)[66] uses variable length frames. A frame is composed of three periods, the length of which can be adjusted dynamically:

- Broadcast period, during which the BS tells the nodes about the current frame structure, length, transmission schedules on the uplink etc.
- Reserved period, further divided into uplink and downlink transmission periods.
- Contention period, for the nodes to send new requests to the BS using S-Aloha.

MASCARA is similar to the multiple access protocol used in HiperLAN-2, described in Chap. 4. However, unlike MASCARA, the contention resolution protocol in HiperLAN-2 is BEB-like that adapts better to the number of contending nodes dynamics.

2.5 Conclusion

In this chapter we showed the evolution of wireless medium access control protocols and further classified them into distributed or centralized, random access, guaranteed access or hybrid access protocols. Several protocols were proposed for each class, we described some of them. Each of these shows different characteristics, throughput efficiencies, fairness and overhead. Some of these protocols are used in wireless networking standards nowadays. Others are still research topics. Recent MAC protocol proposals support array antennas which would solve, among other problems, the exposed node problem etc. Supporting array antennas enhances throughput efficiency and power consumption as well, to the cost of protocol complexity. Recent distributed multiple access protocols also focus on QoS support. They will be detailed in Part II, Chapters 5 and 6.

Chapter 3

IEEE 802.11 Wireless LANs

Contents

3.1 Introduction

In 1997, the IEEE adopted the first wireless local area networks (WLAN) standard, IEEE 802.11-1997 which covers the medium access control (MAC) sub-layer and the physical layer (PHY) of the OSI (Open System Interconnection) reference model (Fig. 3.1). The architecture provides a level of indirection, transparent to higher level users: stations may move, roam through a WLAN and still appear as stationary to layers LLC (logical link control), e.g. IEEE 802.2, and above. This allows existing network protocols to run over IEEE 802.11 without any special consideration, just like if Ethernet (IEEE 802.3) were deployed. This fact, along with the reduced cards' cost, boosted IEEE 802.11's rapid deployment and success worldwide. In 1999 the IEEE adopted two PHY extensions to 802.11: 802.11a, an OFDM-based (orthogonal frequency division multiplexing) PHY and 802.11b, a high rate DSSS PHY. A detailed description of the IEEE 802.11 standard is available in [67, 6].

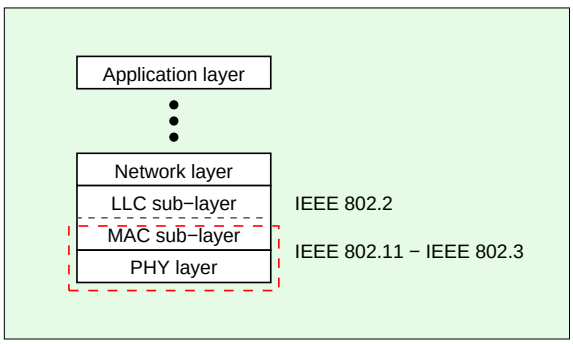


Figure 3.1: IEEE 802.11 position in the OSI stack model.

Section 3.2 of this chapter describes the working modes in IEEE 802.11. Section 3.4 describes the MAC sub-layer. Section 3.5 describes the standard's various physical layers. Section 3.6 describes power save mechanisms. Finally section 3.7 briefly shows security issues related to the standard.

3.2 Working modes

A group of *Wireless Terminals* (WTs) forms a *Basic Service Set* (BSS), as in Fig. 3.2, and the area it covers is called *Basic Service Area* (BSA). A BSS can either be an independent *ad-hoc* network (IBSS) or an *infrastructure network*, in which an *Access Point* (AP) links the WTs to a *Distribution System* (DS), therefore extending their range to other BSSs via other APs. The whole system is then called *Extended Service System* (ESS). The DS can be any kind of fixed or wireless LAN, unspecified in the standard.

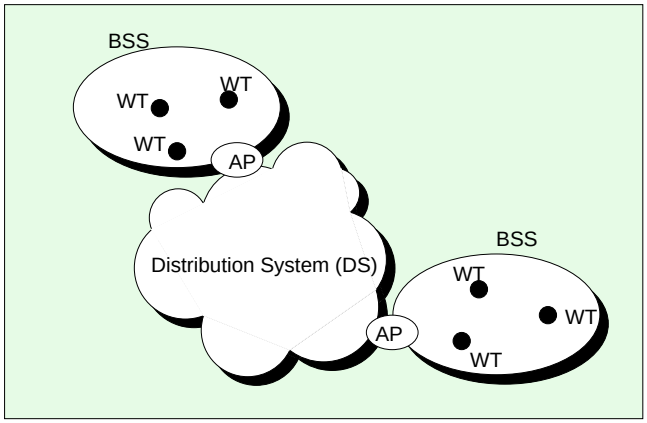


Figure 3.2: An Extended Service System (ESS).

In an IBSS, WTs typically communicate directly with one another. However, when two WTs are out of range of each other, multi-hop paths may be formed and relaying intermediate WTs forward frames from source to destination. This relaying function is built above the MAC sub-layer, therefore it is out of the scope of the standard.

A BSS includes an AP within WTs, giving those connection to the wired LAN, if any, and the local relay function for the BSS. All communications within a BSS go through the AP. Therefore, if two WTs in the same BSS want to communicate, frames are first sent to the AP then to the destination WT, wasting the double of the necessary data rate in direct communication. However, the benefit from this relaying approach is an efficient power saving mechanism, which provides the capability of buffering frames at the AP when the destination node is in dormant mode. See section 3.6 for details about power saving.

One of the major benefits of WLANs is the seamless mobility it provides. In a BSS, mobility is confined to the BSS range. An ESS expands mobility to several BSSs through the DS, which can be a wired or a wireless network. When a WT moves from one AP's range to another, frames will be forwarded by the old AP to the new destination allowing higher protocols to function normally even in presence of high mobility. This is done on the MAC sub-layer level and requires no support from the IP level, such as Mobile IP [68, 69, 70]. The communication between APs was not specified in the standard, and was left up to the vendors.

3.3 States and services

Figure 3.3 shows the different states in which a WT can be and the types of frames it is allowed to transmit in each state.

A station starts at state 1, where it is un-authenticated and un-associated. Only class 1 frames are allowed to be transmitted, i.e. authentication frames. Once the station is authenticated by the AP, it goes to state 2 where an additional class of frames is allowed, class 2, for association and re-association. When a station is both authenticated and associated, i.e. state 3, it is allowed to transmit class 3 frames also, which are data frames.

We should note that in ad-hoc (IBSS) mode, a station is allowed to transmit data frames when it is in state 1, neither authenticated nor associated.

3.4 The MAC sub-layer

The MAC supports the following functionalities:

- Providing reliable data delivery service.

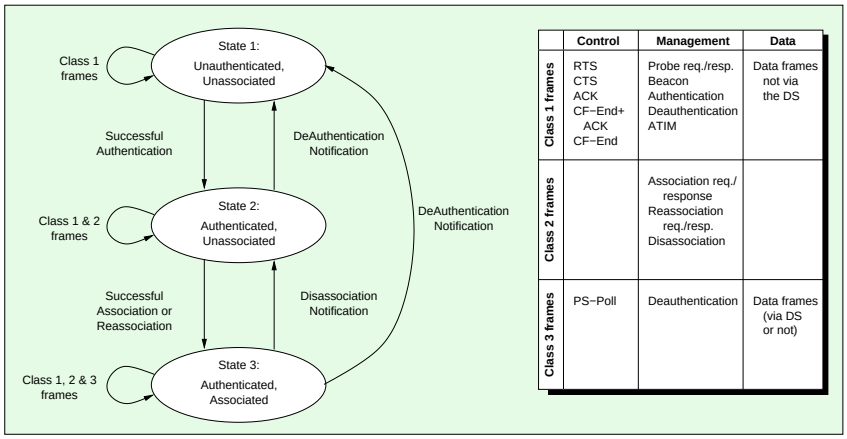


Figure 3.3: Relationship between state variables and services.

- Fairly control the access to the wireless channel. Two different access methods are supported: DCF (Distributed Coordination Function) and PCF (Point Coordination Function).
- Protect data against eavesdropping using encryption.

Those points are detailed in the following.

As seen in the previous chapter, data transmissions over the wireless channel are more exposed to errors than wired transmissions. To reduce this error rate, 802.11 adopts additional functionalities than 802.3. More specifically, when a WT transmits its frame, it cannot consider that the transmission succeeded, without collisions or noise interference. Therefore it should wait for an ACK from the receiving end, without which the sender assumes that the transmission failed and retransmits the frame again. Collision detection in 802.3 was possible due to the fact that the signal is weakly attenuated using cables, which permits the transmitting node to compare its transmitted signal to the one on the cable, and thus detect collisions. On a wireless channel, the transmitting WT cannot compare its signal to the one observed at the receiving node due to the big difference in powers. Therefore it cannot really decide if the frame is well received or not, and should wait for an ACK. An alternative to this is to leave the retransmission decision up to higher layers. However, those layers usually have longer timeouts to detect packet losses. This makes frame retransmissions on the MAC sub-layer more efficient, in spite of the additional overhead it introduces.

As mentioned before, 802.11 supports two services:

- *Distributed Coordination Function* (DCF): which supports delay insensitive data transmissions (e.g. email, ftp).
- *Point Coordination Function* (PCF): this service is optional. It supports delay sensitive transmissions (e.g. real-time audio/video) and can be used in combination with DCF.

In a BSS, WTs and the AP can either work in *contention mode* exclusively, using the DCF, or in *contention-free mode* using the PCF. In the first mode, WTs have to contend for use of the channel at each data frame transmission. In the second mode the medium usage is controlled by a *polling coordinator*, usually situated at the AP, polling the WTs to access the medium, thus eliminating the need for contentions. This last mode is not exclusive, and the medium can be alternated between *contention mode* and *contention-free mode* for CP (contention period) and CFP (contention-free period) respectively.

3.4.1 Distributed coordination function (DCF)

As mentioned earlier, the DCF is an asynchronous data transmission function, which best suits delay insensitive data. It is the only possible function in ad-hoc networks. When used in an infrastructure network, DCF can be either exclusive or combined with PCF. Each WT gets an equal share of the channel through contention, i.e. a WT contends for channel use before each frame waiting for transmission.

The basic scheme for DCF is *carrier sense multiple access* (CSMA)[16, 10] where the principle is to "listen before talking", so the WTs should sense the channel before trying to transmit their data. This protocol has two variants: Collision Detection (CSMA/CD) [71] and Collision Avoidance (CSMA/CA). A collision can be caused by two or more stations using the same channel at the same time after waiting for the channel to become idle. On the physical layer, in *spread spectrum* technology, a channel is the pseudo-random sequence used to

"spread" data. Collisions can also be caused by two or more hidden terminals transmitting simultaneously. Hidden terminals are terminals which cannot hear each other [10].

CSMA/CD is used in Ethernet (IEEE 802.3) wired networks. Whenever a node detects that the signal it is transmitting is different from the one on the channel, it aborts transmission, saving useless collision time. This mechanism is not possible in wireless communications because a WT cannot listen to the channel while it is transmitting, due to the big difference between transmitted and received power levels. To deal with this problem, the sender should wait for an acknowledgment (ACK) from the receiver after each frame transmission, as shown in Fig. 3.4. *Source* axis shows the data transmitted by the source. The destination replies with an ACK, shown on the *Destination* axis. The third axis shows the network status, as seen by *Other* WTs. Note that transmission delays are not shown. The *inter-frame spacings* DIFS and SIFS will be explained later in this section.

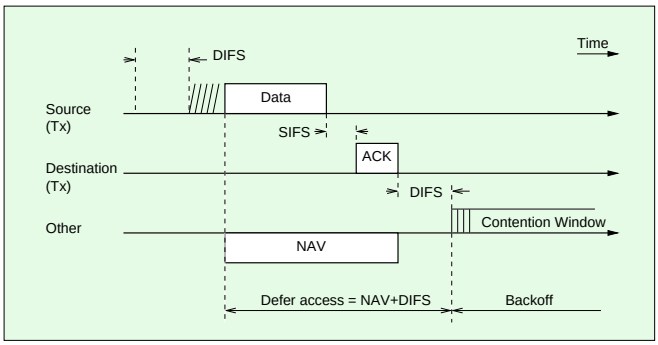


Figure 3.4: Basic access scheme.

If no ACK is returned, a collision must have occurred and the frame is retransmitted. This technique may waste a lot of time in case of long frames, keeping the transmission going on while collision is taking place (caused by a hidden terminal for example). Applying collision avoidance (MACA [20, 21]) minimizes this risk by using an optional RTS/CTS (Request To Send / Clear To Send) scheme in addition to the previous basic scheme, as shown in Fig. 3.5: a station sends an RTS before each frame transmission to reserve the channel. Note that a collision of RTS frames (20 octets) is less severe and less probable than a collision of data frames (up to 2346 octets). The destination replies with a CTS if it is ready to receive and the channel is then reserved for the frame duration. When the source receives the CTS, it starts transmitting its frame, being sure that the channel is reserved for itself during all the frame duration. All other WTs in the BSS update their *network allocation vector* (NAV) whenever they hear an RTS, a CTS or a data frame. NAV is used for *virtual carrier sensing*, as detailed in the next paragraphs.

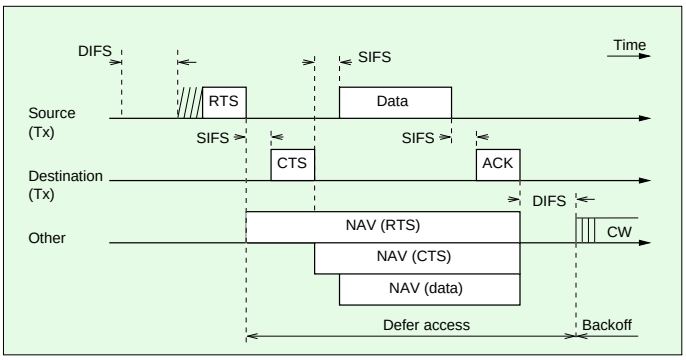


Figure 3.5: RTS/CTS access scheme.

The overhead of sending RTS/CTS frames becomes considerable when data frames sizes are small, and the channel is sub-optimally used. References [29, 72] discuss optimal data frame sizes (*RTS_Threshold*) above which it is recommended to use the RTS/CTS scheme. Very large frames may reduce transmission reliability too. e.g. an uncorrectable error in a large frame wastes more bandwidth and transmission time than an error in a shorter frame. So another optimization parameter is used, which is *fragmentation_threshold*, above which frames are fragmented as shown in Fig. 3.6.

Packet fragments are transmitted on the channel separated by SIFS, so no new packet can interrupt the

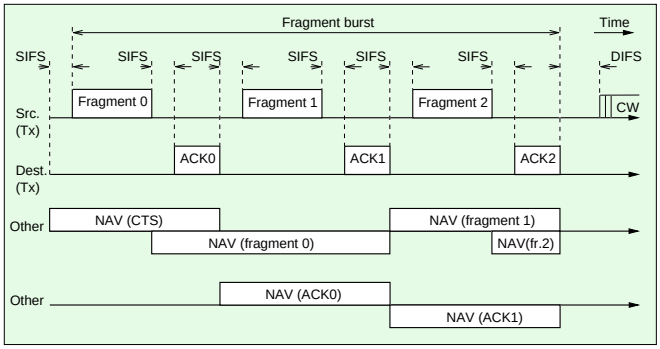


Figure 3.6: Packet fragmentation

current transmission. Each fragment is acknowledged separately, else, the fragment is retransmitted before any other fragments, keeping the sequence in order, and enhancing the throughput efficiency.

Not all frame types have the same priority. For example, ACK frames should have priority over RTS or data frames. This is done by assigning to each frame type a different *inter-frame spacing* (IFS), after the channel turns idle, during which no frames can be transmitted. In DCF, two IFSs are used: Short IFS (SIFS) and DCF IFS (DIFS), where SIFS is shorter than DIFS (Fig. 3.4 and 3.5). As a result, if an ACK (assigned with SIFS) and a new data frame (assigned with DIFS) are waiting simultaneously for the channel to become idle, the ACK will be transmitted before the new data frame (the first has to wait SIFS whereas the data has to wait DIFS). EIFS (extended inter-frame spacing), much larger than all other inter-frame spacings, is used instead of DIFS whenever the PHY layer reports to the MAC sub-layer that the current frame reception contains errors. This allows the MAC frame exchanges to complete correctly before another transmission is allowed.

Carrier sensing can be performed on both physical and MAC layers. On the physical layer, *physical carrier sensing* is done by sensing any channel activity caused by other sources. On the MAC sub-layer, *virtual carrier sensing* can be done by updating a local NAV with the value of other terminals' transmission duration. This duration is declared in data, RTS and CTS frames. Using the NAV, a WT's MAC knows when the current transmission ends. NAV is updated upon hearing an RTS from the sender and/or a CTS from the receiver, so the hidden node problem is avoided.

WTs avoid frame transmission right after the channel is sensed idle for DIFS time, so it does not collide with other "waiting" frames. Instead, a WT with a frame ready to be transmitted waits the channel to become idle for DIFS time, then it waits for an additional random time, *backoff* time, after which the frame is transmitted, as shown in Fig. 3.4 and 3.5. This is applied to data frames in the basic scheme, and on RTS frames in the RTS/CTS scheme. The backoff time of each WT is decreased as long as the channel is idle, during the so called *contention window*(CW). When the channel is busy, backoff time is frozen. When backoff time reaches zero, the WT transmits its frame. If the frame collides with another frame (or RTS), the WT times out waiting for the ACK (or the CTS) and computes a new random backoff time with a higher range to retransmit the frame with lower collision probability. This range increases exponentially as 2^{k+i} where i (initially equal to 1) is the transmission attempt number and k depends on the PHY layer type. Therefore, the backoff time equation is:

$$Backoff\_time = \lfloor 2^{k+i} \times rand() \rfloor \times Slot\_time \tag{3.1}$$

where $Slot\_time$ is function of physical layer parameters, and $rand()$ is a random function with a uniform distribution in [0,1]. There is a higher limit for i, above which the random range (CW_{max}) remains the same. The frame is dropped after a given number of retransmissions so a single frame won't monopolize the MAC. When a packet is successfully transmitted, the CW is reset to CW_{min}.

In all-hear scenarios, all WTs have equal probabilities to access the channel and thus share it equally. But this method has no guarantees for queuing delays, so it is not optimal for time-bounded applications. Time-bounded applications are better supported with the PCF.

3.4.2 Polling coordination function (PCF)

As mentioned before, PCF is based on polling the WTs to access the channel, therefore it is *contention-free*. Details about PCF can be found in [67, 6], nevertheless we briefly cite the main features of this function.

Figure 3.7 shows the PCF operational mode: The AP starts the contention-free period (CFP) periodically by transmitting a *beacon* frame, which updates the NAVs of the WTs with the maximum expected CFP time. Note that, as the AP has to wait for previous transmissions to resume before transmitting the *beacon* frame, the CFP repetition interval may not be constant, as noted in [73]. After sending the *beacon*, the AP starts polling

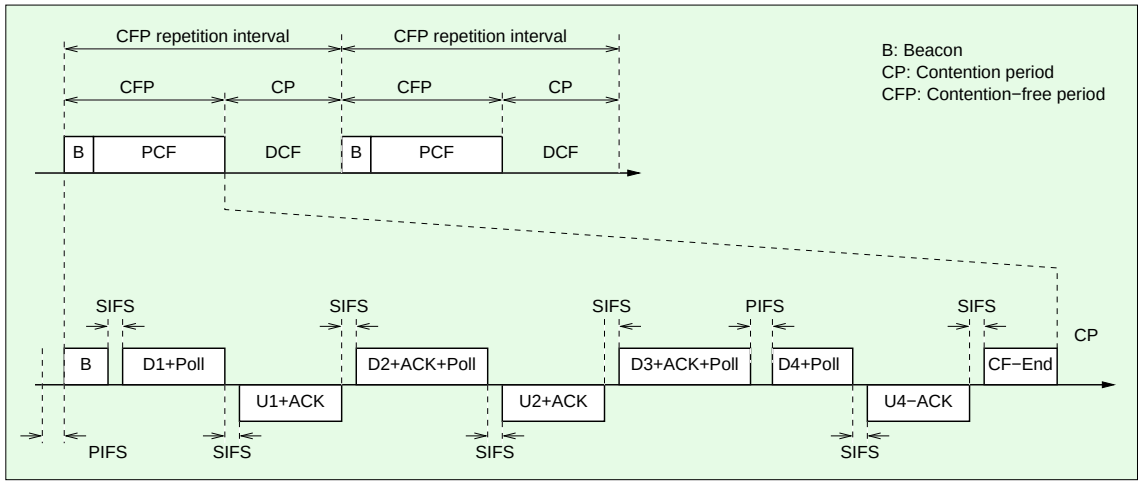


Figure 3.7: Polling coordination function (PCF).

the WTs for consecutive transmissions, according to a polling list. Polls and ACKs can be piggybacked to data frames so bandwidth is efficiently used. As with SIFS and DIFS, PCF gets priority over DCF by waiting the channel being idle for a polling IFS (PIFS) before it grabs the channel. PIFS is shorter than DIFS, giving the AP absolute priority to transmit before any of the WTs try to contend using DCF. This enables the AP to deliver near-isochronous service to the WTs on the polling list.

When the CFP ends, DCF mode starts being used by WTs randomly contending to access the medium each time they detect the medium idle for a period longer than DIFS, as described before.

When using the PCF, only WT-AP or AP-WT frame transmissions are possible. Therefore, a communication between two WTs in the same BSS have to go through the AP, wasting bandwidth. However, one of the advantages of this mechanism is that the AP provides a good *power saving* capability: the AP can store incoming frames in a buffer allowing the destination WT to stay in *sleep mode* during relatively long periods, in order to save battery power.

3.5 The PHY layer

IEEE 802.11 radios operate in the 2.4 GHz ISM band or in the 5.7 GHz U-NII band. In order to coexist with other radio standards operating in the same frequency bands, IEEE 802.11 uses spread spectrum techniques to spread the radiated power over the allowed frequency spectrum. Spread spectrum techniques are known to have the following properties [74, 75], originally designed for military purposes:

1. *Multiple access capability:* Different transmitters using orthogonal *codes* to transmit their data bits will be able to use the radio channel simultaneously, and the receivers will be able to distinguish their signal from others. The codes used to spread data bits must be sufficiently low cross-correlated, so that at de-spreading the receiver rebuilds the "good" signal with a high power while other signals remain spread with low power.

2. *Protection against multi-path interference:* As explained in section 1.3, radio waves propagate following multi paths reflected and refracted by several obstacles. The signals of different paths are all copies of the same transmitted signal but with different amplitudes, phases, delays and arrival angles. The receiver may take advantage of this fact and of spread spectrum modulation to reconstruct the signal using *rakes*. Each *finger* in a rake estimates a single path's parameters and tracks it. The efficiency of this reconstruction typically depends on the frequencies and the types of modulations used.

3. *Privacy:* The transmitted signal can be de-spread and the data recovered only if the code is known to the receiver.

4. *Interference rejection:* When we cross-correlate an interfering signal with the code, the interfering power is spread (divided by the code length) at the receiver, while the desired signal is de-spread to a higher power (Fig. 3.8).

5. *Anti narrow jamming capability:* This is almost the same as interference rejection, except that interference is intentionally introduced.

6. *Low probability of interception (LPI):* Because of its low power density, the spread spectrum signal is difficult to detect and intercept by a hostile listener.

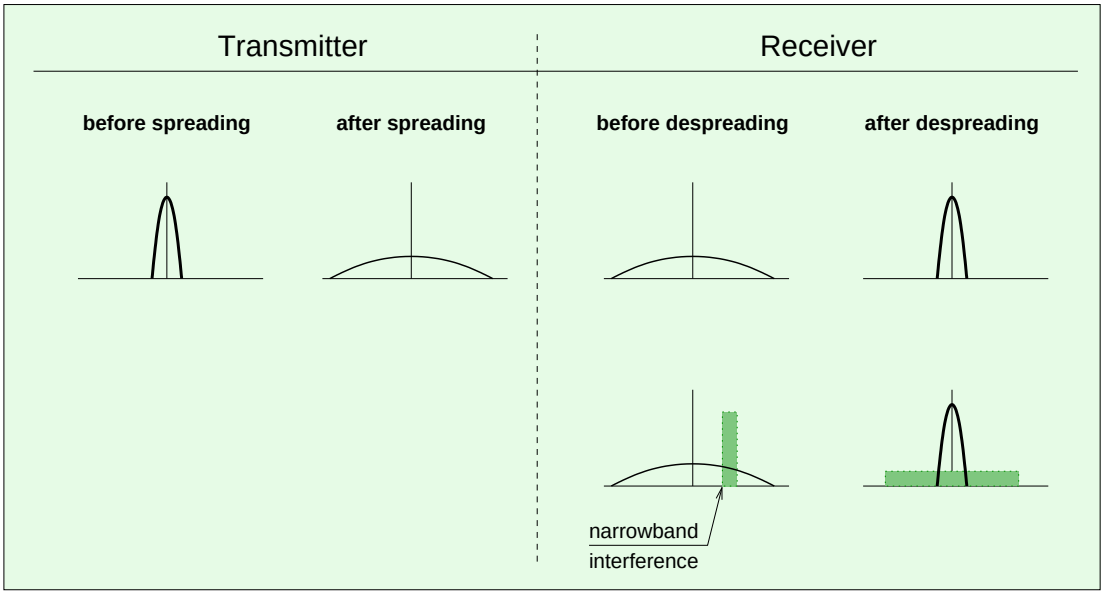


Figure 3.8: Interference rejection by spread spectrum techniques.

The standard defines known/fixed code sequences to be used by transmitters and receivers. Therefore the multiple access property do not really apply between transmitters using the same sequence of the standard. However, multiple access can still be considered between different transmitters using different codes of different standards, e.g. Bluetooth and IEEE 802.11 etc. Furthermore, privacy anti-jamming and LPI do not apply neither. The length and cross correlation of the code also has major influence on the rest of the properties cited above.

In the 1997 version of the standard, three physical layers were specified:

- Frequency Hop Spread Spectrum (FHSS) based: The spreading code defines the frequency at which data bits are to be transmitted. Sender and receiver should synchronously hop using the same frequency hop pattern (defined by equations and tables given in the standard) in order to communicate.
- Direct Sequence Spread Spectrum (DSSS) based: Instead of sending raw data bits, DSSS correlates data with the code *chips* running at higher rate (Fig. 3.9). The code used is an 11-chip known sequence called *Barker code*. The resulting high rate data stream is modulated and transmitted in the air. At the receiver side, the reverse procedure is applied to retrieve the original data.

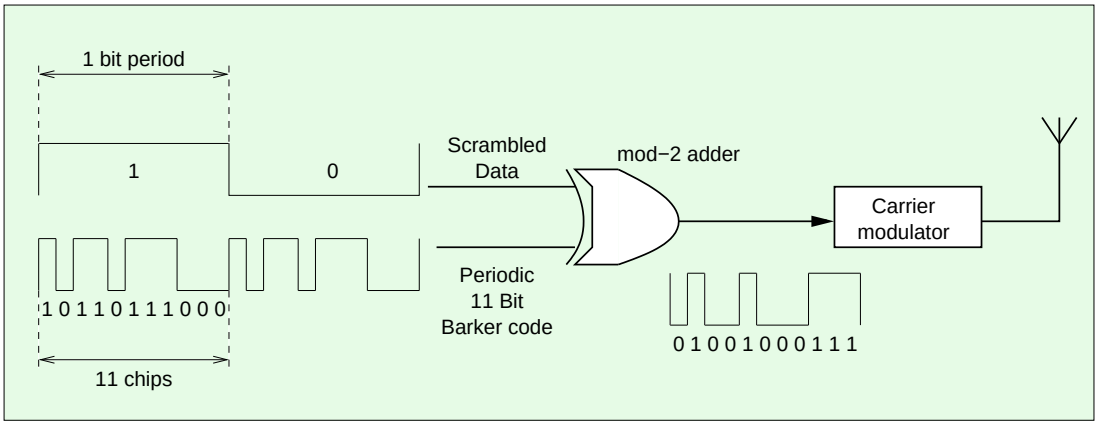


Figure 3.9: DSSS transmitter principle.

- Infra Red (IR) based: Data bits are modulated using *pulse position modulation* where the IR-light pulse position in time slot defines the data bit value.

All three PHY specifications describe operations at 1 and 2 (raw) Mbps. Low data bit rates use relatively robust modulation schemes. As packet headers are more crucial than packet payloads, the formers are modulated and transmitted at low bit rates in order to resist to channel errors. Packet payloads can be transmitted at different data rates, specified in the packet header, therefore they are more vulnerable to channel errors if high bit rates are used.

In 1999, the IEEE standardized two PHY extensions:

- IEEE 802.11a, using Orthogonal Frequency Division Multiplexing (OFDM) in the 5 GHz unlicensed U-NII band. IEEE 802.11a can offer up to 54 Mbps (raw) data rate. OFDM is known to have good properties in indoor radio propagation. The basic principle of OFDM is to divide the data bit stream into several sub-streams. Those data bit sub-streams modulate orthogonal sub-carriers (48 data sub-carriers and 4 pilot sub-carriers) which are combined using inverse fast Fourier transform (IFFT) and transmitted in the air. On the receiver side, the inverse procedure is applied using FFT to separate different sub-carriers and retrieve the original data sub-streams. The specification is similar to that of HiperLAN-2 [7] introduced in the next chapter.

- IEEE 802.11b, which uses high rate DSSS in the 2.4 GHz band, offering up to 11 Mbps (raw) data rates. The high rate DSSS is due to using an enhanced modulation technique, CCK (complementary code keying) while still using low data rate modulations for backward compatibility.

the MAC sub-layer is common for all of the underlying PHY specifications: FHSS, DSSS, IR, 802.11a or 802.11b. However, the MAC parameters' values may change from one PHY specification to another.

3.6 Power save mode

When a mobile station turns off its receiver and its transmitter to save power, it is said to be in *low power* mode. Because of the significant difference between BSSs and IBSSs, two power management mechanisms were specified.

3.6.1 Power management in infrastructure BSS

Power management in an infrastructure BSS is centralized in the AP. This mechanism allows greater power saving than with independent BSS due to the capability of packet buffering in the AP, allowing WTs to stay in low power mode for longer periods.

During association, a MH tells the AP about its low power durations in terms of *beacon* periods. After each low power period the MH awakes and learns if there are any packets waiting at the AP. The MH must also awaken at times determined by the AP for multicast frames to be delivered.

The AP buffers all frames destined to MHs in low power mode which are associated to it. The frames remain at the AP for at least the number of *beacon* periods specified by the WT during association. The AP indicates the presence of buffered frames to the WTs in each beacon, so a WT can ask the AP to deliver its buffered frames. The AP indicates that more frames are to be transmitted by setting the proper *more data* bit in the frame header, until the buffer becomes empty.

3.6.2 Power management in independent BSS

Power management in an IBSS is fully distributed. Before going into low power mode, a station must complete a data frame handshake with any other station, announcing its low power mode state. During this handshake the station must remain in the awake state. In low power mode, the station has to wake up to receive each *beacon* and to stay awaken during a *traffic indication message window* period. During this period, any other station attempting to send frames to the power saving station must announce those frames during this window period so the receiving station stays awake until the next *beacon* transmission.

A station desiring to transmit a frame to another must estimate the power saving state of the destination based on the last data frames received from it. If the destination is in low power mode, the source has to wait for an acknowledgment to its announcement sent during the *traffic indication message window* before it transmits the actual data frame. An exception to this rule are multicast frames whose announcement is not acknowledged before their actual transmission.

This mechanism requires a minimum awaken duty cycle of the senders and the receivers, therefore the power gain that can be achieved in an BSS is limited.

3.7 Security issues

To deal with the open nature of the wireless channel, IEEE 802.11 uses encryption on the MAC sublayer to protect data transmitted in the air. WEP (Wired Equivalent Privacy) was first designed to support a protection level comparable to that of wired networks, at the time of conception. Later on, the encryption algorithm used, RC4 [76, 77], showed to be weak and several attacks that exploit this weakness are recently known [76, 77]. RC4 is a symmetric stream cipher that supports a variable data length (not block cipher) and (variable) key lengths up to 256 bytes. IEEE 802.11 chose 40-bit and 128-bit key lengths.

WEP principle is shown in Fig. 3.10. Data is concatenated with its integrity check value which is the output of an integrity algorithm (CRC-32) applied to these data. Integrity check is used to combat data modification on the wireless channel.

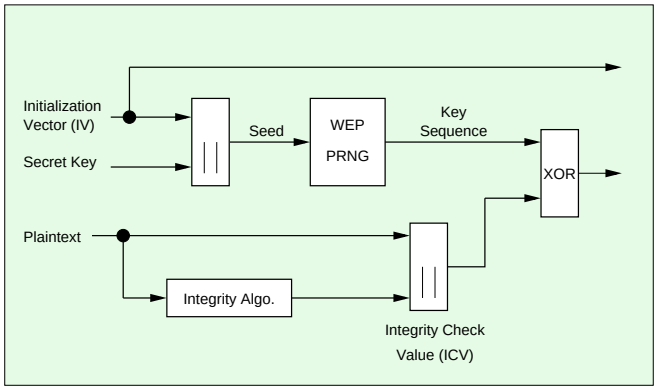


Figure 3.10: Wired Equivalent Privacy (WEP) block diagram.

The resulting data stream is encrypted (XORed) with a key sequence generated by a Pseudo-Random Number Generator (PRNG), using RC4. This key sequence must be changed regularly to avoid data analysis and key breaking. Therefore, the seed of the PRNG is a concatenation of the user secret key, and of an initialization vector (IV). The IV can be changed for each packet for more robustness, and is sent to the destination in the frame header in clear-text.

The secret key can either be selected in a shared list or a negotiated key. The shared list (up to four entries) must be known to all the stations in a BSS or an ESS. The index of the key used for encryption is also sent to the destination, in clear-text, in the frame header. the main drawback of pre-shared lists is that they are known to all stations in the BSS, making keys more vulnerable to be revealed.

An alternative to pre-shared lists keys is negotiated keys. Key negotiation ensures that the keys are only known to station pairs, therefore they have less risks to be revealed. However, no specific negotiation algorithms, such as Diffie-Hellman, were specified in the standard.

When encryption is used, the sender sets the encryption bit in the MAC frame header. At the receiver side, the reverse process is applied: The destination uses the encryption key with the IV to decrypt the contents of the frame, it then applies the integrity check algorithm to ensure that no modification occurred to data.

Last, we should note that encryption is applied to data payload only, leaving the data frame header clear to eavesdroppers.

Chapter 4

HiperLAN-2 and Bluetooth

Contents

In this chapter we describe two more wireless networking standards, HiperLAN-2 and Bluetooth. HiperLAN was designed by ETSI for local area networks. However, Bluetooth was designed for personal area networks, which is not exactly the same application area as for IEEE 802.11 and HiplerLAN. All the three may coexist geographically. HiperLAN-2 may interfere with IEEE 802.11a as they use the same frequency band (5.7 GHz) and the same modulation techniques. Bluetooth may interfere with IEEE 802.11b. They have different data rates, different characteristics and they support different services. In section 4.1 we will describe HiperLAN-2. Section 4.2 describes Bluetooth and section 4.3 compares the three standards described so far.

4.1 HiperLAN-2

HiperLAN-2 [7, 78] is the european standard, developed by ETSI, for wireless LANs. This alternative supports QoS, uses enhanced security algorithms and better radio management in addition to all the features seen in IEEE 802.11.

A HiperLAN network typically has a topology similar to that of *infrastructure mode* described for IEEE 802.11. All mobile terminals (MTs) communicate with the network's AP. The main features of HiperLAN-2 are the following (features not common to IEEE-802.11 are preceded by a star):

- *High-speed transmission:* HiperLAN-2 uses OFDM which is very efficient in time-dispersive environments, e.g. inside buildings where the transmitted radio signal is reflected from many points, leading to different propagation delays before reflections reach the receiver. Data rates are up to 54Mbps. The MAC layer is a form of dynamic time-division duplex, detailed later in this section.
- *(*)Connection-oriented:* HiperLAN-2 uses signaling functions to establish connections between an MT and the AP. Data is transmitted over these connections which can be either point-to-point or point-to-multipoint. Packet broadcast is also possible.
- *(*)QoS support:* As HiperLAN-2 is connection oriented, it is straightforward to implement QoS support. Each connection can be assigned different QoS parameters such as data rate, delay jitter etc. or to apply relative differentiated services. This ensures some flow isolation when different types of flows are being transmitted simultaneously.

- *(*)Dynamic frequency selection (DFS):* Unlike current cellular networks, HiperLAN-2 uses automatic frequency selection/planning. The AP listens to neighboring APs and radio sources and selects its radio channel accordingly, trying to reduce interference.
- *Security support:* HiperLAN-2 supports authentication and data encryption. Both the AP and the MT can authenticate each other to ensure authorized access to the network and to ensure access to a "legal" network respectively. The encryption algorithm used by HiperLAN-2 is DES and 3-DES, which is known to be robust.
- *Network and application independent:* Like IEEE 802.11, HiperLAN-2 is designed to provide a transparent wireless network solution to higher layers. It is also designed to be flexible with easy adaptation and integration with a variety of fixed networks. It may be used to simply replace (wired) Ethernet or to be used as access network to third generation cellular networks.
- *Power save mode:* The centrally-controlled nature of HiperLAN-2 wireless networks makes power saving more efficient. An MT may request the AP to enter in low power mode at any time, and it specifies the sleep period. At the end of each period the MT wakes up and checks for wake up indication from the AP. In case of packets buffered at the AP, the MT keeps awaken to retrieve them. Else, it goes to sleep mode again. Different sleep periods are supported to allow for either short latency requirement or low power requirement.

4.1.1 Hiperlan-2 layer stack

Figure 4.1 shows the HiperLAN protocol stack. It comprises two planes, depicted from the ISDN functional partitioning:

- The control plane includes functions for connection control, establishment, release and supervision.
- The user plane includes functions for transmission of data over established connections.

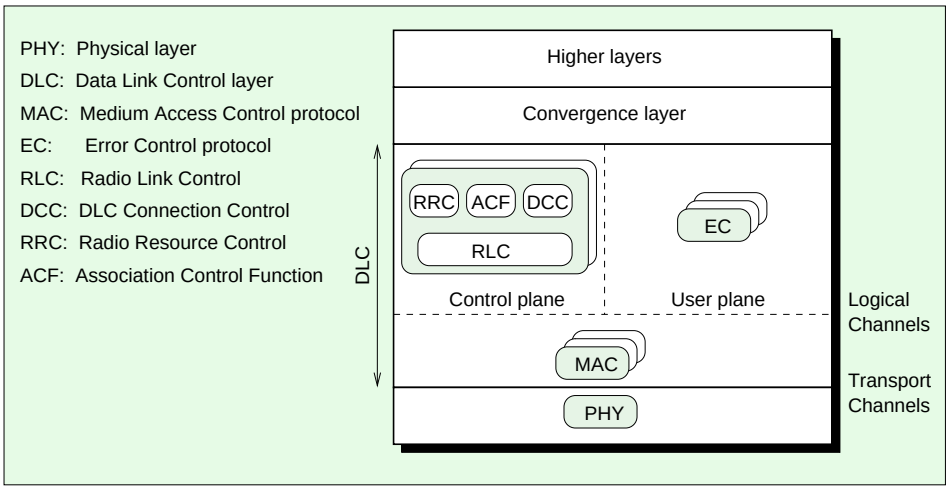


Figure 4.1: HiperLAN-2 stack

HiperLAN-2 has three basic layers: The physical layer (PHY), data link control layer (DLC) and the convergence layer (CL). We will describe each of these layers in the following subsections.

4.1.2 The physical layer

As mentioned in the previous chapter, HiperLAN-2 and IEEE 802.11a have similar physical layer specifications, based on OFDM (originally used for ADSL, called DMT, discrete multitone). HiperLAN-2 operates in the 5.7 GHz unlicensed frequency band with 20 MHz channel spacing which results in 19 separate channels (in Europe) along the allocated band. Each channel is divided into 52 sub-carriers: 48 data sub-carriers and 4 pilot sub-carriers, used as a reference for demodulation.

The high-rate data stream is divided into several low-rate data streams which modulate one of the 48 sub-carriers as shown in Fig. 4.2.

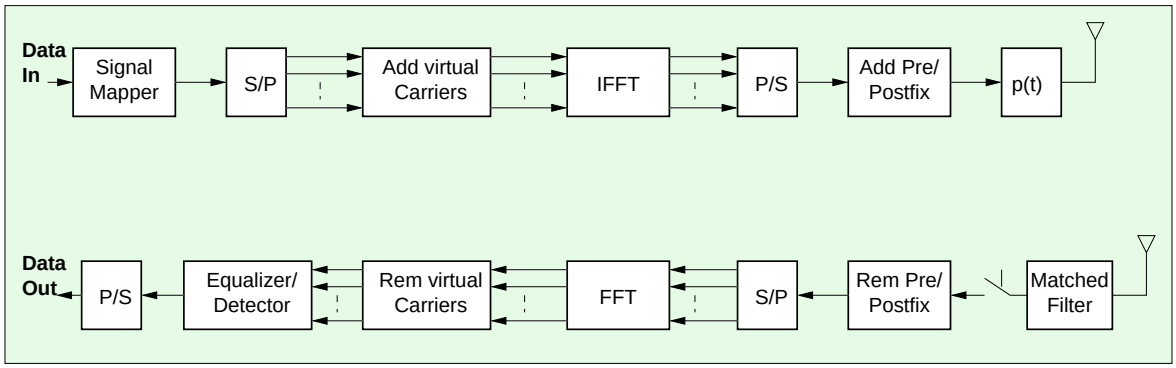


Figure 4.2: HiperLAN-2 modulator and demodulator

The hardware is simplified by the use of IFFT and FFT at the modulator and demodulator respectively. Possible modulation schemes are: BPSK, QPSK and 16-QAM, each with different coding rate shows different robustness against channel errors. The main advantage of OFDM is that it reduces inter-symbol interference caused by multipath propagation.

HiperLAN-2 supports multi-beam antennas to improve the carrier to interference ratio in the radio network. As we will see in the following, multi-beam is supported on the DLC layer, allowing up to seven beams to be used.

4.1.3 The data link control layer (DLC)

The DLC constitutes the logical link between an AP and the MTs. It has functions for both the user plane and the control plane. The corresponding sublayers are described here.

The MAC protocol in HiperLAN-2 is centralized at the AP which informs MTs at what time they are allowed to transmit their data. Each MT requests for resources and the AP adapts time division accordingly. Time is shared among MTs following a TDMA, while AP-MT use TDD as follows:

The basic MAC frame structure (Fig. 4.3) has a duration of 2ms. It comprises a downlink (DL) phase (AP-MTs), an uplink (UL phase) during which data transmission is contention-free, specified using the frame control channel (FCH). Contention among MTs is allowed on the random access channel (RCH) only, over which they send requests for resources to the AP for the coming MAC frame (UL or DL). The AP dynamically adapts the durations of the UL and DL phases accordingly and sends the information about the current frame structure on the frame control channel (FCH).

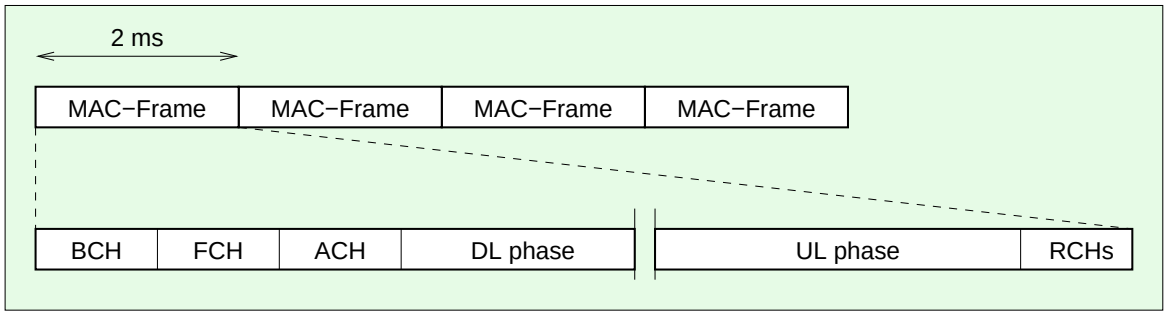


Figure 4.3: HiperLAN-2 MAC frame structure

Contention on the random access channel (RCH) to send resource requests may not be successful in case of collisions. The feedback information is sent using the access feedback channel (ACH) to inform MTs about previous access attempts using RCH. Access to the RCH shall be adapted to the feedback information, controlled by a contention window (CW_a) maintained by each MT. As in IEEE 802.11, in order to maintain stability, the CW value depends on the number of transmission attempts a as follows:

- Initial attempt: $a = 0$, $CW_0 = n$

- Retransmission: $a \geq 1$, $CW_a = \begin{cases} 256 & 2^a \geq 256 \\ 2^a & n < 2^a \leq 256 \\ n & n \geq 2^a \end{cases}$

where n is the number of RCHs in the MAC frame. n may vary from frame to frame. An MT sends its resource request on RCH number r_a where r_a is uniformly chosen in $[1,CW_a]$ which may span over several MAC frames.

The length n of the RCH is advertised to all MTs using the broadcast channel (BCH) at the start of each MAC frame. Other information are also send on the BCH intended for all MTs, such as transmission power level, starting and ending of FCH (and RCH), AP identifiers, the network identifier etc.

During DL and UL phases the traffic consists of either control packet data units (C-PDU) or user packet data units (U-PDU). The former is referenced as short transport channel (SCH) while the latter is referenced as long transport channels (LCH).

SCH, LCH and RCH are called transport channels. They are used by the *logical channels* described hereafter according to a mapping scheme between logical and transport channels:

- *Slow broadcast channel (SBCH)*: Used for broadcast control information concerning the whole radio cell, handover acknowledgments, seeds for encryption etc. SBCH is used once per MAC frame and per antenna element (shown as multiple MAC instances in Fig. 4.1), and can be accessed by all MTs. Obviously, it is not used on the uplink. On the downlink, SBCH uses SCH and LCH transport channels.

- *Dedicated control channel (DCCH)*: It carries radio link control (RLC) information from the AP to specific MTs. Association control and connection control messages are carried within DCCH, which is bidirectional. DCCH uses SCH and LCH on the downlink. It uses SCH, LCH and RCH on the uplink.

- *User data channel (UDCH)*: Conveys user data between the AP and a MT bi-directionally. To establish a connection the DLC uses signaling over the DCCH, then data is sent over UDCH, in sequence. UDCH uses LCH only, in downlinks and in uplinks.

- *Link control channel (LCCH)*: Conveys information between the error control (EC) functions in the AP and the MT for a given UDCH. LCCH is bidirectional and uses SCH in the downlink, SCH and RCH in the uplink.

- *Association control channel (ASCH)*: Carries new association and re-association requests. ASCH is uplink only, it uses RCH.

DLC connections can be unicast, multicast or broadcast and have unique identifiers. MTs send resource requests to the AP, asking for resources to transmit a number of PDUs they specify, so the AP can control the priorities and delays of each MT.

Error control (EC) in HiperLAN-2 is based on selective-repeat ARQ to ensure reliable transmissions on the wireless channel. It also ensures that PDU are delivered in-order to higher layers. For delay sensitive flows, such as voice, PDUs may be discarded after a configurable playback threshold to ensure short packet delays.

On the DLC control plane we find the following functions:

- *Association control function (ACF)*, which controls association, depending on the AP with the best signal received by the MT. An MT then requests an ID from the AP. This is followed by an exchange of link capabilities using the ASCH. HiperLAN-2 supports two authentication mechanisms: pre-shared key and public key (using PKI). Authentication algorithms supported are MD5, HMAC and RSA. If encryption has been negotiated, the MT and the AP will start the Diffie-Hellman key exchange, then data is encrypted using DES or 3-DES encryption algorithms. Disassociation can be either explicit, requested by the MT, or implicit, i.e. when the MT is unreachable for a given period of time.

- *DLC user connection control (DCC)*, used to establish a connection between a MT and the AP. The connection request carries the connection characteristics. If the connection can be established, the AP acknowledges the MT's request.

- *Radio resource control (RRC)*, which controls handovers, dynamic frequency selection, *MT-alive* notifications and the power save mode. An MT measures the signal quality and decides whether to request a handover, which can be of two types: re-association or using the fixed network support. DFS enables the AP to instruct a MT to perform measurements on radio signals received from neighboring APs so the frequency used can be changed for less interference.

4.1.4 The convergence layer

The aim of the convergence layer is to provide transparent wireless support to higher layers/applications. This necessitates two functions: On the control plane, the convergence layer has to translate service requests from higher layer to the DLC control plane. On the user plane, the CL has to adapt packet formats and sizes of higher layers to the DLC user plane.

Two types of convergence layer were defined: Cell-based and packet-based. The former supports ATM networks, whereas the latter supports a variety of packet-based networks, such as Ethernet and IEEE 802.1p for QoS support.

4.2 The Bluetooth technology

In 1994 Ericsson Mobile Communications started looking for alternatives to replace cables connecting mobile phones with accessories, using radio links to avoid IR line-of-sight connection constraints. Requirements included handling data and speech which enables mobile phones to connect to headsets and computer devices. Later on the requirements were developed to include service discovery protocols and applications/profiles, and "Bluetooth" became more than just a cable replacement.

The Bluetooth wireless technology [79, 80, 8] is designed as a short-range connectivity solution for personal, portable and hand-held electronic devices (usually called PAN, personal area network) operating in the 2.4 GHz ISM band. On the other side of the coverage spectrum, IEEE 802.11 connects computer devices to infrastructure networks such as campus LANs or ISP networks, or to other computer devices to form an ad-hoc infrastructureless network. Therefore the application areas of these two standards do not really overlap. However, they may co-exist in the same geographical areas, causing radio interference with each other.

The personal connectivity space looks like a communication bubble, moving and connecting the person inside with all surrounding devices that enter the bubble, each of these with a service to offer. Bluetooth devices are designed to be low-cost, small-size and a user-friendly replacement for interconnection cables.

Handling data enables Bluetooth devices to connect to infrastructure LANs also (via a mobile phone for instance), rising up the concept of "personal gateways" connecting all devices on a person's body to remote services.

Several other manufacturers joined Ericsson to form the Bluetooth Special Interest Group. In 1999, version 1.0 of the Bluetooth specifications saw the light. It kept the temporary name it had, that of king Harald Blåtand, the danish king who united Denmark and Norway in the tenth-century, just like Bluetooth is expected to unify telecommunications and computing industries.

Bluetooth was also chosen to be a baseline of the IEEE 802.15.1 (WPAN, Wireless PAN) standard in July 1999. The IEEE 802.15.2 task group studies coexistence issues between 802 wireless technologies. 802.15.3 task group is developing standards for high-rate radios ($>$ 20Mb/sec), and 802.15.4 is developing standards for low-rate ($<$ 200Kb/sec).

Unlike IEEE 802.11 and HiperLAN, the Bluetooth protocol stack covers all the layers of the ISO reference model. Fig. 4.4 shows the Bluetooth protocol stack along with the profiles layer. Layers belong to three groups: Transport protocols, middleware protocols and the applications/profiles. Transport protocols were developed exclusively for Bluetooth, however some of the middleware protocols were adopted.

4.2.1 Transport protocols

The Bluetooth radio operates in the license-free ISM 2.4 GHz frequency band. It uses fast FHSS (1600 hops/sec) to spread the signal over 79 one-MHz channels in a pseudo-random hopping pattern. The center frequency is defined by $f_c = 2,402 + k$ where $k = 0, ..., 78$. Data is modulated and transmitted on frequencies around f_c using Gaussian frequency shift keying (GFSK), where data bits determines the frequency shift to the upper or the lower side of center frequency f_c. This frequency shift is done smoothly, using a Gaussian distribution, which eliminates the undesirable effects of sudden frequency shifts of simple FSK. The baud rate is 1 Msymbols/sec, raw transmission rate is 1 Mb/s. Bluetooth devices can use one of 3 power classes, therefore covering different area sizes: 20, 4 and 0 dBm (Classes 1, 2 and 3 resp.).

The *baseband* enables a Bluetooth device to communicate with others using defined functions, creating links, *piconets* and *scatternets*. The baseband also controls access to the medium and formats low-level packets.

Each Bluetooth device has a 48-bit unique hardware address, *BD_ADDR*. This address is used to establish communications between devices: A *master* device can communicate with up to seven active *slave* devices to form a *piconet* (Fig. 4.5) with no need for any infrastructure support. Any device can be master (usually the initiator of the piconet) or slave, depending on the piconet creation.

Piconets are formed in two phases:

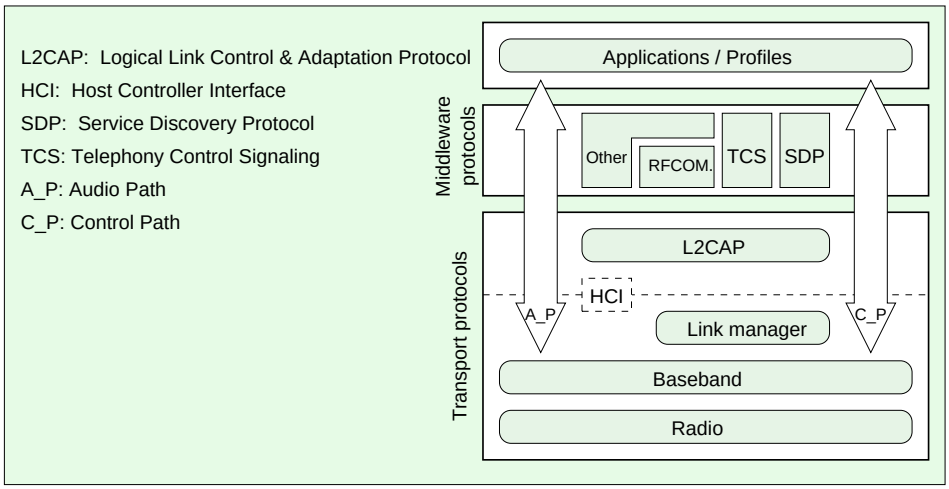


Figure 4.4: The Bluetooth layer stack

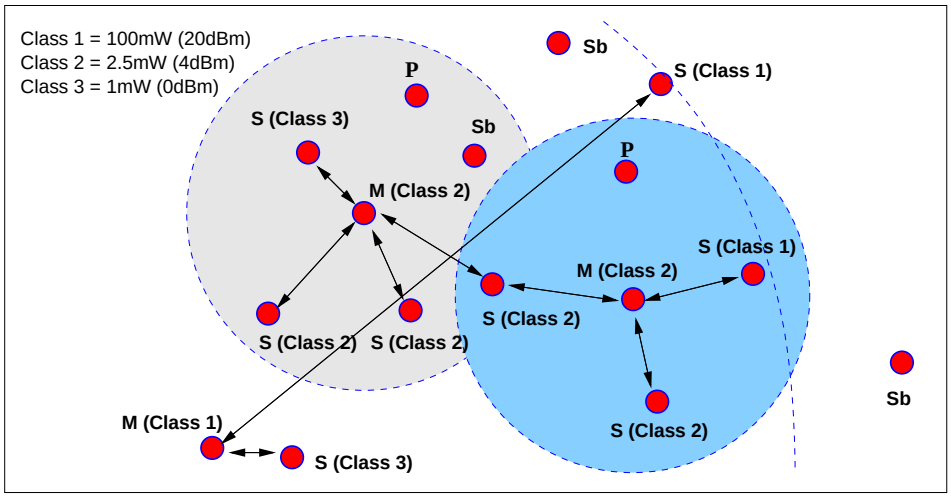


Figure 4.5: Bluetooth piconets

- *Inquiry phase*: During this phase a device (future master of the piconet) gathers information about neighboring devices by transmitting inquiry messages. Other devices may be scanning for inquiries and detect them. They may respond to inquiry messages by response messages that contain, among other information, the device's address. The master builds a list of neighboring devices, to be used during the paging phase.
- *Paging phase*: A master pages the slaves it wants to join its piconet.

After the paging phase, master and slave may swap their roles. Slaves can be in one of the following states:

- *Active:* Those are the Bluetooth devices, up to seven in a piconet, participating in active communications. Each has a temporary address assigned to it by the master of the piconet. The master is the piconet coordinator which polls each slave to transmit.
- *Parked:* These are additional non-active Bluetooth devices that may be registered with the master and invited to become active later on.
- *Stand-by:* These are devices not associated with any piconet.

All devices in a piconet use the same frequency hopping sequence synchronously to communicate with the master. The hopping sequence is defined using the address of the master of the piconet and the offset between the two clocks. Transmit and receive times are slotted, each transmission can be single-slot or multi-slot (3 or 5

slots). During multi-slot packet transmission frequency doesn't hop and at the end of which frequency resumes as if no frequency blocking occurred. Master and slave share the medium using time-division duplex (TDD): Masters use even numbered time-slots and slaves use odd ones.

Piconets can co-exist in time and space independently, as in Fig. 4.5. They may also share common devices to form a *scatternet*. In this case, the common device must be slave in a piconet and master in the other to be able to communicate with both without synchronization problems.

Two link types between master and slave are supported:

- *Synchronous connection-oriented (SCO)*: Up to three SCO links may be used in a piconet. SCO links are convenient for audio transmissions, at 64Kb/s in each direction (master-slave). As audio packets are delay sensitive they are not retransmitted if transmission errors occur. However, FEC (Forward error correction) can be used to recover erroneous audio packets.
- *Asynchronous connectionless (ACL)*: Between a master and a slave only one ACL link may be used. Convenient for (asynchronous) data packet exchange, ACL links retransmit erroneous packets, and optionally use FEC to recover them.

The properties of these links are setup by the *link manager* protocol (LMP). At this level, devices are authenticated and links are optionally encrypted. Furthermore, this protocol learns about the other device's power save mode, if it supports SCO links and what packet sizes are supported. The LMP also establishes the SCO connections and configures polling time intervals. These LMP transactions use ACL links.

While encryption is done on the baseband level, the key exchange is done by the LMP. Both ACL and SCO links may be encrypted using 128-bit long keys. Keys are generated using SAFER+ algorithm [81]. SAFER+ is also used for authentication based on challenge/response and shared keys. Another approach would be using PKI which is not possible in ad-hoc infrastructureless networks like Bluetooth. Whether for authentication or for link encryption, SAFER+ uses the user PIN to generate all subsequent keys. This makes the whole system security depend on the length and randomness of this user-provided string. The challenge/response approach used for authentication is shown in Fig. 4.6.

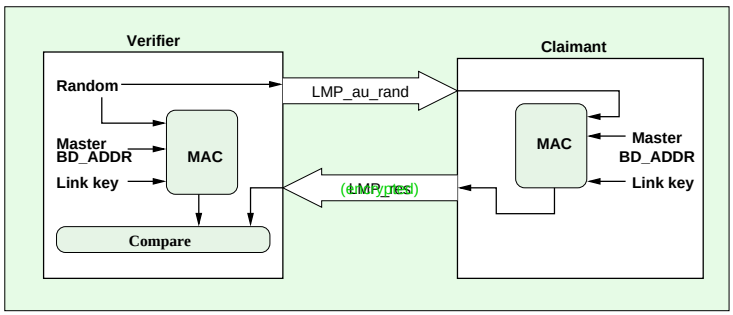


Figure 4.6: The challenge/response approach used for authentication

The *verifier* generates a random number and sends it in clear-text to the *claimant*. This last uses the random number, the master's address and the link key (generated by SAFER+ using the user PIN) to compute its encrypted response. The verifier receives the response and compares it with the same algorithm result on its side. If they match, then both sides have the same (secret) key, unrevealed to eavesdroppers.

The LMP configures the power mode to be in one of the three states:

- *Sniff* mode: The slave listens to the master periodically. The period is configured by the LMP.
- *Hold* mode: The device agrees with its communicating partner to remain silent for a given amount of time.
- *Park* mode: The slave agrees with its master to park until further notice, but still listens to beacon transmissions from the master which may also wake it up using these beacons.

The HCI (Host controller interface) is an interface for host devices to access lower layers of the Bluetooth stack through a standardized interface. Through the HCI host devices can do what the LMP does for higher layers: pass data, link configuration commands, power mode configuration, authentication commands etc.

The L2CAP (Logical link Control and adaptation protocol) provides several logical links to the middleware protocols. It multiplexes several logical channels (connectionless or connection oriented) over the device's ACL links (one per slave), and identifies each logical channel with a two-octet unique channel identifier. L2CAP

also provides packet fragmentation at the source and defragmentation at the destination device, so that large packets of higher layers (up to 65 Koctets) can be passed through the baseband (2744-octet packets maximum). Furthermore, QoS can also be supported and negotiated on this level. However only best effort traffic is currently supported.

4.2.2 Middleware protocols

Not all the middleware protocols are involved in each communication, as it is the case for the transport protocols. Four middleware group of protocols are supported by Bluetooth devices: Service discovery protocol (SDP), RFCOMM protocol, telephony control signaling (TCS) protocol and *other* protocols.

Using SDP, a Bluetooth device can inquire what services are available in neighboring devices and learn how to access them. The SDP provides information about available services and the ways to access them. SDP does not contain the services themselves, neither is their access protocol. The service information at the SDP is encoded using universally unique short service identifiers so bandwidth is used in an efficient way.

The RFCOMM protocol provides a serial communication interface (RS-232-like) over the packet-based transport layers. It also allows the multiplexing of several serial ports over a single transport link, according to the ETSI 07.10 standard. Several legacy applications using serial ports can be used over RFCOMM protocol with no modifications required.

The TCS protocol uses the same set of telephone control commands as modems, the AT command set, to send and receive control signaling over the RFCOMM protocol. Therefore an application can instruct a mobile phone, both equipped with Bluetooth devices, to dial a given phone number using the TCS protocol. TCS can also operate over L2CAP using another set of commands which, unlike the AT command set, supports point-to-multipoint communications, such for a cellular phone which can be used as a cordless phone and is further able to establish direct communications with other cellular phones.

Other protocols were also adopted to support point-to-point (PPP) communications enabling IP over serial lines, OBEX (Object exchange) and IrMC protocols. All these run over the RFCOMM protocol.

4.2.3 Bluetooth profiles

The Bluetooth specifications comprises two parts:

- *The core specification* that defines the radio characteristics and communication protocols, as detailed in the previous subsection.
- *The profile specification:* The application area of Bluetooth is very wide. To ensure interoperability among different implementations, profiles define how Bluetooth protocols have to be used to realize given applications.

The notion of profiles originated from the ISO (ISO/IEC TR10000). Profiles are like vertical slices through the protocol stack, as shown in Fig. 4.7. They provide a set of higher layer procedures and uniform ways of using the lower layers. This reduces implementation options, defines user interface guidelines etc. which makes ad-hoc networking from different manufacturers more functional and increasing market acceptance of new devices.

Each Bluetooth device supports one or more profiles, from the following list:

- *The generic access profile*, is the most basic Bluetooth profile; all other profiles are built upon it and use its facilities. It facilitates establishing baseband links, discovering other Bluetooth devices and defines procedures related to security.
- *The serial port profile*
- *Dial up networking*
- *FAX profile*
- *Headset profile*
- *LAN access profile*
- *Generic object exchange profile*
- *Object push profile*
- *File transfer profile*
- *Synchronization profile*

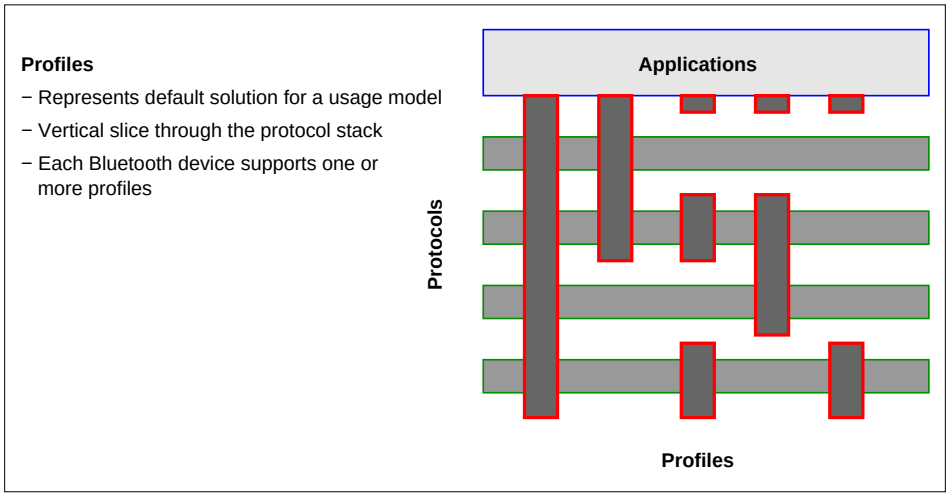


Figure 4.7: Bluetooth profiles

- *Intercom profile*
- *The cordless telephony profile*

Many other draft profiles are being designed for future versions of Bluetooth.

4.2.4 Research topics

Due to the wide application area covered by Bluetooth, many research issues see the light and some are treated by working groups. We should note that some of these issues are common to other wireless networking standards too:

- Coexistence with other wireless standards: As Bluetooth may coexist with other wireless networking standards like IEEE 802.11 and HiperLAN, they may have detrimental effect on each other. Work is in progress to determine and quantify these effects and propose solutions to improve efficiency, such as adaptive frequency hopping.
- IP over Bluetooth and pervasive networking: Issues like addressing, zero-configuration plug-and-play networking are treated.
- Ethernet emulation: In order to provide transparent Ethernet-like capabilities to higher layers, while hiding underlying Bluetooth complexities. This would result in IEEE 802.11-like cards.
- Power aware routing optimization: which tries to reduce the overall transmission power involved in node-to-node communications.

4.2.5 An example

In this subsection we will show, using the example in [8], how a Bluetooth device discovers, establishes links and requests services from other devices, along with the concerned protocols and functions (refer to Fig. 4.4).

Consider a cell phone and a laptop computer, both equipped with Bluetooth devices. The cell phone can act as a modem and periodically scans for inquiries to see if anyone want this service. When an application that needs dial up networking is opened on the laptop computer, this last knows it needs to establish a Bluetooth link to a device providing dial up networking profile. First, the laptop performs an inquiry to find out what Bluetooth devices are in its neighborhood, by transmitting a serie of inquiry packets. The cell phone eventually detects an inquiry message and replies with a *frequency hop synchronization* packet containing all necessary information for the laptop's Bluetooth device to create a connection to the cell phone.

The cell phone may not be the only device in the laptop's neighborhood, therefore other devices scanning for inquiries may also respond with *frequency hop synchronization* packets, so the laptop builds a list of its neighboring devices.

At this step, the laptop could present the user with the list of neighboring devices and their types it found and the user chooses what to do next. It could also be the application which takes the decision of searching for dial up networking devices, depending on the application design.

To check out whether a device supports a particular service, the application needs to connect to the other device's SDP and to interrogate it. First, the laptop pages the cellular phone which, if scanning for pages, responds to the paging and an ACL baseband link is established between the two devices. Now that the ACL link is ready, an L2CAP connection can be set above the ACL link, possibly multiplexing different flows of different protocols over a single ACL link, using identifiers to distinguish packets of different protocols.

The laptop uses the L2CAP channel to set up a connection to the SDP on the cellular phone, which will be interrogated for information about dial up networking by the laptop's SDP client. The cellular phone's SDP replies with the attributes relating to dial up networking. Now the laptop has the necessary information about dial up networking devices in its neighborhood, it may close the connection to the cell phone (to save battery power, or to establish another connection to another device).

At this step, it depends on the application's design again to give the choice to the user or to decide by itself which neighboring device it shall use.

The laptop pages the cellular phone to establish a new baseband ACL link, the same way it did for connecting for SDP. If the application has particular configuration parameters (such as QoS parameters) to be applied to the link, it may use the HCI to configure the Bluetooth device. Next, the LMP configures the link.

Now the ACL connection is set up, an L2CAP connection can be set up to be used by RFCOMM which supports dial up networking. L2CAP uses the specific RFCOMM's identifier to distinguish between multiplexed packet flows. RFCOMM, on its turn, may multiplex several protocols across one connection, each with its own channel number.

Now the dial up networking connection is set up and used by the laptop, without the need to be cable-connected to the cellular phone. If one of the devices moves out of range of the other, the laptop repeats the same procedure to find another device to connect to.

4.3 Comparison

To resume this chapter, Fig. 4.8 compares the three standards detailed in the last two chapters: IEEE 802.11, HiperLAN and Bluetooth. [82] compares HiperLAN-2 and IEEE 802.11a from the performance point of view.

Characteristic	802.11	802.11b	802.11a	HiperLAN−2	Bluetooth
Spectrum	2.4 GHz	2.4 GHz	5 GHz	5 GHz	2.4 GHz
~Max. PHY rate	2 Mbps	11 Mbps	54 Mbps	54 Mbps	1 Mbps
~Max. data rate, L3	1.6 Mbps	5 Mbps	32 Mbps	32 Mbps	723 Kbps
MAC	Carrier Sense MA with Coll. Avoid.(CSMA/CA)			TDMA / TDD	TDMA/TDD
Connectivity	Conn.less	Conn.less	Conn.less	Conn.Oriented	CL, CO
Multicast	Yes	Yes	Yes	Yes	Yes
QoS support	if PCF	if PCF	if PCF	Yes	Yes
Frequency selection	FHSS, DSSS	DSSS	Single carrier	Sing. carr +dyn. select.	FHSS+dyn.sel.
Authentication	Yes	Yes	Yes	Yes	Yes
Encryption	RC4	RC4	RC4	DES, 3−DES	SAFER+
Handover support	Proprietary / not specified in std.			No	No
Fixed network support	Ethernet	Ethernet	Ethernet	Eth.,IP,ATM,PPP ..	IP/PPP
Management	802.11 MIB	802.11 MIB	802.11 MIB	HiperLAN−2 MIB	−
Radio link quality control	No	No	No	Link adaptation	No

Figure 4.8: Comparison table of IEEE 802.11, HiperLAN and Bluetooth

Part II

Service differentiation in IEEE 802.11

Chapter 5

Service differentiation

Contents

The IETF is currently working on service differentiation in the Internet. However, in wireless environments where bandwidth is scarce and channel conditions are variable, IP differentiated services are sub-optimal without support from lower layers.

In this chapter we present four service differentiation schemes for IEEE 802.11 as we did in [83, 84, 85]. The first one is based on scaling the contention window according to the priority of each flow or user. For different users with different priorities, the second, the third and the fourth mechanisms assign different minimum *contention widow values*, different inter frame spacings and different maximum frame lengths respectively. We simulate and analyze the performance of each scheme with TCP and UDP flows.

5.1 Introduction

Wireless communications are an emerging technology and are becoming an essential feature of everyday's life. Not only computer networks are becoming mobile [67], eventually each device will have one or several wireless interfaces (e.g. laptops, cameras, phones etc.) [8]. Simultaneously, multimedia is having an equivalent growth. Multimedia applications impose requirements on communication parameters, such as data rate, drop rate, delay and jitter. Guaranteeing those requirements in wireless environments is very challenging because wireless links have variable characteristics (due to noise). To deal with this problem, many wireless communication standards have been defined. Some of the proposals enhance the Quality of Service (QoS) of the whole system, others differentiate between the priorities of each mobile host, offering them different quality of service parameters (e.g. different data rates or delays etc.)[2, 4]. In this chapter we propose mechanisms for service differentiation for IEEE 802.11. The chapter is organized as follows: Section 5.2 presents simulations and analysis of the IEEE 802.11 when used with TCP (transport control protocol)[86] and UDP (user datagram protocol)[87] transport protocols. Section 5.3 introduces some means of service differentiation on the wireless link with some simulations and mathematical models. Section 5.3 analyzes these mechanisms in noisy environments. Section 5.4 analyzes per-flow differentiation. Finally, section 5.6 gives some hints for future work and section 5.7 concludes this chapter.

5.2 UDP and TCP over IEEE 802.11

In this section, we present simulation results, using *NS* [88], and we analyze the behavior of UDP and TCP when running on top of an IEEE 802.11 MAC sub-layer. More performance analysis of TCP and UDP over CSMA/CA can be found in [89]. The topology of the simulation network is rather simple (see Fig. 7.4): Three WTs, denoted by WT_i where $i = 1, 2$ and 3 respectively, are uniformly distributed around an AP and are sending their packets to a fixed host wire-attached to the AP.

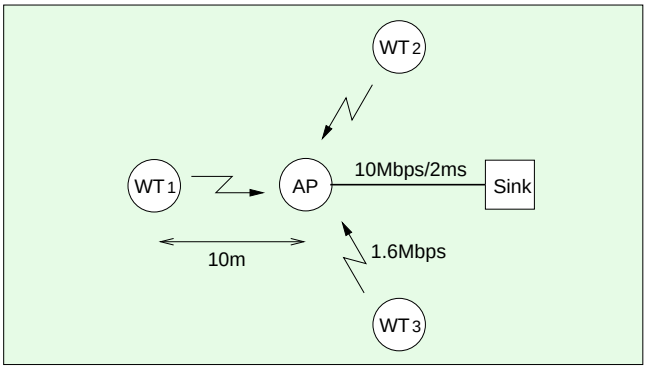


Figure 5.1: Simulation network topology.

5.2.1 UDP flows

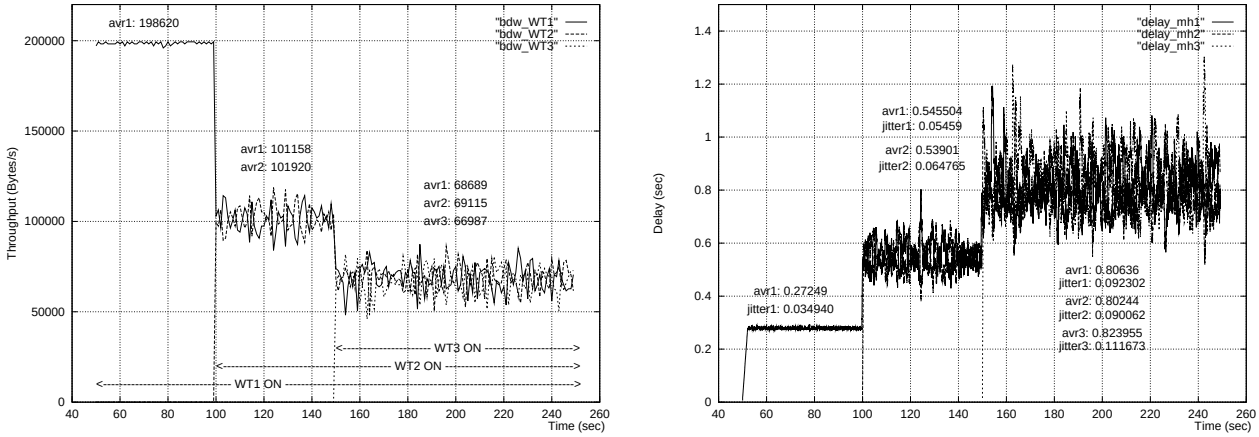
Let us first consider the use of constant bit rate (CBR) traffic sources over UDP. WT_1, WT_2 and WT_3 start sending their CBR/UDP packets at seconds 50, 100, and 150 respectively, using the RTS/CTS scheme. Simulation ends at second 250. During time interval [50,100[, WT_1 can get the desired data rate as long as it does not exceed the effective radio link data rate, i.e. 1.6 Mbps in our simulation (considering 2 Mbps raw data rate). In this example a single traffic overloads the link, sending 1100-byte packets each 0.005 seconds (giving a data rate of 1.76Mbps $>$ 1.6Mbps, so the channel is busy most of the time). As shown in Fig. 5.2(a), WT_1 has a stable throughput. It also has short delays and jitters (Fig. 5.2(b)) (we consider that the jitter is the standard deviation of the delay). The drop rate, which is about 10% in this case, depends on the used bit rate. During the second phase (i.e. between seconds 100 and 150), WT_1 and WT_2 share the data rate almost equally as they both have the same probability to access the medium (Fig. 5.2(a)). The average delays of both traffics are higher than in the first period due to a higher number of RTSs denied: The channel is occupied by one terminal, the other terminal must wait during that time. It can also be the case that RTSs collide. Jitter also gets higher due to the more variable channel usage, caused by a higher number of WTs contending to access the channel. During the third period, between seconds 150 and 250, WT_3 shares the medium with the previous two. Throughput gets lower, since data rate is shared among the three WTs. Delay, jitter and drop rate get higher.

5.2.2 TCP flows

When we replace the UDP transport layer with the TCP one, the throughput, delay and jitter behave the same way as in UDP. However packet dropping due to buffer overflow at the sender is avoided with TCP. We observe absolutely no TCP dropped packets due to its adaptability: When the sender requests to transmit and the channel is idle, no dropping is observed as long as the traffic is adapted to the offered throughput, which is the case of TCP. Some RTSs collide, are dropped, then retransmitted by the MAC sub-layer transparently to the TCP layer.

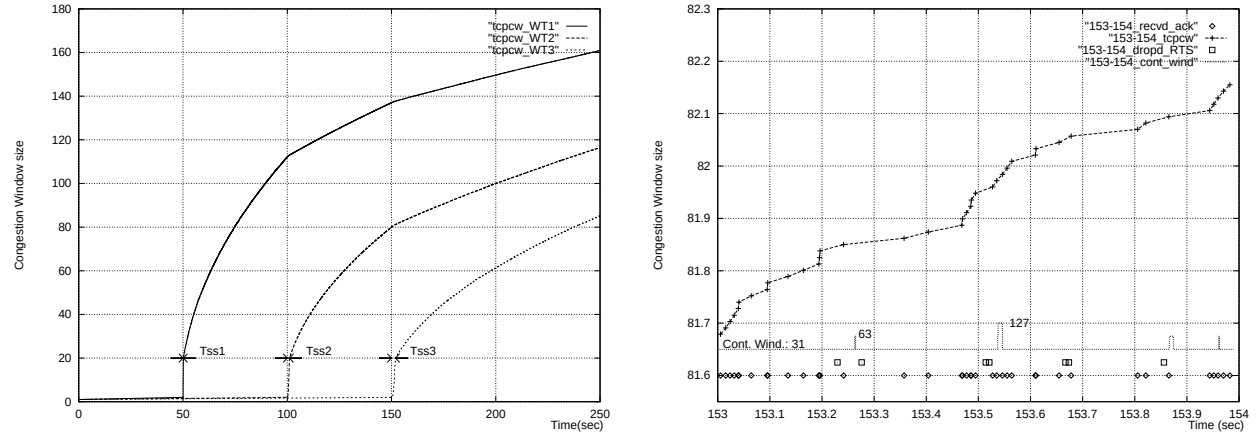
The TCP *congestion window (cwnd)* sizes of all three WTs are shown in Fig. 5.3(a), for the whole simulation time. Even if *cwnd* is a byte counter in TCP, we express *cwnd* in packets for convenience.

At each new period, more congestion occurs and the general slope decreases. However the *congestion window* never decreases during the simulation time, even at the instant values scale. After the *Slow Start* period, in which the *cwnd* increases by 1 at each TCP-ACK reception, the *cwnd* reaches the *ssthreshold* (20 in this case) then the *congestion avoidance* period starts, during which *cwnd* increases by $1/cwnd$ at each TCP-ACK reception. If a packet is dropped, detected by timing out the TCP-ACK or by receiving multiple similar TCP-ACKs, the *ssthreshold* is set to $cwnd/2$ and the *cwnd* is reset to 1 [86].



(a) (b)

Figure 5.2: Throughputs and delays using UDP



(a) Congestion window size of all WTs

(b) Cong. window size of WT_2, during the period 153-154.

Figure 5.3: TCP congestion window sizes.

Fig. 5.3(b) is a "zoom" of WT_2 *congestion window* during the period [153,154]. This figure also shows TCP-ACK packets reception instants, the RTS dropping and the *contention window* sizes.

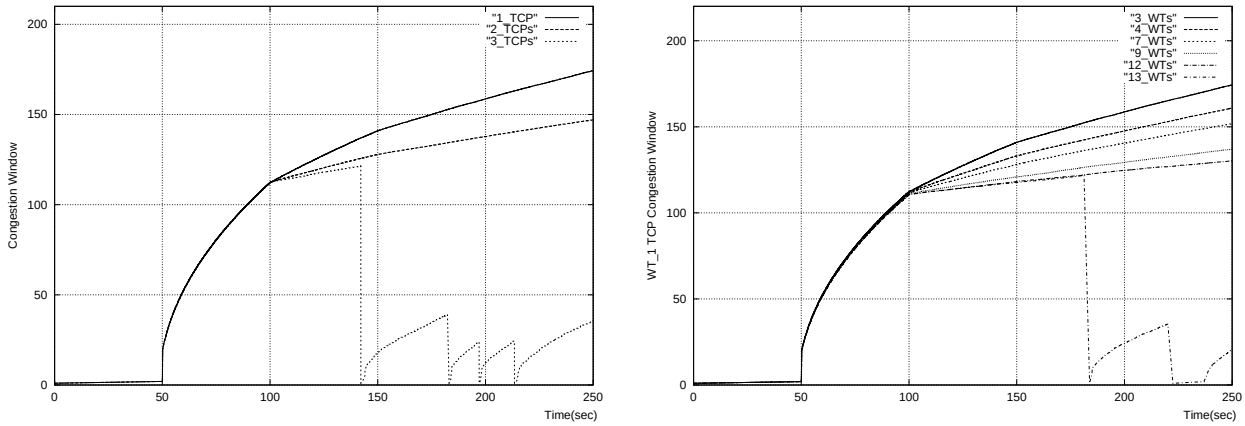
At each TCP-ACK packet arrival, the *congestion window* increases by $1/cwnd$ as we are in the *congestion avoidance* period, and it never decreases because TCP never times out for a TCP-ACK reception: dropped RTSs, for TCP-ACK as for data packets, are retransmitted by the MAC sub-layer much faster than the TCP timeout. When two or more WTs are used simultaneously, the delay between two TCP-ACK packets reception is obviously higher than when using a single WT due to more collisions, less free channel periods etc. Therefore the TCP *congestion window* increases at a slower rate (as seen in Fig. 5.3(a)) and the slope is lower. This can also be noticed when comparing the WTs' respective *slow start* periods Tss_1, Tss_2 and Tss_3 shown in Fig. 5.3(a). Surely, these delays not only affect the *cwnd*, but the data rate too. In fact, when using TCP the data rate is $\lim_{t\to\infty} \frac{1}{t \times RTT} \int_0^t cwnd\, dt$, where RTT is the round trip time [90]. Last, we should note that a TCP source won't receive the TCP-ACK of a packet if:

- after several RTS attempts, the data packet has been dropped by the MAC sub-layer.
- after several RTS attempts, the TCP-ACK has been dropped by the MAC sub-layer.
- either the data packet or the TCP-ACK did not reach its destination, because of *noise* on the channel.

A severe or busy channel could lead to such scenario: Consider the case where WT_1 uses TCP while WT_2 and WT_3 use UDP flows of the same packet size as TCP. Even though each of the CBR/UDP flows is configured to consume all the available data rate, we see that WTs equally share the available data rate. No TCP timeouts

were observed and the contention window keeps increasing during the simulation time. Even when we increase the number of UDP flows in WT_2 and WT_3, we observe no effect on the contention window of WT_1: available channel data rate is shared among WTs and not among different flows. Several flows in a single WT share the same MAC sublayer and so they have the effect of a single flow toward other WTs. Decreasing (resp. increasing) the CBR packet sizes in WT_2 and WT_3 would decrease (resp. increase) the TCP *cwnd* slope in WT_1.

To force TCP timeouts, we increased the number of TCP flows in WT_1 from 1 to 2 and 3, while WT_2 and WT_3 use UDP flows. The congestion window sizes of the TCP connections are shown in Fig. 5.4(a).



(a) Using several TCP flows in WT_1.

(b) WT_1's TCP *cwnd* when using several WTs.

Figure 5.4: TCP congestion windows when using several data flows

When two TCP flows use the same MAC sublayer, each of them will have longer delays before accessing the channel than when acting alone. This reduces the slope of the *cwnd* considerably. Adding a third TCP flow in the same WT introduces more delays for channel access, causing TCP timeouts before receiving the waited ACK. Note that "Full data rate" CBR/UDP flows added in WT_1 would consume the whole available data rate, without sharing it with TCP.

A similar observation is made on TCP *cwnd* (in WT_1) when we increase the number of WTs from 3 to 13, using either UDP or TCP (Fig. 5.4(b)). When the number of WTs is large enough, TCP may also time out after several consecutive collisions. Note that there is no possible congestion at the AP or the fixed host in our simulations.

5.3 Differentiation mechanisms

As mentioned in the introduction, in order to give WTs either statistical or absolute QoS guarantees, we can get differentiated services between WTs by giving them different QoS parameters.

When using PCF (Polling Coordination Function), introducing priority is centralized and somehow simple as in TDMA. We will not get into it in this paper. We aim to introduce priorities in the IEEE 802.11 using the DCF (Distributed Coordination Function). Several parameters can be considered, among which:

1. *Backoff increase function:* Each priority level has a different backoff increase function.
2. CW_{min}*:* Each priority level has a different minimum contention window value.
3. *DIFS:* Each priority level is assigned a different DIFS, after which it can transmit its RTS or data packet.
4. *Maximum frame length:* Each priority level has a maximum frame length allowed to be transmitted at once.

In the following subsections we analyze them separately and show simulation results with corresponding mathematical analysis.

5.3.1 Backoff differentiation

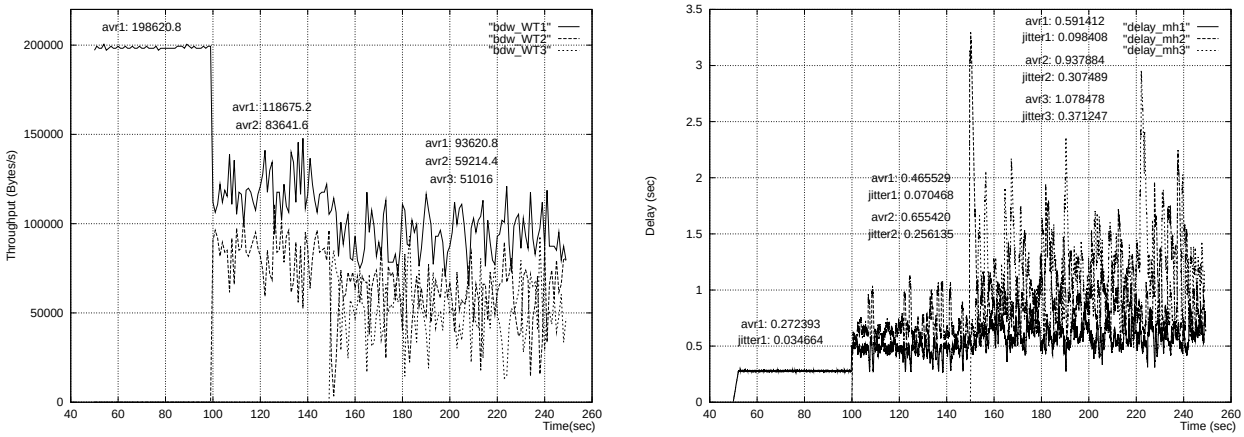
As we have seen in chapter 3, (3.1):

$$Backoff\_time = \lfloor 2^{k+i} \times rand() \rfloor \times Slot\_time$$

the only configurable term in this equation is 2^{k+i}. Our first attempt to introduce priority is to replace it by ${P_j}^{k+i}$ where P_j is a priority factor of WT_j. Therefore, instead of multiplying the range by two at each retransmission, we multiply it by P_j. Here, the higher the priority factor is, the larger is the backoff range, the lower is the chance to first access the channel, the lower is the throughput.

UDP flows

We used this scheme in the same network configuration as section 5.2. WTs send UDP packets, using the RTS/CTS scheme. At second 50, WT_1 starts transmission with a priority factor P_1=2 (meanwhile WT_2 and WT_3 are idle). Then, at second 100, WT_2 starts transmission with P_2=6. Finally, at second 150, WT_3 starts transmission with P_3=8. The AP uses a priority factor of 2. Results are shown in Fig. 5.5(a), 5.5(b) and 5.6.



(a) (b)

Figure 5.5: Throughputs and delays using UDP with priorities

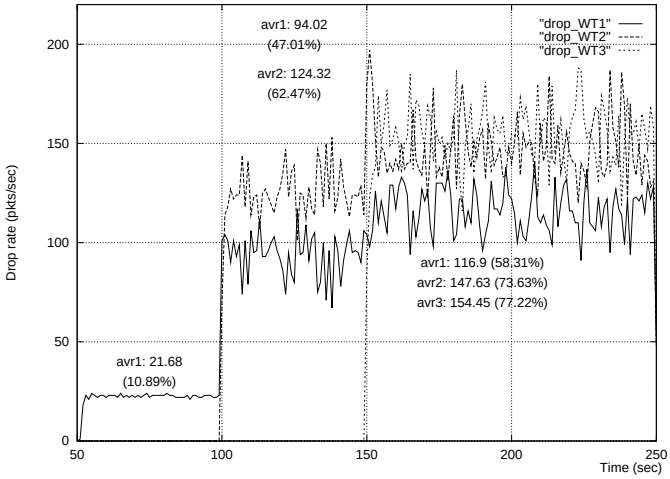


Figure 5.6: Drop rates using UDP with priorities.

When only WT_1 is on, it uses the whole link data rate, exactly as in the case with no priorities (cf. section 5.2). When WT_2 goes on (at second 100), the link is unequally shared between the two WTs, WT_1 having a higher data rate share (1.42:1). At second 150, the third WT goes on and the results show that the three WTs get different data rate shares. Obviously, we can change the ratios P_i/P_j $(i \neq j)$ to obtain other data rate shares with a wider range, therefore better priorities. But as this range increases (high priority ratios) the system becomes unstable, showing more data rate variability and higher jitters[1]. This instability is more visible with low priority traffics (high priority factors, as with WT_3). From the data rate point of view, the whole system efficiency gets slightly better when using more WTs, due to more sensing , "filling" more channel idle

[1]High delays are caused by the channel overload, even with a single WT, see Chap. 8 for a detailed explanation.

periods and getting the channel more busy (comparing the data rates of WT_1, WT_1 and WT_2 together, and all three WTs in Fig. 5.2(a): $(avr_1 + avr_2 + avr_3)_{150-250} > (avr_1 + avr_2)_{100-150} > (avr_1)_{50-100}$). As shown in Fig. 5.2(a) and 5.5(a) these data rate sums remain almost the same after introducing the priority scheme.

TCP flows

Note that when we replace UDP by TCP in all WTs, the results are quite different: they show no considerable differentiation effect, and all three WTs almost equally share the data rate, as shown in Fig. 5.7. In fact, TCP is an adaptive transport protocol based on the feedback control embedded in the reception of ACK packets. In both *Slow Start* and *Congestion Avoidance* periods, TCP sends new data packets only at ACK reception. There are two reasons that explain why using *backoff* differentiation is not efficient for TCP flows:

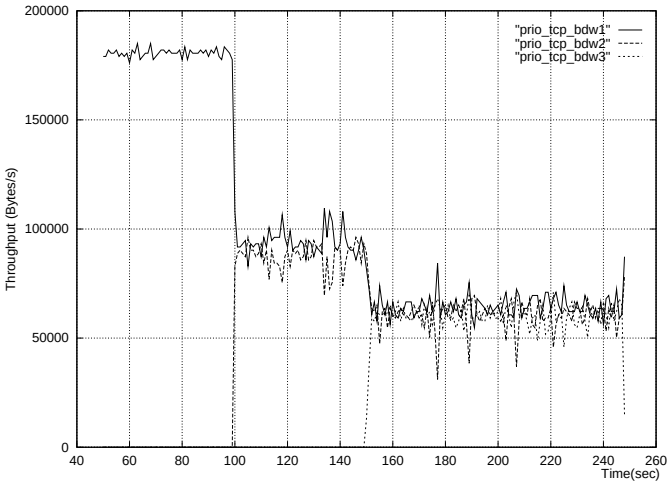


Figure 5.7: Throughputs using TCP with priorities.

- *Common priority for all TCP-ACKS:* The AP sends all TCP-ACKs for all WTs using the same priority (the highest in our simulations) as our differentiation is on a per-station basis, not per-flow basis. In a per-flow differentiation scheme, the AP would have to look into the header of each packet to check the destination address/port. This gives additional load for the AP. Per-flow differentiation is discussed in Section 5.4. It could be also the case that differentiation is made on a packet basis, which supposes that each packet has a priority field that sets the differentiation parameters (similar to DiffServ [91]). The additional field causes overhead for short packets. This approach is left for future work.

- *Slow AP:* The backoff differentiation mechanism works only if a WT does not receive any CTS upon sending an RTS, it then increases its contention window. The contention window increases proportionally to the different priority factors P_i assigned to each WT. Therefore the probability of scaling the contention window size is proportional to the probability of RTS collision which is proportional to the number of contending RTSs. With TCP, during the congestion avoidance phase, a source waits for a new ACK before generating a new packet, i.e. generating an RTS, because of the congestion control algorithm. In our case, these ACKs are generated by a central entity, the AP. This AP tends to become the "coordinator". If the AP is slow, most of the WTs will be waiting for an ACK and therefore the number of contending WTs will be lower. Respectively, if the AP is fast enough, each WT will receive an ACK and will be ready to contend to access the channel.

The number of contending WTs (i.e. ready to send an RTS), is shown in the birth-death chain of Fig. 5.8. The AP succeeds to send a TCP-ACK with a probability β_i, increasing the number of contending terminals. It fails sending its TCP-ACK with a probability α_i, thus increasing the number of waiting TCP-ACKs (the number of waiting packets is therefore reduced).

If the AP sends TCP-ACKs slowly (i.e. with a low priority), α_i are greater than β_{i-1}, and the chain drifts to the state 0: most WTs will be waiting for a TCP-ACK. This leads to a lower number of contending WTs (each with an RTS) and therefore to a low RTS collision probability. In this case our scheme does not work very well. No priority effect can be seen.

If the AP sends TCP-ACKs fast enough (i.e. it has a much higher priority than all the WTs), β_{i-1} are greater than α_i, and the chain drifts to state 3: TCP sources will receive their ACKs very quickly and most of them will be contenting for the medium. This leads to a higher number of RTSs contending to access the

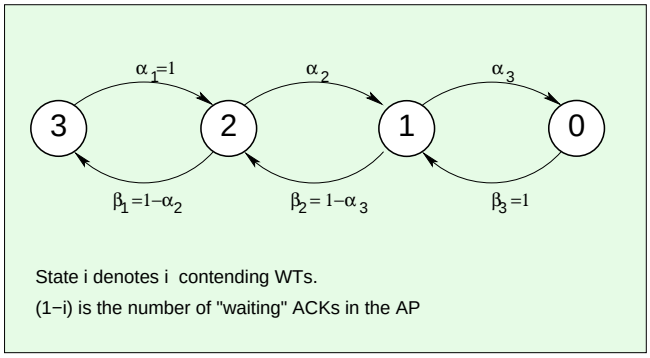


Figure 5.8: State transition diagram for TCP generated packets.

channel, which increases the probability of RTS collisions. In this case, the backoff priority scheme works well and the priority effect is much more visible.

To check out the *slow AP* assumption, we counted each congestion window size occurrence when using UDP and TCP separately. Results are shown in Table 5.1: Using UDP, 37195 RTSs were sent by all terminals, out of which 2313 ($= 34 + 922 + 830 + 527$) collided and the contention windows increased proportionally to each terminal's priority factor, to become 62, 62, 186 and 248 respectively. Using TCP (where the AP also has to send TCP-ACK packets), contention windows did not increase as often as with UDP. Note that more packets are sent on the network because of the TCP-ACKs. Therefore one should compare the ratio: (number of backoffs)/(total number of RTSs) instead of the actual numbers in the tables. With TCP, the contention widow value 62 has been reached more often than with UDP. This is because of an additional node, the AP, contending to access the channel to send the TCP-ACKs.

Table 5.1: Contention window distributions

	CW distrib. for UDP				CW distrib. for TCP			
Cont. Win. Size	AP	WT_1	WT_2	WT_3	AP	WT_1	WT_2	WT_3
(CW_{min})31	555	22718	9182	4740	28969	17099	8794	5076
62	34	922	-	-	1885	1466	-	-
124	0	45	-	-	33	53	-	-
186	-	-	830	-	-	-	940	-
248	0	5	-	527	0	0	-	667
496	0	0	-	-	0	0	-	-
992	0	0	-	-	0	0	-	-
(CW_{max})1023	0	0	75	54	0	0	29	22

This shows why introducing priorities in the backoff time increase has lower effect on TCP than on UDP. In other words, for the same P_i/P_j used with TCP and UDP, the resulting relative priority range width is much higher with UDP.

Combined TCP-UDP flows

When the AP's priority is not high enough, simulations show that when we apply the backoff priority mechanism on different flow types, in different WTs, simultaneously:

- A UDP flow with high priority won't have considerable advantage over a single TCP flow with lower priority, and the common channel data rate is equally shared. In fact, the UDP RTSs are exposed to collision with AP RTSs, while TCP RTSs collide less often.

- On the other hand, when we apply the priority scheme to a WT with high priority using TCP flows, and another with low priority using UDP flows, high priority TCP flows get more throughput than low priority UDP ones. Backoff priorities enhance the TCP throughput without necessarily enhancing the *cwnd* size, as the RTT is considerably reduced relatively to the no-priority scheme.

Mathematical analysis

In this subsection we present a mathematical analysis of the simulation results shown in Fig. 5.5(a). The analysis aims to explain the data rate shares and collision probability in the second period (seconds 100 to 150) when using UDP. Similar but more complex reasoning can be applied to the third period.

During the second period (seconds 100 to 150), where only WT_1 and WT_2 are transmitting at full data rates, each of the WTs' data rate share is proportional to its probability to access the channel, i.e. its random backoff value is lower than the other's ($DIFS + Backoff_1 < DIFS + Backoff_2$). This is similar to comparing two random variables (r.v.) X and Y which bounds are $[a, b]$ and $[a, d]$ respectively. The probability of having $X < Y$ (thus WT_1 accessing the channel before WT_2) is given by:

$$P(X < Y) = \begin{cases} 1 - \frac{1}{2} \times \frac{b+1-a}{d-a} & \text{if } b \leq d \\ \frac{1}{2} \times \frac{d-a}{b-a} & \text{if } b > d \end{cases} \tag{5.1}$$

Subtracting DIFS from a, b and d simplifies the equations without changing $P(X < Y)$. As time is slotted, where a time slot is equal to the contention window unit, a collision occurs when $X = Y$, and both transmitted packets are dropped. The collision probability is given by:

$$P(X = Y) = \frac{1}{max(b, d)} \tag{5.2}$$

Initially, both ranges $[a, b]$ and $[a, d]$ are equal to $[0, CW_{min}]$, b and d denote the contention window sizes cw_1 and cw_2 of WT_1 and WT_2 respectively. As contention windows increase at each collision and decrease at each successful transmission, the combination of subsequent cw_1 and cw_2 values give the 21-state transition diagram of Fig. 5.9 and 5.10. Multiplying the probability of WT_1 success (i.e. $P(X < Y)$ given in (5.1)) in each state by the probability of that state, then summing over all 21 states, gives the WT_1 data rate share (0.59 in this case). Similar computations give the WT_2 data rate share and collision probability. Note that routing packets are not taken into consideration in this analysis, but surely are considered in the simulation, which results in a slight difference between simulation and mathematical results.

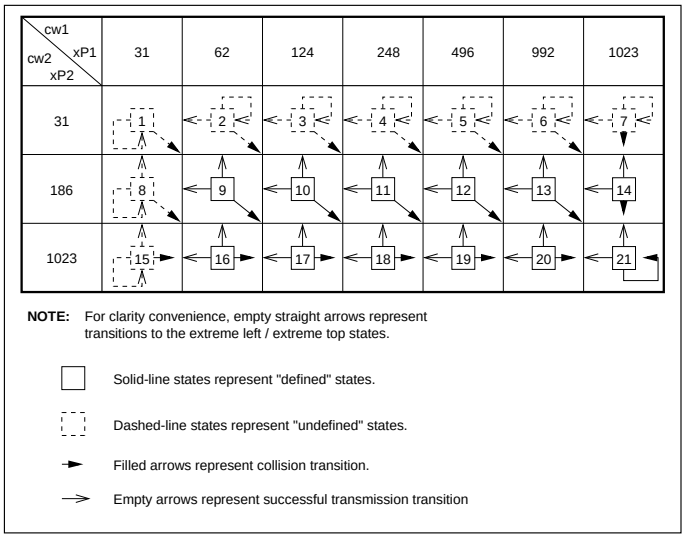


Figure 5.9: Contention windows state transition diagram

The transition diagram of Fig. 5.9 helps finding each state probability and the inter-state transition probabilities: A filled arrow represents a transition due to a collision, after which both contention windows are multiplied by their respective priority factors P_i. In this case, we called the target state a *defined* state, where both *backoff_times* are re-computed. *Defined* states, in which both cw_1 and cw_2 correspond to the indicated values, are shown with a solid line. In *defined* states, applying (5.1) and (5.2) to these values give us the transition probabilities.

On the other hand, and empty arrow indicates a transition due to a successful transmission. For clarity reasons, empty straight arrows represent transitions to the extreme left / extreme top states and not to adjacent states. e.g. in state 19, if WT_1 succeeds accessing the channel, the transition is made to state 15, not state 18. In case of successful transmission, the winning WT resets its contention window (to 31), while the other WT keeps reducing its backoff. This is represented with *undefined* states surrounded by dashed lines. An *undefined* state has one reset (31) contention widow size which bounds the new *backoff* value, and the second *backoff*

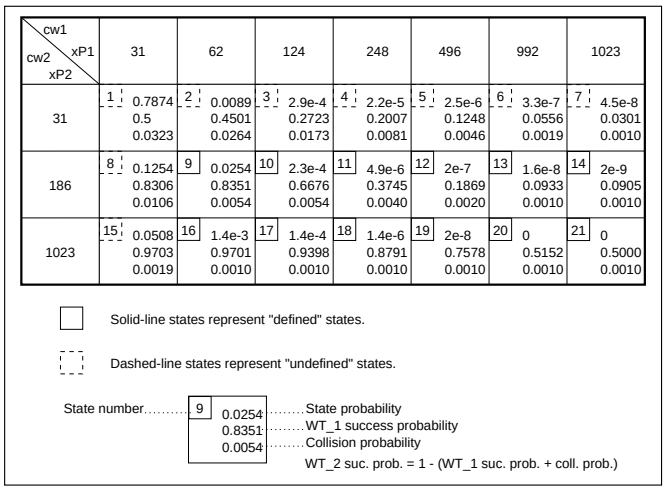


Figure 5.10: Contention windows state transition diagram: numerical values

depends on the previous states. This makes the outgoing transition probabilities function of several previous states, hence the chain is not a Markov chain.

The unknown *backoff* bound in *undefined* states could be replaced by the expected contention window size, taking into consideration previous states probabilities and the corresponding transition probabilities. Applying (5.1) and (5.2) to each state gives a set of equations which, once solved, gives the probability of each state.

One major observation on this chain is that it strongly drifts to state 1 (with probability 0.79), in which both contention windows are reset to $CWmin$, both equal 31. This fact makes the data rate shares slightly dependent of the P_1/P_2 values. To deal with this, we considered $CWmin$ differentiation, in which P_1/P_2 values strongly influence the data rate shares. The resulting data rate difference can be clearly seen in the simulations, when using UDP or even TCP flows.

5.3.2 CW_{min} differentiation

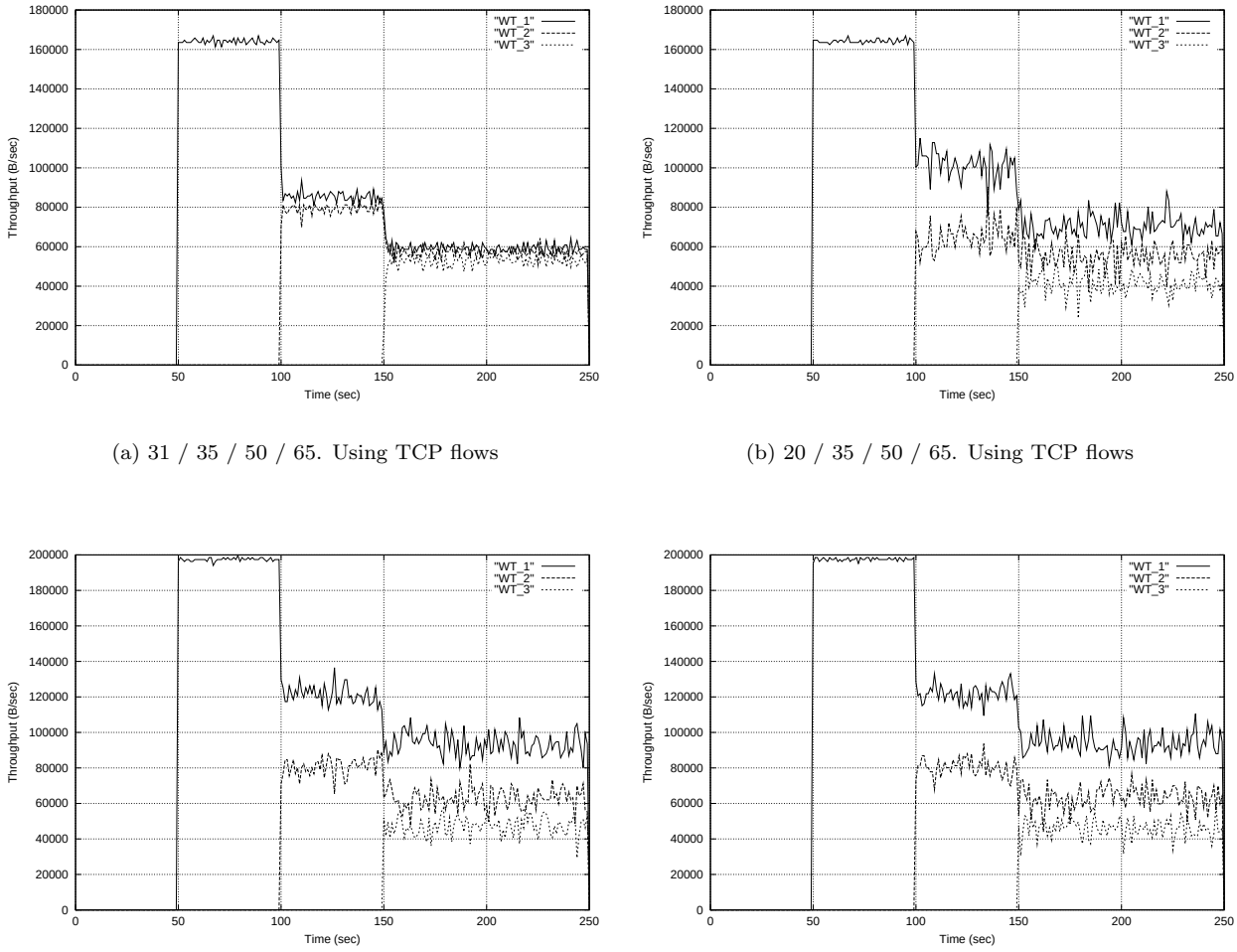
Working on *backoff* differentiation led us to the second differentiation mechanism, the CW_{min} differentiation. The main motivation is that, with a small number of WTs contending to access the channel, CW values are at their minimum value (CW_{min}) most of the time. Therefore, a backoff differentiation mechanism won't be applied correctly since the CWs are rarely increasing, and high CW values are rarely used. This led us to differentiate the most utilized CWs: CW_{min}.

The simulation scenario is the following: WT_1 starts transmitting at second 50, then WT_2 starts at second 100, then WT_3 starts at second 150. Simulation ends at second 250. Packet sizes are 1100 bytes long, sent at 0.005 second intervals when using UDP flows. Results for both UDP and TCP flows are shown in Fig. 5.11(a) to 5.11(d) for comparison convenience. The sets of values shown as $w/x/y/z$ indicate the values of CW_{min} for $AP/WT_1/WT_2/WT_3$ respectively.

When we use TCP flows, with 31/35/50/65 CW_{min} values (Fig. 5.11(a)), there is no noticeable differentiation effect visible. This is due to the slow TCP-ACKs transmissions by the AP. In fact, the AP uses a CW_{min} which is close to that of WT_1. As each WT has to wait for a TCP-ACK before starting a new transmission, a slow AP makes each of the "closed-loop" flows much slower and the different CW_{min} values assigned to the WTs do not have any real effect. When we use a faster AP, with CW_{min}= 20 (Fig. 5.11(b)), the TCP-ACKs are sent much faster, so the different WTs do not have to wait as in Fig. 5.11(a) before transmitting, and the different CW_{min} values they have show much more effect.

On the other hand, when we use UDP flows (Fig. 5.11(c) and 5.11(d)), accelerating the AP from CW_{min}= 31 to CW_{min} = 20 has absolutely no effect on the differentiation scheme. This is obviously true as WTs do not wait for any feedback from the AP, so the data rate shares remain the same, whatever is the CW_{min} value the AP has.

Comparing the figures vertically (compare Fig. 5.11(a) to 5.11(c), and Fig. 5.11(b) to 5.11(d)), shows that, for the same sets of CW_{min} values, UDP flows get more differentiation effect than TCP flows. Consequently, the data rate shares that UDP flows get can be considered as the maximum data rate shares that TCP flows can get, when we accelerate the AP indefinitely.



(a) 31 / 35 / 50 / 65. Using TCP flows

(b) 20 / 35 / 50 / 65. Using TCP flows

(c) 31 / 35 / 50 / 65. Using UDP flows

(d) 20 / 35 / 50 / 65. Using UDP flows

Figure 5.11: CW_{min} differentiation

5.3.3 DIFS differentiation

We have seen in the previous paragraphs that using *backoff* differentiation does not always apply to TCP flows. CW_{min} differentiation partially solves this problem, but it cannot provide strict priorities. An alternative solution would be to use DIFS for differentiation.

As shown in chapter 3, IEEE 802.11 ACK packets get higher priority than RTS packets, simply by waiting SIFS which is shorter than DIFS (for RTS). We will use the same idea to introduce priorities for data frames (in the basic scheme) and for RTS frames (in the RTS/CTS scheme). In this approach we give each priority level a different DIFS, say $DIFS_j$ where $DIFS_{j+1} < DIFS_j$. So the WTs having priority j waits $DIFS_j$ idle period before transmitting the packet. To avoid same priority frames collision, the backoff mechanism is maintained in a way that the maximum contention window size added to $DIFS_j$ is $DIFS_{j-1} - DIFS_j$ as illustrated in Fig. 5.12. This ensures that no WT of priority $j+1$ has queued frames when WT of priority j starts transmission. Low priority traffic will suffer as long as there are high priority frames queued.

It could also be the case that the maximum random range (RR_j) after $DIFS_j$ can be made greater than $DIFS_{j-1} - DIFS_j$, so the previous rule becomes less severe. In this case, a packet which failed to access the channel at the first attempt will probably have its priority reduced after consecutive attempts, depending on the DIFSs and the RRs values. This technique may be useful for real-time application, where we have more constraints on delays than on packet drops. Simulation results show the following:

- This mechanism offers a very wide range of relative priority: It can be a 1:1 when DIFSs are equal and RRs are equal. The relative priority can be infinite when $DIFS_j \geq (DIFS_{j+1} + RR_{j+1})$.
- Applying DIFS differentiation shows no efficiency loss, as seen in Fig. 5.13 (here, the packet size is 2312 bytes).

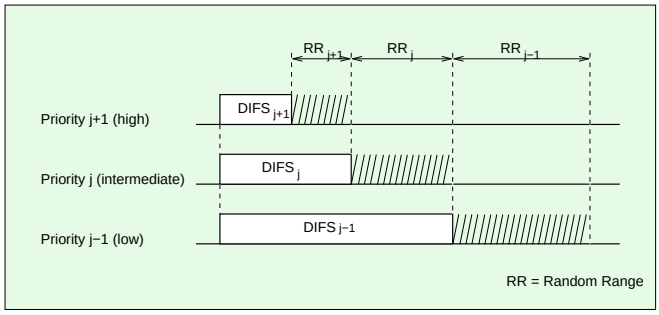


Figure 5.12: Including priority in DIFS

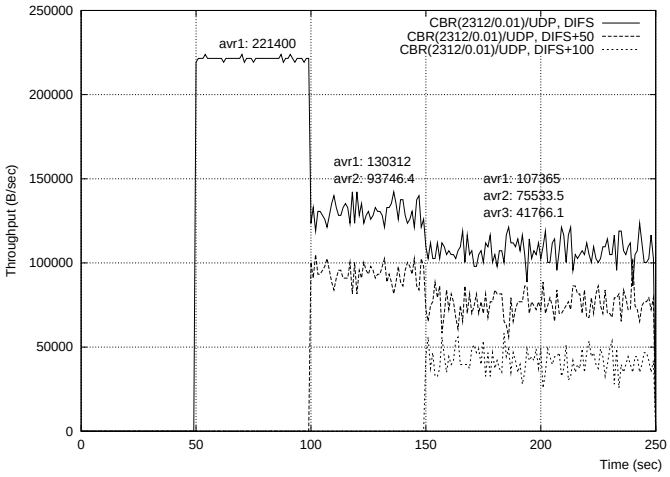


Figure 5.13: Using CBR/UDP, $DIFS_{AP} = 50\mu s$, $DIFS_{WT_1} = 50\mu s$, $DIFS_{WT_2} = 100\mu s$, $DIFS_{WT_3} = 150\mu s$

- For the same $DIFS_j$ sets, UDP shows more priority effect (e.g. throughput ratios) than TCP. Here there is no backoff problem with TCP (as when applying *backoff* differentiation), but TCP-ACK packets of several WTs are still sent with the same priority, which reduces the priority effect. When we accelerate TCP-ACK transmission by reducing the AP DIFS (low bounded by SIFS), differentiation becomes more visible, as shown in Fig. 5.14.
- For TCP flows, as we increase the AP DIFS, the relative priority decreases.
- We can apply this mechanism to give UDP priority over TCP (which was not always applicable with *backoff* differentiation) and vice versa, same $DIFS_j$ result in the same throughput ratios.

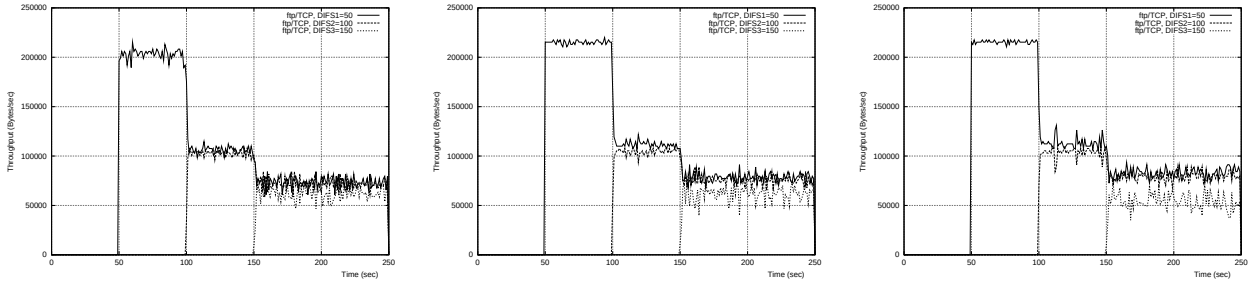
Mathematical analysis

In order to find the interpretation for the data rate shares of the various WTs when using UDP, let us start by analyzing the second period, with two active WTs, then we will move to period three and generalize the analysis.

With two active WTs, and as packet types are equal, we can say that the data rate share of a given WT (say WT_1) is equal to the probability that WT_1 accesses the channel first. That is the corresponding (*DIFS + backoff*) value is less than the others. This leads us to the problem of two random variables (r.v.) X_1 and X_2 with different bounds $[a, b]$ and $[c, d]$ respectively, uniformly distributed over these ranges (see Fig. 5.15(a)). We can easily show that, the probability of having $X_1 \leq X_2$ is:

$$P(X_1 \leq X_2) = \begin{cases} 1 - \left(\frac{1}{2} \times \frac{b-c}{d-c} \times \frac{b-c}{b-a}\right) & \text{if } b \geq c \\ 0 & \text{if } b \leq c \end{cases} \tag{5.3}$$

This equation complies, with a difference of only 0.7%, to the data rate shares of WT_1 and WT_2 of our simulation during the second simulation period (seconds 100 to 150). In fact, the initial contention window size is 31, a $Slot\_time$ is $20\mu s$ which gives a random range of $620\mu s$, for both of the WTs. $DIFS_{WT_1} = 50\mu s$



(a) $DIFS_{AP} = 50\mu s$ (b) $DIFS_{AP} = 35\mu s$ (c) $DIFS_{AP} = 21\mu s$

Figure 5.14: DIFS differentiation with TCP flows

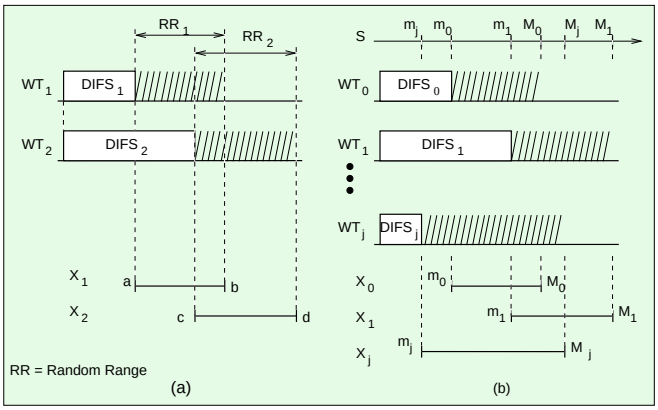


Figure 5.15: Corresponding r.v. for DIFS

and $DIFS_{WT_2} = 100\mu s$, together with the random ranges, give a=50, b=670, c=100 and d=720 for (5.3). The slight difference of 0.7% between (5.3) and the simulation results can be due to:

- the non perfect random number generator of the simulator.
- not taking subsequent backoffs into consideration in the mathematical analysis.

In order to apply the analysis to the third period (seconds 150 to 250), we have to consider more than two r.v. and the analysis becomes less intuitive and more complex. We also generalize the case to any disposition of the various r.v. bounds, as in Fig. 5.15(b).

Let $N+1$ be the number of WTs (as well as the number of r.v.). Let m_i and M_i be the lower bound and the upper bound respectively of r.v. X_i. Let S be the ordered set of the bounds (lower and upper) of all the r.v. For all $i = 0, .., N$, let S_i be the ordered set of the bounds (lower and upper) of all the r.v. such that $s_i \in S_i, m_i \leq s_i < M_i$.

Given a r.v. X_0, we show that the probability that X_0 is less than all other r.v. $X_i, \forall i \neq 0$ is given by:

$$P(X_0 \leq X_{k\neq 0}) = \sum_{s_j \in S_j, j=0,..,N} \left(\prod_{i=0}^{N} \frac{s_i^+ - s_i}{M_i - m_i} \times \delta_s \right) \tag{5.4}$$

where,
s_i^+ is the element succeeding s_i in S and

$$\delta_s = \begin{cases} 1 & \text{if } s_0^+ \leq s_i \; \forall i \neq 0 \\ 0 & \text{if } s_0 \geq s_i^+ \; \exists i \neq 0 \\ 1/(n+1) & \text{otherwise, where } n \text{ is the number} \\ & \text{of "i"s where } s_i = s_0 \end{cases}$$

Equation (5.4) is useful to explain the data rate shares of several WTs when using DIFS differentiation. Inverting this equation is useful to determine $DIFS_i$ function of the desired data rate shares among the WTs,

e.g. when using *DiffServ* or to optimize *end-to-end* parameters. The number of operations (divisions and multiplications) needed when applying (5.4) directly grows as N^N, which shows that further computing optimization is needed to be applied in real-time admission control for large numbers of WTs. Equation (5.4) is not applicable to WTs with TCP flows, where we should take the base station flow (TCP-ACKs) into consideration, as well as packets of different sizes, which adds new factors to the equation.

5.3.4 Maximum frame length differentiation

The fourth mechanism that can be used to introduce service differentiation into IEEE 802.11 is to limit the maximum frame length used by each WTs. Here, we should distinguish between two possibilities:

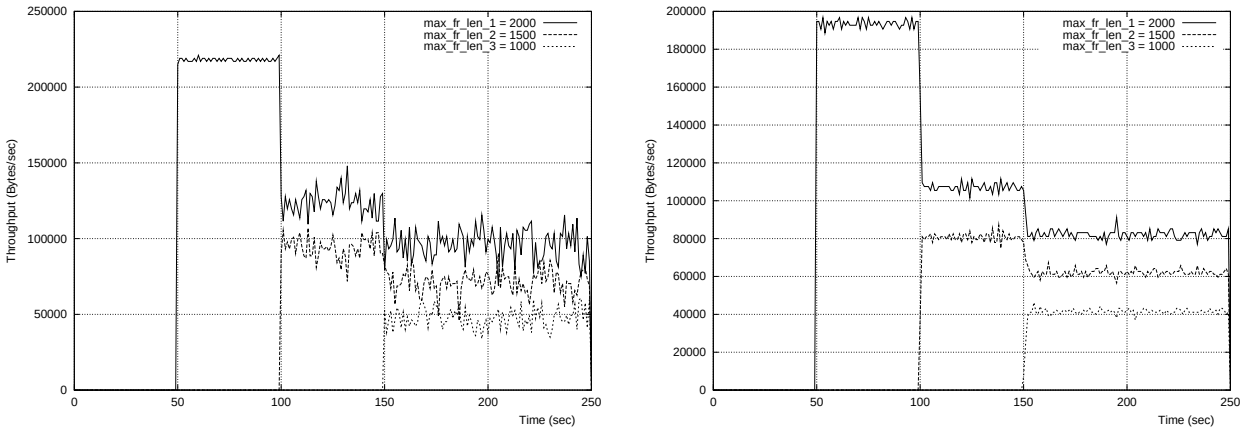
- Either to drop packets that exceed the maximum frame length assigned to a given WT (or simply configure it to limit its packet lengths), or
- To fragment packets that exceed the maximum frame length. As mentioned in chapter 3, this mechanism is actually used to increase transmission reliability, we will also use it for differentiation.

Fig. 3.6 shows how a WT would send a fragmented packet. We can see there are no RTSs between packet fragments, so a given WT keeps sending its packet fragments as long as it is receiving the corresponding ACKs. Meanwhile, all other WTs are "quiet". This leads us to almost the same data rate shares as if there were no fragmentation, unless there is fragment loss (thus a new RTS), due to a noisy channel for example. In the case of no fragment loss, both above cases can then be described by the former one, i.e. limiting packet lengths to a given value.

Simulations showed, as one would intuitively expect, that data rate shares are directly proportional to the maximum frame lengths allowed for each WT. That is, for a given WT_0:

$$\frac{B_0}{\sum_{i=1}^{N} B_i} = \frac{L_0}{\sum_{i=1}^{N} L_i} \tag{5.5}$$

where B_i and L_i are the throughput and the maximum frame length respectively of the WT_i.



(a) using UDP flows

(b) using TCP flows

Figure 5.16: Maximum frame length differentiation.

Results in Fig. 5.16 comply perfectly with (5.5). Using UDP, WT_1 has 4/3 the throughput of WT_2, whether during period 2 or during period 3, because it has the same ratio of frame lengths. The same rule applies for TCP flow throughput ratios. Note that the overall throughput of TCP is lower than UDP, due to TCP-ACK overhead.

Maximum frame length differentiation gives an infinite priority range, with no cost over system stability for high priority ratios P_i/P_j. Equation (5.5) shows no computing or inversion problems in order to apply it in real-time admission control with *DiffServ* or *end-to-end* optimization. Note that (5.5) applies to WTs with UDP flows as well as to TCP flows. This scheme also applies properly to give TCP flows priority over UDP one and vice versa.

5.4 Service differentiation with noisy channels

In the previous sections we only considered perfect channels, without noise (interference, fading or multi-path). This section provides a brief description of our simulation results with noisy channels. Simulations show that noise changes the performance of the four described schemes. Consider the *packet error rate* (PER) to be

$$PER = 1 - (1 - BER)^L$$

where BER is the *bit error rate*, and L is the packet length in bits. We first consider a channel with a BER of 10^{-6}. This leads to a low PER (for a 1100-bytes packet, the PER is 0.9%), and no considerable effects can be seen on any of the four mechanisms.

However, when we apply a $10^{-4}BER$ to all packets, simulations show that:

- With no priorities, the data rates of all WTs drop almost proportionally to the PER, and the data rate ratios remain the same, equal to unity, for both UDP and TCP flows respectively.
- With *backoff* differentiation, two effects can be seen: The first is the data rate drop due to packet errors. The second is that the data rate ratios increased dramatically, even with TCP flows (such differentiation couldn't be seen without channel errors). When a data frame is corrupted, the sending WT times out waiting for the corresponding ACK. The WT then increases its contention window for collision avoidance as if there was a collision. As different WTs increase their contention windows differently, because they have different priorities, they get different data rate shares. As a result, the priority that each WT gets depends directly on the channel conditions. This property is of course not desirable.
- With *DIFS* differentiation, the data rates drop proportionally to the PER, and the relative priority of each WT remains the same.
- With *maximum frame length* differentiation, long packets are more likely to be corrupted than short ones. This decreases the priority effect of the *maximum frame length* scheme.

5.5 Per-flow differentiation

In the previous sections we used differentiation on a WT basis, i.e. each WT has its own MAC sub-layer with its differentiated parameters. This fact limits the performance one would expect from the network. In fact, if a WT has different flows to different destinations, all flows share the same MAC sub-layer with equal parameters without being able to be differentiated.

Furthermore, we have seen in the previous sections that when we use closed-loop flows such as TCP, the differentiation effect is reduced due to the fact that different TCP-ACKs sent back from the shared AP with a single priority. Whether it is *DIFS*, CW_{min} or *backoff* differentiation, they all suffer from the single priority common destination (Fig. 5.11). Note that the *maximum frame length* differentiation cannot be applied to TCP-ACKs.

Apart from service differentiation, other facts also require require per-flow separation. Consider two data flows transmitted by a single sender WT_0 to two different destinations, WT_1 and WT_2. If WT_1 is located in a congested or noisy environment, the flow to WT_2 will also suffer from the situation, even if WT_2 is in a clear environment. In fact, the flow to both WTs share the same MAC sub-layer in WT_0. When WT_0 receives no ACK (or a CTS if RTS/CTS is used) for a frame transmitted to WT_1 either because of congestion or because of packet loss, it increases its CW to avoid future collisions and retransmits the frame after timeout. Meanwhile, frames to destination WT_2 have to be queued in WT_0 until the frame to WT_1 is retransmitted, possibly several times, until successfully received and acknowledged, or that the maximum retry counter is reached. This reduces the data rate to WT_2 considerably even though WT_2 suffers no congestion or noise on its side.

As we did in [85], in this section we investigate in detail the need for per-flow differentiation of closed-loop flows such as TCP (the second case cited above), and we show the need for queues separation after then.

5.5.1 Single queue per-flow differentiation

All the differentiation mechanisms described in the previous sections suffered one major common problem when trying to differentiate TCP flows: The AP always uses its own priority to send back TCP-ACKs to different WTs. This reduces the differentiation effect. In fact, whether we use *DIFS*, CW_{min} or *backoff* differentiation, WT_1 and WT_2 wait on average t_1 and t_2 respectively before transmitting their packets, and the resulting data rate ratio is proportional to t_2/t_1.

However, when we use TCP flows, the TCP-ACK transmission introduces an additional delay t_0 to the closed-loop flows. So the data rate ratio between WT_1 and WT_2 becomes proportional to $(t_2 + t_0)/(t_1 + t_0)$. When the AP is slow (low priority parameter), t_0 is high, which reduces the fraction $(t_2 + t_0)/(t_1 + t_0)$ considerably.

If we try to compensate the differentiation loss by increasing t_1 and t_2, we loose efficiency by making WT_1 and WT_2 wait more than necessary. Therefore, the AP must be as fast as possible, so t_0 is as small as possible, but t_1 and t_2 still have to compensate the small differentiation loss.

Another alternative is to make the AP use different priorities for different destinations, i.e. *per-flow differentiation*. Therefore, instead of waiting a fixed time t_0 before transmitting a packet, the AP should wait t_{01}, t_{02} or t_{03} according to the destination of the packet. The resulting data rate ratio between WT_1 and WT_2 becomes $(t_2 + t_{02})/(t_1 + t_{01})$ which is equal to t_2/t_1 when $t_{02}/t_{01} = t_2/t_1$, therefore no differentiation loss needs to be compensated. In other words, optimally, the AP should send back the TCP-ACKs with different priorities, proportional to the priorities of the destinations. In the following subsections, we are going to apply the *per-flow* differentiation to the mechanisms briefly described in the previous sections. The waiting times t_j and t_{0j} defined above would designate the expected waiting time when we apply:

- different $DIFS_j$ in DIFS differentiation.
- different CW_{minj} values in CW_{min} differentiation, and
- different backoff increase factors P_j in *backoff* differentiation.

Among these mechanisms, CW_{min} and *DIFS* differentiations show similar behavior when the AP uses per-flow differentiation. However, *backoff* differentiation does not show much effect due to the low number of collisions. Therefore the effect introduced by per-flow differentiation is even less visible. In the following we only show *DIFS per-flow* differentiation. The same reasoning applies to the other mechanisms.

Per-flow DIFS differentiation

We ran ten simulations with per-flow *DIFS* differentiation and we show the results in this subsection. The topology is the same as in Fig. 7.4 and the scenario is the same as in the previous sections. Table 5.2 shows the DIFS values that the AP and the WTs use respectively: The column AP_j shows the $DIFS_j$ values the AP uses to send the TCP-ACKs to WT_j, while the column WT_j shows the $DIFS_j$ values each WT_j uses to send its data packet. The last line describes the observations made on data rate differentiation using each set of DIFS values. The corresponding simulation results are shown in Fig. 5.17.

Set I shows the case where we wanted to check out the "elementary" effect of data rate differentiation due to the AP only.

Simulation showed no differentiation at all. Two possible reasons:

- AP_1 is relatively fast, but AP_2 and AP_3 are slow. As the TCP-ACKs are sent "serially" by the AP, the fast TCP-ACK still have to wait for the slow one and the overall AP speed is still slow.
- WT_1 TCP data are sent slowly. Accelerating the corresponding AP_1 would not accelerate the loop flow unless we accelerate WT_1.

Set II eliminates the second possible reason: Even though we accelerated WT_1, no differentiation took place, the AP is still globally slow, fast packets still have to be queued with slow ones to be transmitted.

Set III shows the situation where the WTs require differentiation, and the AP is fast enough. The resulting data rate differentiation is good. WT_2 and WT_3 get equal data rates, lower than the data rate WT_1 gets.

Set IV and Set V provide redundant information: the AP is slow in Set IV, therefore the differentiated WT_j do not result in differentiated data rates. In Set V, the differentiated AP does not result in flow differentiation because it still is globally slow.

Sets VI to X show the AP DIFS values, where the AP is always fast, so we can observe the per-flow differentiation effect. Set VI shows a situation similar to the one in Set III, we replaced the DIFS = 90 by DIFS = 150, so the differentiation is bigger. Keeping the AP with the same average speed (DIFS = 50), we changed the AP_j values to 30/50/70 as in Set VII. Even though the AP average speed is the same[2] as in Set VI, we observe more differentiation because the AP sends the TCP-ACKs with a speed proportional to the destination speed. We keep the same AP average speed, but we inverse them, 70/50/30, as in Set VIII. The resulting differentiation is lower than in Set VII, because the AP uses differentiated DIFS values, but in the wrong order.

Sets VII, IX and X show the case where WT_2 DIFS decreases from 150, 120 then to 100. All the rest of AP_j, WT_1 and WT_3 remain the same. The throughput ratio (differentiation) between WT_1 and WT_2 decreases also. One interesting observation is that, in Set X, AP_1 and AP_2 have very different values, as in WT_1 and WT_2. However, the throughputs are almost equal, which was not the case in Set IX. In fact, when we reduced WT_2 from 120 to 100, we invoked more utilization of $AP_2 = 50$, which led to a slower AP than in Set XI, so

[2]This is not really the same average, 50. WT_1 has a short DIFS, therefore it sends more packets than the other two WTs, so the $AP_1 = 30$ is more used than the $AP_2 = 50$ and $AP_3 = 70$, and the resulting AP average DIFS is lower than 50.

Table 5.2: Per-flow DIFS differentiation with TCP

	Set I		Set II		Set III		Set IV		Set V	
j	AP_j	WT_j	AP_j	WT_j	AP_j	WT_j	AP_j	WT_j	AP_j	WT_j
1	50	90	50	50	50	50	90	50	50	50
2	90	90	90	90	50	90	90	90	90	50
3	90	90	90	90	50	90	90	90	90	50
Obs.	Bad diff.		Bad diff.		Good diff.		Bad diff.		Bad diff.	
	Set VI		Set VII		Set VIII		Set IX		Set X	
j	AP_j	WT_j	AP_j	WT_j	AP_j	WT_j	AP_j	WT_j	AP_j	WT_j
1	50	50	30	50	70	50	30	50	30	50
2	50	150	50	150	50	150	50	120	50	100
3	50	150	70	150	30	150	70	150	70	150
Obs.	Good		Very good		Good		Bad		Very bad	

the differentiation is less visible. We should also note that the throughput of WT_3 remained almost the same in all the three simulation sets, VII, IX and X.

Mathematical models which better describe the differentiation behavior function of the differentiation parameters are under construction.

5.5.2 MAC sub-layers with per-priority queues.

In the previous subsection we saw that all packets are put in the same queue, independent of their priority. This introduced mutual interferences between priorities: When the AP serves a low priority flow, the AP global speed depends on the utilization of this flow. If it is highly used, the AP gets slow, and differentiation gets lower.

A possible solution is to assign to each priority (or to each WT) a different queue. Simulation showed a total independence between priorities: Even if a low priority flow exists, it won't slow down the AP (the shared node), and differentiation is much more clear.

Note that when using this approach with CW_{min} differentiation, the shared node (e.g. the AP in our scenarios) will be avoiding collisions less than other WTs do. In fact, when a single queue per MAC sub-layer is used, we just have one packet/node contending to access the channel, during a CW period. However, when a node uses n queues (for n TCP connections), we have n packets per CW period, as if in a shared node, collision avoidance decreases as the number of connections increases.

The remarks made above are not restricted to the differentiation mechanisms we simulated. They can be generalized to any shared node trying to differentiate between its outgoing flows.

5.6 Future work

Beyond the results presented in this paper, future work should address the following issues:

- *Mapping DiffServ to MAC differentiation* [92]. i.e. How must DiffServ parameters be mapped to MAC differentiation in order to get the optimal performances, including end-to-end ones.
- *Modeling the system,* for TCP flows and for per-flow differentiation.
- Parameters distribution between the WTs. i.e. how to establish the differentiation parameters between the WTs, in a distributed way, while taking the hidden nodes problem into consideration.

5.7 Conclusion

This chapter presents some results of our work on introducing service differentiation mechanisms into IEEE 802.11 MAC sublayer. We propose a scheme based on the contention window variation, another based on CW_{min} differentiation, a third based on *DIFS* differentiation and a fourth one based on the *maximum frame length* allowed to each wireless terminal. The first scheme consists of scaling the contention window according to the priority of each flow or user. We show via simulations that this scheme performs well with UDP but does not always work with TCP flows. The second mechanism, CW_{min} differentiation, partially solves the TCP problem but cannot offer strict priorities. The third mechanism, which consists of assigning different DIFSs for different priority WTs, showed better results as it can be applied to TCP and UDP flows and can provide strict/absolute

priorities. The fourth mechanism, which assigns different maximum frame sizes to different priorities, showed less complex results and works well with both kinds of flows too. The four different mechanisms do not introduce any efficiency loss: the data rate sums remain almost the same after introducing the priority schemes. On the other hand, the whole system is much less stable with *backoff* priority, but keeps the same stability level with CW_{min}, *DIFS* and *maximum frame length* priorities. We also drew some remarks on per-flow differentiation, where a common access point gives different priorities to different flows, which enhances TCP flows differentiation.

We show that in noisy environments, the *backoff* and *maximum frame length* schemes do not perform well anymore, while the performance of DIFS-based schemes remains unchanged. The data rate ratios increase for *backoff* mechanisms due erroneous backoffs. These ratios decrease for *maximum frame length* mechanism, but they keep the same values with *DIFS* mechanism which shows to have the best general properties among the four.

Flows must be differentiated on a per-flow basis, and furthermore separated into different priority queues to reduce frame delay interference. One issue characterizes wireless networks (PHY layer), which does not exist in wired networks (network layer): frame retransmissions. As long as a frame is not successfully received and acknowledged, the IEEE 802.11 MAC sub-layer keeps retransmitting it, making frames waiting in the same queue suffer from undesirable consequences.

As a final conclusion we would recommend to use the DIFS based schemes for service differentiation and that flows must be separated as much as possible, so the interference between flows is reduced.

In Chapter 6 we will describe the draft standard IEEE 802.11e which applies most of principles cited in this chapter. However, the draft standard does not apply queue separation, for a single class, to reduce inter-flow interference.

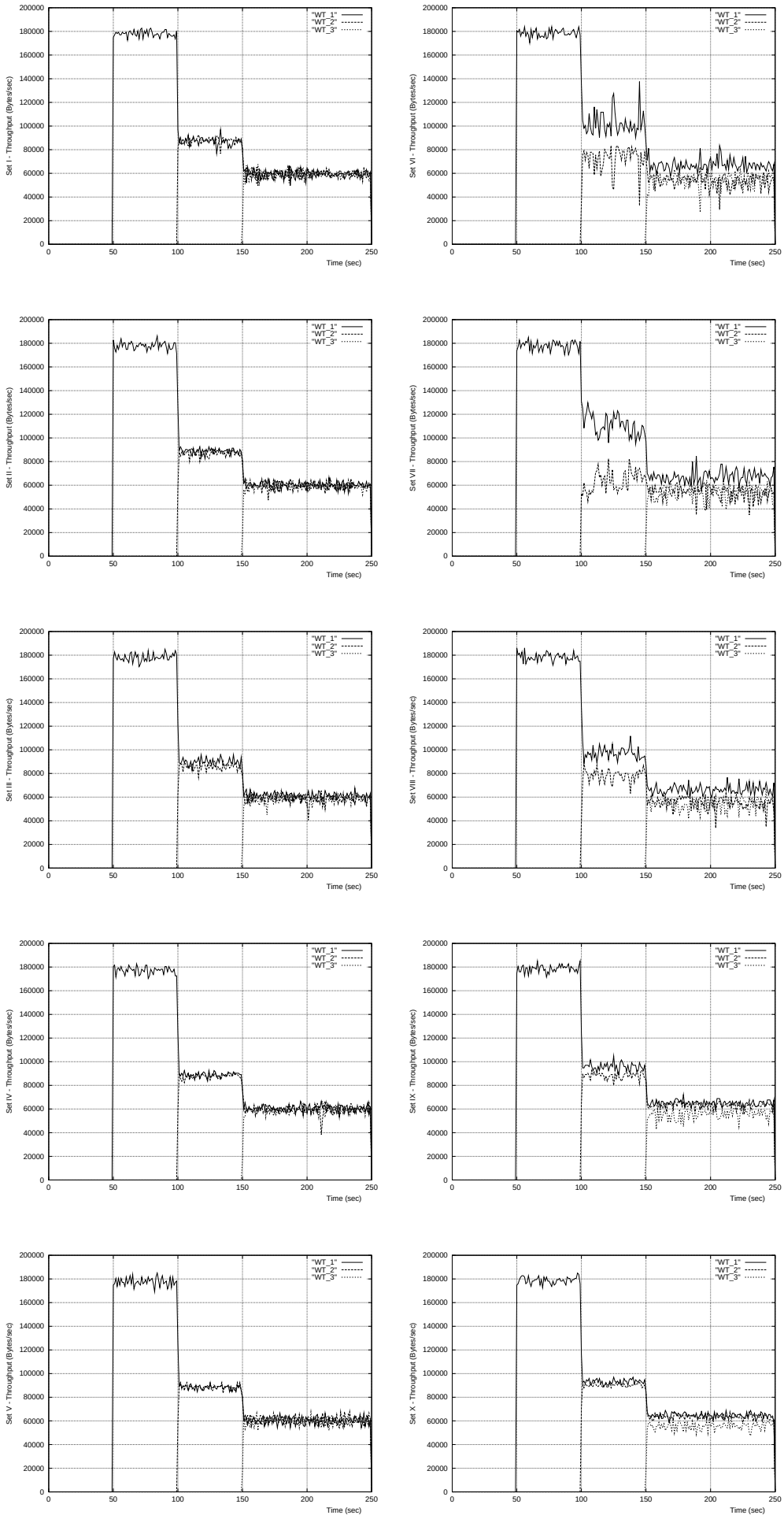


Figure 5.17: 'textitPer-flow DIFS differentiation

Chapter 6

Related work

Contents

In the recent years, many proposals were enriching the literature of QoS support for wireless networks [1, 2, 3, 4, 93, 94, 95, 96, 97, 98, 99, 12, 100]. They can be divided into many categories: Centralized ([12]) or distributed algorithms, for single-hop ([2]) or multi-hop ([3, 97]) networks, stateful ([97]) or stateless routing, QoS is IP-based ([93]) or MAC-based etc. An overview of some of these approaches can be found in [101].

In the following sections we are going to describe four of those approaches which are of interest to our previous work. Section 6.1 describes the current draft standard IEEE 802.11e for QoS extensions. Section 6.2 describes one of the approaches for real-time traffic support called *Black burst*. Section 6.3 describes a mechanism which supports service differentiation in multi-hop ad-hoc networks, trying to eliminate DIFS differentiation overhead introduced in the previous chapter. Last, section 6.4 describes two algorithms that estimate the channel occupation and therefore can be used to tune application parameters for service differentiation support and for admission control.

6.1 IEEE 802.11e draft standard [1]

The IEEE 802.11 task group E is currently considering extensions to the legacy 802.11 standard to support QoS. A brief description and performance evaluation of the proposed draft standard [1] can be found in [73]. The draft standard uses combinations of the differentiation mechanisms proposed in [83, 85, 84, 22] and detailed in the previous chapters. This section briefly describes the draft standard 802.11e by emphasizing the difference with the legacy 802.11 standard only.

The proposed standard introduces an enhanced DCF (EDCF) and a hybrid coordination function (HCF). Stations able to support 802.11e are called *enhanced stations* and may act as a centralized controller for other stations within the BSS. A centralized controller is called *hybrid coordinator* (HC) and typically resides in the AP. EDCF can be applied during CPs only, while CFPs can still be used alternated in time with CPs. HCF can be used in both CPs and CFPs.

6.1.1 Enhanced distributed coordination function (EDCF)

Figure 6.1 shows the main features of the IEEE 802.11e MAC sub-layer. One MAC sublayer supports up to eight traffic categories (TCs) mapped into eight independent backoff instances according to the following rules:

- Backoff starts decreasing, to contend to access the channel when it reaches zero, after detecting the channel idle for an AIFS (arbitration IFS) period of time. AIFS depends on the corresponding TC, and is at least equal to the legacy standard DIFS. In the legacy standard, all traffic flows wait for a $DIFS$ idle time before decreasing the backoff.

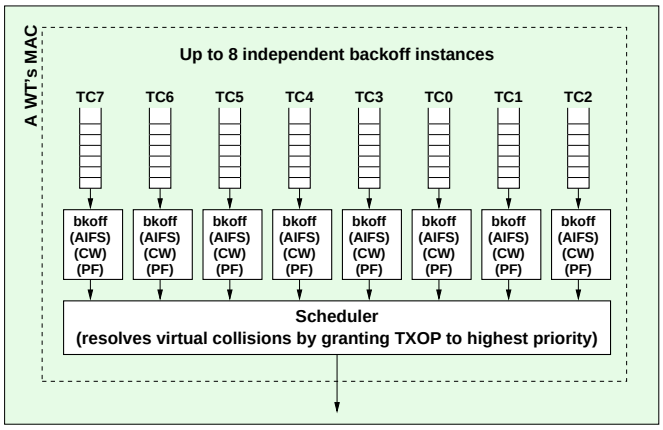


Figure 6.1: IEEE 802.11e MAC sub-layer.

- The backoff is drawn from a uniform distribution in the interval $[1, CW[TC]+1]$. $CW[TC]$ is the contention window of a given TC, and is differentiated using its lower bound value $CW_{min}[TC]$. In the legacy standard, all traffic flows have the same CW_{min} value.
- Upon each unsuccessful frame transmission, the CW[TC] is multiplied by a persistence factor PF[TC] which also depends on the TC. In the legacy standard, the CW is always doubled (PF=2) after each unsuccessful transmission.
- Upon several unsuccessful transmission, $CW[TC]$ keeps increasing but does not exceed $CW_{max}[TC]$ which also depends on the corresponding TC. In the legacy standard, a single CW_{max} value is used.

The $CW[TC]$ computation for a given TC can be written as:

$$newCW[TC] = \min\{((oldCW[TC] + 1) \times PF[TC]) - 1, CW_{max}[TC]\}$$

QoS parameters; $AIFS[TC]$, $PF[TC]$, $CW_{min}[TC]$ and $CW_{max}[TC]$ can be distributed by the hybrid coordinator using the *beacon* frames.

TCs can be seen as virtual stations inside a station. To deal with *virtual collisions* between different TCs whose backoff reached zero at the same time, a scheduler resolves this kind of collisions by granting access to the TC with the highest priority among the colliding TCs. The transmitted frame can still collide with other stations transmitting at the same time and the corresponding backoff values are therefore multiplied by their respective PFs.

6.1.2 Hybrid coordination function (HCF)

The HCF extends the EDCF access rules. HCF can operate in contention periods (CP) and in contention free periods (CFP). During CP, a station can transmit its frame when its backoff reaches zero or when it receives a special polling frame, CF-Poll from the HC. To send a CF-Poll the HC waits the channel to be idle for PIFS time, giving it priority over other contending stations waiting for $AIFS[TC] > PIFS$.

During the CFP, the HC may specify the transmission duration and starting time for the polled station using CF-Poll frames. As the HC uses $PIFS$ before sending its CF-Polls and the polled station waits SIFS time before sending its frame, none of the other stations can get access to the channel since they wait for $AIFS > PIFS > SIFS$.

The mechanism used for stations to request for polling is somehow similar to that used in HiperLAN-2 on the random access channel RCH. It is called *controlled contention* in 802.11e. The HC starts the controlled contention period by sending a special control frame. This forces legacy stations to update their NAV and remain silent until the end of this interval. The control frames specifies a number of controlled contention opportunities and a filtering mask containing the TCs in which resource requests may be placed. A station with queued frames matching the TC filter chooses one opportunity interval and transmits a resource request frame containing the TC and transmission duration. The HC then generates another control frame with a feedback field to acknowledge resource requests so that contending stations can detect possible collisions, similar to the information on the access feedback channel (ACH) used in HiperLAN-2 and to the mechanism introduced in GAMA (group allocation multiple access)[99, 102].

When BSSs overlap, consecutive polling frames from different APs may collide, degrading the system performance of both BSSs. Several solutions are under discussion, among which is dynamic frequency selection (DFS), which is also used in HiperLAN-2.

Performance evaluation of several scenarios with different EDCF parameters, combined EDCF and HCF, and with overlapping BSSs can be found in [73].

6.2 Black burst [2]

To support the requirements of real-time applications, such as bounded end-to-end delays,[2] proposes a multiple access scheme called *Black burst* (BB). BB can be overlaid on IEEE 802.11 implementations without requiring changes to the access procedures of data nodes. Real-time nodes require few changes to the IEEE 802.11 standard. With this scheme stations with real-time traffic contend to access the channel by sending pulses of energy which durations are proportional to the delay a frame observed before the channel became idle. This ensures collision-free access to the channel and gives real-time frames priority over data frames. The random access scheme is turned off and substituted by the BB. The performance of the scheme is claimed to approach perfect time division multiplexing via a distributed algorithm. One main feature of BB is that it applies only to networks with no hidden nodes.

In [2], DIFS, PIFS and SIFS are substituted with t_{long}, t_{med} and t_{short} ($t_{long} > t_{med} > t_{short}$). Stations with real-time flows wait t_{med} for the channel being idle before trying to transmit their frames, while nodes with data frames wait for t_{long}, with no chances to grab the channel before real-time frames.

Right after t_{med}, and instead of transmitting its frame, a node with real-time frames starts jamming the channel with its BB which length is an increasing function of the contention delay experienced by the node measured from the instant of the attempt to access the channel until the channel became idle for t_{med}, i.e. the start of the transmission of the BB. The node starts transmitting its BB and senses the channel for a period t_{obs} to determine whether it had the longest BB, in which case it starts transmitting its frame and schedules the next transmission to t_{sch} in the future, where t_{sch} is the same for all nodes. Other nodes who failed the contention (shorter BBs) wait for the channel to become idle again for t_{med} to transmit longer BBs. At the end, the mechanism reaches a steady state where nodes with real-time frames appear to share the medium on a *time division multiplexing* basis in a distributed manner, with no need for synchronization or explicit slot assignment.

Figure 6.2 shows an example of BB operation between two nodes contending to send their real-time frames on the channel.

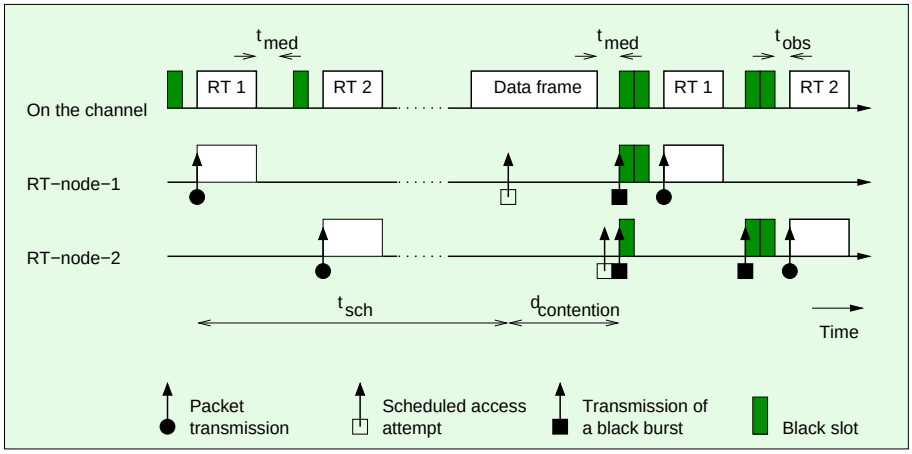


Figure 6.2: Time diagram illustrating a BB contention example.

Nodes 1 and 2 have their attempts delayed by a data packet transmission, after which the channel goes idle for t_{med}. Therefore both nodes start jamming the channel with BBs. Node 1 transmits a longer burst as it has been waiting longer, and therefore wins the contention. It observes the channel idle for t_{obs} to realize it won the contention, and transmits its frame thereafter. Node 2 waits the channel to become idle for t_{med} again. At this time node 2 sends a longer BB reflecting the longer delay it has been waiting to access the channel. As its BB is the longest, node 2 transmits its packet after sensing the channel for t_{obs} time.

This scheme gives real-time packets absolute priority over data packets. Furthermore, contention among real-time packets converges to a round-robin-like distributed scheduler. BB can further be enhanced to support real-time sessions with different bandwidth requirements. For instance, one can imagine different t_{sch} scheduling valued for different nodes, resulting in different shares in the available data rate.

The authors also proposed a method called *chaining* which aims to reduce the number of contending nodes in a wireless LAN. The idea consists of forming a chain of transmitting nodes. Each time a real-time node transmits its packet, it polls/invites another real-time node to transmit thereafter. The invited node has to transmit its packet after t_{short} so other nodes won't start transmitting their BBs before, and the chain stays in-sequence. Chains enhance the throughput efficiency. Each time a chain breaks into sub-chains, other nodes may still contend with BBs, reducing the channel efficiency. Chains may also be concatenated. Chains construction and concatenation algorithms must avoid closed loops, and must avoid long number of packets in a chain so data packets can still have reasonable access to the channel. Concatenation constraints mainly rely on the tail node which must take chain sizes into consideration.

Simulation showed that BB can handle more real-time nodes than CSMA/CA, with stable data and real-time traffic operation, due to the absence of collisions. Furthermore, chaining increases the number of real-time nodes. Specifically, the maximum number of real-time nodes that can be supported increases with the number of nodes in a chain, because it reduces the contention overhead. From the delay point of view the authors show that, for several fractions of real-time load, BB offers lower (and bounded) delays and lower jitters than CSMA/CA, even at higher traffic loads. However, chaining does not bring considerable enhancement for packet delays.

6.3 Busy tone priority scheduling (BTPS) [3]

Ad-hoc networks are typically multi-hop networks, where each node does not necessarily hear every other node. Location dependent contention and hidden terminals make priority scheduling in multi-hop networks significantly different from that in wireless LANs [3]. Black burst cannot be applied in multi-hop ad-hoc networks where hidden terminal exist, as cited in the previous section. Furthermore, in multi-hop networks *DIFS* differentiation, *backoff* differentiation and CW_{min} differentiation [1, 83] are claimed to offer sub-optimal results by the authors in [3], who propose a new scheduling scheme using two narrowband signals to ensure medium access for high priority source stations. The proposed protocol is called BTPS (busy tone priority scheduling) and will be briefly described in this section.

The example in Fig. 6.3 shows how BTPS operates in a 3-hop scenario.

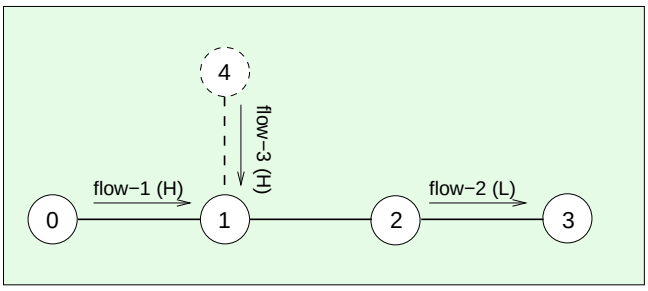


Figure 6.3: A simple BTPS scenario.

Node 0 has high-priority packets to send to node 1, while node 2 has low-priority packets to send to node 3. The key point is that node 2 should be aware of the high priority packets at node 0 so it defers its low-priority transmission. Moreover, when node 0 has no packets to send, node 2 should maximize its own throughput.

This is achieved by using two narrow-band busy tone signals, $BT1$ and $BT2$: When node 0 has high priority traffic to transmit, it starts transmitting $BT1$ each M time slots (during DIFS and backoff periods) before it acquires the channel, where M is a parameter of BTPS. Each node that hears $BT1$, i.e. node 1 in our example, starts transmitting $BT2$ each M time slots (See Fig. 6.4). This ensures that node 0 will transmit its high-priority packets first. Each node that hears $BT1$ or $BT2$ (but the source node, node 0) differs its transmission for some duration.

When node 0 has no high-priority packets to send, this mechanism also ensures no efficiency loss since node 2 can get the whole bandwidth, with no additional overhead due to BTPS.

Consider now a fifth node, node 4, hidden to node 2, with high-priority traffic contending with node 0 to send its traffic to node 1 (dashed items in Fig. 6.3). A collision between high-priority packets may occur, and therefore node 2 may grab the channel to send low-priority packets. BTPS avoids this situation since node 2 is always aware of high-priority packet transmissions, whether colliding or not. Nodes 0 and 4 will detect the collision after *CTS-timeout*, during which node 2 must not access the channel. This is why the time deferal, upon hearing BT1 or BT2, to transmit low-priority packets is required to be equal to *CTS-timeout*.

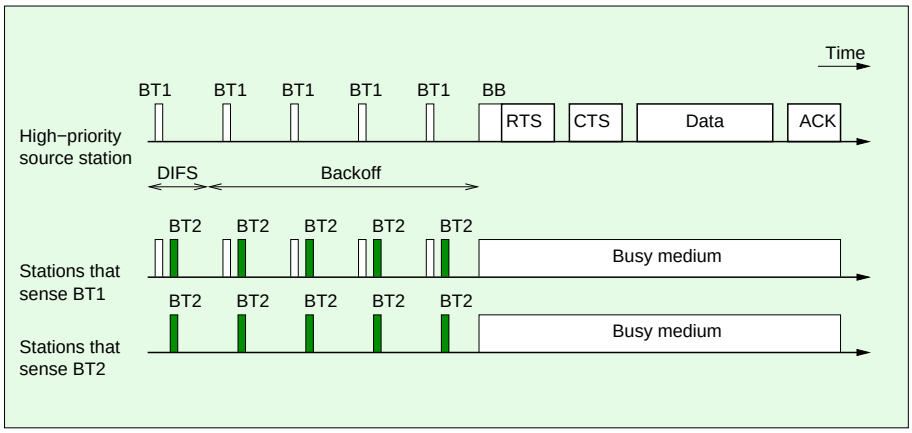


Figure 6.4: Behavior of BTPS protocol.

It may be the case that, while transmitting $BT2$, node 1 misses a high-priority packet (or its RTS) transmitted from node 4. To deal with this, a short 2-slot *black burst* (Fig. 6.4) is transmitted prior to the actual data (or RTS) so the receiving node (node 1) can keep listening to data instead of switching to $BT2$ transmission.

The authors also show performance evaluation comparing BTPS to our *DIFS* differentiation and to the legacy standard IEEE 802.11. It mainly shows that both differentiation mechanisms provide satisfactory differentiation relative to IEEE 802.11, but BTPS shows less overhead and more efficiency than *DIFS* differentiation from the throughput point of view, especially for short data packets where *DIFS* differentiation overhead becomes considerable.

Further advantages of BTPS over *DIFS* differentiation are:

- Absolute priorities are possible, without the DIFS overhead needed to ensure absolute priorities.
- There is no priority decrease upon collisions between high-priority packets.

However, the drawbacks of BTPS relative to *DIFS* differentiation are:

- BTPS needs two out-of-band narrow frequency bands, requiring relatively more complex hardware.
- BTPS supports only two priority classes.

Furthermore, since radio signal attenuation typically depends on the frequency, using out-of-band signals leads to *busy tone* ranges different from the ranges reached with data. This fact leads to sub-optimal results for BTPS.

6.4 Virtual MAC (VMAC) and virtual source (VS) algorithms [4]

Unlike the approaches in the previous sections, the algorithms shown here do not aim to provide service differentiation, but to provide estimated performance metrics, used at the application level for admission control. The authors in [4] propose two novel algorithms that extend the DCF in IEEE 802.11 in order to support service differentiation, in a distributed way: Virtual MAC and virtual source algorithms. VMAC passively monitors the radio channel and establishes local estimations of delays, jitters, packet collisions and packet losses taking into account both local conditions and interference caused by external effects or overlapping cells. VMAC estimations are passive in order to avoid putting additional load on the channel.

Using the estimations of VMAC, VS tunes the application parameters in response to the dynamic radio channel conditions and determines whether a new session with a particular service level requirement should be admitted or not.

The authors first investigate CW_{min} differentiation for small number of contending nodes. Simulation showed a good separation, from the delay point of view, between the two classes: delay-sensitive CBR flows and best-effort TCP flows. Then they check if this separation is maintained across a wide range of traffic loads, which would increase the interference among traffic classes. In fact, CW_{min} differentiation offers statistical, non-deterministic service separation. Delays remained differentiated, even at high loads. However, they increase with traffic load for both high-priority and low-priority flows. Throughput is not completely left for high-priority flows when the number of high-priority flows reaches the channel saturation. A small part of the available data rate is shown to still be used by TCP (low-priority) flows. Real-time applications do not only require that high-priority flows get better services then low-priority-flows. They mainly require bounded delays and absolute (not relative) priorities which VMAC and VS try to estimate.

To estimate the channel free capacity, the idle channel time after DIFS is measured. VMAC and VS operate in parallel to the real application and the MAC at a node and estimate the service level. They emulate the behavior of a real traffic source and its MAC by generating virtual packets. However, no actual data transmission is done. Packets are time-stamped and placed into a *virtual buffer*, scheduled for transmission on the channel (after backoff) as when using a real MAC. However, instead of sending the virtual packet, the VMAC estimates the probability of collision if these virtual packets were sent. In case of collision (detecting a packet on the channel), VMAC backs off as a real MAC would do. If no collision occurs, VMAC estimates the total packet delay and its overhead. All other MAC aspects are emulated, e.g. retransmissions, CW increase etc.

VMAC continuously keeps track of packet delays, packet loss rates and collisions. These estimates can be used by the application before the actual transmission of a real packet. The estimated delays showed to be very close to the simulated ones on a wide range of traffic loads and in the saturation region. With or without differentiation, simulated delays and jitters are close to their estimated values. Thus, the approach is suitable for evaluating the admissible capacity of the channel for real-time traffic.

Delays due to packet collisions and retransmissions are called *MAC delays* and are estimated at the VMAC level. However, a packet observes higher delays due to packetization, interface queuing, which are also function of the packet sizes, bit rates etc. Those are not taken into consideration by the VMAC, but at the VS level which estimates what the application would observe helping to optimize the application performance.

For instance, considering a constant bit rate (e.g. audio application), increasing packet rates decreases packetization delays but also increases overhead, collisions and CW sizes. Therefore, the overall delay is a tradeoff between packetization delays and MAC layer delays. VS can build estimated delay graphs for all possible packet rates, which helps the application choose the optimal point reducing the packet delays it observes. This estimated delay graph depends on the channel conditions and existing background traffic flows.

The applicability of VMAC and VS go beyond application tuning. Their estimates can also be used for admission control for supporting real-time traffic flows using differentiated MAC. Simulation showed that applying VMAC and VS mechanisms maintained the observed real-time packet delays below a given value most of the time in a random simulation scenario, all real-time flows that would break other flows' constraints being rejected to maintain stability.

Last, we should note that VMAC and VS can also be coupled with our differentiation mechanisms proposed in the previous chapter to provide and efficient admission control for service differentiation.

Part III

Enhancing IEEE 802.11 in noisy and in congested environments

Chapter 7

Enhancing IEEE 802.11 performance in noisy environments

Contents

7.1 Introduction

Many wireless medium access control protocols use CSMA. When a frame collides with another, both are retransmitted at future random times, and the contention windows are increased. A host detects that a frame collided if it does not receive any acknowledgment from the destination before a given timeout. However, in wireless environments, frame losses can be due to collisions or to noise interference, but a source cannot identify the real cause. So it increases its contention window for both events. This is obviously sub-optimal: Contention windows should not be increased to avoid collisions when losses are due to noise. This mechanism introduced undesirable impact of noise on service differentiation, namely *backoff* differentiation in Chapter 5, and it lacks fairness and efficiency as noted in [22] and shown in the next sections.

In this chapter we start by identifying and analyzing the problem, then we propose a basic scheme to statistically distinguish collision losses from noise losses. We then search for an optimal solution that adjusts the contention window values to reduce the overhead by taking the noise effect into consideration, while still avoiding collisions. We evaluate our scheme by simulation, comparing it to the current CSMA scheme and theoretical optimal ones.

Section 7.2 shows the motivations for this work, through actual tests and simulations, after which we analyze the problem and the advantage of solving it in section 7.3. Finally we propose our solution in section 7.4. Section 7.5 shows many future work and optimization to be done, then section 7.6 concludes this chapter.

7.2 Motivations

In this section we start by showing how does CSMA/CA behave in a noisy environment, using simple simulation scenarios in *NS* [88]. Then we proceed by citing two practical situations.

The first simulation briefly shows how do CWs behave in presence of noise. Figure 7.1 shows the contention windows distribution of a WT coexisting with other WTs, using a clear channel. As the number of WTs increases, CW sizes get lager more frequently in order to avoid collisions. If we consider the case of two WTs, Fig. 7.2 compares the CW distributions before and after applying a 10% packet error rate (PER). After applying noise to the channel, we can see that the CW distribution is almost similar to the case of five WTs in a clear channel: The WTs are trying to avoid high collision rates by increasing their CW sizes, uselessly.

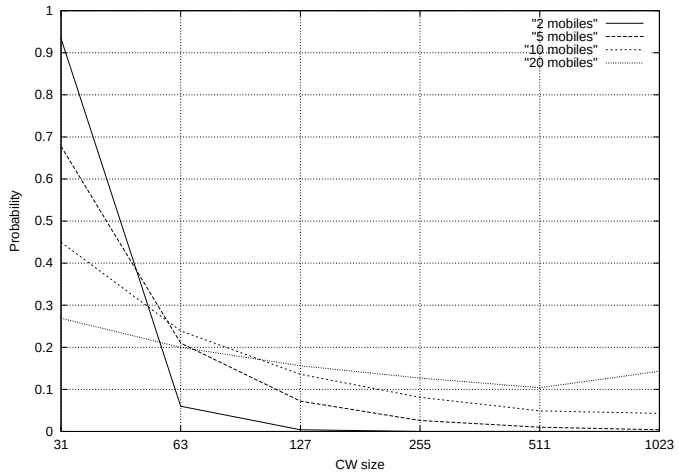


Figure 7.1: Contention windows distribution using a clear channel.

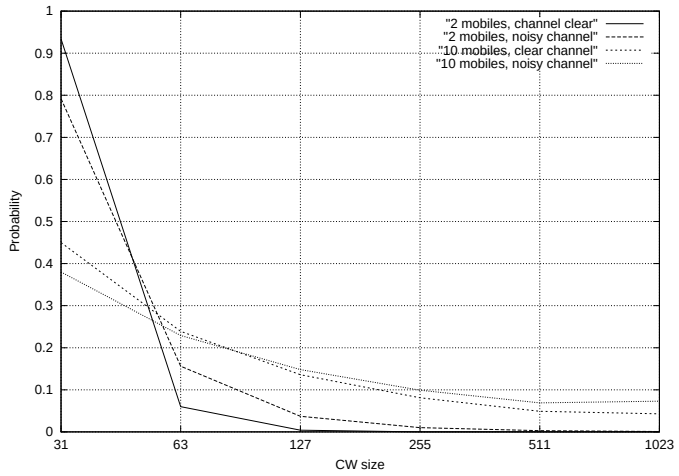


Figure 7.2: Contention windows distribution using a noisy channel (PER=10%).

The next simulation shows the side effects of increasing the CWs due to noise losses. The network topology is shown in Fig. 7.3. Two WTs are placed at equal distances from the access point (AP), which is wire-connected to a fixed host S. No possible congestion is possible on the wired links. We should note that this also applies to ad-hoc networks, when two WTs are communicating to a common one (instead of the AP).

In the first scenario, two traffic sources are placed in S. At second 50, the first traffic source starts transmission to WT_1 via the AP. At second 150, the second traffic source starts its transmission to WT_2, through the same AP. Each source sends 1100-Byte UDP (User Datagram Protocol) [87] frames at 0.005 seconds intervals. The transmission periods have been slightly changed from 0.005 to avoid transmission synchronization, which would lead to unfair shares. Results are shown in Fig. 7.4 and table 7.1.

When there is no noise on the channel (first 2 rows of table 7.1), the traffic toward WT_1 can obtain the whole available data rate exclusively during the first period. Few collisions with other routing packets are observed, which caused the contention window to increase a few times. During period II, the traffic toward WT_2 turns on, and the available data rate is equally shared. No significant collisions nor contention window increases are observed, as the AP is the only node transmitting.

If we consider the case where noise can corrupt the transmitted frames at a high constant BER, such as 10^{-4}, several observations can be made (refer to the middle 2 rows of table 7.1). Even though there is no reason for more collisions than in the case with no noise, the contention windows reached high values very often, degrading the performance dramatically. In fact, receiving an ACK for a transmitted frame is the only way for a WT to know that the frame was received successfully. When no ACK is received, CSMA/CA mechanism assumes that a collision took place, and doubles its contention window to avoid more collisions in future retransmissions. In our case, there is no considerable collisions to cause such CWs increase as only one node is using the channel at a time. Obviously, one can see that the high contention window values are due to the introduced noise: Noise

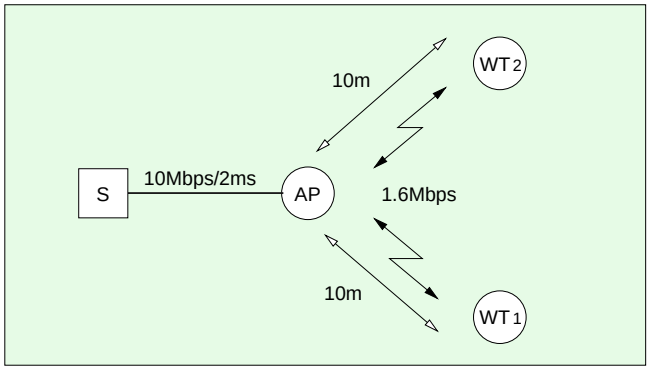


Figure 7.3: The simulation network topology.

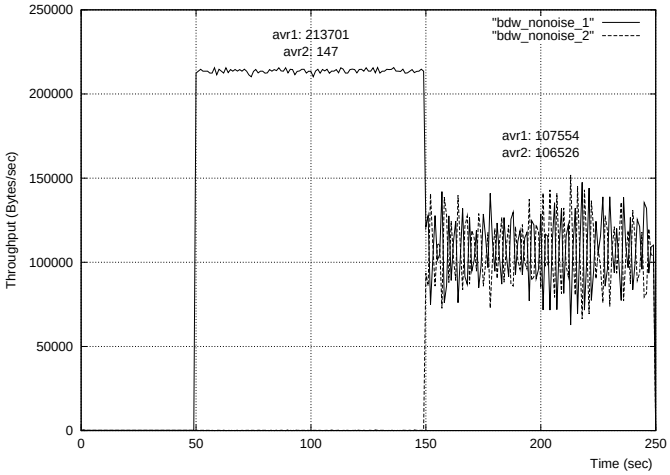


Figure 7.4: Scenario 1 throughputs.

can corrupt either the transmitted data frame or the corresponding ACK (with a much lower probability). The data source times out waiting for the ACK and assumes a collision took place, which is not really the case, and it doubles its contention window. Increasing the contention window, which is supposed to reduce collisions, introduces several side effects such as lower data rates and higher delays. In fact, with a $BER = 10^{-4}$ and 1100-byte long frames, the PER is 58% ($PER = 1-(1-BER)^L$, where L is the frame length in bits). However, the data rate drop from 213701 Bps to 64867 Bps is 69%, considerably higher than 58%, due to high CW values. This noise effect also helped the *backoff* differentiation mechanisms in Chapter 5, where TCP flows tend to have low CW values. But noise forced CW increase, showing more differentiation effect.

Consider the case where only frames directed to WT_1 may be corrupted by noise (last 2 rows of table 7.1). Two new interesting observations can be made. First, as both of the flows share the same MAC sub-layer which does not distinguish between flows, a CW increase due to flow 1 (and the following retransmissions) also makes flow 2 packets wait longer in the interface queue. This reduces the data rate offered to WT_2, from 106526 Bps to 48721 Bps even though none of its frames are corrupted by noise. This observation was also made in [22], which motivated the flow separation mechanism, without dealing with "noise-or-collision" identification. Second, the flow directed to WT_1 increased from 31759 Bps (in *global noise*) to 51784 Bps (in Noise/1), even though its frames are corrupted at the same rate in both cases. This is because the traffic to WT_2 sees no more noise, and so it does not slow down (by increasing the CW and retransmitting frames) the shared AP MAC sublayer anymore. The AP increases its CW less frequently, retransmits less frames, therefore frames to WT_1 and WT_2 are sent faster than in the *global noise* situation, and the data rate gets better.

As mentioned in the Chapter 1, different locations of a given area have different signal strengths, hence different BERs. We can observe this by measuring the round trip time (RTT) of frames between a WT and another. RTT variation reaches hundreds of microseconds from one location to another. This is due to some factors such as:

- Forward error correction (FEC) processing time.

Table 7.1: Data rates and contention window distribution for scenario 1.

		Data rates (Bps)		Contention window size distribution					
		WT_1	WT_2	31	63	127	255	511	1023
Clear	I	213701	147	19942	129	8	1	0	0
	II	107554	106526	19951	53	4	0	0	0
Global noise	I	64867	124	6597	3789	2341	1386	827	1019
	II	31759	31957	6494	3722	2286	1405	883	1069
Noise/1	I	66676	141	6669	3740	2240	1331	829	1024
	II	51784	48721	9717	2888	1729	1035	633	807

- Packet processing time at the receiver.
- Packet retransmission on the MAC sub-layer, in case the frame was not acknowledged.
- The increase of CW values when a frame is not acknowledged.

As mentioned before, our work deals with optimizing the last point only.

FHSS and DSSS introduced in the previous section both operate in the ISM (Industrial, Scientific and Medical) frequency band, 2.4 GHz, but were not designed to operate without interfering with each other. UCLA simulation results in [103] show such a situation: A Bluetooth (using FHSS) slave operating close to a WLAN AP (using IEEE 802.11, DSSS) causes a very high frame drop rate, up to 46%, and high access delays on the WLAN AP side. In such a common situation, CW increase avoidance seems mandatory and can compensate a considerable part of the lost data rate.

The data rate drop due to noise can be enhanced using FEC mechanisms and is out of the scope of this paper. However, the side effect of noise, which is the useless increase of CWs, can be avoided, and the overall performance can be enhanced. This is detailed in the next sections.

7.3 Problem analysis

In this section we analyze how is the data rate related to the CW increase. This increase can be due to collisions or to noise, hence we can deduce how data rate is related to the frame loss rate.

Figure 7.5 shows the case of a single mobile, transmitting a single frame and receiving the corresponding ACK, with the corresponding IFS and backoff. Let T be the overall time and L the frame size. Then the useful data rate (udr) is:

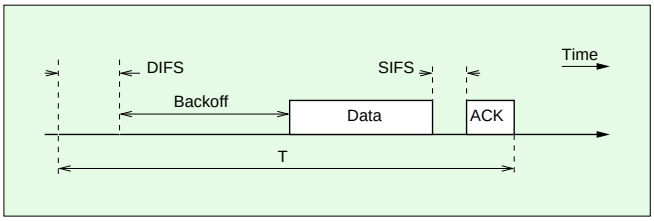


Figure 7.5: Packet transmission timing.

$$udr(L) = \frac{1}{T} \times L = \frac{1}{T_{DIFS} + T_{bkf} + T_{pkt} + T_{SIFS} + T_{ACK}} \times L$$

where $T_{DIFS}, T_{bkf}, T_{pkt}, T_{SIFS}$ and T_{ACK} are the DIFS time, the backoff time, the frame transmission time, the SIFS time and the ACK transmission time respectively.

On the long run, the average useful data rate is:

$$\begin{aligned} E[udr(L)] &= E[\tfrac{1}{T_{DIFS}+T_{bkf}+T_{pkt}+T_{SIFS}+T_{ACK}} \times L] \\ &= L \times E[\tfrac{1}{T_{DIFS}+T_{bkf}+T_{pkt}+T_{SIFS}+T_{ACK}}] \\ &= L \times E[\tfrac{1}{K+T_{bkf}}] \end{aligned}$$

where $K = T_{DIFS} + T_{pkt} + T_{SIFS} + T_{ACK}$ is a constant.

$$E[udr(L)] = L \times \sum_{i=1}^{1023} \frac{1}{K + T_i} \times P_s(bkf = i) \tag{7.1}$$

where T_i is i time slots duration, and $P_s(bkf = i)$ is the probability of a successful frame transmission (without collision) at backoff value i. $P_s(bkf = i)$ depends on the CW values and their respective probabilities:

$$P_s(bkf = i) = \sum_{j; 2^{4+j}-1 \geq i} \frac{1}{2^{4+j} - 1} \times P_s(CW = 2^{4+j} - 1)$$

where $P_s(CW = c)$ is the probability of a successful frame transmission with CW value c which is the main complex component of this equation. For instance:

$$P_s(CW = 63) = P(CW = 31) \times P_c(CW = 31) \times [1 - P_c(CW = 63)]$$

where $P(CW = l)$ is the probability of having $CW = l$ and $P_c(CW = l)$ is the probability of a collision when $CW = l$. These depend directly on the CW values of other WTs contending to access the channel. The last paragraph of this section shows how to compute P_s for all CW sizes, in the case of two WTs, and we use the result to compute $E[udr(L)]$ shown here.

Fig. 7.6 shows that at $PER = 0$, $E[udr(L, PER)]$ converges toward 1 when L grows indefinitely, i.e. the overhead of a frame transmission becomes negligible when the frame sizes increases. We can also see the amplitude of the useful data rate drop for a given PER due to the increasing CWs, regardless of the frame retransmission. For instance, for $L = 100B$ and $PER = 0.1$, the $E[udr]$ is almost 0.45, this means 10% less than what $E[udr]$ is in a clear channel ($PER = 0$) due the the extra delays introduced only. Therefore, if we find some means to avoid increasing the CWs when a frame is lost due to noise, we can gain up to 10% of the data rate, regardless of the lost frames.

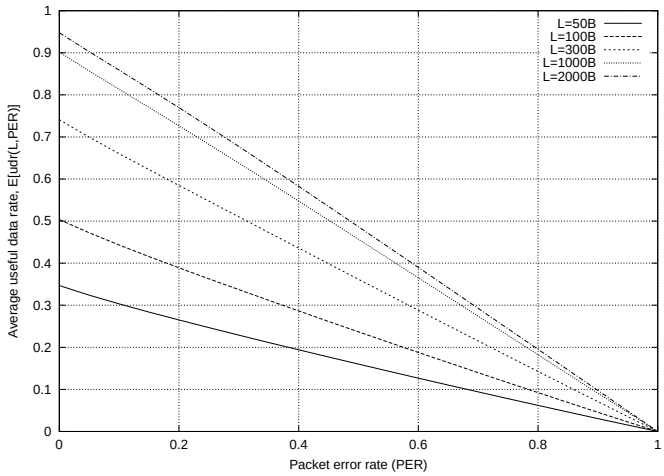


Figure 7.6: Average useful data rate ($E[udr]$) function of PER and L(in Bytes), using a 2Mbps data rate.

For all L values, the curves tend to zero when the PER tends to one, i.e. when PERs get high, CWs keep increasing and the overhead of a frame transmission becomes predominant.

Last, we should note that as L increases, the curve turns from convex-up to convex-down, especially for low PER values. That means, once more, that frame losses causes more overhead for short frames than for long ones, from the "CW-increase" point of view.

Computing $P_s(CW = n)$:

In this paragraph we compute the different probability values needed, i.e. :

- $P(CW = i)$: the probability of having a CW size i.
- $P_s(CW = i)$: the probability of transmitting data successfully with a CW size i.
- $P_c(CW = i)$: the probability of having a collision with a CW size i .

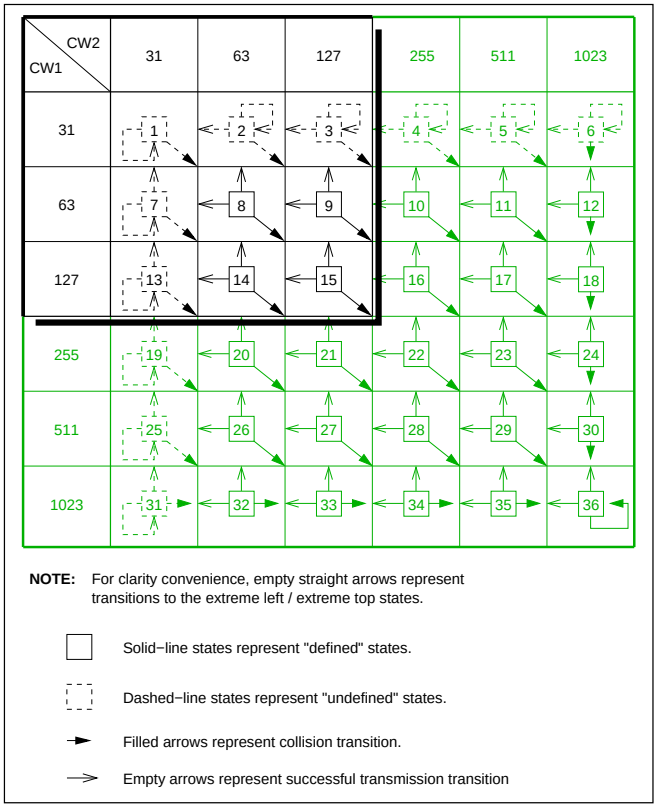


Figure 7.7: CWs transition diagram.

In the following we only deal with the two WTs case. Similar but more complex analysis can be done for a higher number of WTs.

As we proceeded in Chapter 5, we will use the transition diagram of Fig. 7.7.

The combination of the CW values of two WTs gives a 36-state diagram. The transition between different states is triggered either by a collision (represented with a filled arrow) or by a successful transmission (represented by an empty arrow). For clarity reasons, empty straight arrows represent transitions to the extreme left / extreme top states and not to adjacent states. e.g. in state 10, if WT_2 succeeds accessing the channel, the transition is made to state 7, not state 9.

After each collision, both of the CWs are increased, therefore their values are known and the corresponding state is *defined* as we called it (represented with solid lines). In a *defined* state ($CW_1 = b, CW_2 = d$), the collision probability $P_c(b,d)$ is given by:

$$P_c(b,d) = \frac{1}{max(b,d)} \tag{7.2}$$

and the success probability $P_s(b,d)$ of WT_1 is given by:

$$P_s(b,d) = \begin{cases} 1 - \frac{1}{2} \times \frac{b+1}{d} & \text{if } b \leq d \\ \frac{1}{2} \times \frac{d-1}{b} & \text{if } b > d \end{cases} \tag{7.3}$$

However, after a successful transmission, the winning WT resets its CW to 31, and the other keeps reducing its backoff time. We called the corresponding state *undefined*, represented with dashed lines. An *undefined* state has one reset CW size which bounds the new backoff time, and the second backoff time (of the other WT) depends on the previous states. The corresponding CW value can be replaced by the estimated CW size, taking into consideration the previous states probabilities and the corresponding transition probabilities. Therefore P_c and P_s defined above can be applied to these CW values.

When we apply these equations to each of the states of Fig. 7.7, we obtain a set of equations. Solving this set gives us the probability of each state. To simplify the task, we reduced the transition diagram to the nine upper left states (hi-lighted in Fig. 7.7). This approximation is acceptable since high CW values are rarely reached when deploying two WTs only.

Let P^i be the probability of state i in Fig. 7.7, P_c^i the probability of a collision in state i, and P_s^i the probability that WT_1 makes a successful transmission in state i. According to Fig. 7.7 we have:

$$P^8 = P^1 \times P_c^1$$
$$P^9 = P^2 \times P_c^2$$
$$P^{14} = P^7 \times P_c^7$$
$$P^{15} = P^8 \times P_c^8$$

$$P^2 = P^2 \times P_s^2 + P^8 \times P_s^8 + P^{14} \times P_s^{14}$$
$$P^3 = P^3 \times P_s^3 + P^9 \times P_s^9 + P^{15} \times P_s^{15}$$
$$P^7 = P^7 \times (1 - P_c^7 - P_s^7) + P^8 \times (1 - P_c^8 - P_s^8) + P^9 \times (1 - P_c^9 - P_s^9)$$
$$P^{13} = P^{13} \times (1 - P_c^{13} - P_s^{13}) + P^{14} \times (1 - P_c^{14} - P_s^{14}) + P^{15} \times (1 - P_c^{15} - P_s^{15})$$

$$\sum_i P^i = 1$$

The solution of this set of equations is:

$$\begin{array}{lll} P^1 = \frac{1}{E}; & P^2 = \frac{B}{E}; & P^3 = \frac{D}{E}; \\ P^7 = \frac{A}{E}; & P^8 = \frac{P_c^1}{E}; & P^9 = \frac{P_c^2 \times B}{E}; \\ P^{13} = \frac{C}{E}; & P^{14} = \frac{P_c^7 \times A}{E}; & P^{15} = \frac{P_c^1 \times P_c^8}{E}; \end{array}$$

where:

$$A = P_c^1 \times \frac{(1-P_s^2)\times(1-P_c^8-P_s^8)+P_s^8\times(P_c^2-P_c^2\times P_c^9-P_c^2\times P_s^9)}{(P_c^7+P_s^7)\times(1-P_s^2)-(P_c^7\times P_s^{14})\times(P_c^2-P_c^2\times P_c^9-P_c^2\times P_s^9)}$$

$$B = \frac{P_c^1\times P_s^8+A\times P_c^7\times P_s^1 4}{1-P_s^2}$$

$$C = \frac{A\times P_c^7\times(1-P_c^{14}-P_s^{14})+P_c^1\times P_c^8\times(1-P_c^{15}-P_s^{15})}{P_c^8+P_s^9}$$

$$D = \frac{\frac{P_c^1\times P_s^8+A\times P_c^7\times P_s^{14}}{1-P2suc}\times P_c^2\times P_s^9+P_c^1\times P_c^8\times P_s^{15}}{1-P_s^3}$$

$$E = (1 + A + B + C + D + P_c^1 + B \times P_c^2 + A \times P_c^7 + P_c^1 \times P_c^8)$$

and obviously:

$$P_s(CW = i) = \sum_{state_j;CW_1=i} P^j \times P_s^j$$

Until now we considered that a transition to higher CW values is caused by collisions only. To add the noise loss probabilities to these transitions, (7.2) should be replaced by:

$$P_{loss}(b, d) = 1 - (1 - P_c(b, d)) \times (1 - PER)$$

and (7.3) should be replaced by:

$$P_s(b, d) = \begin{cases} 1 - \frac{1}{2} \times \frac{b}{d} - \frac{1}{2} \times P_{loss}(b, d) & \text{if } b \leq d \\ \frac{1}{2} \times \frac{d}{b} - \frac{1}{2} \times P_{loss}(b, d) & \text{if } b > d \end{cases}$$

leading to the results shown in Fig. 7.6.

7.4 Proposal

Consider the same network topology of section 7.2 in which we apply a different scenario: instead of placing both of the traffic sources in the fixed host, we place each of them in a different WT. The main reason for this change is to avoid the mutual influence between flows when sharing the same MAC sub-layer, which is shown in "Noise/1" part of the scenario of section 7.2.

At second 50, WT_1 starts sending 1100-byte UDP packets to node S each 0.005 seconds. At second 150, WT_2 starts the same procedure, using the same packet parameters. Results (Table 7.2) show no mutual influence between WTs. In fact, the UDP traffic flows do not share the same MAC sub-layer. Instead, each of them contends to access the channel separately. When the channel is clear, both flows have equal shares during period 2. This is also the case when *global noise* is applied. If only WT_1 is exposed to noise, it will have its throughput reduced, but noise has no effect on the throughput of WT_2 at all, which was the case in the scenario of section 7.2. The side effect of noise, which is retransmissions and the increase of the CWs is still clearly visible.

Table 7.2: Data rates and contention window distribution for scenario 2

		Data rates (Bps)		Contention window size distribution					
		WT_1	WT_2	31	63	127	255	511	1023
Clear	I	214200	0	19626	43	3	0	0	0
	II	106837	106355	19425	1204	48	1	0	0
Global noise	I	63000	0	6456	3702	2307	1381	845	1046
	II	35884	36590	6779	4092	2532	1597	1015	1294
Noise/1	I	66161	0	6432	3698	2276	1386	819	1005
	II	12577	182123	17709	877	479	277	173	209

Figure 7.8 shows the contention window transition diagram used by CSMA/CA. Each time frame loss is detected the contention window is increased (as $2^i - 1$) to avoid further collisions, regardless of the frame loss cause, whether it is noise or collision. But this technique is not efficient when the frame losses are due to noise only. This would lead to higher values of $\alpha_i, i \neq 0$ which makes the chain drift to high CW values, resulting in high CW average. When CWs are high, delays are high and throughputs are low (tables 7.1 and 7.2), without enhancing any collision avoidance in compensation.

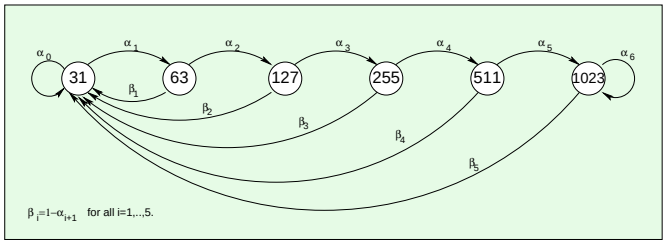


Figure 7.8: Contention window transition diagram.

A CW increase transition is triggered by either a collision-loss or a noise-loss event. A CW reset transition (to 31) is triggered by a successful transmission event. Our aim is to distinguish collision-losses from noise losses. This reduces the number of events triggering the CW increase, which therefore reduces the average CW size, leading to better performances without increasing the collision rates.

Our approach is statistical and totally independent of lower layers. The global idea is to learn about the channel status and number of contending terminals by observing the CW evolution.

In our first simulation we assume that the loss rate due to noise is constant with time, such as when a WT is far enough from the AP, with fixed fading and multi-path, or when FHSS and DSSS systems coexist in the same area, as described in section 7.2 and in Chapter 1.

7.4.1 Noise frame loss.

Consider the case where just noise can cause frame losses and there is no possible collisions, such as during period I of scenario 2. Therefore the frame loss rate is constant, whatever the CW size is. For each CW size i, we set two counters: the number of transmitted frames tx_i and the number of lost frames d_i (in practice, we count the received ACKs, which corresponds to (transmitted - lost)). The loss rate at each CW size i is given by d_i/tx_i. If this loss rate keeps constant over all CW sizes i, we can deduce that the frame losses are due to noise and we must not increase the CW. If the frame losses are due to collisions, the loss rate d_i/tx_i must have decreased when increasing the CWs. In practice, we consider that the loss rate is constant if the standard deviation of d_i/tx_i remains lower than a given value ϵ. This parameter will be more useful later in this section. Applying this mechanism during period I of the simulation leads to the results shown in Fig. 7.9 and Table 7.3. Previous simulation results are still shown in the table for comparison convenience.

During period I, we can see that the CW average value dropped considerably, which resulted in a good data rate enhancement. Data rate of WT_1 increased from 63000 Bps to 77638 Bps.

Applying the basic enhancement scheme to period II where collision frame losses also exist is not appropriate as it is intended for noise frame losses only. We kept it for analysis curiosity: At the beginning of period 2, WT_1 already had some statistics about the loss rates for each CW size, which made it recognize that most of the frame losses are due to noise (the standard deviation of the loss rates is $\leq \epsilon$). Therefore WT_1 limits its CW to lower values than those of WT_2, resulting in an unfair data rate distribution in favor of WT_1. After a while,

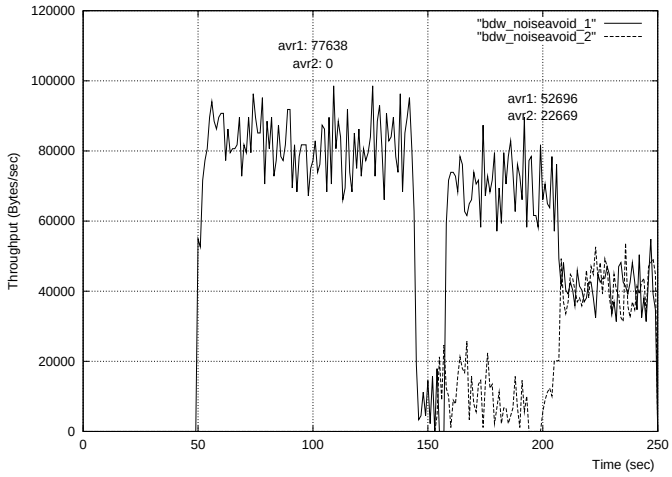


Figure 7.9: The basic enhancement scheme effect on bandwidths.

Table 7.3: Data rates and contention window distribution for the basic enhancement scheme.

		Data rates (Bps)		Contention window size distribution					
		WT_1	WT_2	31	63	127	255	511	1023
Clear	I	214200	0	19626	43	3	0	0	0
	II	106837	106355	19425	1204	48	1	0	0
Global noise	I	63000	0	6456	3702	2307	1381	845	1046
	II	35884	36590	6779	4092	2532	1597	1015	1294
Noise avoidance	I	77638	0	19135	152	88	53	34	46
basic scheme	II	52696	22669	18711	377	224	137	82	95

WT_2 gets enough loss rates values that converge, and the variation becomes lower than ϵ. It adopts small CW values, which give him equal data rate share with WT_1.

7.4.2 Combined noise and collision frame losses.

When we consider more than one active WT (as in periods II of scenario 2), collision will coexist with noise causing frame losses. Applying the basic scheme of the previous sub-section is somehow limited and sub-optimal. In fact, consider the two extreme cases:

- Noise and collisions are totally correlated (Fig. 7.10-a): This is when all of the noise corrupted frames collide, or all of the collided frames get noise corrupted, depending which rate is higher. In either cases, collision or noise frame loss, or both simultaneously, the frame will be lost. So the total frame loss rate is the maximum of the two rates. Applying the scheme described above will check if the total loss rate is constant over all CW values, which is not the case, and so the CW will keep increasing. However, the CW should not get higher values than the one corresponding to point A, i.e. cw_A. Above cw_A, the frame losses are due to noise and there is no use of increasing the CW. So, instead of computing the standard deviation of the loss rate over all CW values, the scheme should start computing it from high values down. Whenever the standard deviation goes above ϵ, point A is detected, above which the CW should not be increased.

- Noise and collisions are totally independent (Fig. 7.10-b): This is when none of the colliding frames is noise corrupted and none of the noise corrupted frames collide. Therefore the total loss rate is the sum of the two rates. As noise is assumed to be constant over all CW values, the total loss rate curve is a vertical shift-up of the collision rate curve. Applying the enhanced scheme of "total correlation" would find some cw_B as an optimal maximum CW value. However, cw_B is relatively high, due to the vertical shift of the total loss rate curve, due to noise. If we eliminate the noise effect by shifting down the collision-loss curve, point B would correspond to point B'. The shift amount must be equal to the (unknown) noise loss rate. However, we assume that the collision loss rate is negligible at $CW = 1023$, so we can shift that point down to zero. After eliminating the noise effect, B' is the point that gives the same collision loss rate as the "equilibrium" point B with both noise and collision loss rates, but with lower CW values than B.

Lower CW values corresponding to B' give better data rates than with B. (Note that if we set $\epsilon = 0$ then $A \equiv B'$).

The values of cw_B and $cw_{B'}$ are the upper bound and lower bound respectively of the optimal maximum CW size. Finding the optimal maximum CW size between the two depends on the correlation between noise and collision, which also depends mainly on the frame sizes.

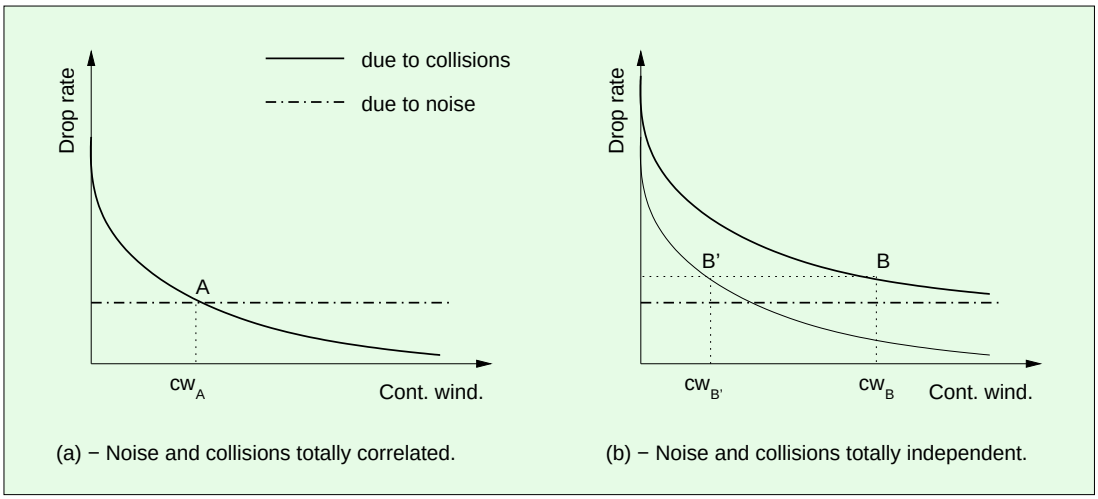


Figure 7.10: Combined noise and collision avoidance extreme cases.

When we adopt low CW value limits, such as $CW_{B'}$, to increase udr we also increase the number of collisions and frame retransmissions, which decrease udr. Therefore, as one may intuitively think, optimal CW size selection is a tradeoff between collisions and time overhead. In the previous section, equation (7.1), we only considered one period T in $E[udr]$ regardless of the retransmissions: as our aim was to compute the side effect (overhead) of increasing the CW due to a noise frame loss.

If we take retransmissions into considerations, we should add a parameter to the denominator of (7.1), T_i^0 which is the average time previously spent for retransmissions, prior to the current transmission. $E[udr(L)]$ becomes:

$$E[udr(L)] = L \times \sum_{i=1}^{1023} \frac{1}{K + T_i^0 + T_i} \times P_s(bkf = i) \tag{7.4}$$

For the two WTs case, T_i^0 is computed at the end of this section.
In (7.4) we can see that:

- When T_i decreases, $E[udr]$ increases. This is the case when we limit CW size to low values to reduce data transmission overhead.
- When T_i^0 increases, $E[udr]$ decreases. This is a negative effect of limiting CW size to low values, therefore increasing the collision rate, the retransmission rate and consequently T_i^0.
- When $P_s(bkf = i)$ decreases, $E[udr]$ decreases. This is another negative effect of limiting CW size to low values. In fact, $P_s(bkf = i)$ is proportional to $P(CW = m) \times P_s(CW = m)$. The probability of a successful transmission P_s is low for low CW values, which is more frequent (high $P(CW)$) when we limit CW size to low values, such as $CW_{B'}$.

To observe the $E[udr]$ variation with the CW size limit, we should recompute the $P_s(bkf = i), \forall i$, as we did in Section 7.3 taking into consideration the limitation of CW sizes. Therefore we can find the optimal CW size limit, above which CW increases overhead and below which collisions increase. This value, CW_{opt} optimizes the useful data rate.

As mentioned before, we limit our computations of $P_s(bkf = i)$ and T_i^0 to the case of two WTs only. This results in the lowest CW size limit (31) where we cannot observe how CW_{opt} avoids additional collisions. We just can observe how it avoids additional overhead (i.e. the CW increase) in favor of increasing the useful data rate. We are currently working on general forms of $P_s(bkf = i)$ and T_i^0, taking into consideration the number of WTs.

Applying this enhanced mechanism to a third scenario, where both WTs start transmission at second 50, lead to the results shown in Table 7.4. In this scenario, we avoided the unfairness period detailed above by letting the WTs start transmission at the same time.

Table 7.4: Data rates and contention window distribution for scenario 3

	Data rates (Bps)		Contention window size distribution					
	WT_1	WT_2	31	63	127	255	511	1023
Clear	106848	106378	38810	2394	106	5	0	0
Noise	35414	37626	13730	8277	5150	3200	1975	2467
Noise + Enhanced Mechanism	39244	41160	37701	772	551	176	139	99

Using the enhanced scheme, we get a total data rate of 80404 Bps instead of 73040 Bps without it. A perfect scheme would give 87423 Bps. We should note that we used a relatively high bit error rate (10^{-4}) to "magnify" the noise-backoff effects. The mechanism keeps performing well with lower bit error rates.

Computing T_i^0:
For a given backoff value, i, T_i^0 represents the average overhead due to retransmissions. Always in Fig. 7.7, consider the case of WT_1. As in Section 7.3, we just have to consider the states with $CW_1 \geq i$. For each of these states, compute the overhead introduced by each path leading from state 1 to this state, that is:

$$T_i^0 = \sum_{aa}(\prod_{bb} P^s P_{c/s2}) \sum_{bb}(K + CW_1/2)$$

where
aa: "Each path, R, leading from state 1 to a state with $CW \geq i$".
bb: "Each state s on R".
K is the same as in (7.1), and $P_{c/s2}$ is either P_c or P_s of WT_2 according to the path.

7.4.3 Dynamic environments.

Until now, we assumed that some parameters do not change with time. However, WTs may change location and the noise loss rate would change accordingly. The number of WTs may also vary, which would change the frame loss rate. Clearly, this effect can also be observed with the same number of WTs, while changing the frame transmission rate. The above mechanism can get adapted to these variations when:

- We limit the lost frame counter and transmitted frame counter to a fixed time window in the past. This window must not be very large, because "history" information is useless when the channel (fading etc.) or the number of mobiles changes rapidly. The window must not be very small neither, so the computed loss rate remains statistically valid. The time-window approach can also be replaced by the filter:

  $$new\_avr\_d_i/tx_i = \alpha \times d_i/tx_i + (1 - \alpha) \times old\_avr\_d_i/tx_i$$

  where α should be optimized, instead of the time-window size. Using the filter instead of the time-window reduces the amount of memory needed for each CW size.

- We change the value ϵ, below which the noise variation is considered negligible, according to eventual noise frame loss variation. For instance, when frames of different lengths are transmitted, a constant BER would lead to a variable PER. ϵ should be kept higher than this PER variation.

- We refresh the mechanism periodically. Consider the case of Fig. 7.10-a, if the constant noise level gets lower, the optimal point A goes toward high CW values, and more collisions must be avoided. Hence our mechanism should be refreshed periodically, to find new optimal maximum CW values higher than the actual fixed one. It should also be refreshed occasionally, when any of the $d_i/tx_i (i <= opt.max.CW)$ changes, due to noise level getting higher for example.

7.5 Future work

Beyond the results presented in this paper, future work should address the following issues:

- *Searching for optimal CW size limits for a higher number of WTs:* Equation (7.4) can be used to find the optimal CW size limit which reduces the overhead while avoiding collisions, in order to maximize the useful data rate. However, some terms of the equation were computed for two WTs only. Future work should consider a higher number of WTs, so we can generalize our optimization scheme.
- *Optimizing ϵ,* the threshold below which the noise loss rate is considered as constant. This parameter, which depends on several others such as WT movement frequency, frame sizes variability etc., has major influence on the scheme performance.

7.6 Conclusion

This chapter presents some results of our work on enhancing collision avoidance when deployed in noisy environments. We showed, through simulation, that our mechanism enhances the efficiency and fairness, without degrading performance parameters such as collisions. The main idea is to statistically distinguish frame losses that are due to noise from those due to collisions, so CSMA/CA will not increase its contention window uselessly. When frame losses are due to noise exclusively, a basic scheme has been introduced, which enhances performance considerably. When frame losses are due to combined collision and noise, the basic scheme takes the risk of neglecting collisions in favor of low contention window values. An optimal extension to the basic scheme was made which showed good performances as well. Last, we introduced more extensions to the scheme to be able to perform correctly in a dynamic, more general case environments.

Chapter 8

Enhancing IEEE 802.11 performance in congested environments

Contents

8.1 Introduction

In the previous chapter we explored mechanisms to avoid useless CW increase. Typically, this applies to noisy environments where stations should not increase their CWs when frame losses are due to noise not to collisions. This would lead to a considerable decrease of the backoff overhead in presence of noise. However, some drawbacks have to be mentioned:

- Noise may vary more frequently than we assumed, leading to efficiency degradation.
- The mechanisms introduced are relatively complex for a MAC sub-layer.
- Most importantly, all we try to win by limiting the CW size is some backoff time slots. However, what we risk is one or several collisions, and their corresponding retransmissions, which costs much more than a single backoff. This makes the "gamble" more risky:

These facts made us think of the other side of the "gamble":

How can we avoid collisions and retransmissions at the risk of some backoff overhead ?

The answer may be by avoiding the sudden CW decrease. Typically, when the contention level is high and a station succeeds to transmit a frame, it resets its CW and re-experiment contention "from scratch". This leads to new collisions and retransmissions, wasting bandwidth. A slow CW decrease would not suffer the drawbacks of CW increase limitations because (in contrast to the above three points):

- On a short time scale, the contention level (number of contending stations, or contending flows) is most likely to be the same.
- Slow CW decrease mechanisms can be simple (e.g. linear), or much more complex (using feedback control theory)
- All what we risk is some backoff overhead, but we avoid more collisions and their respective retransmissions, which is obviously more worthy.

The slow CW decrease was first introduced in [22], among several other extensions to CSMA and MACA (like backoff copying and per-flow backoff counters). The main idea was to increase the CW at each collision by multiplying it by 1.5, and to decrease it linearly (-1) at each successful frame transmission. The approach was called MILD (multiplicative increase, linear decrease), and did not explore the effect of other decrease (or

increase) factors on efficiency. Furthermore, short simulation results were shown for WT-AP communications only.

In [104], the slow CW decrease was considered, but from the fairness point of view. [104] tries to establish local utility functions in order to achieve system-wide fairness, with no explicit global coordination. Then it "translates" a given fairness model into a corresponding backoff-based collision resolution algorithms that probabilistically achieve the fairness objective. These algorithms include different backoff increase/decrease factors.

[104] tried to enhance the fairness properties of IEEE 802.11, MACAW [22] and CB-Fair proposed in [105]. Always aiming to establish fair contention algorithms, [105] uses slow CW increase and decrease functions. Each station i contends to access the channel to send a frame to station j with a probability p_{ij}, computed in two ways using time-based and connection-based methods. These methods are pre-established using information broadcast by each station about the number of logical connections and the contention time.

One can also find some similarity between working for fairness on the MAC sub-layer and working on fairness on the transport layer. TCP Reno [106] uses additive increase and multiplicative decrease based on [107] in order to attain fairness among flows.

In this chapter our main aim is to investigate the CW decrease functions from the data rate and delay efficiency point of view not the fairness point of view. We consider various network topologies and schemes to validate our analysis.

Section 8.2 introduces the approach of multiplicative CW decrease using a simple simulation scenario. Section 8.3 considers a more general scenario, with an ad-hoc network. It also evaluates CW decrease mechanisms using the throughput gain and the throughput settling time metrics. Section 8.4 briefly shows linear CW decrease performance and section 9.6 concludes this chapter.

8.2 Multiplicative CW decrease, single destination

First, consider 50 wireless terminals (WTs) uniformly distributed in a 100x100m square area. WT_1 is the common receiver for all other 49 WTs.

Simulation starts at second 44, we increase the number of transmitting WTs by one each two seconds: WT_i starts transmission at second $40 + 2i$, $i \geq 2$, WT_1 being the common receiver. All nodes are within the range of each other.

Each transmitting WT sends 1050-byte CBR packets each 5ms, providing a full data rate. At second 150, all traffic sources stop but one ($WT_2 \rightarrow WT_1$). At second 260, all sources stop sending data.

Optimally, when the number of WTs n increases, each WT would get $1/n$ of the available data rate. However, due to the increasing collisions, frames would be corrupted, not acknowledged and retransmitted, which would decrease the actual date rate observed by each WT.

The dashed curve in Fig. 8.1 shows how the total throughput decreases as the number of contending terminal increases (e.g. seconds 44-150). In fact, after each collision, the source has to wait for a timeout to realize that the frame collided, increases its contention window (to reduce further collision risks) then retransmits the frame. After a successful transmission the source resets its contention window.

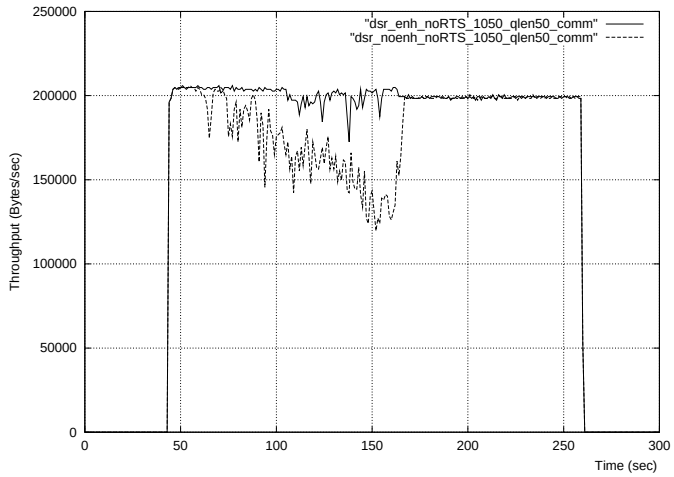


Figure 8.1: Total throughput comparison, without RTS/CTS.

As a node resets its CW after a successful transmission, it "forgets" about collision experience it had. If all WTs keep transmitting with the same data rate, most probably the new transmission will observe contention

and collisions as before. This can be avoided by keeping some history of the observed collisions: Instead of resetting the CW to CW_{min}, we set the CW to 0.8 times its previous value (low bounded by CW_{min}, i.e. $CW_{new} = max\{CW_{min}, 0.8 \times CW_{prev}\}$). The solid curve in Fig. 8.1 shows the considerable throughput enhancement we get (up to 53%), especially with high number of transmitting nodes (second 150): When we decrease the CW slowly, we waste more backoff time in favor of collision avoidance. Furthermore, throughput is more stable, due to lower/smoother variations of CW values.

The slow CW decrease is a tradeoff between wasting some backoff time and risking a collision followed by the whole frame retransmission. As the time of the latter is much larger than the backoff time, slow CW decrease is much better on the average. The average overhead due to backoff and retransmissions can be written as:

$$E[overhead] = O_{bkof} \times (1 - P_{col}) + O_{retx+bkof} \times P_{col}$$

where P_{col} is the probability of a collision, O_{bkof} is the overhead due to backoff time and

$$O_{retx+bkof} = \sum_{i=1}^{r} (bkof + data)$$

is the overhead due to retransmissions and their corresponding backoffs, r being the number retransmissions until a successful frame reception.

The worst case for slow CW decrease would be when we consider high CW values, but no congestion is taking place. This is the case at second 150, when we stop all but one transmission ($WT_2 \rightarrow WT_1$) in order to observe the remaining throughput. Fig. 8.1 shows that the slow CW decrease still behaves better than resetting the CW; after few successful transmissions, the slow CW decrease would reach CW_{min} value which CW reset would have directly reached. However, the overhead of the slow CW decrease is still negligible compared to a single frame retransmission.

The above analysis is not completely correct. In fact, all traffic sources (but one) stop at second 150, but the effect is shifted to around second 168. This is due to the residual packets queued in the interfaces of all 48 WTs (the interface queue length is 50). After sources stop, these remaining packets will continue contending to access the channel, possibly collide and get retransmitted resulting in the following:

- Smooth the sudden sources stop, therefore we cannot observe the real overhead of slow CW decrease when traffic sources suddenly stop.
- As congestion still exists, the slow CW decrease still shows better performance.

To avoid this residual queuing effect, consider now the same scenario as before, but with shorter interface queue lengths (= 2), in order to eliminate the smoothed sources stop and observe the maximum overhead due to slow CW decrease. Fig. 8.2 shows that the above queuing effects are eliminated, and the overhead due to slow CW decrease can be observed at its worst: no congestion, high CW values, i.e. second 150.

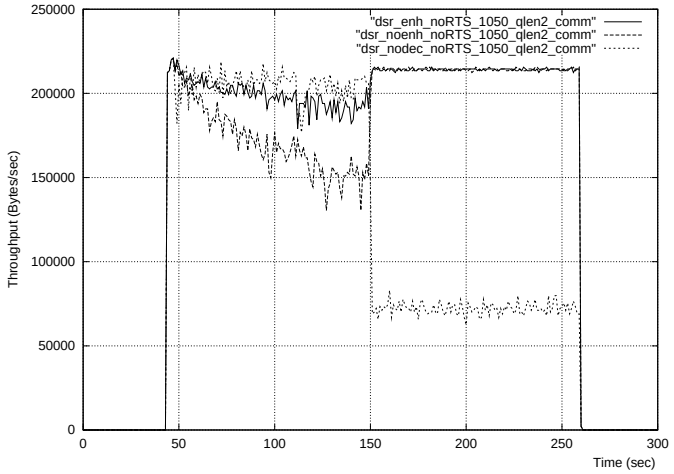


Figure 8.2: Total throughput comparison, without RTS/CTS, qlen = 2.

This shows that slow CW decrease (solid-line curve, dsr_enh_noRTS_1050_qlen2_com) performs as good as CW reset (dashed curve, dsr_noenh_noRTS_1050_qlen2_comm) at low congestion, even right after high congestion. This can be considered as the response of the mechanism to the congestion changing frequency at

its maximum. Slow CW decrease performs as well at intermediate congestion variation frequencies, when the number of transmitting sources changes up and down more smoothly.

For comparison convenience, we added a third curve (dsr_nodec_noRTS_1050_qlen2_comm) to Fig. 8.2, showing the overall throughput when we do not decrease the CW at all, i.e. keeping it at its maximum reached values. This shows that the CW time cannot be absolutely considered as negligible and must be reduced upon successful transmissions. Else, the performance decreases considerably at low congestion and high CW values, as we can see for flow $WT_2 \rightarrow WT_1$ after second 150.

Figure 8.3 shows the delays observed for the same simulation scenarios. We can see how the delays increase with the number of contending nodes for both slow CW decrease (solid curve) and for CW reset (dashed curve). However the slow CW decrease scheme shows relatively lower delays and jitters. Since the CW decreases slowly, we are avoiding more collisions and retransmissions, which is shown by lower average delays. And since the CW varies slowly, staying more adapted to the actual congestion level, the jitter is lower than the one with CW reset by tens of milliseconds. The contention success chances varies with the CW variation, therefore using sudden CW reset after each successful transmission leads to very high jitters. Slow CW decrease has lower jitters, showing the convenience of this approach typically at high congestion levels.

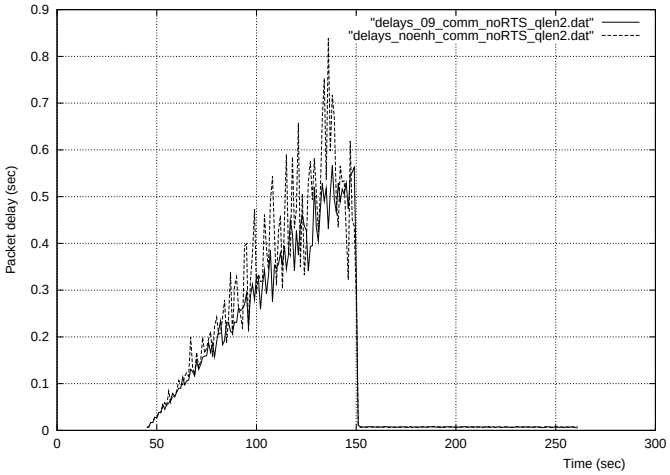


Figure 8.3: Packet delays comparison, without RTS/CTS, qlen = 2.

For completeness, we show in Fig. 8.4 the packet delays when we do not decrease the CW at all (solid curve). It has similar behavior to that of slow CW decrease at high congestion levels. However, after the sudden congestion level drop (second 150), this mechanism keeps high CW values leading to high delays and low throughput. These delays existed before the congestion level drop, but to the advantage of collision avoidance, increasing the throughput and lowering the overall packet delays. We should note that when we consider longer interface queues (e.g. 50), delays become orders of magnitude higher than the delays in Fig. 8.3 and 8.4.

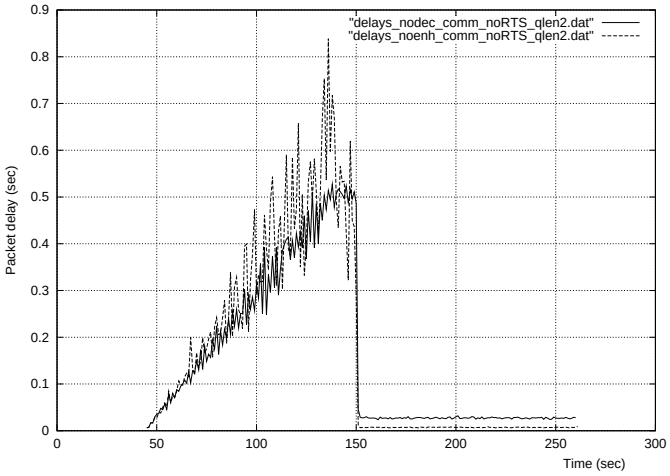


Figure 8.4: Packet delays comparison, without RTS/CTS, qlen = 2, the no-decrease scheme.

When we use short data packets, the relative gain decreases and the slow CW decrease becomes less efficient:

the time overhead introduced by the slow CW decrease becomes comparable to the packet payload. To the extent, consider the RTS/CTS exchange before a data packet transmission. Slow CW decrease would avoid (short) RTS collisions which are less probable (in case of hidden nodes) and less severe, from the data rate point of view. Therefore we observe low gain of slow CW decrease over CW reset.

This can be seen in Fig. 8.5. When congestion is low, we observe no gain, slow CW decrease performs as good as CW reset. At high congestion level (second 150), we observe a 6.8% throughput enhancement. This gain will be shown greater in the next section, when we consider ad-hoc scenarios. Obviously, RTS/CTS adds overhead and performs less than the basic scheme, whether using CW decrease or CW reset.

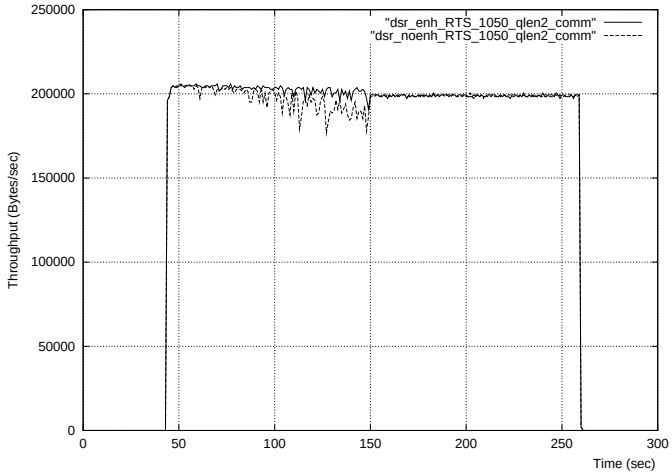


Figure 8.5: Total throughput comparison with RTS/CTS.

8.3 Ad-hoc, all-hear scenario

Consider now a different scenario, with 50 WTs sending data to 50 different WTs, all within the range of each other, uniformly spread over a 100mx100m area. The RTS/CTS scheme is used.

Fig. 8.6 shows a simulation with two similar phases showing different results: In the first phase (seconds 40 to 150) we increase the number of flows by one each 2 seconds. At second 150 we reset the number of flows and then start increasing it again (seconds 150 to 260). The throughput in the first phase is lower and varies more than in the second phase, whether using CW decrease or CW reset. This is due to routing information exchange during the first period which, once established, interferes less with throughput results in the second phase.

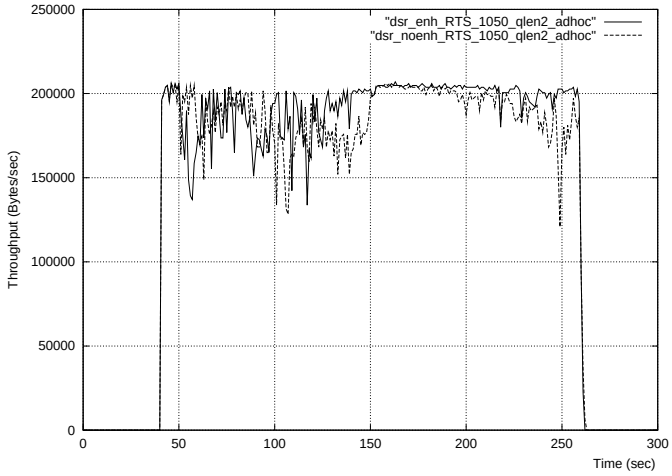


Figure 8.6: Ad-hoc all-hear, with RTS/CTS, scenario 1.

To avoid this transient effect, we added a “warm up” phase to our simulations, seconds 5-40, during which all WTs are active. We then consider the same scenario as in the previous section: At second 40 all WTs are

inactive, then we activate an additional flow each two seconds, until second 150. Congestion is at its most (second 150) when we turn all WTs off but one flow keeps running in order to observe the its CW behavior. Results are shown in Fig. 8.7.

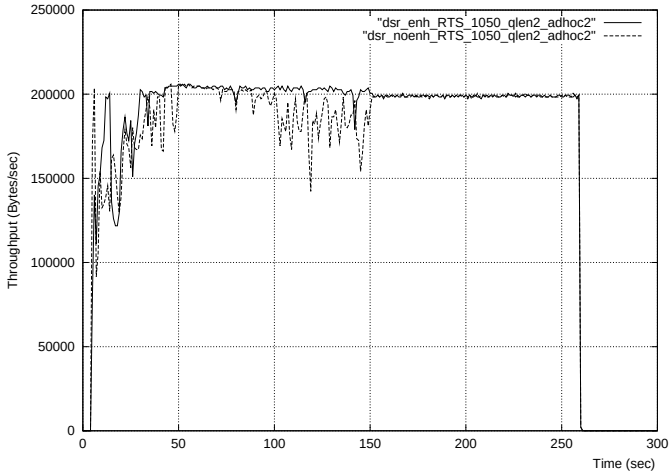


Figure 8.7: Ad-hoc all-hear, with RTS/CTS, scenario 2.

We observe that, in ad-hoc topologies and the random scenario we used, the gain of CW decrease over CW reset reaches 15% at high congestion, in contrast to 6.8% obtained with the single destination scenario, even when RTS/CTS is used.

In order to evaluate the performance of the CW decrease approach, we introduce two metrics used in feedback control theory [108]:

- *Throughput gain (G)*: This is the ratio of the throughput obtained by applying CW decrease over the throughput obtained by applying CW reset.
- *Settling time (T_s)*: After a sudden decrease of active WTs number (e.g. second 150), T_s is the time it takes a single flow to reach its throughput steady state, with small CW values. T_s characterizes the system *response time* using CW decrease.

In the following we will use different CW decrease factors δ and different data rates λ to evaluate G and T_s. Fig. 8.8 shows the throughput gain G function of the CW decrease factor δ. We can see that:

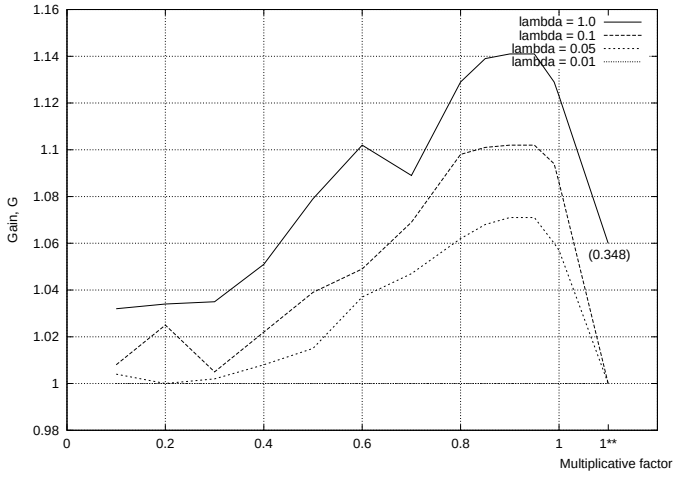


Figure 8.8: Throughput gain, G, vs. CW decrease factor δ.

- When δ decreases, the CW decrease becomes closer to CW reset and shows no enhancement over this last, $(G \rightarrow 1)$.

- However, when the multiplying factor δ is high, CW decreases slowly upon each successful frame transmission, still avoiding future collisions and retransmissions, therefore the throughput is higher than with CW reset ($G > 1$). For all λ values, the maximum gain G_{max} is around $\delta_{max} = 0.9$.

- As λ decreases (lower data rates) the gain G converges to unity. In fact, when data rates decrease, we observe fewer collisions leading to fewer CW increase and CW decrease. Therefore the advantage of slow CW decrease over CW reset gets lower and converges to one.

- When $\delta = 1$, we observe a considerable gain $G > 1$ when the channel is highly congested (as seen in Fig. 8.2). However, when the channel becomes less congested, the CW value keeps constantly high, increasing overhead, and decreasing throughput efficiency. This is what we denoted by (1**) in Fig. 8.8. For low data rates, this overhead (when $\delta = 1$) is negligible relative to the idle channel periods between consecutive packets. Therefore the gain $G = 1$. However, when $\lambda = 1$, this overhead becomes considerable leaving large idle gaps between packets, reducing efficiency, therefore the gain drops to $G = 0.348$.

- When using $\delta < 1$, the CW size and overhead progressively decrease upon each successful transmission. Therefore the overhead cited above, with $\delta = 1$, will still exist but for a transient period only, the duration of which is function of δ, the frame data rate λ and the corresponding successful transmissions. This transient period is characterized by T_s, the settling time we defined above.

To measure T_s with acceptable precision, we cannot proceed as in Fig. 8.1 right after all traffic flows but one stop and simply measure the time it takes the remaining flow to get to its stable state. In such a scenario, flow-1 is contending to access the channel with other flows. It has a non-zero probability to access the channel right before second 150, and therefore start its transient period with a short CW, unsuitable to measure T_s.

We proceed using a different simple scenario[1], as in Fig. 8.9: A single flow is considered. It starts at second 5, then from second 40 to second 60 we force the CW to its maximum, 1023, as it would be in highly congested environments. This reduces its throughput considerably. At second 60 we let the CW use slow CW decrease and CW reset respectively, and measure the settling times T_s. We used a large δ value, 0.9999, so T_s would be visible enough on the figure's scale. For lower δ values we will "zoom" into second 60.

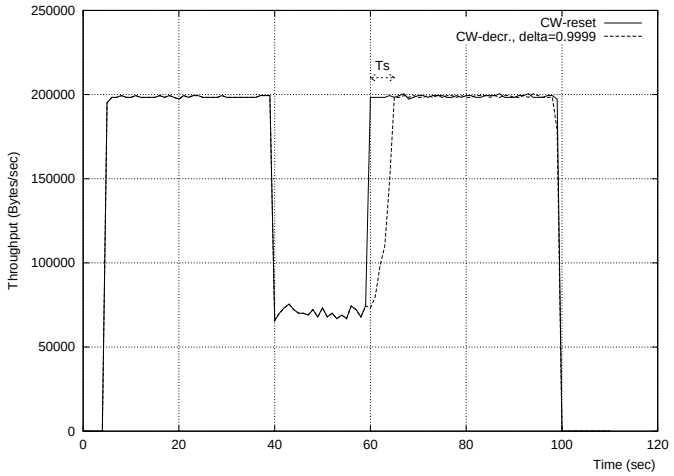


Figure 8.9: Throughput using forced CWmax between second 40 and 60.

Figure 8.10 shows that, as one would intuitively think, when δ increases, we need more successful transmissions before throughput reaches its steady state, that is T_s increases. This increase is much higher than linear, especially for high δ values. The reader should distinguish the settling time T_s from the frame transmission delays. The first concerns throughput stability, while the second concerns transmission delays. In the previous examples, a T_s of one second simply means that 200 frames should be sent successfully before the throughput reaches its high steady state. However, evaluating the user perception of T_s is out of scope of this work.

Choosing the right multiplicative decrease factor δ is a compromise between having a high throughput gain G and a short settling time T_s, for the case of sudden congestion decrease. Intermediate δ values like 0.6-0.8 would satisfy such a tradeoff. For smoother congestion decrease, which is practically hard to predict, one would choose higher δ values to get higher throughput gains, without much care about T_s.

[1]This scenario corresponds to the system response to an impulse input, from the feedback control point of view.

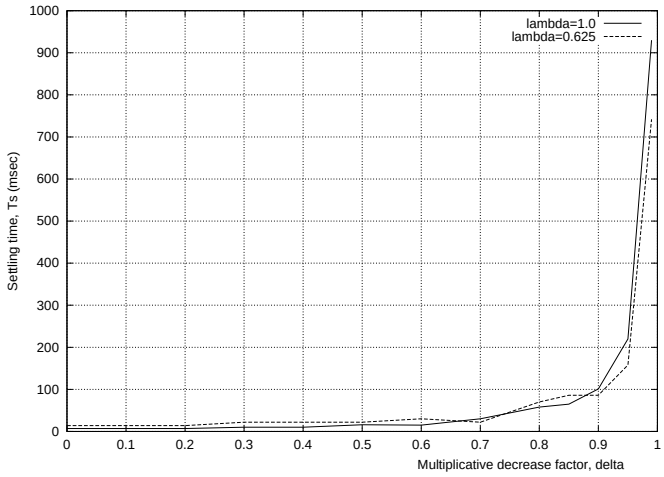


Figure 8.10: Settling time T_s vs. δ.

8.4 Linear CW decrease

In this section we repeat the same analysis of the previous section but with linear CW decrease, i.e. upon each successful frame transmission, CW is decreased by a constant value α.

From the throughput gain G point of view, Fig. 8.11 shows that linear CW decrease can reach the same gain values as multiplicative CW decrease. When α is small, CW decreases slowly, avoiding future collisions and retransmissions, leading to a throughput enhancement, just like high δ values of the previous section.

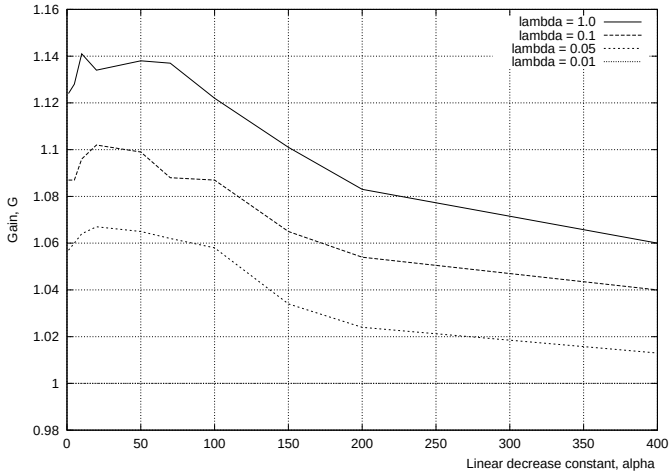


Figure 8.11: Throughput gain, G, vs. CW decrease constant α.

However, the settling time T_s shown in Fig. 8.12 is higher than with multiplicative CW decrease, especially for small α values ($\alpha < 100$) that would result in good throughput enhancements ($G > 1.12$).

Finally we should note that in [22], the authors use linear CW decrease with $\alpha = 1$. This surely enhances throughput, as would very high δ values do with multiplicative CW decrease. However, Fig. 8.10 and 8.12 show that very high δ values and very low α values would lead to unacceptable settling times T_s, if one considers sudden congestion level drops. From the user point of view, high settling time values (T_s) mean longer delays before the user gets the maximum throughput after moving from a highly congested area to a low congested one, or when all of his neighbors suddenly end their transmissions.

We aim to push our analysis further to investigate the influence of slow CW decrease on battery consumption,

8.5 Conclusion

In this chapter we investigated slow CW decrease instead of CW reset after each successful frame transmission. This would avoid future collisions, considering that congestion level is likely to stay constant. It also reduces the

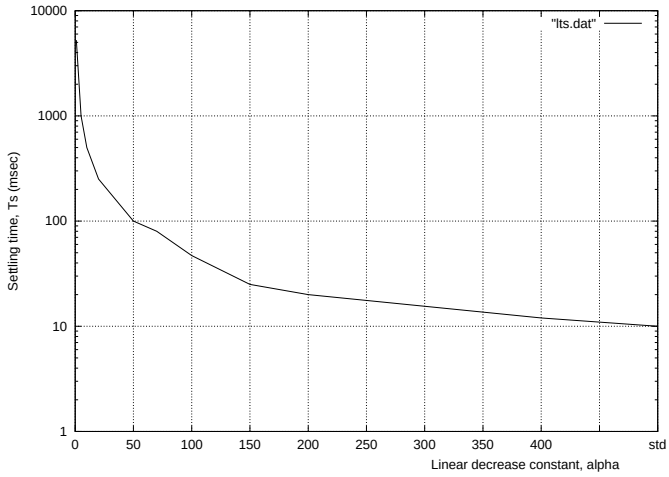


Figure 8.12: Settling time T_s vs. α.

number of frame retransmissions, which would also reduce congestion on the channel, increasing the throughput considerably. This throughput gain is function of frame lengths and data rates. We showed the considerable gain when using large data frames (53%), and extended the analysis for the worst gain values, that is for short data frames, e.g. when using RTS/CTS. Both multiplicative and linear CW decrease showed considerable throughput gains at $\delta = 0.9$ and $\alpha = 50$ respectively, with relatively low settling times after sudden congestion level drops. This settling time can be enhanced by using more complex CW decrease schemes using feedback control theory. However, this adds much more complexity on the MAC sub-layer, slightly enhancing the settling times without any throughput gain enhancement. Future work should consider modeling these schemes, trying to establish relations between the gain, settling time, data rates, frame lengths, number of active WTs and the CW decrease parameters. We also aim to explore adaptive CW decrease algorithms, in which decrease parameters change with the congestion load level. Slow CW decrease also enhances battery power consumption, which we also plan to explore in the near future.

Part IV

IEEE 802.11-based ad-hoc multi-hop networks

Chapter 9

Modeling IEEE 802.11-based ad-hoc multi-hop networks

Contents

In the previous chapter we noticed that the optimal maximum throughput and delays do not necessarily correspond to the maximum sending data rate at the sources. In fact, for a fixed number of nodes, increasing the sending data rate at the sources puts more data on the channel, trying to increase the throughput. However, this also increases collisions on the channel and their corresponding retransmissions, wasting time, decreasing individual throughputs and the overall channel efficiency as well. This led us to the question:

How can we estimate delays and throughputs in IEEE 802.11-based ad-hoc multi-hop networks ?

Estimating throughput and delays in ad-hoc networks helps to optimize the performance of the entire network as well as of individual paths. However such estimations are challenging tasks given the dynamism of the flows, varying channel conditions, interference and contentions between terminals. Major optimization topics are interference reduction between neighboring terminals, throughput and transmission power optimization.

In this chapter we propose a new method for modeling IEEE 802.11-based ad-hoc multi-hop networks. We investigate the conditions which makes applying queuing schemes to ad-hoc networks feasible which allows us to better estimate various parameters such as delays, throughputs and drop rates. Such estimates are used in data rate control at traffic sources so transmission power can be optimized, saving battery life, reducing interference and the offered loads to other nodes, which also makes the network scale better. We validate the approach through simulations.

9.1 Introduction

Ad-hoc networks provide convenient infrastructure-free communication media. Nodes cooperate by forwarding each other's packets until the final destination, without needing to go through a pre-established wired infrastructure.

Each packet reaches its destination following a multi-hop path. The main advantage of multi-hop forwarding (vs. single-hop) is the reduced transmission power. Electromagnetic signals are attenuated in free space with the exponent of the distance, not linearly. Therefore, the overall energy needed to reach a given destination in one hop is much higher than the sum of energies needed to forward it on several shorter hops. Furthermore,

transmitting with a reduced power causes less interference, allowing sufficiently distant nodes to transmit concurrently. Therefore the total amount of data that can be transmitted simultaneously increases linearly with the total geographic area of the ad-hoc network.

However, as nodes have to forward each other's packets, the data rate available to each single node will be limited by both the channel capacity and the load generated at other distant nodes. The number of these distant nodes increases when we increase the geographic area of the ad-hoc network, assuming a constant density, which on the counter part allowed the reuse of the spectrum.

This problem was considered by Gupta and Kumar [109]. As the network area increases, the average number of hops between source and destination also increases with the spatial diameter of the network, i.e. with $\Theta(\sqrt{n})$ where n is the total number of nodes. Therefore the total end-to-end capacity is $\Theta(n/\sqrt{n})$ and each node can have an end-to-end throughput of $\Theta(1/\sqrt{n})$. In other words, the throughput available to each node approaches zero as the total number of nodes increases.

$\Theta(1/\sqrt{n})$ is an upper limit. A scheduling mechanism which achieves $\Theta(1/\sqrt{n \log n})$ is also presented in [109] assuming a uniform random static network with random traffic patterns.

Li et al. [110] claimed that the assumption of a random traffic pattern, where each pair of nodes is equally likely to communicate, may not reflect reality: In large networks, users may communicate mostly with physically nearby nodes and go through a wired infrastructure to reach far correspondents. Therefore, path lengths (on the wireless channel) remain constant as the network grows, leading to a constant per-node available throughput.

Grossglauser and Tse [111] exploited the fact that nodes in an ad-hoc network, so far considered static, are mobile. For non delay-sensitive packets, a relaying scheme is proposed: a node transmits the packet to the different closest nodes which will relay the packet to the final destination whenever they get close to it. This keeps a constant number of hops on each path, allowing the network to scale without decreasing the average per-node throughput.

All of the above analysis focused on the statistical evaluation of capacities. From another point of view, this chapter aims to evaluate throughputs on arbitrary static paths in an IEEE 802.11-based ad-hoc multi-hop network, taking into consideration interfering and crossing-through traffic flows. The approach is to assimilate the network to a queues network and to establish the issuing relationships between parameters. This allows us to control data rates at traffic sources in order to optimize throughputs, delays, to reduce interferences and contentions in order to obtain a better throughput per node. It also reduces data retransmissions after collisions or interferences, considerably saving battery power.

We describe our modeling approach in Section 9.2. Section 9.3 analyzes the backoff's influence on the processing rate, not considered in Section 9.2. Section 9.4 compares the simulation results obtained so far. Section 9.5 shows future work plans and finally Section 9.6 concludes this chapter.

9.2 Initial steps toward modeling an IEEE 802.11 multi-hop network

In this section we consider elementary schemes and describe their response to different input data flows. The next section deals with more complex schemes, also taking into consideration the backoff time before transmitting packets, which we consider as negligible in this section.

We compare the analysis results to simulations results using *NS 2.1b8* [88]. The propagation model we consider is *free space*, which assumes the radio signal is constant with time, exponentially attenuated with distance. More complex propagation models such as *shadowing* is considered for future work.

The channel raw capacity is 2 Mbps, which gives a net data rate of approximately 1.6 Mbps given the packet size we used (1050 bytes) and the corresponding headers. Constant bit rate (CBR) traffic sources and exponential traffic sources showed similar results, therefore we only show results corresponding to CBR flows.

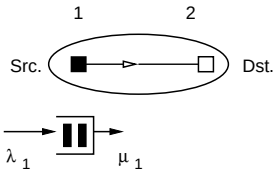
To illustrate communication ranges, we adopt two notations: a solid-line oval means nodes within the receive range of each other (i.e. the received power is above the interface's *receive threshold*). A dashed-line oval means nodes within the carrier sensing range of each other (i.e. the received power is above the interface's *carrier sense threshold*). Traffic sources are illustrated as filled boxes, traffic sinks are empty boxes, and routing nodes are illustrated as dots (cf. Fig. 9.3).

A packet can be received properly when the received power is higher than *receive threshold* and it still can be sensed as long as the received power is higher than *carrier sense threshold*. We configured the corresponding receive range to be 120m, while the sensing/interfering range is 160m. Without loss of generality, we do not consider the capture effect. One main assumption in this section is that we consider the source sending rate to be constant, even when its backoff increases. In Section 9.3 we show how our estimations change when we consider the backoff influence.

For comparison convenience, the simulation results for the following schemes are shown in Section 9.4.

9.2.1 Single-hop scheme

Figure 9.1 shows a single hop scheme. We assimilate this basic element of a multi-hop path to a single queue server, where the server processing rate is the available channel data rate and the incoming rate is the data rate at the input of the wireless interface. The queue length is the wireless interface's queue length.



Source data rate			Throughput (B/sec)		Delay (sec)		Drop rate
λ	PktSize	Interval	Avrage	Std. Dev.	Avr.	Jitter	(pkt/sec)
λ=1	1050	0.005	214133	731	0.231	0.057	1
λ/2	1050	0.010	107800	0	46e−3	0	0
λ/3	1050	0.015	71872	506	46e−3	0	0

Figure 9.1: The single-hop model.

The first simulation set shows the behavior of the single-hop with a varying incoming data rate λ_1 from sender node 1 to destination node 2 (Fig. 9.2). As long as λ_1 is lower than the channel capacity μ_1, the queue does not fill up and the average packet delay is equal to the packet transmission time (neglecting the backoff delay). The output data rate is equal to the incoming data rate, and no packets are dropped.

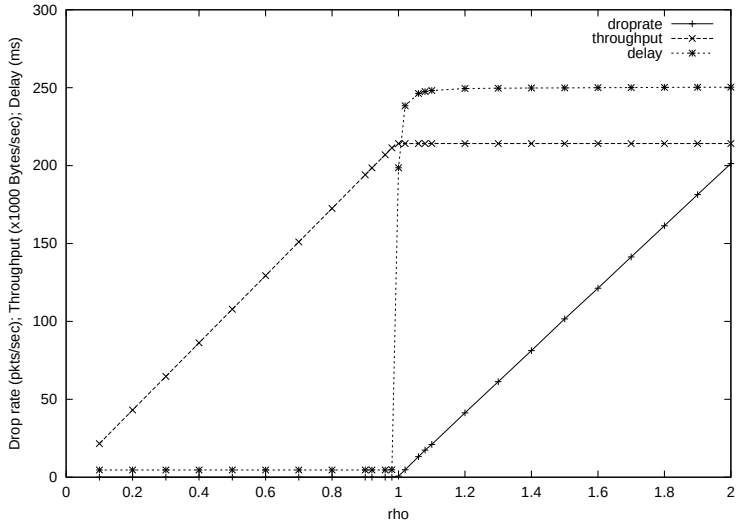


Figure 9.2: The single-hop model properties.

When λ_1 is close to μ_1, the queue builds up and the average packet delay grows with the average number of packets in the queue ($= \rho/(1-\rho)$ where $\rho = \lambda_1/\mu_1$, assuming an M/M/1/K model).

Beyond μ_1, the packet rate exceeding μ_1 will be dropped, the delay is constant (230ms), equal to the queue length (50) times the packet transmission time (4.6ms), and the throughput is constant ($= \mu_1$).

So far, the analysis is a typical single-queue single-server problem, no special issues are due to the wireless link which will be the case in the two-hop scheme and thereafter.

9.2.2 Two-hop scheme

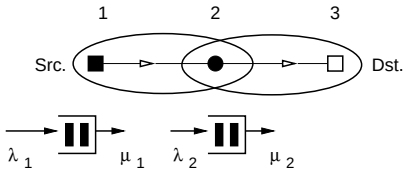
Consider now the two-hop path shown in Fig. 9.3. λ_i and μ_i denote the packet arrival rate and the packet departure rate of node i respectively.

The channel is shared between node 1 and node 2, therefore the following saturation relationship applies:

$$min(\lambda_1, \mu_1) + min(\lambda_2, \mu_2) = 1 \tag{9.1}$$

(9.1) denotes that, when both servers are saturated, the processing rates μ_i are complementary: all the processing rate not used by one is available for the other. Since we use no differentiation mechanisms (e.g. EDCF [1]), both nodes have equal chances to access the channel, therefore $\mu_1 = \mu_2 = 1/2$ when $\lambda_1 \geq 1/2$ and $\lambda_2 \geq 1/2$.

Generally, if $\lambda_1 < 1/2$ and $\lambda_2 > (1 - \lambda_1)$, e.g. an additional flow running through node 2, maintaining saturation, (9.1) gives $\lambda_1 + \mu_2 = 1$.



Source data rate			Throughput (B/sec)		Delay (sec)		Drop rate
λ	PktSize	Interval	Avrage	Std. Dev.	Avr.	Jitter	(pkt/sec)
λ=1	1050	0.005	109991	2538	0.527	0.031	98
λ/2	1050	0.010	108200	620	0.013	0.002	0
λ/3	1050	0.015	72122	514	0.009	0.000	0

Figure 9.3: The two-hop model.

If $\lambda_1 > (1 - \lambda_2)$ and $\lambda_2 < 1/2$ (i.e. the output from node 1 is higher than the input to node 2, for instance if the traffic has different destination interfaces), (9.1) gives $\mu_1 + \lambda_2 = 1$.

Note that (9.1) cannot be applied when the channel is not saturated. In this case, the output from node i is λ_i, with no considerable queuing delays or packet drops.

When we apply a $\lambda_1 = 1$ data rate to the two-hop path, (9.1) gives the throughput $\mu_2 = \lambda_2 = \mu_1 = 1/2$.

We observe that the average packet delay is almost the double of what it is in the single-hop case. In fact, the available processing rate is divided by two which doubles the waiting time in the queue and, most importantly, no considerable queuing in node 2 is taking place. Changing the queue length of node 2 shows no effect on the delays, queue length of node 1 is the main delay factor. Even though the incoming data rate is equal to the outgoing one, the queue in node 2 does not build up due to a quasi-synchronization between the two flows: When a packet arrives from node 1, it will be transmitted by node 2 most probably shortly after receiving it. This is shown in the next paragraph.

When the channel is under-saturated ($\lambda_1 < 1/2$), delay drops considerably as it becomes equal to transmission delays only. No queuing and no packet drops are observed.

Note that adding the RTS/CTS mode does not enhance the data rate since it adds overhead without avoiding any collisions here.

Quasi-synchronization between the two flows:

Now we show why, in the two-hop scheme, the queue at node 2 cannot build considerably.

Consider that, at time t, the two nodes 1 and 2 have new packets to transmit. They choose random backoff values $X_1(t)$ and $X_2(t)$ respectively in $[0; CW_{min}]$. The probability that node 1 transmits its packet first is

$$P[X_1(t) < X_2(t)] = 1/2$$

Assume that node 1 transmits its packet, while node 2 keeps decreasing its backoff value. The probability that node 1 transmits another packet consecutively is given by

$$P[X_1(t+1) < X_2(t+1)] = P[X_1(t+1) < (X_2(t) - X_1(t)]$$

After i successful contentions, the probability that node 1 succeeds to access the channel again before node 2 is given by:

$$P[X_1(t+i) < X_2(t+i)] = P[X_1(t+i) < (X_2(t) - \sum_{j=0}^{i-1} X_1(t+j)]$$

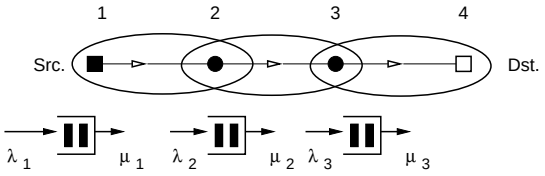
Therefore, for a given node, we can see that the success probability after consecutive successes decreases rapidly with the number of attempts. That means long queue builds at node 2 are rare, therefore we can neglect its queue waiting delays. Note that the above analysis shows that data rate shares between contending terminals is more fair than when we consider the transition probability between success events to be constant leading to a Markov chain model.

9.2.3 Three-hop scheme

Consider now the three-hop scheme of Fig. 9.4. The new issue in this configuration is the collisions at node 2, when the RTS/CTS handshake is not used. In fact, each hop is 100m long and the sensing range is 160m. Therefore nodes 1 and 3 are out of sensing range of each other, and eventually can be transmitting simultaneously causing collisions at node 2.

Let us first check out the throughput to node 4, when we use full data rate at node 1 ($\lambda_1 = 1$), without using the RTS/CTS scheme. The channel at the first hop is saturated due to the full data rate used at node 1. Furthermore, as there are no additional flows going through node 2, and as the flow at node 3 collides with the flow into node 2: $min(\lambda_2, \mu_2) = \lambda_2$ therefore (9.1) gives:

$$\mu_1 + \lambda_2 = 1$$



Source data rate			Throughput (B/sec)		Delay (sec)		Drop rate
λ	PktSize	Interval	Avrage	Std. Dev.	Avr.	Jitter	(pkt/sec)
λ=1	1050	0.005	68418	1021	0.813	0.017	136
λ/2	1050	0.010	68239	987	0.811	0.016	65
λ/3	1050	0.015	68418	1112	0.589	0.222	1

Figure 9.4: The three-hop model.

On the other hand, the channel on the second hop is not saturated, no extra flow goes through node 3, and the flow goes to a unique destination, therefore

$$\lambda_2 = \lambda_3$$

An additional equation can be drawn from the collisions at node 2:

$$\lambda_2 = \mu_1 \times (1 - \lambda_3)^2 \tag{9.2}$$

Solving the set of the three equations gives

$$\lambda_2 \approx 0.318$$

which is equal to the throughput to node 4. This is slightly higher than the throughput when using RTS/CTS, i.e. 1/3, explained later in this section, of which an extra overhead must be subtracted. Intuitively, one can roughly think of the throughput $\lambda_3 (= \lambda_2)$ as bounded by 1/3, obtained by applying RTS/CTS, and 1/2 which is the data rate shares in each hop considered separately. Practically, simulations show that the throughput is ≈ 0.319.

We should note that all colliding packets at node 2 are retransmitted by node 1, and should not be reduced from λ_2. However, the transmission at node 1 is limited to μ_1, already saturated, therefore equation (9.2) still applies. (9.2) denotes that a good packet reception occurs when no transmissions starts at node 3 for two packet durations.

As one can see, collisions at node 2 are proportional to the incoming data rate, reducing the overall throughput. In other words, the optimal/maximum throughput may not correspond to the maximum incoming data rate. We will check this out in the following.

Consider the case where $\lambda_1 < 1 - \lambda_2$, i.e. the channel is not saturated. The incoming flow at node 2 can be written as:

$$\lambda_2 = \lambda_1 \times (1 - \lambda_2)^2 \Rightarrow \lambda_2 = \frac{1 + 2\lambda_1 - \sqrt{1 + 4\lambda_1}}{2\lambda_1}$$

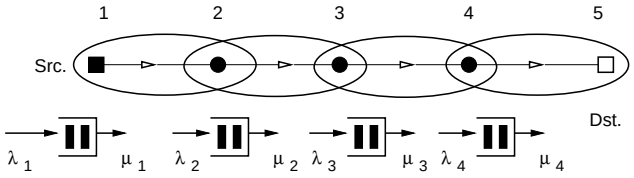
which derivative is strictly positive, i.e. shows no maximum point. Therefore the throughput is an increasing function of the incoming data rate.

From the delay point of view, reducing the incoming data rate also reduces the number of collisions and the corresponding retransmissions. Therefore we can reduce queuing delays at node 1, while still maintaining the same throughput. This explains the considerable delay decrease (from 0.813 to 0.589 seconds, see table in Section 9.4) when we reduced λ_1 to 1/3, while maintaining the same throughput.

When we consider the three-hop scheme using RTS/CTS, (9.1) should be replaced by

$$min(\lambda_1, \mu_1) + min(\lambda_2, \mu_2) + min(\lambda_3, \mu_3) = 1 \tag{9.3}$$

when the channel is saturated. (9.3) takes into consideration 3 consecutive nodes instead of 2 nodes in (9.1), owing to the fact that just one of the three can transmit at a time, due to the RTS/CTS handshake. When $\lambda_1 = 1$, considering no interference/collision takes place at the intermediate nodes, and no additional flows go through them, one can see that $\lambda_2 = \lambda_3 = \mu_1 = \mu_2 = \mu_3 = 1/3$.



Source data rate			Throughput (B/sec)		Delay (sec)		Drop rate
λ	PktSize	Interval	Avrage	Std. Dev.	Avr.	Jitter	(pkt/sec)
λ=1	1050	0.005	48894	15060	1.025	0.599	155
λ/3	1050	0.015	48664	16045	1.174	0.920	22
λ/4	1050	0.020	54500	0	0.019	0.000	0

Figure 9.5: The four-hop model.

9.2.4 Four-hop scheme

Consider the four-hop scheme in Fig. 9.5. This is a typical scenario which shows that the optimal throughput and delay through this four-hop path does not necessarily correspond to the maximum input data rate at the source.

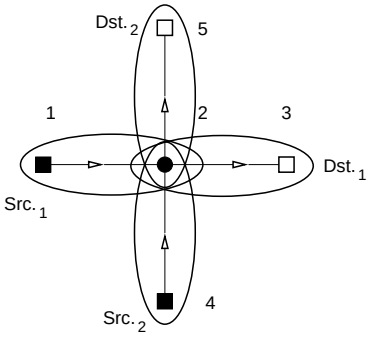
At input saturation, we observe big throughput variations, high delays and high jitters. When we reduce λ_1, the variations remain the same as well as the relatively low throughput. Simulation showed that maximizing λ_4 with respect to λ_1 gives $\lambda_{1opt} \approx 0.256$. At this point collisions and retransmissions are at their minimum, allowing data to flow "smoothly" along the four-hop path, with a considerably low delay and high throughput. Below λ_{1opt} the delay remains low but the throughput is sub-optimal.

Using RTS/CTS, we observe the same behavior. However the optimal throughput is lower than the throughput without RTS/CTS due to the overhead of RTS/CTS.

9.2.5 Crossing flows scheme

In the previous subsections the interference and collisions are due to various nodes routing the same data flow. Note that RTS/CTS may avoid theses collisions and save retransmission power, but the problem persists with interference, especially when we consider a high range interference (higher than 160m considered here): A node may not properly hear an RTS/CTS, to properly update the NAV, but it can still cause interference at the receiver.

Figure 9.6 shows two two-hop schemes sharing the middle router. Two different data flows contend to cross through node 2.



Source data rate			Throughput (B/sec)		Delay (sec)		Drop rate
λ	PktSize	Interval	Avrage	Std. Dev.	Avr.	Jitter	(pkt/sec)
λ=1	1050	0.005	51876	8404	1.402	0.166	152
λ/3	1050	0.015	53921	7339	1.353	0.144	16
λ/4	1050	0.020	54100	848	0.023	0.006	0

Figure 9.6: The crossing flows model.

Here, optimization is done on each data rate with respect to the data rate of the other source, not only to itself (as in the three- and four-hop schemes). Interference and collisions may be caused by one flow itself and by the other coexisting flow, as one would intuitively deduce.

If we consider equal data rates at both sources, the optimal point corresponds to $\lambda_{1opt} = \lambda_{2opt} = 1/4$. Even without scheduling, just data rate optimization at the sources makes data run smoothly on each path, with very low delays (0.023 instead of 1.4 seconds since no queues fill up), low jitters and a slightly better throughput, relatively to any $\lambda_1 > \lambda_{1opt}$ or $\lambda_2 > \lambda_{2opt}$.

9.3 Further considerations: Backoff's influence on processing rates

In the previous section we considered the delay due to backoff as negligible. However this assumption typically depends on the considered scenario. For instance, consider the two sources S_1 and S_2 in Fig. 9.7, transmitting with respective data rates λ_1 and λ_2 to the same destination D situated in the receive range of both S_1 and S_2, without using RTS/CTS. S_1 and S_2 cannot sense each other's transmission.

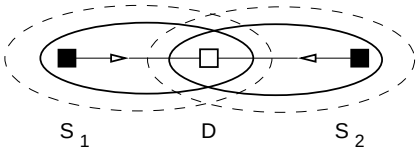


Figure 9.7: Typical scheme where the backoff effect must be considered.

If we consider the backoff to be negligible, the throughput from source S_1 to destination D is what is received from S_1 without colliding with the traffic from S_2 i.e.

$$\lambda_1 \times (1 - \lambda_2)^2$$

Now assume both sources transmit at full data rates, that is $\lambda_1 = \lambda_2 = \mu_1 = \mu_2 = 1$. This would result in no throughputs at all, from neither sources: both sources are transmitting continuously, causing continuous collisions at the destination which hears nothing but erroneous bits.

However, when a source does not receive an ACK from the destination, it doubles its CW to avoid future collisions. Upon several CW increases, the "gap" due to the backoff along with the preceding DIFS may fit a transmitted frame from the other source, without causing any collisions, so the destination D receives a full throughput from both sources, right after the transient period where the first collisions occur.

Since the CW increases at each collision as

$$CW = 2^i - 1; i = 5, .., 10$$

the average CW size can be written as

$$E[CW] = \sum_{i=5}^{10} (2^i - 1) \times P_i$$

where P_i is the probability of collision at $CW = 2^i - 1$. P_i is function of the number of nodes contending to access the channel, the data rates transmitted by each, and of the packet lengths. Therefore, as long as CW is small, P_i is high which increases CW so it may fit a data packet from another transmitter.

We should note that in this particular scheme, the full throughput to the destination D may come from a single source monopolizing the transmission. This source is likely to be the first to succeed to access the channel and transmit its packet successfully, keeping its CW relatively small. On the other side, the other source will keep high CW values leaving the source with very few chances to grab the channel again. This processing rate variation is not only due to collisions, but to interference also. The major difference is that the interfering traffic source does not necessarily increase its CW.

Therefore, to have a more accurate queuing model, we should take the backoff into consideration when evaluating the processing rates.

In the next example we consider two two-hop schemes (cf. Section 9.2.2) of which the middle routing nodes are within the range of each other, as in Fig. 9.8.

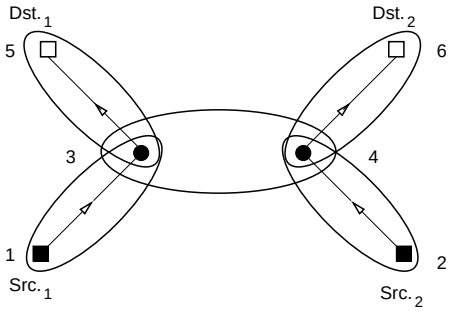
The two traffic sources transmit at full data rates toward the respective destinations. Note that the middle nodes are not necessarily within the receive range of each other (120m), the same applies when they can simply sense each other. RTS/CTS is not used, however the final results are similar.

If we do not consider the CW increase, our estimation would result in

$$\mu_3 = \mu_4 = 1/2$$

But the CW increase has a considerable effect here also:

The input data rates (=1) used at node 1 and 2 (1/2 at node 3 and 4) start causing continuous collisions at nodes 3 and 4. This causes the CW to increase considerably, letting some packets from one be successfully transmitted during the backoff time of the other, and vice versa. The first one among 3 and 4 to fail accessing the channel backs off, giving more chances to the other, who is likely to keep accessing the channel for long



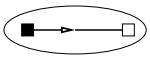
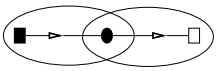
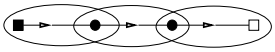
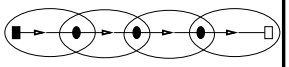
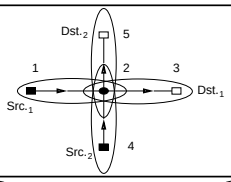
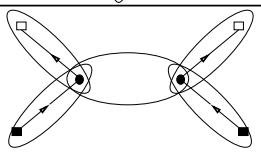
Source data rate			Throughput (B/sec)		Delay (sec)		Drop rate
λ	PktSize	Interval	Avrage	Std. Dev.	Avr.	Jitter	(pkt/sec)
λ =1	1050	0.005	48634	44696	0.625	0.474	155
λ/2	1050	0.010	49367	46880	0.616	0.484	54
λ/3	1050	0.015	43313	38489	0.413	0.698	26
λ/5	1050	0.025	31972	28660	0.616	0.945	10

Figure 9.8: Composed scheme (two 2-hops).

periods, until the first grabs the channel again. This results in series of long bursts alternated between node 3 and node 4 randomly.

During each burst period, data flow goes on a single two-hop path. Therefore, on the short term, one of the throughputs is 1/2 while the other is 0. On the long run, throughputs are equally shared between the two flows, resulting in an average of 1/4 each. Since each of the throughputs is either 1/2 or 0 during a given burst (and never =1/4), the standard deviation of the throughput is very high. The same idea also applies to delays: packets are either dropped or forwarded on a single two-hop path (delays are close to delays on an isolated two-hop path).

9.4 Simulation results for various schemes

Scheme	Source data rate			Throughput (B/sec)		Delay (sec)		Drop rate
	λ	PktSize	Interval	Avrage	Std. Dev.	Avr.	Jitter	(pkt/sec)
	λ =1	1050	0.005	214133	731	0.231	0.057	1
	λ/2	1050	0.010	107800	0	46e−3	0	0
	λ/3	1050	0.015	71872	506	46e−3	0	0
	λ =1	1050	0.005	109991	2538	0.527	0.031	98
	λ/2	1050	0.010	108200	620	0.013	0.002	0
	λ/3	1050	0.015	72122	514	0.009	0.000	0
	λ =1	1050	0.005	68418	1021	0.813	0.017	136
	λ/2	1050	0.010	68239	987	0.811	0.016	65
	λ/3	1050	0.015	68418	1112	0.589	0.222	1
	λ =1	1050	0.005	48894	15060	1.025	0.599	155
	λ/3	1050	0.015	48664	16045	1.174	0.920	22
	λ/4	1050	0.020	54500	0	0.019	0.000	0
	λ =1	1050	0.005	51876	8404	1.402	0.166	152
	λ/3	1050	0.015	53921	7339	1.353	0.144	16
	λ/4	1050	0.020	54100	848	0.023	0.006	0
	λ =1	1050	0.005	48634	44696	0.625	0.474	155
	λ/2	1050	0.010	49367	46880	0.616	0.484	54
	λ/3	1050	0.015	43313	38489	0.413	0.698	26
	λ/5	1050	0.025	31972	28660	0.616	0.945	10

9.5 Future work

In order to generalize the modeling approach we are considering the following points for future work:

- *More complex network topologies:* In this chapter we just analyzed elementary network schemes to be able to validate the approach. Future work should consider more complex and realistic topologies.

- *More complex propagation model:* The radio propagation model we used is the *free space* model where the signal attenuation is considered constant with time. Models closer to reality should be considered, taking into account the shadowing effects.
- *Evaluation of battery power saving:* As we have seen in the previous paragraphs, reducing data rates at traffic sources also reduces interference, collisions and the corresponding retransmissions. One of the resulting advantages is the considerable battery power saving, which depends mainly on the amount of transmitted (and retransmitted) data.
- *Backoff's influence on processing rates:* As shown in Section 9.3, the backoff each node has cannot be considered as negligible in many scenarios. Future work should also consider this issue in order to have better estimations for more general topologies and scenarios.

9.6 Conclusion

Power control reduces interference and tends to optimize the throughput available to each node in an ad-hoc network, by reducing the interfering radiated signals.

On the other hand, source data rates can also be optimized in a way to enhance the path's throughput or the average throughput available to each node in the whole network. Controlling the offered load at each source enhances the available throughput at other nodes, making the ad-hoc network more scalable.

In this chapter we investigated the applicability of queuing properties to IEEE 802.11-based ad-hoc multi-hop networks. Several elementary network topologies were considered in order to establish the appropriate relationships between the queues' parameters and the ad-hoc network parameters.

Such an approach enables us to better optimize throughputs and delays by controlling data rate at traffic sources, taking into consideration neighboring interfering flows, crossing-through flows and other offered loads. This approach also reduces battery consumption by avoiding useless data transmissions and eventual retransmissions caused by interference and collisions.

Simulations and analysis give close results, showing that the approach is valid so we can proceed into analyzing more complex topologies in the future.

Chapter 10

Conclusion

Internet access is increasingly reaching mobile wireless terminals e.g. PDAs, cellular phones, enriching our daily life with more applications and facilities. From work to home and in the street, wireless access to the Internet is becoming reality, and soon becomes essential. A wide set of standards come along to support these access technologies, giving us the freedom to move while still being connected.

Due to the technological advances in DSPs and microelectronics in general, wireless access data rates are increasing significantly, allowing more applications such as email, browsing, audio or video to be deployed in wireless terminals in addition to traditional voice communications. Therefore, wireless LANs (WLANs), personal area networks and pervasive computing in general are attracting more research and industry attention, preparing a surely promising wireless era, with unlimited application fields. The number of wireless terminal users is obviously expected to keep increasing, as shown previously with second generation telephone networks. Moreover, the variety of applications in these wireless terminals requires several devices per user and ubiquitous connectivity.

The increased number of users, of wireless devices per user and connection time result in a considerable load on the radio channel. In fact, these wireless terminals/applications have to operate in unlicensed frequency bands like ISM and U-NII constrained by regulatory bodies. Wireless access standards are supposed to cope with this bandwidth and power limitations, while still making efficient use of the radio channel, in spite of various competing standards and high loads.

Real-time applications like audio and video require minimum QoS guarantees to operate properly. Those requirements cannot be filled by using best-effort protocols, especially under high loads. Furthermore, the wireless nature of the medium gives another set of challenges to provide QoS for wireless applications. Noise, interference, fading etc. are basic elements of radio channels, and do not go in the same direction with the QoS objectives.

The previous issues make QoS support in wireless networks a hot topic for many research groups around the world. QoS support can be done at different levels of the layer stack. *DiffServ* and *IntServ* of the IETF proposed QoS solutions at the network (IP) layer. However, for wireless networks, these QoS solutions would be sub-optimal if not coupled by QoS support at lower layers, i.e. at the MAC sub-layer.

Circuit switching, inspired from classical telephony networks, simplifies QoS management and separation between flows due to the fact of centralized control and simple admission control of voice connections. However, circuit switching is not appropriate for Internet applications, based on IP packet switching. Packet switching is claimed to be more appropriate for the wide range of different applications Internet supports. With no signaling used, packet switching currently offers best-effort services only, un-appropriate for many "demanding" applications.

In this thesis we focused our work on providing QoS in packet-switching-based wireless LANs, e.g. IEEE 802.11. The work is divided into several parts which deal with different QoS issues:

- *Service differentiation*
  In this part we propose several service differentiation mechanisms for IEEE 802.11. All are based on simple differentiation of the MAC parameters: Backoff increase factor, DIFS, CW_{min} and the maximum allowed frame length. We show simulation results of these differentiation mechanisms operating with TCP flows and UDP flows. TCP showed reduced differentiation effects due to its closed-loop nature, which we tried to enhance using per-flow differentiation later in the same chapter. We also show and analyze the channel noise effect on these differentiation mechanisms. Most of the results of this chapter comply with the recent draft standard IEEE 802.11e proposed for QoS extension.

- *Enhancing IEEE 802.11 performance in noisy environments*
  Noise showed undesirable effects on service differentiation. Furthermore, since nodes cannot distinguish noise packet drops from collision packet drops, noise increases contention window values and packet overheads uselessly, leading to throughput decrease. We first show the noise effects in various scenarios. We then propose a mechanism to statistically detect noise on the channel and avoid unnecessary contention window increase. The mechanism defines upper and lower bounds of the optimal contention window value which need to be refined in future work.

- *Enhancing IEEE 802.11 performance in congested environments*
  Avoiding contention window increase when detecting noise on the channel is a sensitive operation since the compromise is between contention window time slots and collision increase, with their corresponding retransmissions. This fact led us to the inverse tradeoff: how to avoid retransmissions at the cost of contention window overhead. This is typically the case in congested environments, after a successful frame transmission. We explored two alternatives of contention window resetting: Multiplicative contention window decrease and linear contention window decrease. Both showed considerable throughput and delay enhancement over the legacy standard, using contention window reset. This is due to more collision (and retransmission) avoidance which also reduces the load on the channel, optimizing packet delays and the throughput.

- *Modeling IEEE 802.11-based ad-hoc multi-hop networks*
  In ad-hoc networks, putting more packets on the channel does not necessarily enhance the throughput since it increases the load on the channel, packet collisions and retransmissions. In this part we introduce a preliminary method for estimating throughputs and delays in IEEE 802.11-based ad-hoc multi-hop networks, and we compare it to simulation results. Estimating throughput can be used for various optimization issues, such as battery consumption, interference reduction, network scaling and for packet delay optimization as well. We apply our method to simple network topologies, aiming to validate it for more complex topologies in the future.

Future work

The issues we explored in this thesis can further be enhanced and extended. Among these enhancements we envision:

- *Mapping DiffServ to MAC differentiation*: i.e. How must DiffServ parameters be mapped to MAC differentiation in order to get the optimal performances, including end-to-end ones.

- *Modeling the system* for service differentiation with TCP flows, combined TCP and UDP flows and for per-flow differentiation.

- *Parameters distribution* for service differentiation between the WTs. i.e. how to establish the differentiation parameters between the WTs, in a distributed way, while taking the hidden nodes problem into consideration.

- *Enhancing the mechanism for noisy environments* to find more precise CW_{max} values. In this thesis we limited our approach to finding lower and higher bounds of the optimal CW_{max} value.

- *Exploring slow CW decrease impact* on packet delays and battery power saving. Both parameters must be enhanced when using slow CW decrease due to the considerable decrease of collision and retransmission rates.

- *Establishing relationships, when using slow CW-decrease* between the throughput gain, settling time, data rates, frame lengths, number of active WTs and the CW-decrease parameters.

- *Extending throughput and delay estimations* to more complex topologies than those considered in Chapter 9. The radio propagation model we used is the *free space* model where the signal attenuation is considered constant with time. Models closer to reality should also be considered, taking into account the shadowing effects.

- *Evaluating the battery power saving* when using throughput optimization in ad-hoc networks. Here also, controlling the source data rates decrease the number of packets sent on the channel, therefore reducing the collision and retransmission rates, considerably decreasing the battery power consumption.

Résumé en Français / Summary in French

Chapitre 1

Introduction

Le concept des communications par commutation de paquets a commencé en 1962, avec le but de construire des réseaux très robustes. En 1968, l'Internet a vu la lumière, connectant quatre ordinateurs dans quatres universités aux États-Unis, et son taux de croissance est beaucoup plus grand que l'on ne pouvait immaginer à l'époque. Les ordinateurs étaient typiquement larges et très couteux. L'idée des réseaux personnels (PAN), où une personne porte plusieurs appareils connectés entre eux ainsi qu'à un réseau fixe, était encore une fiction. Maintenant, c'est devenu une réalité, due au progrès apportés par les nouvelles technologies ainsi qu'à la concurrence et à la demande d'appareils plus puissants à des coûts de plus en plus réduits.

Depuis le dédut du vingtième siècle, les communications sans-fil ont commencé à paraître. Typiquement inspirées des réseaux téléphoniques, ces communications étaient orientées-connexion. En 1970, Pr. Abramson de l'université d'Hawaii voulait connecter les ordinateurs de l'université dans divers bâtiments sur différentes îles de l'archipel. Le protocole de communication, premier système sans-fil non-orienté-connexion, était baptisé sous le nom d'"Aloha". En 1997, l'IEEE lança le premier standard pour réseaux locaux sans-fil, IEEE 802.11. Celui-là a connu un grand succès de déploiment dû au fait qu'IEEE 802.11 est conçu pour remplacer les cartes Ethernet (IEEE 802.3) utilisées dans les réseaux filaires, d'une manière transparente aux protocoles des couches supérieures.

Les applications sans-fil s'incrustent dans notre vie quotidienne et deviennent parfois un besoin essentiel, que ça soit au niveau social, professionnel, scientifique, medical ou militaire. Une grande variété de standards de communications sans-fil suit cette evolution pour satisfaire à ses besoins. De nos jours, plusieurs de ces standards existent, supportant des débits de données de plus en plus grands: IEEE 802.11, HiperLAN, Bluetooth, HomeRF etc. (voir Fig. 1.1). Cette richesse en standards ainsi que leur bonnes performances est due aux avances technologiques en microelectronique, qui rend la théorie de communications plus proche de la réalité, et nous offre de meilleurs types de modulation pour mieux combattre les problèmes de communications radio.

Vus leurs champs d'applications, ces standards doivent opérer, sans permis, dans des bandes de fréquences appropriées comme la bande ISM et U-NII (Fig. 1.2). Par suite, plusieurs standards coexiteront dans les mêmes bandes de fréquences et possiblement dans les même zones géographiques, sans coordination au préalable. Ceci produira des conflits qui font le sujet de recherche de plusieurs groupes de travail.

Les communications radio se distinguent des communications filaires par le fait que les ondes électromagnétiques se propagent dans l'air, ou dans le vide, au lieu des câbles. Ceci est caractérisé par une forte atténuation du signal dans le medium (Fig. 1.4), les réflexions multiples (Fig. 1.3) du signal sur différents obstacles, le bruit canal, et les interférences diverses.

En combinant ces diverses propriétés du canal radio, on obtient un medium très hostile (Fig. 1.5); le niveau du signal varie en temps et en espace d'une manière imprévisible, traditionellement modélisé stochastiquement, en se basant sur de réelles mesures. Le taux d'erreur dans les canal radio est considérablement supérieur au taux d'erreur dans les réseaux filaires, typiquement protégés par des écrans ou en torsadant les fils. Le taux d'erreur de bits sur un canal radio est grand, de l'ordre de 10^{-3}, tandis que celui des réseaux filaires est de l'ordre de 10^{-6}. Pour pallier à ce problème, on a souvent recours à utiliser des paquets relativement courts, à des mécanismes de correction d'erreurs (FEC) ou à des mécanismes de retransmission des paquets erronnés. Ceci améliore considérablement la fiabilité du canal radio, avec un taux d'erreur de bits acceptable pour les diverses applications.

Cependant, le manque de support de la qualité de service (QoS) se révèle autant que le nombre d'utilisateurs augmente, et que les applications deviennent plus exigeantes en termes de débits et de courts délais. C'est le cas dans le cœur de l'Internet ou dans les réseaux d'accès, où l'accès sans-fil est en croissance notable. Au niveau de la couche réseau, les travaux sur *DiffServ* et *IntServ* visent à assurer un certain niveau de QoS dans les réseaux filaires ou sans-fil. Le support de la QoS pourra aussi être appliqué au niveau de la couche de contrôle d'accès au medium (MAC), pour pallier aux problèmes du canal radio mentionnés ci-dessus, ainsi que pour fournir un support des couches basses, afin d'obtenir une performance globale amélioré.

Comme dans les réseaux filaires, les réseaux sans-fil se classent en deux catégories: à commutations de circuits ou à commutation de paquets. Le support de la QoS s'avère facile à fournir dans la première catégorie, vue sa nature souvent centralisée et le maintient d'états dans les nœuds. D'autre part, les communications à commutations de paquets, telles que dans l'Internet actuellement, n'offrent qu'un service "meilleur-effort", c'est-à-dire sans aucune garantie. Cependant, cette approche s'est montrée très robuste et ne pose pas de problème de passage a l'échelle, grâce à l'absence d'états dans les routeurs et à la signalisation simplifiée. C'est dans ce sens que nous orientons notre travail dans cette thèse, tout en visant une amélioration de la QoS dans les communications sans-fil à commutation de paquets, pour le standard IEEE 802.11.

Le travail est divisé en quatre parties: la première présente les divers standards de communications sans-fil actuels. La deuxième partie présente nos travaux sur la différentiation de services dans IEEE 802.11, ainsi que d'autres travaux sur la QoS dans les réseaux sans-fil. La troisième partie propose des améliorations pour les protocols d'accès dans les environnements bruités et les environnements congestionnés. La dernière partie, présente notre travail préliminaire sur l'estimation des débits utiles et des délais dans les réseaux ad-hoc.

Chapitre 2

Les protocoles de contrôle d'accès au médium sans-fil

2.1 Introduction

De nos jours, plusieurs types de médiums de communications sont utilisés pour l'échange de voix ou de données: la lumière pour les communications à travers les fibres, le son pour les communications sous l'eau, et les ondes électromagnétiques (EM) pour les communications par câbles. La lumière, le son et les ondes EM sont utilisés aussi pour les communications dans l'air.

Indépendamment du type du médium, les transmetteurs ont besoin d'un protocole de contrôle d'accès pour un partage équitable des ressources et d'une manière efficace. Comme dans la vie de tous les jours, les gens communiquent/parlent en utilisant des protocoles convenables aux situations; on demande la permission avant de parler, ou simplement on écoute s'il y a quelqu'un qui parle avant de commencer sa propre conversation etc.

Cette métaphore peut décrire les protocols MAC encore mieux; dans un région donnée, une seule personne est sensée parler en même temps. Si non, la personne qui écoute n'entendra que du bruit, sauf si l'un parle beaucoup plus fort que les autres. Cependant, parler haut dérange plus du monde dans une région plus large, et épuise la personne qui parle. Parler à voix basse interfère avec moins de personnes, permettant à ceux qui sont suffisamment loin de communiquer en même temps. Par contre, parler à voix basse est vulnérable aux bruits. Deux personnes dans une même région qui lancent des conversations simultanées résultent par un bruit incompréhensible, et elles doivent répéter ce qu'elles viennent de dire. Ceci consomme du temps, de l'énergie des personnes qui parlent et celles qui écoutent aussi. Pour éviter tels conflits, les personnes doivent attendent des temps différents pour relancer leurs discussions, ou elles attendent un "coordinateur" à leur demander de parler de nouveau, selon la situation.

2.2 MAC, les éléments de base

Dans le chapitre précédant nous avons cité plusieurs aspects des canaux radio qui ont des effets destructifs sur les signaux EM. Le signal radio transmis d'une station se propage dans l'air en s'atténuant. En-dessus d'un certain seuil de réception l'information transportée n'est plus compréhensible. La distance correspondante à ce seuil s'appelle la "portée" du transmetteur. Deux autres seuils éxistent: le seuil d'interférence et le seuil de détection de la porteuse, en-dessus desquels le signal peut causer des interférences avec d'autre transmission, et le signal peut être détecté, respectivement. Le seuil de réception est évidemment plus grand que celui d'interférence qui, à son tour, est supérieur au seuil de détection du signal. Dans ce chapitre nous ne considérons que le seuil de réception.

L'atténuation des signaux radio rend la transmission et la réception dépendantes de la position du récepteur par rapport à l'émetteur. Notons qu'un transmetteur détecte s'il y a des transmissions en cours sur le canal avant de commencer sa transmission pour éviter les collisions au niveau du récepteur. Nous distinguons deux situations possibles (cf. Fig. 2.1).

- Les nœuds cachés: Un nœud caché est un nœud qui est à la portée du récepteur, mais hors portée de l'émetteur [10, 11]. e.g. dans Fig. 2.1, A transmet pour B. Entre-temps, C a un paquet à transmettre, détecte un canal libre puisqu'il est hors portée de A. Par suite C commence sa transmission qui génère une collision au niveau de B, qui est à la portée de A et de C simultanément. A est caché pour C. Les nœuds cachés augmentent les chances de collisions, réduisant ainsi l'éfficacité de l'utilisation du canal.
- Les nœuds exposés: Un nœud exposé est un nœud à la portée du transmetteur, mais hors portée du récepteur [11]. e.g. dans Fig. 2.1, considérons que B transmet vers A. C détecte un canal occupé et

diffère sa transmission pour éviter une collision. Cependant, C peut initier sa transmission sans causer de collision puisque A est hors portée de C. C est exposée pour B. Cet aspect réduit l'éfficacité de l'utilisation du canal aussi.

Les collisions ont lieu quand un nœud commence une transmission durant une transmission en cours d'un autre nœud caché. Ca peut être aussi le cas de deux nœuds, à portée l'un de l'autre, qui commencent leurs transmissions en même temps. Cependant, une bonne réception de l'un des deux signaux peut avoir lieu si sa puissance à la réception dépasse largement le signal de l'autre.

Le fait que le signal s'atténue vite avec la distance dans un canal radio fait que la détection des collisions soit impossible dans les réseaux sans-fil. En effet, quand un nœud transmet son signal, une grande fuite a lieu vers le circuit récepteur du même nœud. Vû que le signal transmis dépasse par des ordres de grandeur le signal reçu, le récepteur est "aveuglé" par sa propre transmission. Par suite, un nœud qui transmet ne peut pas écouter le canal simultanément, comme c'est le cas des détections des collisions dans les réseaux Ethernet. Un canal de retour peut être donc utilisé pour informer les stations sur une éventuelle collision, comme on va voir dans la section suivante. Puisque les collisions ne peuvent pas être détectées immédiatement, des protocoles qui évitent les collisions doivent être utilisés pour rendre l'utilisation du canal plus éfficace.
Les réseaux sans-fil peuvent être distribués ou centralisés. Les réseaux sans-fil distribués, appelés réseaux ad-hoc, n'ont aucun coordinateur/administrateur centralisé, ce qui les rend plus robustes que les architectures centralisées. Les réseaux ad-hoc fonctionnent en TDD seulement. D'autre part, les réseaux sans-fil centralisés sont souvent connectés à des infrastructures filaires, dont le dernier saut est le lien sans-fil. Elles ont des stations de base (BS), appelées également points d'accès (APs), qui font l'interface entre la partie filaire et la partie sans-fil du réseau. Le fait que ces réseaux sont centralisés fait que le support de la QoS est facile à fournir. Cependant, ils sont moins robustes et plus compliqués à mettre en œuvre que les réseaux ad-hoc. Pour multiplexer les communications ascendante et descendante, les réseaux centralisés utilisent FDD ou TDD.

Pour chacune de ces architectures plusieurs protocoles MAC ont été proposés. Chacun de ces protocoles a ses caractéristiques qui conviennent à des scénarios plus qu'à d'autres. Nous allons décrires quelques uns de ces protocoles dans la suite. Les paramètres souvent utilisés pour évaluer les performances de ces protocoles sont:

- *Le délai:* Les trafics du type temps-réel sont sensibles aux délais des paquets. Le délai est le temps que prend un paquet pour arriver à sa destination, prenant en compte les délais des files d'attentes et des retransmissions.
- *Le débit utile:* On compare souvent les protocoles MAC par leurs éfficacités en utilisation du canal. Le débit utile est la fraction de la capacité du canal utilisée pour transmettre les données. Pour maintenir cette fraction suffisamment haute, on doit réduire les coûts de transmission et les collisions (et les retransmissions).
- *Equité:* C'est la mesure de l'équité entre les nœuds en compétition que peut fournir un protocole MAC [12]. Cette définition peut être biaisé quand nous prenons en considération le support de la qualité de service et la différenciation de services. Dans ce dernier cas, l'équité est la capacité de distribuer les débits utiles proportionellement aux allocations prévues.
- *Stabilité:* A cause du surcoût d'un protocole MAC, un système supporte souvent des charges qui sont beaucoup plus petites que la capacité du canal. Un système stable peut supporter des charges instantanées plus grande que la capacité du canal si la charge globale moyenne est inférieure à cette dernière.
- *Consommation d'énergie:* La consommation d'énergie est un paramètre important pour les appareils sans-fil puisqu'elle puise la batterie. Elle et composée de deux facteurs: L'énergie de traitement et l'énergie de transmission. L'énergie de transmission peut être optimisée en réduisant le surcoût des transmissions, les collisions et les retransmissions, ainsi que par le support du mode de veille.

2.3 L'évolution des protocoles MAC

La recherche dans le domaine des protocoles de contrôle d'accès au médiums sans-fil a commencé dans les années 1970. Aloha [13, 14] fut le permier protocol MAC, conçu en 1970 par le Pr. Abramson pour connecter les ordinateurs de l'université placés sur des îles diffŕentes, en utilisant des trasmissions radio.

Le meilleur débit utile qu'Aloha peut atteindre est 18%, quand la probabilité de transmission d'un paquet à un moment donné st 0.5 (cf. Fig. 2.2). Ce débit utile peut être doublé $(1/e)$ si on considère que le temps est discret, et que les transmissions peuvent avoir lieu au début de chaque unité de temps uniquement. La partie vulnérable de transmission est ainsi éliminée, doublant ainsi l'éfficacité du système [15], au coût de la complexité de synchronisation entre transmetteurs. Ce protocol est appelé "slotted Aloha" (S-Aloha).

[16] proposa d'écouter le canal avant d'initier une transmission (CSMA, carrier sense multiple access). Ce protocole montre une amélioration considérable du débit utile par rapport à Aloha et S-Aloha. CSMA dépend

typiquement de la position du transmetteur. CSMA ne détecte pas les nœuds cachés, et évite de transmettre si le transmetteur est exposé. Cependant, les avantages apportés par CSMA dépassent ces inconvénients, par suite il est utilisé dans tous les protocoles qu'on va citer dans la suite.

CSMA a trois versions: 1-persistant, non-persistant et p-persistant, où le prefixe indique la probabilité qu'un nœud transmette un paquet juste après avoir détecté le canal libre.

S'il y a collision, les nœuds concernés continuent leurs transmissions sans le savoir. Ceci résulte en une perte de temps et de débit utile puisqu'aucun des nœuds réussit à livrer son paquet (si on considère que les puissances à la réception sont comparables). Cette perte devient plus grave quand les paquets sont grands. Pour pallier à ce problème CSMA peut être amélioré en utilisant la détection des collisions (CSMA/CD). Quand un transmetteur détecte que le signal sur le canal est différent de celui qu'il transmet, il arrête la transmission, gagnant ainsi du temps et de la bande passante.

Ce mécanisme s'avère irréalisable dans les réseaux sans-fil, comme détaillé précédemment. Par suite il est remplacé par un mécanisme qui évite les collisions (CSMA/CA). CSMA/CA peut être appliqué en utilisant une signalisation hors-bande, comme dans BTMA [10] et RI-BTMA [19], ou une signalisation en-bande comme dans MACA [20, 21] et ses extensions [11, 22, 23, 24].

2.4 Les types des protocoles MAC

Les protocoles MAC peuvent être classifiés comme dans la Fig. 2.4. Au premier niveau, c'est les protocoles distribués et les protocoles centralisés. Les premiers peuvent être utilisés dans n'importe quelle architecture, tandis que les deuxièmes ne s'appliquent qu'aux architectures centralisées. Les protocoles distribués utilisent typiquement des techniques d'accès aléatoire. Les protocoles centralisés, à leur tour, utilisent une plus grande variété de techniques: accès aléatoire, accès guaranti et les techniques hybrides. Plusieurs protocoles sont cités dans la suite, qui sont à la base de plusieurs standards de communications sans-fil.

Parmis les protocoles MAC distribués nous citons:

- Distributed foundation wireless medium access control (DFWMAC)[25].
- Elimination yield - Non-preemptive priority multiple access (EY-NPMA) [40, 41, 42].

Parmi les protocoles MAC centralisés nous citons les suivants:

- Centralized random access protocols: ISMA (idle sense multiple access) [43], R-ISMA (reservation idle sense multiple access) [44], RAP (Randomly addressed polling) [48], GRAP (Group RAP) [50], GRAPO (GRAP optimized) [51] and RAMA (resource auction multiple access) [52, 53].
- Guaranteed access protocols: A la "round-robin" [55], DTMP (disposable token MAC protocol) [56]
- Les protocoles d'accès hybrides:
  - RRA (Random reservation access): PRMA (Packet reservation multiple access) [58],
  - Demand assignment protocols: C-PRMA (Centralized PRMA) [64], DQRUMA (distributed-queuing request update multiple access) [65] et MASCARA (mobile access scheme based on contention and reservation for ATM)[66].

Pour plus de détails se referer à la version anglaise de cette thèse.

Chapitre 3

Les réseaux sans-fil IEEE 802.11

3.1 Introduction

En 1997, l'IEEE adopta le premier standard pour les réseaux locaux sans-fil (WLAN), IEEE 802.11-1997 qui couvre la sous-couche du contrôle d'accès au médium (MAC) et la couche physique (PHY) du modèle OSI (Fig. 3.1). En 1999 l'IEEE adopta deux extensions pour la couche physique, lui permettant des transmissions à des débits supérieurs: 802.11a, utilisant la technologie OFDM (multiplexage en frequences orthogonales) et 802.11b, utilisant DSSS à haut débit. Dans ce chapitre nous décrivons ce standard en détail.

3.2 Mode d'opération

Dans un IBSS, les terminaux sans-fil (WT) communiquent directement entre eux. Cependant, quand la destination est hors portée de la source, des chemins multi-sauts peuvent être formés à l'aide des WTs intermédiaires entre la source et la destination. Cette fonction s'effectue au-dessus du niveau MAC.

Un BSS contient un point d'accès (AP) qui donne aux WTs un accès au réseau filaire (LAN) et assure le relai pour toutes les transmissions entre WTs; les paquets sont transmis à l'AP en premier qui les fait suivre vers la destination finale, consommant ainsi le double du débit nécéssaire avec une communication directe. Cependant, cette approche donne aux WTs la possibilité de passer en mode de veille, réduisant considérablement la consommation d'énergie.

Un avantage majeur des WLANs est le support de mobilité qu'ils fournissent d'une manière transparente aux niveaux supérieurs. Les APs prennent à leur charge la livraison des paquets aux WTs déplacés, permettant aux protocols des couches supérieures un fonctionnement normal, même sous haute mobilité.

3.3 La sous-couche MAC

La sous-couche MAC fournit les fonctions suivantes:

- La livraison fiable des paquets de données.
- Le contrôle d'accès au canal d'une manière équitable. Deux méthodes d'accès sont définies: La fonction de coordination distribuée (DCF) et la fonction de coordination par élection (*Polling*) (PCF).
- Protection des données par chiffrage.

Dans le chapitre précédant nous avons montré que les transmissions sur un canal radio sont soumises à plus d'erreurs que les transmissions filaires. Pour pallier à ce problème, on dope 802.11 de plus de fonctionnalité que 802.3. Plus précisément, quand un WT transmet un paquet, il ne peut pas détecter s'il y a eu une collision ou si le paquet est bien reçu. L'atténuation d'un signal dans le canal radio est très forte, ce qui rend la détection des collisions impossible pour un transmetteur actif. Par suite, la source doit attendre un acquitement (ACK) de la destination, confirmant la bonne réception du paquet. Ce mécanisme est similaire à celui des protocoles des couches supérieures telle que TCP pour la fiabilité des transmissions de bout-en-bout. Cependant, cette fonctionnalité au niveau MAC rend les retransmissions plus efficaces vûs les délais relativement courts.

Dans un BSS, les WTs et l'AP peuvent opérer en "mode contention" exclusivement (i.e. utilisant DCF), ou en "mode sans-contention" (i.e. PCF). Dans le premier mode, les WTs luttent pour accéder au canal avant la transmission de chaque paquet. Dans le deuxième mode, le contrôle d'accès se fait au niveau de l'AP qui élit le prochain WT à transmettre, sans avoir recours à la contention entre les nœuds. Ce dernier mode peut être utilisé en alternance avec le premier, durant des périodeis alternées qu'on appelle "période de contention" (CP) et "période sans-contention" (CFP) réspectivement.

3.3.1 La fonction de coordination distribuée (DCF)

La DCF est une fonction de transmissions asynchrones, convenable pour les transmissions de paquets de données qui n'ont pas de contraintes sur les délais. En mode ad-hoc, c'est la seule fonction de coordination possible. En mode infrastructure, elle peut être combinée avec la fonction PCF. Avec DCF, tous les WTs luttent pour accéder au canal avant chaque transmission de paquet.

Le mode basic de DCF est le CSMA [16, 10] où chacun des nœuds écoute le canal avant de transmettre. Ce protocol a deux variantes: détection de collisions (CSMA/CD) [71] et évitement de collisions (CSMA/CA). Une collision est due à deux ou plusieurs nœuds transmettant simultanément sur le même canal radio. Dans IEEE 802.3, quand un transmetteur détecte une collision, il interrompe sa transmission pour réduire le temps perdu. Ce mécanisme n'est pas applicable aux réseaux sans-fil, vu la grande différence de niveaux entre le signal transmit et le signal à détecter. La source attend donc un paquet d'acquitement (Fig. 3.4) de la destination, sans lequel la source retransmettra le paquet initial.

Pour éviter les collisions et les pertes de temps/débits correspondants, on a recours à transmettre des paquets RTS/CTS, relativement courts, respectivement avant et après les paquets de données (Fig. 3.5) [20, 21]. Ceux-ci réservent le canal pour le temps de transmission nécessaire, et évitent les longues durées de collisions éventuelles. Afin de respecter l'ordonnancement de paquets (RTS/CTS/DATA/ACK), on a recours à des espacement inter-paquets différents ($SIFS < DIFS$). Un paquet DATA doit attendre DIFS, tandis qu'un ACK attend SIFS, permettant à l'ACK un accès prioritaire au canal.

Les WTs ne transmettent pas juste après un DIFS pour éviter les collisions avec d'autres paquets qui attendent simultanément. Ils atendent un temps supplémentaire aléatoire appelé *backoff*, choisi entre 0 et CW (fenêtre de contention). Le *backoff* est décrémenté quand le canal est libre, et est blocqué quand le canal est occupé. Le paquet est transmis quand le *backoff* atteint zéro. S'il y a collision, les sources doublent les tailles de leur CWs pour diminuer les chances des futures collisions, supposant un nombre plus élevé de nœuds. Si non, en cas de bonne transmission, la source remet son CW à CW_{min}.

3.3.2 La fonction de coordination par élection (PCF)

Figure 3.7 montre le fonctionnement du PCF; l'AP lance périodiquement la fonction sans-contention (CFP) en transmettant un paquet balise contenant la durée maximale de la PCF. En suite il choisit les WTs à transmettre à tour de rôle. PCF utilise un espace inter-paquet PIFS plus court que DIFS, pour donner à la PCF la priorité d'accès absolue avant tout autre WTs (fonctionnant après DIFS). A la fin de la CFP les WTs peuvent utiliser DCF comme décrit dans la sous-section précédente, jusqu'à l'émission du paquet balise suivant, qui annonce la nouvelle période CFP.

3.4 La couche physique

IEEE 802.11 partage les mêmes bandes de fréquence (ISM et U-NII) avec d'autres technologies radio. Pour pouvoir coexister sans s'interférer, IEEE 802.11 utilise les techniques d'étalement de spectre qui fournissent les propriétés suivantes:

1. *Possibilité d'accès multiple simultané,* en utilisant des codes d'étalement orthogonaux.
2. *Protection contre les interférences des réflexions multiples:* La technique d'étalement de spectre peut prendre avantage de la diversité en temps des réflexions multiples d'un signal pour reconstruire le signal d'origine.
3. *Protection contre les écoutes indésirables:* Le signal étalé ne peut être récupéré que par la destination qui a le bon code.
4. *Réduction des interférences:* Une éventuelle interférence au récepteur sera étalée dans le spectre de fréquence, et sa puissance sera affaiblie, tandis que le signal d'origine sera reconstruit avec une grande puissance.
5. *Anti-brouillage:* C'est similaire à la proporiété précédente, sauf que l'interférence est introduite volontairement.
6. *Faible chance d'interception:* Vû que le signal transmis est étalé, et que sa puissance est affaiblie, le signal sur le canal est difficile à détecter par un écouteur hostile.

Le standard défini un code fixe et connu pour tous. Par suite, seule la réduction des interférence reste applicable parmi les propriétés citées ci-dessus. La possibilité d'accès multiple simultané reste applicable si on considère la coexistence de 802.11 avec d'autres standards.

Au niveau physique, trois spécifications ont été faites en 1997:

- L'étalement de spectre par saut en fréquence (FHSS): Le code d'étalement définit la fréquence de transmission. L'émetteur et le récepteur doivent utiliser le même code d'une manière synchrone pour pouvoir communiquer entre eux.
- L'étalement de spectre par séquence directe (DSSS): A la place des bits de données bruts, on envoi les bits corrélés avec la séquence du code, à plus haut débit (Fig. 3.9). A la réception, la procédure inverse a lieu reconstituant ainsi la séquence de bits d'origine.
- L'infra rouge (IR): La modulation utilisée est la PPM (modulation par position de pulse).

Ces trois types fonctionnent à 1 ou à 2 Mb/s (brut). La modulation à bas débit est plus résistante aux erreurs canal. Par suite les entêtes des paquets sont modulées avec 1 Mb/s.

En 1999 l'IEEE lança deux extensions pour la couche physique, offrant des débit supérieurs:

- IEEE 802.11a, qui utilise le multiplexage en fréquences orthogonales (OFDM) opérant dans la bande de fréquence libre U-NII (5 GHz). IEEE 802.11a offre des débits allant jusqu'à 54 Mb/s (brut).
- IEEE 802.11b, qui utilise le DSSS haut débit, opérant dans la bande de fréquence ISM (2.4 GHz). IEEE 802.11b offre des débits allant jusqu'à 11 Mb/s (brut).

Le même protocole MAC s'applique à toutes les couches physiques sous-jacentes: FHSS, DSSS, IR, 802.11a ou 802.11b. Cependant, les paramètres de la couche MAC diffèrent d'une couche PHY à l'autre.

3.5 Mode de veille

Quand un nœud éteint ses circuits de réception et de transmission, on dit qu'il est en *mode de veille*. Vue la grande différence entre BSS et IBSS, deux techniques de mode de veille ont été définies:

3.5.1 Mode de veille dans les architectures à infrastructure (BSS)

Le mode de veille dans les architectures à infrastructure est centralisé dans l'AP. Ce mode assure une bonne éfficacité en économie d'énergie grâce à la possibilité de stocker les paquets dans l'AP, permettant ainsi aux WTs d'aller en mode de veille pour de longues durées.

Durant son association, le WT informe l'AP de la durée de veille, et se met en mode actif périodiquement pour récupérer les paquets de ce dernier. l'AP garde les paquets à destination des WTs en veille, et les informe périodiquement en utilisant les paquets balises.

3.5.2 Mode de veille dans les architectures sans infrastructure (IBSS)

Le mode de veille dans les architectures sans infrastructure est complètement distribué. Avant de se mettre en mode de veille, un WT informe un de ses voisin de son état de veille. Il se met en mode actif périodiquement pour recevoir les paquets balises et il reste en ce mode durant la *fenêtre de message d'indication de trafic*. Tout WT ayant des paquets à transmettre à un autre en mode de veille doit l'annoncer durant cette fenêtre pourque ce dernier reste en mode actif jusqu'au prochain paquet balise. Il attend ensuite l'acquitement de son annonce avant de transmettre le paquet de données. Vû que les émetteurs et les récepteurs sont en mode actif la plupart du temps, le gain du mode de veille sans infrastructure est limité.

3.6 La sécurité dans IEEE 802.11

Vue la nature sans-fil du canal de transmission, IEEE 802.11 crypte les données au niveau MAC pour les protéger des éventuelles écoutes. Le mécanisme de cryptage introduit s'appelle WEP (Wired Equivalent Privacy) qui offre un niveau de sécurité comparable à celui des réseaux filaires. Plus tard, l'algorithme de cryptage utilisé, RC4 est devenu cassable [76, 77]. RC4 supporte des blocs de données à taille variable. Il utilise le chiffrement symétrique, dont les clés peuvent atteindre 256 octets. WEP utilise des clés de 40 bits et de 128 bits.

Le principe de WEP est montré dans Fig. 3.10. Les données sont concaténées avec leurs valeurs de vérification d'intégrité, résultante de l'algorithme CRC-232 appliqué à ces données. Le résultat de la concaténation est additionné (XOR) avec une séquence clé pseudo-aléatoire générée par RC4. Cette clé doit être changée régulièrement pour éviter son analyse. Alors le générateur de nombre aléatoire est initialisée par une valeur (IV), passée en clair sur le canal.

La clé secrète peut être choisie d'une liste établie au préalable, ou bien négociée en utilisant Diffie-Hellman par exemple. En fin, notons que les entêtes des paquets ne sont pas cryptées, seul les contenus le sont.

Chapitre 4

HiperLAN-2 et Bluetooth

Dans ce chapitre nous décrivons deux standards de réseaux sans-fil, HiperLAN-2 et Bluetooth. HiperLAN est un standard de l'ETSI pour les réseaux locaux, tandis que Bluetooth est un standard des réseaux personnels sans-fil. HiperLAN-2 peut interférer avec IEEE 802.11a puisqu'ils occupent la même bande de fréquences et utilisent la même modulation. Bluetooth, d'une autre part, occupe la même bande de fréquences qu'IEEE 802.11b.

4.1 HiperLAN-2

HiperLAN-2 [7, 78] est le standard européen pour les réseaux locaux sans-fil développé par l'ETSI. En plus des propriétés citées pour IEEE 802.11, HiperLAN-2 supporte la QoS, utilise des algorithmes de sécurité et de gestion des canaux radio améliorés.

La topologie d'un réseau HiperLAN est similaire à celle d'IEEE 802.11 en mode infrastructure. Tous les terminaux mobiles (MTs) comuniquent avec le point d'accès (AP). Les principales propriétés d'HiperLAN-2 sont les suivantes (celles non communes avec 802.11 sont précédées par un astérisque):

- *Transmission haut débit:* HiperLAN-2 utilise la technologie OFDM, très efficace dans les environnements à diversité en temps (tels que les bâtiments). Les débits atteignent 54 Mb/s.
- *(*)Orienté connexion:* HiperLAN-2 utilise des fonctions de signalisation pour établir des connexions entre les MTs et l'AP.
- *(*)Support de la QoS:* Vue la nature orientée-connexion d'HiperLAN-2, le support de la QoS devient plus facile à fournir. Les connexions peuvent avoir différentes garanties en termes de débit, délai, gigue etc. sans qu'elle ne s'interfèrent mutuellement.
- *(*)Allocation automatique des fréquences:* Au contraire des réseaux cellulaires actuels, HiperLAN-2 utilise la sélection automatique des fréquences. L'AP éoute les APs voisins et choisit son canal radio de manière a réduire les interférences.
- *Support de la sécurité:* HiperLAN-2 supporte l'authentification et le cryptage des données. Les MTs et l'AP peuvent s'authentifier mutuellement. Les algorithmes de cryptage utilisés sont DES et 3-DES, bien connu par leurs robustesses.
- *Transparence pour les couches supérieures:* HiperLAN-2 peut aussi remplacer plusieurs types de réseaux fixes (Ethernet, ATM etc.) d'une manière transparente aux couches supérieures.
- *Mode de veille:* Comme dans le mode infrastructure d'IEEE 802.11, la nature centralisée d'HiperLAN rend son mode de veille efficace en terme d'énergie.

4.1.1 La pile d'Hiperlan-2

Figure 4.1 montre les couches d'HiperLAN. Elles consistent en deux plans, hérités de la partition ISDN:

- Le plan de contrôle, responsable de l'établissement, le contrôle, la supervision et la fermeture des connexions.
- Le plan de l'utilisateur, responsable des transmissions à travers les connexions déjà établies.

HiperLAN-2 a trois couches de base: la couche physique (PHY), la couche de contrôle du lien de données (DLC) et la couche de convergence (CL).

4.1.2 La couche physique

Comme mentionné dans le chapitre précédent, HiperLAN-2 et IEEE 802.11a ont des spécifications très similaires au niveau physique, à base d'OFDM (à l'origine utilisé pour l'ADSL sous le nom de DMT, "discrete multitone"). HiperLAN-2 fonctionne dans la bande de fréquences libre 5.7 GHz, avec des canaux espacés de 20 MHz, donnant place à 19 canaux (en Europe). Chaque canal est divisé en 52 sous-porteuses: 48 pour les données et 4 pilotes, utilisées comme références pour la démodulation.

Le flux de données haut-débit est divisé en plusieurs flux bas-débit, chacun modulant une des 48 sous-porteuses, comme montré dans la Fig. 4.2.

La circuiterie est rendue simple par l'utilisation de la transformée de Fourier rapide (FFT) et la transformée de Fourier inverse rapide (IFFT) dans le démodulateur et dans le modulateur respectivement. Les modulations utilisées sont: BPSK, QPSK et 16QAM, chacune avec un taux de codage différent. L'avantage principal d'OFDM est qu'il réduit les interférences inter-symbols causées par la propagation multi-chemin des signaux.

HiperLAN-2 supporte les antennes multi-lobe qui améliorent la qualité du signal dans le réseau sans-fil. Nous verrons dans la suite que les antennes multi-lobes sont supportées au niveau de la couche DLC, qui permet l'utilisation jusqu'à sept lobes.

4.1.3 La couche de contrôle du lien de données (DLC)

Le DLC constitue un lien logique entre l'AP et les MTs. Il comporte des fonctions du le plan utilisateur et du plan de contrôle. Les sous-couches correspondantes sont décrites ici.

Le protocol MAC d'HiperLAN-2 est centralisé dans l'AP qui informe les MTs de leurs instants de transmission. Chaque MT demande des ressources a l'AP qui adapte les divisions de temps TDMA convenablement. Les communications MT-AP sont duplexées en division de temps. La structure d'un paquet MAC, de durée 2ms, est montrée dans la Fig. 4.3. Elle comprend un lien descendant (DL)(AP-MT) et un lien ascendant durant lequel les transmissions sans contention sont spécifiées dans le canal de contrôle du paquet (FCH). La contention entre MTs se fait sur le canal d'accès aléatoire (RCH), sur lequel les MTs envoient leurs demandes de ressources à l'AP. Les résultats des contentions dans le paquet précédent sont transmis sur le canal d'accès (ACH).

4.1.4 La couche de convergence

Le but de la couche de convergence est de fournir un support sans-fil transparent pour les couches supérieures et les applications. Ceci nécéssite deux fonctions: sur le plan de contrôle, la couche de convergence traduit les demandes de services des couches supérieures à la couche de contrôle DLC. Sur le plan utilisateur, elle adapte les formats des paquets des couches supérieures au plan DLC utilisateur.

Deux types de couches de convergence ont été définis: basé-cellule et basé-paquet. Le premier supporte les réseaux ATM, tandis que le deuxième supporte divers réseaux à base de paquets, tels qu'Ethernet et IEEE 802.1p pour le support de la QoS.

4.2 La technologie Bluetooth

En 1994, Ericsson Mobile Communications commença la recherche d'alternatives pour remplacer les câbles qui racordent les téléphones mobiles aux accessoirs avec des liens radio, pour éviter le poblème d'alignement des appareils infra-rouge. Le but initial était de communiquer la voix et les données entre les téléphones, les casques et les ordinateurs. La vue s'est élargie plus tard pour inclure le support de découverte de services, et les applications/profiles. Ainsi, "Bluetooth" [79, 80, 8] est devenu plus qu'un remplacement de câbles.

L'espace de connectivité personnel ressemble à une boule de communication, en mouvement, qui connecte la personne au milieu à tous les appareils voisins qui entrent dans la boule, chacun avec un service à fournir. Les appareils Bluetooth sont conçus pour être bon marché, petits, et un replacement facile à manipuler des câbles de connexion.

Le support de paquets de données donne à Bluetooth la possibilité de se connecter à des réseaux locaux (LAN) aussi, en utilisant un téléphone mobile par exemple. Ceci fait surgir la notion de "passerelles personnelles" qui connectent les appareils portés par une personne à des systèmes distants.

En 1999, Bluetooth fut choisi comme base du standard IEEE 802.15.1 pour les réseaux personnels sans-fil (PANs). Le groupe de travail IEEE 802.15.2 cherche l'effet de la coexistance entre différentes technologies 802.

A la différence d'HiperLAN et 802.11, Bluetooth couvre toutes les couches du modèle ISO (Fig. 4.4). Les couches sont réparties en trois groupes: les protocols de transport, les ptotocols intergiciels et les applications/profiles.

4.2.1 Les protocols de transport

Bluetooth utilise la bande de fréquences libre ISM 2.4 GHz. Il utilise le saut rapide en fréquences F-FHSS (1600 sauts/seconde) pour étaler le signal sur les 79 canaux de 1 MHz chacun, d'une manière pseudo aléatoire. Une fréquence centrale est définie par $f_c = 2,402 + k$ où $k = 0, ..., 78$. Les données sont modulées par saut en fréquence Gaussien, de part et d'autre de la fréquence centrale f_c, à 1 Msymbols/seconde. Le débit brut de transmission est de 1 Mb/s. Bluetooth a trois niveaux de puissance de transmission: 20, 4 et 0dBm (Classe 1, 2 et 3 respectivement).

Avec le *baseband*, les appareils Bluetooth communiquent entre eux en créant des liens, des *piconets* et des *scatternets*. Le *baseband* controle aussi l'accès au médium et s'occupe de la mise en forme des paquets aux nivaux bas.

Un périphérique Bluetooth a une adresse unique de 48 bits, le *BD_ADDR*. Cette adresse est utilisée pour établir les communications entre différents périphériques. Le *maître* peut communiquer avec sept *esclaves* au plus, formant ainsi le *piconet* (Fig. 4.5) sans avoir besoin de support d'aucune infrastructure. Tout périphérique peut être *maître* ou *esclave*, selon la création du *piconet*.

Tous les périphériques dans un *piconet* utilisent la même séquence de sauts en fréquences, d'une manière suynchronisée avec le *maître*. La séquence des sauts est définie en fonction de l'adresse du *maître* du *piconet* et du déphasage entre les horloges. Les transmissions entre *maître* et *esclaves* se font en muliplexage en temps (TDD); le *maître* utilise les intervals de temps pairs, et les *esclaves* utilisent les intervals impairs.

Les *piconets* peuvent coexister en temps et en espace d'une manière indépendante, comme dans la Fig. 4.5. Ils peuvent aussi avoir des périphériques en commun pour former un *scatternet*. Dans ce dernier cas, le périphérique en commun doit être *maître* dans un *piconet* et *esclave* dans l'autre pour pouvoir communiquer avec les deux sans problème de synchronisation.

Deux types de lien entre *maîtres* et *esclaves* existent:

- *Synchrone orienté-connexion (SCO)*: On peut avoir jusqu'à trois liens SCO dans un *piconet*. Les liens SCO sont convenables pour les transmissions de voix, à 64 Kb/s dans chaque direction (*maître-esclave*). En cas d'erreur, les paquets ne sont pas retransmis. Par contre, on peut appliquer le FEC pour les récupérer.

- *Asynchrone non orienté-connexion (ACL)*: Entre un *maître* et un *esclave*, un seul lien ACL peut être établi. Les liens ACL sont convenables pour les échanges (asynchrones) de paquets. Les paquets erronés sont retransmis, comme on peut utiliser le FEC pour les récupérer aussi.

Le cryptage est fait au niveau *baseband*. Par contre, l'échange des clés se fait au niveau LMP. Les liens ACL et SCO peuvent être cryptés avec des clés de 128 bits. Les clés sont générés en utilisant l'algorithme SAFER+ [81]. Le même algorithme est aussi utilisé pour l'authentification.

Le l'interface de contrôle (HCI) permet à des interfaces hôtes d'accéder aux couches Bluetooth. Via le HCI, les interfaces hôtes peuvent avoir ce que le LMP fournit aux couches supérieures: échange des données, commandes de configuration des liens, configuration de la puissance, commandes d'authentification etc.

Le protocol de contrôle du lien logique et d'adaptation (L2CAP) fournit plusieurs liens logiques pour les protocols intergiciels. Il multiplexe plusieurs canaux logiques (SCO ou ACL) sur le lien ACL. Il fournit aussi les mécanismes de fragmentation et de défragmentation pour adapter les grands paquets (jusqu'à 65 KOctets) au *baseband* (2744 Octets maximum).

4.2.2 Protocols intergiciels

A la différence des protocols de transport, les protocols intergiciels ne sont pas tous utilisés pour chaque communication. Quatre groupes de protocols existent: découverte de service (SDP), RFCOMM, signalisation de contrôle de téléphonie et *autres* protocols.

Avec SDP, une interface Bluetooth peut demander auprès de ses voisines sur les services disponibles et comment y accéder. SDP ne contient pas les services eux même, il n'est pas leur protocol d'accès non plus. Les informations sur les services dans SDP sont encodés avec des identificateurs universels, uniques et courts, pour économiser la bande passante nécéssaire pour les transmettre.

RFCOMM fournit une interface de communication série (similaire à RS-232) sur les couches de transport à base de paquets. Il donne aussi la possibilité de multiplexer plusieurs ports série sur un seul lien de transport. RFCOMM fournit un port série pour les applications sur-jacente d'une manière complètement transparente.

Le protocol TCS utilise les mêmes commandes de contrôle de téléphonie que les modems, appelées commandes AT, pour envoyer et recevoir la signalisation de contrôle, utilisant RFCOMM.

Les *autres* protocols ont été adoptés pour fournir les communications PPP, permettant l'utilisation d'IP sur des lignes séries, OBEX et IrMC. Ils utilisent tous le protocol RFCOMM.

4.2.3 Les profiles

Les spécifications de Bluetooth sont formées de deux parties:

- *La spécification cœur* qui définit les caractéristiques radio et les protocols de communications, décrits dans les sections précédentes.
- *La spécification de profiles:* Le domaine d'application de Bluetooth est très large. Pour s'assurer d'une bonne interopérabilité entre diverses implémentations, les profiles définissent comment les protocols de Bluetooth doivent être utilisés pour mettre une certaine application en œuvre.

La notion de profiles surgît de ISO/IEC TR10000. Les profiles ressemblent à des tranches verticales traversant les couches de protocols, comme le montre la Fig. 4.7. Ils fournissent aux couches supérieures un ensemble de procédures et des méthodes uniformes pour l'utilisations des couches inférieures. Ceci réduit les options dans l'implémentation permettant à plusieurs interfaces de différents fabriquants d'interopérer d'une manière fonctionelle.

4.3 Comparaison

Figure 4.8 montre une comparaison entre les trois standards cités dans les deux derniers chapitres: IEEE 802.11, HiperLAN et Bluetooth.

Chapitre 5

Différentiation de service

Dans ce chapitre nous présentons des mécanismes de différentiation de service au niveau MAC. Ces mécanismes pourront être utilisés pour rendre les mécanismes de différentiation de service au niveau IP, tels que *Diff-Serv* et *IntServ*, plus efficaces. A la base de ces mécanismes de différentiation, l'idée est de différentier les paramètres du standard 802.11 au niveau MAC. Le résultat de la différentiation obtenue dépend des paramètres de différentiation utilisés ainsi que des protocols de transports utilisés sur les couches supérieures (TCP ou UDP).

5.1 UDP et TCP en-dessus de 802.11

Avant d'aborder les mécanismes de différentiations, analysons tout d'abord le comportement des protocols de transport UDP et TCP en-dessus de la couche MAC du standard actuel, c-à-d sans différentiation.

La topologie du réseau simulé, avec NS, est simple (Fig. 7.4); 3 nœuds WT_i sans-fil transmettent leur paquets vers une destination dans le réseau filaire, à travers un point d'accès AP. WT_i sont à égales distances du point d'accès, et sont tous l'un à la portée de l'autre.

On place des sources de trafic UDP dans les divers WT_i. Chaque source envoi des paquets de 1100 octets chaque 5 ms, pouvant ainsi saturer le canal à elle seule. Durant une première phase, secondes $[50, 100[$, seule la source WT_1 transmet des paquets et occupe toute la bande passante du canal (Fig. 5.2(a)). Le débit utile est stable, les délais et la gigue sont relativement petits (Fig. 5.2(b)). Le taux de paquet rejeté est égal au surplus de débit que la source envoi sur le canal, dépassant la capacité de celui-là. Durant la deuxième phase, i.e. les secondes $[100, 150[$, le nœud WT_2 entre en compétition avec WT_1 pour accéder au canal et transmettre ses paquets. La Figure 5.2(a) montre qu'ils partagent équitablement la bande passante, puisque les deux ont les mêmes chances d'accéder au médium. Cependant, les délais de réception des paquets deviennent plus grands puisque chaque nœud doit attendre la fin de la transmission de l'autre avant de transmettre ses paquets. Ca peut être aussi causé par des collisions de paquets (données ou RTS) nécéssitant des retransmissions. La gigue augmente aussi durant la deuxième phase, dû à un niveau de contention supérieur à celui de la première phase. Durant la troisième phase, secondes $[150, 250[$, WT_3 entre en compétition avec les deux premiers, partageant équitablement le débit utile avec eux, et augmentant les délais et la gigue.

Quand on utilise des sources TCP au lieu de UDP, on a les mêmes observations sur les débits utiles ainsi que les délais et la gigue. Cependant, on observe un taux de paquet rejeté nul. Ceci est dû au fait que TCP est adaptatif, réduisant son débit quand le canal est congestionné. En plus, les paquets entrant éventuellement en collision seront retransmis au niveau MAC, avant que TCP ne constate la perte. Cette hypothèse est confirmée par la Fig. 5.3(a), montrant les fenêtres de congestion des sources TCP, en croissance continue. Quand un nouveau nœud entre en compétition avec les autres, on observe une diminution de la pente de la fenêtre de contention, due à l'augmentation des délais de réception des acquitements TCP. Figure 5.3(b) est un agrandissement de la fenêtre de congestion de WT_2 durant la seconde 153. Elle montre comment une éventuelle perte de paquet est retransmise au niveau MAC avant que TCP ne la constate et qu'il ne reduise la taille de sa fenêtre de congestion. Ceci n'est pas toujours le cas quand le niveau de contention sur le canal est grand. Par exemple, observons la fenêtre de contention de WT_1 quand on y place des sources TCP supplémentaires. La Figure 5.4(a) montre qu'au bout de trois sources TCP dans WT_1, en compétition entre eux mêmes et les autres WTs, les fenêtres de congestion subissent des chuttes, dues à des délais de retransmission MAC considérables, entrainant des *timeout* de TCP.

Une observations similaire se fait quand on augmente le nombre de nœuds en compétition (Fig. 5.4(b)). Au bout de treize nœuds en compétition, on commence à observer des *timeout* TCP.

5.2 Mécanismes de différentiation

Pour assurer une certaine différentiation de service offert à différents nœuds, on introduit la différentiation dans divers paramètres de la fonction de coordination distribuée (DCF) du standard:

1. Différents facteurs d'incrémentation du *backoff* pour différentes priorités .
2. Différentes tailles minimales des fenêtres de contention, CW_{min}.
3. Différents espacements inter-paquets DIFS.
4. Différentiation par limitation des tailles des paquets des différents nœuds, où l'on permet aux différentes priorités de transmettre des paquets de tailles différentes.

5.2.1 Facteurs d'incrémentation du *backoff*

Après une éventuelle collision, un WT_j multiplie la taille de sa fenêtre de contention par P_j au lieu de 2 (du standard). Quand P_j est grand, la fenêtre de contention de WT_j est grande en moyenne, parsuite il a moin de chances d'accéder au canal, et son débit utile est réduit.

Quand on applique ce mécanisme avec $P_{AP} = 2$, $P_1 = 2$, $P_2 = 6$ et $P_3 = 8$ sur des flux UDP, on obtient les résultats des Fig. 5.5(a), 5.5(b) et 5.6

On constate une différentiation des débits utiles, des délais et des taux de paquet rejeté entre les trois nœuds. L'éfficacité du système en terme de débits est la même qu'avant la différentiation. On pourra augmenter le rapport P_i/P_j pour une différentiation amplifiée, cependant ceci entraine de grandes variations des débits, des délais et des taux de rejet.

Quand on utilise des flux TCP, le comportement est différents. On observe une très petite différentiation, tel que dans la Fig. 5.7.

Ceci est dû à deux causes:

- *L'utilisation de la même priorité pour l'envoi des TCP-ACK depuis l'AP*, ce qui réduit considérablement la priorité relative entre les flux en boucle. Ceci nous a motivé a exploiter la différentiation *par-flux* au lieu de *par-nœud* plus tard dans ce chapitre.
- *Un AP lent,* ce qui réduit le taux d'envoi des TCP-ACK en retour pour les sources TCP. Ceci réduit le nombre de paquets TCP mis sur le canal, réduisant par suite le taux de collision et l'utilisation des paramètre différentiés P_i. Quand l'AP est accéléré, il envoi plus d'acquitements en retour (par unité de temps) et on observe plus de collisions des paquets TCP transmis, plus d'augmentations des fenêtres de contention, par suite une différentiation améliorée.

(Cette dernière supposition est vérifiée dans la version anglaise).

La même logique s'applique aussi aux flux TCP combinés avec des flux UDP:

- Un flux UDP ne peut pas avoir une priorité plus haute qu'un flux TCP, vu que la fenêtre de contention des nœuds utilisant TCP augmente rarement, donnant à ces nœuds une grande priorité d'accéder au canal.
- D'autre part, un flux UDP avec une basse priorité acquiert un débit utile inférieur à celui d'un TCP de hautre priorité. Le taux de collision des paquets du premier étant supérieur aux deuxième, UDP utilisera ses (grands) P_i plus souvent, reduisant ainsi ses chances d'accès au médium par rapport à TCP.

Dans la suite nous présentons une analyse mathématique pour interpréter le rapport des débits utiles observés durant la deuxième phase $[100, 150[$ dans la Fig.5.5(a), avec des flux UDP. Le débit utile de chacun de WT_1 et WT_2 est proportionnel à la probabilité qu'il accède au canal avant l'autre. i.e. que la valeur de son *backoff* soit inférieure à celle de l'autre. Ceci revient à comparer deux variables aléatoires X et Y bornées par [a,b] et [a,d] respectivement. La probabilité que X soit plus petite que Y est donnée par:

$$P(X < Y) = \begin{cases} 1 - \frac{1}{2} \times \frac{b+1-a}{d-a} & \text{if } b \leq d \\ \frac{1}{2} \times \frac{d-a}{b-a} & \text{if } b > d \end{cases} \tag{5.1}$$

La probabilité d'un collisions, i.e. les deux v.a. soient égales est:

$$P(X = Y) = \frac{1}{max(b, d)} \tag{5.2}$$

Initialement, les deux marges [a,b] et [a,d] sont égales à $[0, CW_{min}]$, et changent avec les collisions et les bonne transmissions selon le diagramme dans la Fig. 5.9. Là, une collision est représentée par une flèche pleine, et une bonne transmission est représentée par une flèche vide. En appliquant l'équation 5.1 sur chacun des 21

états de la figure, multipliée par la probabilité de chacun des états, nous donne l'espérance de succès de WT_1, i.e. $X < Y$.

Notons aussi que l'état 1 est prédominant dans cette chaine, avec une probabilité 0.79, puisque le nombre de nœuds (2) est très petit. Dans cet etat les fenêtres de contention (CW) sont égaux, limitant ainsi la différentiation. Ceci nous a conduit à considérer la différentiation des CWs les plus probables, i.e. dans l'état 1, CW_{min}.

5.2.2 Differentiation CW_{min}

Le scénario et la topologie des simulations sont les mêmes que ceux de la section précédente. Les résultats des simulations sont résumés dans la Fig. 5.11. La notation $w/x/y/z$ indique les valeurs respectives des CW_{min} de $AP/WT_1/WT_2/WT_3$.

Dans la Fig. 5.11(a), on observe le même phénomène de l'AP lent vu précédemment avec les flux TCP, qui empêche la différentiation. En accélerant l'AP, i.e. en réduisant son CW_{min}, Fig. 5.11(b) montre une différentiation considérable des flux TCP. Cependant, la même accélération de l'AP n'a aucun effet sur la différentiation des flux UDP, comme les montres les Fig. 5.11(c) et 5.11(d). Evidemment, puisque l'AP n'envoit pas de paquets en retour, son accélération n'a aucun effet sur le partage des débits utiles.

En comparant les figures "verticalment", on remarque que les flux UDP montrent une meilleure différentiation que les flux TCP pour les mêmes valeurs des CW_{min}. On pourra ainsi penser de la différentiation des flux UDP comme étant la limite que la différentiation des flux TCP pourra atteindre.

5.2.3 Différentiation DIFS

Quand on associe différents DIFS à différents nœuds, on aboutit à des débit différentiés, plus stables qu'avec la différentiation *backoff* (voir Fig. 5.13), pouvant s'appliquer aussi aux flux TCP, puisque le problème de *backoff* non-incrémenté différentiellement n'existe plus.

Ce mécanisme peut être utilisé aussi pour affecter des priorités absolues aux nœuds, l'un par rapport à l'autre, comme dans la configuration de la Fig. 5.12. Mais ceci laisse les basses priorités souffrir tant que les hautes priorités ont des paquets à transmettre.

La différentiation DIFS montre des débits plus stables, et pourra s'appliquer aux flux UDP, TCP ainsi qu'aux deux combinés.

Pour pouvoir interpréter le rapport des débits obtenus dans la phase 2, utilisant des flux UDP, dans la Fig. 5.13, on a recours à une analyse similaire à celle de la section 5.2.1. Le problème est réduit à la comparaison de 2 v.a. X_1 et X_2 limitées par [a,b] et [c,d] respectivement (voir Fig. 5.15(a)). La probabilité que $X_1 \leq X_2$ est donnée par:

$$P(X_1 \leq X_2) = \begin{cases} 1 - \left(\frac{1}{2} \times \frac{b-c}{d-c} \times \frac{b-c}{b-a}\right) & \text{if } b \geq c \\ 0 & \text{if } b \leq c \end{cases} \tag{5.3}$$

Cette équation donne le même rapport de débits obtenu par simulation, avec une erreur de 0.7%

On peut aussi généraliser l'équation 5.3 pour N nœuds (et N v.a.), comme dans la Fig. 5.15(b). Considérons m_i et M_i les limites inférieures et supérieures resp. de la v.a. X_i. Soit S l'ensemble ordonné des limites de toutes les v.a. On obtient:

$$P(X_0 \leq X_{k\neq 0}) = \sum_{s_j \in S_j, j=0,..,N} \left(\prod_{i=0}^{N} \frac{s_i^+ - s_i}{M_i - m_i} \times \delta_s \right) \tag{5.4}$$

où,
s_i^+ est l'élément qui succède s_i dans S et

$$\delta_s = \begin{cases} 1 & \text{if } s_0^+ \leq s_i \; \forall i \neq 0 \\ 0 & \text{if } s_0 \geq s_i^+ \; \exists i \neq 0 \\ 1/(n+1) & \text{sinon, où } n \text{ est le nombre} \\ & \text{de "i"s t.q. } s_i = s_0 \end{cases}$$

L'inversion de cette équation est utile pour l'affectation des divers DIFS, étant donnés les rapports des débits utiles désirés.

5.2.4 Différentiation par limitation des tailles des paquets

Quand on contraint les différentes priorités à transmettre des paquets de tailles différentes, le débit utile obtenu par une priorité est proportionnel à la taille de ses paquets, i.e.:

$$\frac{B_0}{\sum_{i=1}^{N} B_i} = \frac{L_0}{\sum_{i=1}^{N} L_i} \tag{5.5}$$

où B_i est le débit utile obtenu par WT_i et L_i est la taille de ses paquets. Cette équation est facile à inverser pour pouvoir affecter les L_i étant donnés les débits utiles désirés. Elle s'applique pour les flux UDP aussi bien que pour les flux TCP, comme le montre la Fig. 5.16.

Ici, le rapport des débits obtenus par WT_1 et WT_2 est 4/3, étant donnés que les tailles des paquets respectives sont 2000 et 1500 octets. Ce rapport est le même durant la deuxième et la troisième phase. C'est aussi le même rapport en utilisant des flux TCP, en notant la baisse des débits due au surcoût des acquitement TCP.

Ce mécanisme peut aussi être appliqué pour donner à des flux TCP la priorité sur des flux UDP et vice vers çà.

5.3 Effet du bruit sur le canal

Dans les analyses précédentes, le canal était considéré clair. Si on considère que les paquets peuvent être bruités par le canal, le taux de perte des paquets est:

$$PER = 1 - (1 - BER)^L$$

où BER est le taux d'erreur des bits. Nous notons les observations suivantes, tirés des simulations:

- Sans différentiation, les débits de divers WT_i diminuent tous proportionellement à au PER. Cette diminution est due aux paquets rejetés, et à un effet secondaire qui est l'augmentation du *backoff* losqu'un paquet est bruité, et dont l'ACK correspondant n'est pas reçu.
- Avec la différentiation *backoff*, les paquets bruités vont contribuer à l'augementation des divers backoff proportionellement aux divers facteurs P_i, puisque les paquets bruités sont considérés comme dûs à des collisions. Ceci résulte en une amplification aléatoire de la différentiation. Cet effet est evidemment indésirable.
- Avec la différentiation DIFS, les débits utiles des divers WT_i diminuent tous proportionellement au PER, par suite leur rapport (la différentiation) reste constant.
- Avec la différentiation par limitation des tailles des paquets, les différentes tailles de paquets subissent des PER différents. Par suite, la différentiation est indésirablement fonction du BER aléatoire.

5.4 Différentiation par-flux

Dans les sections précédentes, nous avons considéré que divers flux dans un même nœud partagent les mêmes paramètres de la couche MAC. Ceci limite les mécanismes de différentiation; on ne peut pas offrir des services différenciés à des flux issus d'un même nœud. C'est aussi le cas de l'AP qui enovoie les TCP-ACK à diverses sources TCP, en utilisant la même priorité.

Indépendemment des différentiations, un autre phénomène révèle la nécéssité de séparer les flux dans un certain nœud. Considérons deux flux issus du même nœud WT_0, vers deux destinations différentes WT_1 et WT_2. Si WT_1 se situe dans un environnement congestionné ou bruité, le flux correspondant va causer des incrémentations du *backoff* dans WT_0, ralentissant même l'envoi des paquets vers WT_2 qui pourrait être situé dans un environnement "clair". Ceci est du au fait que tous les paquets dans un nœud partagent la même file d'attente avant d'être transmis. Dans ce qui suit, on va distinguer entre la différentiation à file unique ou à files multiples.

5.4.1 Différentiation par-flux à file unique

La différentiation des flux UDP par *backoff*, DIFS ou CW_{min} pourrait être vue comme une introduction d'un délai moyen différentié t_i avant l'envoi d'un paquet du nœud WT_i. Le rapport des débits ainsi obtenu par WT_1 et WT_2 est proportionnel à t_2/t_1. L'utilisation de flux TCP introduit, par l'utilisation des ACK en retour, un délai constant t_0 dû à l'AP. Ceci réduit la différentiation désirée à $(t_2 + t_0)/(t_1 + t_0)$. t_0 est grand quand l'AP est lent. L'accélération de l'AP diminue t_0 et améliore la différentiation, mais t_0 ne peut être jamais annulé pour obtenir t_2/t_1 avec les flux TCP.

Une autre possibilité sera que l'AP traite les ACK différemment, introduisant de délais moyens t_{01} et t_{02} proportionels à t_1 et t_2. Ceci rend la différentiation optimale puisque:

$$t_{02}/t_{01} = t_2/t_1$$

t_i est le délai moyen introduit par CW_{min}, DIFS, ou par les *backoffs* différentiés. Dans la suite, nous considérons seulement la différentiation DIFS.

Nous tournons dix simulations avec une différentiation DIFS par-flux. La table 5.2 résume les résultats. La colonne AP_i désigne le DIFS utilisé par l'AP pour envoyer les ACK vers WT_i. La colonne WT_i désigne le DIFS utilisé par le nœud WT_i.

Les simulations I à V montrent que, tant que l'AP contient des éléments lents, la différentiation entre flux est mauvaise. Les simulations VI à X montrent des cas où l'AP a la même vitesse moyenne. La différentiation s'améliore tant que le rapport des vitesses d'envoi des ACK AP_i/AP_j se rapproche du rapport des vitesses d'envoi des paquets TCP WT_i/WT_j.

5.4.2 Différentiation à files multiples

Afin de réduire l'interférence entre divers flux partageant la même MAC d'un nœud, nous considérons différentes files pour différents nœuds. Ceci revient revient à supposer que chacune des files est remplacée par un WT, et le débit utile observé par chaque nœud est la somme des débits obtenus par ses files. Reste à noter qu'un nœud à files multiples possède en moyenne un plus grand nombre de paquets par *time_slot* qu'un nœud a file unique, utilisant ainsi une contention plus “agressive”.

Chapitre 6

Autres travaux

Dans les dernières années, plusieurs propositions ont enrichi l'état de l'art avec des propositions de support de la QoS dans les réseaux sans-fil [1, 2, 3, 4, 93, 94, 95, 96, 97, 98, 99, 12, 100]. Elles peuvent être classées selon plusieurs critères: centralisés ([12]) ou distribués, pour les réseaux à saut-unique ([2]) ou multi-sauts ([3, 97]), avec routage à maintient d'état ([97]) ou sans maintient d'état, le support de la QoS se fait au niveau IP ([93]) ou au niveau MAC, etc. [101] cite ces aspects avec plus de détails.

Dans les sections suivantes nous allons décrire quatre de ces approches qui sont relativement proches de nos travaux.

6.1 La proposition de standard IEEE 802.11e [1]

A l'IEEE on travail actuellement sur l'extension de 802.11 pour supporter la QoS au sein du groupe de travail E. Elle utilise une combinaison des mécanismes de [83, 85, 84, 22] décrits dans les chapitres précédents.

Le standard proposé introduit le DCF amélioré, EDCF, et une fonction de coordination hybride HCF. On appelle *nœuds améliorés* les nœuds qui supportent 802.11e, et qui peuvent agir comme des contrôleurs centralisés. Un contrôleur centralisé est appelé *coordinateur hybride* (HC) et réside dans l'AP. EDCF peut être appliqué durant les CPs seulement, tandis que les CFPs sont utilisés en combinaison avec CPs. HCF est utilisé durant les CPs et les CFPs.

6.1.1 La fonction de coordination distribuée améliorée (EDCF)

Figure 6.1 montre les propriétés de la sous-couche MAC d'IEEE 802.11e. Une sous-couche MAC supporte jusqu'à huit catégories de trafics (TCs) chacune avec un *backoff* indépendant, un AIFS (qui remplacent les DIFS du standard actuel), un CW_{min} et un facteur de persistence PF (au lieu de 2) indépendants de ceux des autres catégories de classes. Les paramètres $AIFS[TC]$, $PF[TC]$, $CW_{min}[TC]$ et $CW_{max}[TC]$ peuvent être distribués par le coordinateur hybride en utilisant les paquets balises.

On peut assimiler les TCs à plusieurs nœuds vituels dans un nœud. Pour résoudre les problème des *collisions virtuelles* entre les TCs d'un nœud on a recours à un ordonnanceur qui résoud les collisions en donnant l'accès aux classes supérieures. Le paquet transmis pourra toujours faire une collision avec un paquet d'un autre nœud, suite à laquelle les deux nœuds multiplient leur *backoff* par les PFs respectifs.

6.1.2 La fonction de coordination hybride (HCF)

HCF fonctionne durant les CPs et les CFPs. Durant les CPs, une station transmet son paquet quand son *backoff* atteint zéro ou quand elle reçoit un message spécial de *polling*. Au cours de la CFP, le HC peut spécifier l'instant et la durée de transmission du nœud qu'il invite à transmettre.

Pour faire une demande de *polling*, les nœuds utilisent un mécanisme similaire à celui utilisé dans HiperLAN-2 sur le canal d'accès aléatoie RCH. Il est appelé *contention contrôlée* dans 802.11e. Un paquet de contrôle définit un nombre d'opportunités de contention contrôlée et un masque de filtrage contenant les TCs dans lequel les demandes de resources peuvent être placées. Un nœud qui a des paquets à transmettre qui correspondent au filtre de TC choisit un interval d'opportunité et y transmet sa demande de ressources contenant le TC et la durée de transmission. Le HC génère donc un autre paquet de contrôle en retour pour acquiter la réception.

Quand plusieurs BSSs se chevauchent, les paquets de *polling* de diffŕents APs pevent entrer en collision, ce qui dégrade les performances des BSSs concernés. Plusieurs solutions sont discutées parmi lequelles la sélection dynamique de fréquence, utilisé également dans HiperLAN-2.

L'évaluation de performance de EDCF dans plusieurs scénarios, de EDCF et HCF, ainsi que des BSSs chevauchées se trouvent dans [73].

6.2 Black burst [2]

Pour le support des applications temps-réel, tel que les délais limités de bout-en-bout, [2] propose un protocole d'accès multiple appelé *Black burst* (BB). BB peut être déployé en dessus des implémentation de 802.11 sans devoir changer les procédures d'accès pour les nœuds avec des paquets de données, et avec des changements mineurs pour les nœuds avec des trafics temps-réel.

Avec BB, les nœuds à trafic temps-réel luttent pour accéder au canal en envoyant des pulses d'énergie, dont les durées sont proportionelles aux différents temps d'attente des paquets dans les files des nœuds, avant de détecter le canal libre. Si un nœud constate qu'il a le plus long BB, il transmet son paquet, et prépare la transmission du paquet suivant dans un temps t_{sch}, qui est le même pour tous les nœuds. Si son BB n'est pas le plus long, le nœud attend le canal se libérer de nouveau pour envoyer un plus long BB. Ce mécanisme converge ainsi vers un TDMA distribué, sans avoir besoin de synchronisation ni d'attribution explicite des temps de transmission. Il assure aussi un accès sans collision et donne aux paquets temps-réel la priorité sur les paquets de données. Le mécanisme d'accès aléatoire est donc remplacé par BB. La performance de ce mécanisme est proche de celle d'un multiplexage en temps parfait. Cependant, il ne s'applique pas aux les réseaux avec des nœuds cachés.

Figure 6.2 donne un exemple de deux nœuds qui utilisent BB pour accéder au canal. Les nœuds 1 et 2 ont leur paquets retardés par la transmission de paquet de données, à la fin duquel ils attendent un certain temps t_{med} puis commencent à transmettre leurs BBs. Nœud 1 transmet un BB plus long que celui du nœud 2 puisqu'il a attendu plus longtemps. Nœud 1 commence la transmission et nœud 2 attend le canal libre pour un temps t_{med} apres la fin de la transmission pour envoyer un nouveau BB plus long.

Les auteurs proposent aussi une méthode qu'ils appellent *enchaînement* (*chaining*) qui vise à réduire le nombre de nœuds en contention pour accéder au canal d'un LAN sans-fil. Après avoir transmis son paquet, un nœud invite/appelle un autre à transmettre le sien juste après un temps t_{short} pour éviter que d'autres nœuds commencent la transmission de leurs BBs, et la chaîne reste en ordre. Le mécanisme d'enchaînement améliore l'utilisation du canal. Les chaînes peuvent être divisées en sous-chaînes (réduisant ainsi l'éfficacité), ou concaténées.

Les résultats de simulation montrent que BB peut supporter plus de nœuds temps-réel que CSMA/CA, avec plus de stabilité vu l'absence de collisions. En outre, le nombre maximal de nœuds temps-réel qui peut être supporté augmente avec le nombre de nœuds dans une chaîne, puisque l'enchaînement réduit les surcoûts. Les auteurs montrent aussi que les délais et les gigues sont aussi réduits avec BB par rapport à CSMA/CA, même sous grandes charges. Par contre, l'enchaînement n'apporte aucune amélioration aux délais des paquets.

6.3 Busy tone priority scheduling (BTPS) [3]

Les réseaux ad-hoc sont typiquement des réseaux multi-sauts, où tous les nœuds n'entendent pas nécéssairement tous les autres nœuds. Le problème des nœuds cachés rend l'ordonnancement dans les réseaux multi-hop très différent de celui dans les réseaux locaux sans-fil [3]. Par suite, BB ne peut pas être utilisé dans les réseaux ad-hoc multi-hop avec des nœuds cachés, comme vu dans la section précédente. En outre, les auteurs dans [3] réclament que les mécanismes de différentiations dans [1, 83], i.e. différentiation des DIFS, des *backoff* et des CW_{min} offrent des résultas sous-optimaux. [3] propose un nouveau mécanisme d'ordonnancement qui utilise deux signaux à bandes étroites pour guarantir l'accès au canal pour les nœuds prioritaires. Le protocole proposé est appelé BTPS (*busy tone priority scheduling*).

L'exemple de la Fig. 6.3 montre comment BTPS fonctionne dans une topologie à trois sauts. Le nœud 0 a des paquets de haute priorité à envoyer à 1, tandis que 2 a des paquets de basse priorité à envoyer à 3. Le problème consiste à informer 2 de la transmission de paquets haute priorité par 0, pour que 2 diffère sa transmission de basse priorité. En plus, quand 0 n'a pas de paquets à envoyer, 2 devrait maximiser son débit. Pour aboutir à ces fins, on utilise deux signaux à bandes étroites, BT1 et BT2; quand 1 a un trafic haute priorité à transmettre, il transmet le signal BT1 toutes les M unités de temps durant le DIFS et le *backoff* avant d'accéder au canal. M est un paramètre du protocole. Chaque nœud qui entend BT1, i.e. nœud 1 dans notre exemple, transmet le signal BT2 toutes les M unités de temps (voir Fig. 6.4). Ceci garantit que 0 transmet ses paquets haute priorité en premier. Tout nœud qui entend BT1 ou BT2 diffère sa transmission. Quand 0 n'a pas de paquets à transmettre, le mécanisme ne présente pas de perte en éfficacité puisque le nœud 2 peut avoir toute la bande passante, sans surcoût dû à l'utilisation de BTPS.

Pour évaluer les performances de BTPS, les auteurs le compare à notre mécanisme de différentiation DIFS et au standard actuel IEEE 802.11. Les résultats montrent que les deux mécanismes de différentiation fonctionnent mieux que IEEE 802.11, et que BTPS a moins de surcoûts que DIFS, le rendant plus éfficace en débit utile, surtout pour les petits paquets où le surcoût de la différentiation DIFS est grand.

Les avantages de BTPS par rapport à la différentiation DIFS sont:

- BTPS peut fournir des garanties absolues, sans surcoût supplémentaire (ce qui est le cas pour DIFS).

- En cas de collision entre deux (ou plusieurs) paquets de haute priorité, leur priorité reste la même.

Cependant, les inconvénients de BTPS par rapport à DIFS sont:

- BTPS a besoin de deux fréquences hors-bande, nécéssitant des transmetteurs plus complexes.
- BTPS ne fournit que deux classes de priorité.

En outre, vu que l'atténuation des signaux dépend typiquement des fréquences, les signaux hors-bande ont des protées différentes que celle du signal des données. Ce fait réduit l'efficacité de BTPS.

6.4 MAC virtuel (VMAC) source virtuelle (VS)[4]

A la différence des approches des sections précédentes, les algorithmes ici ne visent pas à fournir la différentiation de service, mais plutôt l'estimation des mesures de performances, utilisées au niveau applicatif pour le contrôle d'admission. Les auteurs dans [4] proposent deux nouveaux algorithmes pour le mode DCF d'IEEE 802.11: MAC virtuel et source virtuelle. VMAC observe passivement le canal radio et établit des estimations locales des délais, des gigues, des collisions et des pertes de paquets, en prenant en compte les conditions locales et les interférences des cellules voisines. En utilisant les estimations de VMAC, VS ajuste les paramètres de l'application et détermine si une nouvelle session demandant un certain niveau de service peut être admise.

Les auteurs commencent par exploîter la différentiation CW_{min} pour un petit nombre de nœuds. Les simulations montrent une bonne séparation, du point de vue délais, entre les deux classes: des flux CBR avec des contraintes de délais, et des flux TCP *best-effort*. La différence de délais entre les deux classes reste considérable, même sous grandes charges. Cependant, les délais augmentent avec la charge sur le canal radio pour les deux classes. Le débit utile n'est pas entièrement consommé par les flux de haute priorité quand ils devaient saturer le canal. Une petite partie du débit est toujours utilisée par les flux TCP (basse priorité). Les applications temps-réel demandent souvent des délais limités et des priorités absolues, que VMAC et VS tentent d'estimer.

Pour estimer la capacité libre du canal, on mesure le temps libre après DIFS. VMAC et VS fonctionnent en parallèle avec l'application réelle et le protocol MAC du nœud concerné pour estimer le niveau de service. Ils émulent le comportement d'un trafic réel et son MAC en générant des paquets virtuels. Cependant, on ne transmet pas de données réellement. Les paquets sont estampillés et mis dans une mémoire tampon virtuelle. En suite les paquets sont mis en ordre pour être transmis sur le canal (après un cerain *backoff*) comme si on utilisait un MAC réel. Cependant, au lieu de transmettre le paquet virtuel, le VMAC estime la probabilité de collision si ces paquets étaient envoyés. En cas de collision (détection d'un paquet transmis sur le canal), VMAC double le *backoff* comme un MAC réel aurait fait. S'il n'y a pas de collision, VMAC estime le délai total et le surcoût. Tous les autres aspects MAC sont émulés, e.g. les retransmissions, l'incrémentation et la décrémentation du CW etc.

Les délais estimés sont très proches des délais simulés sur une vaste marge de charges, avec ou sans différentiation. Par suite l'approche est convenable pour l'évaluation de la capacité admissible du canal pour les trafics temps-réel. Notons enfin que VMAC et VS peuvent être appliqués également à nos mécanismes de différentiation vus dans le chapitre précédant pour fournir un contrôle d'admission pour la différentiation de services.

Chapitre 7

Amélioration d'IEEE 802.11 dans les environnements bruités

7.1 Introduction

Dans IEEE 802.11, quand un paquet entre en collision avec un autre, les deux paquets sont retransmits dans des temps aléatoires dans le futur, choisis dans des fenêtres de contention (CWs) plus grandes. Cette collision est indiquée implicitement quand un nœud ne reçoit pas d'acquitement avant un certain temps limite. Cependant, dans les communications sans-fil, une perte de paquets peut être due à une collision aussi bien qu'au bruit sur le canal sans que le nœud source puisse distinguer l'une de l'autre. Par suite le nœud source augmente sa fenêtre de contention. Ce comportement est certainement sous-optimal: Les CWs ne doivent pas être augmentées, pour éviter les collisions quand les pertes sont dues au bruit.

Dans ce chapitre nous commençons par l'analyse du problème, et nous proposons par la suite une méthode basique pour la distinction statistique entre pertes par collision ou pertes dues au bruit canal. Nous cherchons aussi une solution optimale d'incrémentation des CWs qui ajuste leurs tailles convenablement pour réduire le surcoût dû au bruit, tout en évitant les collisions. Nous évaluons les performances de cette méthode par simulation, en la comparant au standard actuel, ainsi qu'a une méthode théorique optimale.

7.2 Motivations

La première simulation montre le comportement des CWs en présence du bruit canal. Figure 7.1 montre la distribution des tailles des CWs d'un nœud donné en présence d'autres nœuds: Quand le nombre de nœuds augmente, les tailles des CWs augmentent aussi pour éviter les collisions qui deviennent plus probables. Considérons le cas de deux nœuds, Fig. 7.2 compare la distribution des CWs avec et sans l'application d'un taux d'erreur paquet (PER) de 10%. En présence du bruit canal, la distribution des CWs est similaire à celle de cinq nœuds dans un canal sans bruit: les nœuds essaient d'éviter les collisions en augmentant les tailles de leurs CWs, inutilement.

La simulation suivante montre les effets secondaires des incrémentations des CWs à cause du bruit. Deux nœuds WT_1 et WT_2 sont placés à égales distances d'un point d'accès (AP) connecté par câble à un nœud S. Aucune congestion n'est possible sur la connexion filaire.

Dans le premier scénario, deux sources de trafic sont placées en S. A la seconde 50, la première source commence à transmettre des paquets à destination WT_1, passant par l'AP. A la seconde 150, WT_2 commence une transmission vers WT_2, en passant par le même AP. Chaque source émet des paquets UDP de 1100 octets toute les 5 ms. Figure 7.1 montre les résultats.

En absence du bruit sur le canal (les 2 premières lignes de la table 7.1), le trafic à destination WT_1 peut obtenir toute la bande passante disponible durant la première période. On observes quelques collisions avec des paquets de routage, ce qui fait incrémenter les CWs. Durant la période II, les transmissions vers WT_2 commencent et la bande passante du lien est équitablement partagée entre les deux flux. On n'observe pas de collisions considerables, puisque l'AP est le seul nœud à transmettre.

Considérons maintenant le cas où l'on applique un taux de perte de bits (BER) de 10^{-4} sur le canal radio, on peut tirer plusieurs observations (se référer aux deux lignes du milieu dans la table 7.1). On constate que les CWs atteignent de grandes valeurs très souvent, bien qu'il n'y a pas plus de collisions que dans le scénario précédent. Ce qui dégrade les performances considérablement. L'incrémentation des CWs, qui est sensée réduire les collisions, introduit plusieurs effets secondaires, comme les débits réduits et des délais considérables. En effet, avec un $BER = 10^{-4}$ et des paquets de 1100 octets, le taux d'erreur paquet (PER) est de 58%. Cependant, la réduction du débit observée est de 69%, bien supérieure à 58%, à cause des grandes tailles des CWs.

Si on considère le cas où seuls les paquets à destination WT_1 sont corrompus par le bruit (les deux dernières lignes de la table 7.1). Deux observations à tirer. Premièrement, puisque les deux flux partagent la même couche MAC de l'AP, qui ne distinguent pas entre les deux flux, une incrémentation du CW du premier flux (et les retransmissions successives) font attendre les paquets à destination WT_2 également, dans la file d'attente de l'interface. Ce qui a réduit le débit reçu par WT_2 de 106526 O/s à 48721 O/s, bien que aucun des paquets à destination WT_2 n'est atteint par le bruit.

Deuxièmement, le débit du flux à destination WT_1 a augmenté de 31759 O/s (dans "global noise") à 51784 O/s (dans Noise/1), bien que ses paquets ont le même taux d'erreur dans les deux cas. En effet, le trafic vers WT_2 n'est plus atteint par le bruit, par suite il ne ralentit plus l'AP partagé. L'AP incrémente son CW moins souvent, retransmet moins de paquets, par suite les paquets des deux flux sont envoyés plus vite qu'avec "global noise", augmentant ainsi le débit.

Le débit utile peut être amélioré utilisant les mécanismes correcteurs d'erreurs (FEC). Cependant on va se focaliser sur l'élimination des effets secondaires des pertes dues au bruit canal, qui est l'incrémentation inutile des CWs. On va détailler cet aspect dans la suite.

7.3 Analyse du problème

Figure 7.5 montre la transmission d'un seul paquet et la réception de son acquitement, avec les intervals de temps entre-paquets (IFS) correspondants et le *backoff*. Soit T le temps total et L la taille du paquet. Le débit utile (udr) est donné par:

$$udr(L) = \frac{1}{T} \times L = \frac{1}{T_{DIFS} + T_{bkf} + T_{pkt} + T_{SIFS} + T_{ACK}} \times L$$

où $T_{DIFS}, T_{bkf}, T_{pkt}, T_{SIFS}$ et T_{ACK} sont le temps de DIFS, temps du *backoff*, la durée de transmission du paquet, le temps de SIFS et le temps de transmission de l'ACK respectivement.

La Fig. 7.6 montre qu'avec un $PER = 0$, le débit utile moyen $E[udr(L, PER)]$ converge vers 1 quand L augmente infiniment, c'est-à-dire le surcoût de transmission devient négligeable quand la taille du paquet augmente. Elle montre aussi la décroissance des débits utiles pour un un PER donné, due à l'augmentation des CWs, sans prendre les retransmissions en considération. Par exemple, pour $L = 100$ Octets et $PER = 0.1$, $E[udr]$ est proche de 0.45, c-à-d 10% de moins qu'avec un canal sans bruit ($PER = 0$) à cause des délais supplémentaires introduits seulements. Donc, si on arrive à éviter l'incrémentation des CWs quand les pertes sont dues au bruit, on peut récupérer jusqu'à 10% du débit.

Pour toutes les valeurs de L, les courbes tendent vers zéro quand PER tend vers 1, c-à-d quand le PER augmente, les CWs augmentent et le surcoût de transmission des paquets devient prédominant.

7.4 Proposition

Considérons la même topologie que celle de la section 7.2 mais dans laquelle on applique un autre scénario: On place les sources de trafic dans les différents WTs plutôt que dans le nœud fixe S. Ainsi on évite les influences mutuelles entre les flux qui partagent le même MAC de l'AP, comme vue dans la section 7.2.

A la seconde 50, WT_1 commence la transmission de paquets UDP de 1100 Octets, toute les 5 ms. A la seconde 150, WT_2 commence la même procédure. Les résultats (Table 7.2) ne montrent aucune influence mutuelle entre les deux flux, puisqu'il ne partagent pas la même couche MAC. Par contre, ils luttent pour accéder au canal indépendemment. Durant la période II, quand le canal est clair, les deux flux partagent équitablement la bande passante. C'est aussi le cas quand on applique le bruit canal sur les deux sources. Quand seul WT_1 est exposé au bruit, il voit son débit diminuer sans influer sur le débit de WT_2, comme c'était le cas de la section 7.2. Les effets secondaires et l'incrémentation des CWs restent encore visibles.

Une incrémentation de CW peut être causée par une collision ou par du bruit. Une remise du CW à sa valeur initiale (31) est causée par une bonne transmission de paquet. Notre but est de pouvoir distinguer les pertes dues aux collisions de celles dues au bruit. On réduit ainsi les évènements causant l'incrémentation du CW, réduisant la taille moyenne des CWs, ce qui donne de meilleures performances sans nécéssairement augmenter le taux de collision.

Notre approche est statistique, sans interaction avec les couches réseaux plus sous-jacentes. L'idée générale est d'estimer l'état du canal et des nœuds participants en observant l'évolution des CWs.

Dans notre première simulation, nous supposons que le taux de perte dû au bruit reste constant avec le temps, e.g. le cas d'un terminal assez loin de l'AP.

7.4.1 Pertes dues au bruit

Considérons d'abord le cas où seul le bruit peut causer des pertes, pas de collisions possibles. C'est le cas de la période I du scénario 2 ci-dessus. Par suite, le taux de perte reste constant, quelle que soit la taille du CW. Pour chacune des valeurs i du CW nous attribuons deux compteurs: le nombre de paquets transmis tx_i et le nombre de paquets perdus, d_i. Le taux de perte observé pour un CW de taille i est donc d_i/tx_i. Si cette dernière valeur reste constante sur toutes les valeurs i du CW, nous pouvons déduire que les pertes sont dues au bruit, et que le CW ne devrait pas être incrémenté. Si les pertes étaient dues aux collisions, le taux d_i/tx_i devait diminuer en incrémentant les CWs. En pratique, nous considérons que le taux de perte est constant si l'écart type des d_i/tx_i reste inférieur à une valeur ϵ donnée. Ce paramètre sera plus utilisé plus tard dans cette section. Figure 7.9 et Table 7.3 montrent les résultats de cette méthode quand on l'applique durant la période I des simulations. Les résultats des simulations précédentes sont gardées pour comparaison.

Durant la période I, nous pouvons voir que la taille moyenne des CW est plus petite, ce qui améliore considérablement le débit. Celui de WT_1 a augmenté de 63000 O/s à 7738 O/s.

Si on applique cette méthode basique à la période II, où les collisions peuvent aussi avoir lieu, le résultat est sous-optimal, mais on le montre pour pouvoir l'analyser; au début de la période II, WT_1 a déjà fait des statistiques sur le taux de perte pour les différentes tailles du CW, ce qui lui a permis d'apprendre que la plupart des pertes sont dues au bruit. Par suite, WT_1 limite les tailles de ses CWs, inférieures à celles de WT_2, et obtient plus de débit que ce dernier. Après un certain moment, WT_2 a plus de valeurs des taux de perte qui convergent, dont la variation est inférieure à ϵ. Ainsi, il adopte des petites tailles de CW, lui donnant ainsi un débit similaire à celui de WT_1.

7.4.2 Pertes dues au bruit et aux collisions combinés

Quand on considère plusieurs WTs (période II du scénario 2), les collisions et le bruit canal coexistent. Le mécanisme de base déjà présenté n'est pas approprié à cette situation. Considérons les deux cas extrêmes suivants:

- Pertes dues au bruit et pertes dues aux collisions totalement corrélées (Fig. 7.10-a):

  C'est le cas où tous les paquets atteints par le bruit font des collisions, ou tous les paquets qui font des collisions sont atteints par le bruit, selon quel taux est plus grand. Dans les deux cas, bruit et/ou collision, le paquet est perdu. Le taux de perte total est par suite le maximum des deux taux. Le mécanisme décrit précédement verifie si le taux de perte (total) est constant quelle que soit la taille des CWs, ce qui n'est pas le cas. Par suite l'incrémentation du CW est illimitée. Cependant, la taille du CW ne devrait pas dépasser la valeur correspondante au point A, c-à-d cw_A. Au delà de cw_A, les pertes sont dues au bruit et l'incrémentation du CW n'est plus utile. Par suite, au lieu de calculer la variation de taux d'erreur sur tous les CWs, le mécanisme devrait examiner la variation depuis les valeurs supérieures des CWs vers les valeurs inférieures. Quand la variation dépasse ϵ, le point A est détecté, au delà duquel on ne devrait plus incrémenter le CW.

- Bruit et collision totalement indépendants (Fig. 7.10-b):

  C'est le cas où aucun des paquets atteints par le bruit fait une collision et inversement. Par suite le taux de perte total est la somme des deux taux. Puisqu'on considère que le taux de perte dû au bruit est constant sur toutes les valeurs du CW, le taux de perte total est une transition verticale de la courbe de taux de perte due aux collisions. Quand on applique le mécanisme amélioré décrit ci-dessus, on trouve un point B correspondant à la limite supérieure du CW, cw_B. cw_B est relativement grand, à cause de la transition verticale de la courbe des collisions, due au bruit. Si on envisage l'élimination de l'effet du bruit, on devrait appliquer une transition verticale descendante de la courbe des collisions, d'un montant égal au taux de perte bruit, et le point B est ramené au point B'.

  Après l'élimination de l'effet du bruit, B' est le point qui donne le même taux de perte que le point B' avec les pertes bruit et collision, cependant, avec un CW plus petit. Les petites valeurs du CW correspondant a B' donne un meilleur débit que celles du point B.

Les valeurs de cw_B et $cw_{B'}$ sont respectivment les limites supérieure et inférieure de la taille optimale du CW maximal. Trouver la valeur optimale exacte entre ces deux limites dépend de la corrélation entre de taux de perte bruit et taux de perte collision.

Quand on adopte des limites résuites de CW, telles que $cw_{B'}$, afin d'augmenter udr, nous augmentons aussi le nombre de collisions et les retransmissions correspondantes, ce qui réduit udr. Par suite, la taille optimale du CW maximal est un compromis entre le taux de collision et le surcoût du *backoff*.

Quand on applique ce mécanisme amélioré sur un troisième scénario, où tous les deux WTs commencent leur transmissions à la seconde 50, on obtient les résultats de la Table 7.4. Dans ce scénario, on évite la période

de non-équité décrite ci-dessus en lançant les deux flux en même temps. Avec ce mécanisme, on obient un débit de 80404 O/s au lieu de 73040 O/s. Un mécanisme parfait donne un débit de 87423 O/s. Notons que le taux d'erreur de bit considéré est relativement grand, pour but d'amplifier l'effet de bruit sur les CWs.

7.4.3 Environnements dynamiques.

Les paramètres utilisés jusqu'à présent sont tous considérés constant avec le temps. Cependant, les WTs peuvent changer de positions et le taux d'erreur bruit change en conséquence. Le nombre de WTs peut aussi changer, ce qui fait varier le taux de collision. Les mécanismes déjà cités peuvent s'adapter à ces variations quand:

- On limite l'historique des compteurs de transmissions et de pertes à une fenêtre de temps limitée dans le passé. Cette fenêtre ne peut pas être très large, puisque les "vieilles" informations ne sont pas utile quand les conditions du canal radio changent vite. La fenêtre ne doit pas être très étroite non plus pourque les statistiques sur le taux de perte soient toujours valides.

  L'approche de la fenêtre de temps peut aussi être remplacée par le filtre:

  $$new\_avr\_d_i/tx_i = \alpha \times d_i/tx_i + (1-\alpha) \times old\_avr\_d_i/tx_i$$

  où α doit être optimisée. L'approche du filtre au lieu de la fenêtre de temps réduit la taille de mémoire nécéssaire pour nos statistiques.

- On change la valeur de ϵ en fonction de la variation du taux de perte dû au bruit. Par exemple, si on considère la transmission de paquets de différentes tailles, un BER constant provoque un PER variable. ϵ doit être supérieure à cette variation de PER.

- On rafraichit le mécanisme périodiquement. Considérons le cas de la Fig. 7.10-a, si le niveau du taux de perte bruit baisse, le point optimal A tend à changer vers des valeurs de CW élevées. Par suite on aura besoin d'éviter les collisions encore plus. Par conséquent, notre mécanisme doit être rafraîchi périodiquement pour trouver de nouvelles limites de CW optimales. Le mécanisme doit être rafraîchi occasionellement aussi, quand une valeur des $d_i/tx_i(i <= opt.max.CW)$ change, due à un changement du niveau de bruit par exemple.

Chapitre 8

Amélioration d'IEEE 802.11 dans les environnements congestionnés

8.1 Introduction

Dans le chapitre précédent nous avons introduit des mécanismes pour éviter les des augmentations inutiles des fenêtres de contention (CW) dans 802.11. Ceci est le cas des environnements bruités où un paquet peut être rejeté à cause du bruit sur le canal, ce qui ne nécessite pas l'augmentation de la taille du CW, sensée eviter les collisions. Ce mécanisme risque des collisions et des retransmissions, pour pouvoir gagner un peu du temps perdu par les *backoffs*. Le risque est grand sachant que le temps de *backoff* gagné est considérablement inférieur au temps pour retransmettre un paquet perdu par collision.

Par suite, inversons le raisonnement; comment pourra-t-on éviter les collisions, tout en prenant le risque de grands *backoffs* ?

La réponse est: en évitant les décrémentations brusques des CWs. Typiquement, quand les niveau de congestion est haut, 802.11 remet son CW au CW_{min} après chaque bonne transmission, oubliant l'historique des collisions et recommençant l'expérience de nouveau. Sachant que le niveau de congestion varie lentement, ceci se traduit par de nouvelles collisions et de nouvelles retransmissions, jusqu'à atteindre la bonne taille du CW. Une décrémentation plus lente du CW après une bonne transmission pourra mieux éviter les collisions.

L'idée de décrémentation lente fut introduite dans [22] qui propose une multiplication par 1.5 du CW après chaque collision et une décrémentation de -1 après chaque bonne transmission. L'analyse des décrémentations lentes y était limitée aux décrémentations linéaires, et l'évaluation des performaces était réduite.

Dans ce chapitre nous développons l'analyse des décrémentations linéaires et multiplicatives, tout en évaluant leur performances, en considérant plusieurs métriques et différentes topologies de réseaux.

8.2 Destination unique

Considérons 50 WTs répartis uniformément dans une région de 100x100m. WT_1 est la destination des flux sortants des 49 autres. A la seconde 44, on commence par augmenter le nombre de WTs qui entre en compétition à raison de un WT toutes les 2 secondes. Tous les nœuds sont à la portée l'un de l'autre. A la seconde 150, le niveau de contention diminue brusquement, i.e. tous les WTs sauf un arrêtent les transmissions de leurs paquets. La simulation se termine à la seconde 260.

Quand le nombre n de WTs augmente, l'idéal sera que chacun obtienne $1/n$ du débit disponible. Cependant, les collisions augmentent avec n, suivis par les retransmissions, diminuant ainsi le débit total, comme le montre la courbe pointillée dans la Fig. 8.1, secondes 44-150. Quand on décrémente le CW plus lentement après chaque bonne transmission, e.g. en multipliant CW par 0.8, on évite mieux les futures collisions. La courbe continue dans la Fig. 8.1 montre un gain qui arrive jusqu'à 53% quand le niveau de contention est haut, i.e. à la seconde 150. Quand on décrémente le CW lentement, on évite les collisions et les multiples retransmissions très coûteuses, au détriment des grands *backoffs* relativement moins coûteux. En plus, on observe moins de variations du débit.

Le plus grand surcoût de la décrémentation lente se manifeste quand le CW est grand mais le niveau de contention est petit, comme c'est le cas à la seconde 150. Cependant, la simulation dans la Fig. 8.1 à la seconde 150 ne montre aucune baisse en débit. En effet, après quelques bonnes transmissions, le CW recupère sa valeur minimale en un temps bien réduit (qui dépend du taux d'envoi des paquets).

L'analyse précédente n'est pas complètement correcte; la baisse du niveau de contention n'est pas aussi brusque qu'il le faut pour montrer le vrai surcoût du mécanisme. En effet, les files dans les nœuds sont de taille 50. A la seconde 150, les sources arrêtent l'envoi des paquets, mais le nombre résiduel de paquets dans les files

assure une transition "douce" du niveau de contention (qui traine jusqu'à la seconde 168 au lieu de 150, vue avec la courbe pointillée)

Pour éviter cet effet, on raccourci les files à 2 paquets par WT afin d'avoir une baisse du niveau de contention plus brusque, qui révèle mieux le surcoût du mécanisme. La Fig. 8.2, seconde 150, montre que le mécanisme récupère vite les petites valeurs du CW sans aucune dégradation du débit utile total.

La réponse du mécanisme à la baisse brusque du niveau de contention, à la seconde 150, pourra être considérée comme la réponse à la frequence maximale de variation du nombre de WTs. Le mécanisme montre des performances aussi bonnes quand les variations sont moins brusques.

La troisième courbe, dsr_nodec_noRTS_1050_qlen2_comm de la Fig. 8.2 montre le cas où on ne décrémente pas les CWs. Quand le niveau de contention est haut, ce mécanisme montre une bonne efficacité puisqu'il évite bien les collisions. Cependant, quand le niveau de contention baisse, le surcoût de la taille des CWs devient considérable par rapport à la taille des paquets, réduisant ainsi l'efficacité.

La décrémentation lente des CWs est moins avantageuse quand les paquets sont plus courts, puisque les collisions/retransmissions sont moins graves. A la limite, l'utilisation de RTS/CTS montre le cas extrême des petits paquets. La Fig. 8.5 montre un gain en débit, toujours positif, de 6.8%, sous haut niveau de contention.

8.3 Réseaux ad-hoc

Considérons le cas de 100 nœuds distribués dans un région de 100x100m, tous à la portée l'un de l'autre. 50 sources différentes envois des flux UDP à 50 destinations différentes, utilisant RTS/CTS. Le scénario est le même que celui de la section précédente, cependant, on ajoute une phase "d'échauffement", 0-40, pour éviter les effets transitoirs dûs aux premiers paquets de routage transmis. La Fig. 8.7 montre les résultats de simulation.

On observe un gain de 15% en utilisant la décrémentation lente, par rapport à la décrémentation brusque, quand RTS/CTS est utilisé.

Dans la suite, nous introduisons deux métriques pour évaluer les performances de la décrémentation lente.

- *Le gain en débit utile (G)*: C'est le rapport entre le débit utile obtenu en utilisant la décrémentation lente et le débit utile obtenu par la décrémentation brusque.
- *Temps de stabilisation (T_s)*: Après une baisse brusque du niveau de contention, tel qu'à la seconde 150, T_s est le temps pris par un nœud donné pour récupérer le débit maximal possible (avec des CWs petits).

Figure 8.8 montre le gain G en fonction du facteur de décrémentation δ et du débit à la source λ. On constate que:

- Quand δ diminue, le mécanisme de décrémentation lente tend à ressembler à la décrémentation brusque, avec des performances similaires ($G \rightarrow 1$)
- Cependant, quand δ augmente, on évite mieux les collisions et le gain G devient plus considérable. Pour tous les λ, le gain maximal est aux alentours de $\delta_{max} = 0.9$.
- Quand λ diminue, le gain tend vers l'unité. En effet, quand λ diminue on observe moins de collisions, parsuite l'avantage de la décrémentation lente de CW devient moins visible.

Quant au temps de stabilisation T_s, Fig. 8.10 montre ce qu'on pourra penser intuitivement; quand δ augment, on a besoin de plus de transmissions et de temps pour récupérer les petits CWs et atteindre son état stable, i.e. T_s augmente. Cette augmentation est plus que linéaire, surtoût pour les grandes valeurs de δ.

Le choix d'une bonne valeur de δ est un compromis entre et grand gain en débit G et un temps de stabilistation T_s réduit. Les valeurs de δ entre 0.6 et 0.8 satisfont ce compromis.

8.4 Décrémentations linéaires des CWs

Considérons maintenant le cas où l'on décrémente les CWs linéairement d'une valeur α après chaque bonne transmission.

Figure 8.11 montre qu'on peut atteindre des gains en débits égaux à ceux obtenus par décrémentation multiplicatives. Quand α est petit, les CWs décroissent lentement, évitant les collisions, ce qui montre un gain considérable, similaires à ceux des grands δ.

Cependant, les temps de stabilistion sont plus grands avec les décrémentations linéaires, comme le montre la Fig. 8.12, surtout pour les petites valeurs de $\alpha(< 100)$ qui correspondent à un bon gain $G(> 1.12)$

Enfin notons que les auteurs dans [22] proposent une décrémentation linéaire avec $\alpha = 1$. Evidemment, ceci résulte en un gain considérable, cependant le temps de stabilisation après une baisse brusque du niveau de contention est très grand, comme le montre la Fig. 8.12.

Des mécanismes encore plus complexes, tels que ceux utilisés en automatiques, pourront montrer des améliorations en temps de stabilisation. Cependant, ceux-ci rendent le MAC complexe afin de réduire les T_s, sans pouvoir améliorer le gain en débits.

Chapitre 9

Modélisation des réseaux ad-hoc multi-sauts IEEE 802.11

Dans le chapitre précédent nous avons constaté que les débits (max.) et délais (min.) optimaux ne correspondent pas nécéssairement au débit maximal à la source. En effet, pour un nombre fixe de nœuds, quand on augmente les débits aux sources on injecte plus de paquets dans le canal, en essayant d'augmenter le débit. Cependant, ceci augmente les collisions et les retransmissions correspondantes, augmentant les délais, et diminuant les débits utiles et l'efficacité du canal. Ce qui nous fait poser la question suivante:

Comment peut-on estimer les débits et les délais dans les réseaux ad-hoc multi-sauts IEEE 802.11?

Dans un réseau ad-hoc, les différents nœuds routent les paquets des autres. Par suite, le débit disponible pour un nœud donné dépend de la capacité du canal ainsi que de la charge générée par les autres nœuds.

L'estimation des débits et des délais dans les réseaux ad-hoc pourra servir aussi à l'optimisation des performances du réseau tout entier, ainsi que celles d'un chemin spécifique: la réduction des interférences entre nœuds, l'optimisations des débits et des puissances de transmissions aussi. Cependant, cette tache s'avère dure vu le dynamisme des flux, le conditions variables du canal radio, les interférences et les contentions entre les nœuds.

Dans ce chapitre nous proposons une nouvelle méthode basique pour modéliser les réseaux ad-hoc multi-sauts IEEE 802.11. Nous investigons les conditions sous lesquelles on peux appliquer cette méthode, ce qui nous permet de mieux estimer les paramètres tels que délais, débits, et taux de perte. Ceci nous permet aussi de faire un contrôle de débits aux sources, afin de mieux optimisez ces derniers paramètres, ainsi que les retransimissons (après d'éventuelles collisions), pour économiser l'énergie des batteries.

9.1 Approche initiale pour modéliser un réseau multi-sauts

Dans cette section nous considérons des scénarios élémentaires et nous décrivons leur comportement avec des flux de débits différents. La section qui suit aborde des scénarios plus élaborés, en prenant en considérations les temps de *backoff* considérés négligeables dans cette section.

Nous considérons des sources à débits constants (CBR) et des sources à débits exponentiels. Ils montrent des résultats similaires, par suite nous donnons les résultats des trafics CBR seulement.

Nous adoptons les notations suivantes pour illustrer les portées des transmetteurs: une ovale pleine représente la zone de réception d'un nœud. Une ovale hachurée représente la zone d'interférence d'un nœud. Les sources de trafic sont les carrés pleins, les destinations correspondantes sont les carrés vides, et les routeurs intermédiaires sont représentés par les points (Fig. 9.3).

Une seule hypothèse est faite dans cette section: la source émet avec un débit constant, même quand son *backoff* augmente. Dans la section 9.2, nous montrons comment cette hypothèse change quand on prend le *backoff* en considération.

9.1.1 Scénario à saut unique

Figure 9.1 montre le scénario à saut unique. Nous assimilons cette brique élémentaire d'un chemin multi-sauts à une file d'attente d'un serveur, où le taux de service du serveur est la capacité du canal radio, et le taux d'arrivée est égal au taux d'arrivée des paquets à l'interface. La taille de la file est égale à la taille de la file de l'interface.

La première simulation montre le comportement du saut unique en variant le débit de la source λ_1 (Fig. 9.2).

Tant que λ_1 est largemet inférieure à la capacité du canal μ_1, la file d'attente ne se remplit pas, et le délai moyen d'un paquet est égal à son temps de transmission (en négligeant les délais dus au *backoff*). Le débit à la sortie est égal au débit à l'entrée. On n'observe pas de perte de paquets.

Quand λ_1 est proche de μ_1, la file commence à se remplir et le délai moyen augmente proportionellement au nombre de paquets dans la file d'attente.

Au delà de μ_1, le taux de paquet à l'arrivée excédant μ_1 est rejeté, le délai est constant (230ms), qui est égal à la taille de la file d'attente (50) multipliée par le temps de transmisison d'un paquet (4.6ms). Le débit à la sortie est constant.

Jusqu'à ce point, l'analyse est typiquement celle d'un serveur avec une file d'attente. On n'a aucune propriété due à la nature radio partagée du canal, ce qui est le cas dans le scénario suivant.

9.1.2 Scénario à deux sauts

Considérons maintenant le scénario à deux sauts de la Fig. 9.3. λ_i et μ_i désignent le taux d'arrivée des paquets et le taux de départ des paquets du nœud i respectivement. Le canal est partagé entre les nœuds 1 et 2, par suite on peut établir la relation suivante:

$$min(\lambda_1, \mu_1) + min(\lambda_2, \mu_2) = 1 \tag{9.1}$$

(9.1) montre que, quand les deux serveurs sont saturés, les taux de service sont complémentaires; le taux de service non utilisé par l'un est disponible pour l'autre. Les deux nœuds ont les mêmes chances d'accéder au canar, par suite $\mu_1 = \mu_2 = 1/2$ quand $\lambda_1 \geq 1/2$ et $\lambda_2 \geq 1/2$.

En général, si $\lambda_1 < 1/2$ et $\lambda_2 > (1 - \lambda_1)$ (c'est le cas d'un flux supplémentaire qui passe par le nœud 2, et qui maintient la saturation), (9.1) donne $\lambda_1 + \mu_2 = 1$.

Si $\lambda_1 > (1 - \lambda_2)$ et $\lambda_2 < 1/2$ (c'est le cas où le débit sortant du nœud 1 est supérieur à celui à l'entrée du nœud 2, en cas de plusieurs destinations par ex.), (9.1) donne $\mu_1 + \lambda_2 = 1$.

Notons que (9.1) ne peut pas être appliquée si le canal n'est pas saturé. Dans ce dernier cas, le débit à la sortie d'un nœud i est λ_i, et les temps d'attente dans les files, ainsi que les taux de perte paquets sont négligeables.

Quand nous appliquons un débit $\lambda_1 = 1$ sur le chemin à deux sauts, (9.1) donne un débit $\mu_2 = \lambda_2 = \mu_1 = 1/2$.

Nous obsrvons que le délai moyen est approximativement le double de ce qu'il est dans le cas du saut unique. En effet, le taux de service disponible est divisé par deux, ce qui double le temps d'attente des paquets dans la file d'attente et notons surtout qu'il n'y a pas de temps d'attente dans la file du nœud 2; bien que le débit entrant est égal au débit sortant, la file d'attente du nœud 2 ne se remplit pas à cause de la quasi-synchronisation entre les deux flux; quand un paquet arrive du nœud 1, il sera transmit par le nœud 2 très probablement juste après sa réception.

Quand le canal est sous-saturé ($\lambda_1 < 1/2$), on observe une forte diminution des délais. On n'observe plus de perte de paquets, et le délai devient égal à la durée de transmission seulement.

Notons que l'utilisation de RTS/CTS ici n'apporte pas d'amélioration au débit puisqu'il ajoute un certain surcoût sans éviter aucune collision.

9.1.3 Scénario à trois sauts

Considérons maintenant les scénario à trois sauts de la Fig. 9.4. La nouveauté dans ce scénario est les collisions au nœud 2, quand on n'utilise pas de RTS/CTS. Les nœuds 1 et 3 sont hors portée l'un de l'autre, et peuvent éventuellement transmettre des paquets simultanément, ce qui résulte en des collisions au niveau du nœud 2.

Vérifions tout d'abord le débit vers le nœud 4, quand on utilise une débit maximal à la sortie du nœud 1 ($\lambda_1 = 1$), sans utiliser RTS/CTS. Le canal sur le premier saut est saturé vu le débit à la sortie du nœud 1. Et puisqu'aucun flux supplémentaire passe à travers le nœud 2, et que le flux du nœud 3 entre en collision avec le flux entrant au nœud 2: $min(\lambda_2, \mu_2) = \lambda_2$ par suite (9.1) donne:

$$\mu_1 + \lambda_2 = 1$$

D'une autre part, le canal du deuxième saut n'est pas saturé, aucun flux supplémentaire passe à travers le nœud 3, et le flux a une destination unique, par suite:

$$\lambda_2 = \lambda_3$$

Une équation supplémentaire peut être établie aussi, considérant les collisions au nœud 2:

$$\lambda_2 = \mu_1 \times (1 - \lambda_3)^2 \tag{9.2}$$

La résolution de ces 3 équations donne

$$\lambda_2 \approx 0.318$$

qui est égal au débit entrant au nœud 4. Celui-là est supérieur au débit avec utilisation de RTS/CTS (= 1/3, voir explication plus tard dans cette section), duquel on doit soustraire un surcoût supplémentaire. Intuitivement, on peut estimer que $\lambda_3(=\lambda_2)$ est limitée par 1/3 (obtenue avec RTS/CTS) et 1/2 si on considère les deux sauts séparément. Pratiquement, les simulations donnent un débit ≈ 0.319.

Notons que toutes les collisions au nœud 2 entrainent des retransmissions par le nœud 1 (et ne doivent pas être soustraites de λ_2). Cependant, les transmissions du nœud 1 sont limitées par μ_1 (déjà saturé), par suite l'équation (9.2) s'applique toujours. (9.2) indique qu'une bonne réception d'un paquet a lieu quand il n'y a pas de transmission du nœud 3 pour la durée de deux paquets.

One peut voir que les collisions au nœud 2 sont proportionelles au débit entrant, réduisant ainsi le débit de bout-en-bout. En d'autres termes, le débit optimal/maximal ne correspond pas forcément au débit maximal à la source. Nous le vérifions dans la suite.

Considérons maintenant le cas où $\lambda_1 < 1 - \lambda_2$ (canal non-saturé). Le flux entrant au nœud 2 vaut:

$$\lambda_2 = \lambda_1 \times (1 - \lambda_2)^2 \Rightarrow \lambda_2 = \frac{1 + 2\lambda_1 - \sqrt{1 + 4\lambda_1}}{2\lambda_1}$$

dont la dérivée est strictement positive, c-à-d dont le maximum est à l'infini. Par suite le débit de bout-en-bout est une fonction croissante du débit entrant.

Du point de vue délai, si on réduit le débit entrant on évite plus de collisions et les retransmissions correspondantes. Le délai au nœud 1 peut donc être réduit, tout en maintenant le même débit de bout-en-bout. Ceci explique la baisse considérable des délais (de 0.813 à 0.589 seconde, quand on réduit λ_1 à 1/3), le débit de bout-en-bout étant toujours le même.

Quand on considère le scénario à tois sauts, utilisant RTS/CTS, (9.1) doit être remplacée par:

$$min(\lambda_1, \mu_1) + min(\lambda_2, \mu_2) + min(\lambda_3, \mu_3) = 1 \tag{9.3}$$

Quand le canal est saturé, (9.1) prend en considération trois nœuds consécutifs au lieu de deux dans (9.1), du fait que juste un des trois nœuds peut transmettre à un moment donné. Quand $\lambda_1 = 1$, et si on considère qu'il n'y a pas de collisions/interférences dans les nœuds intermédiaires, et que aucun flux supplémentaires passe à travers ces derniers, on peut voir que $\lambda_2 = \lambda_3 = \mu_1 = \mu_2 = \mu_3 = 1/3$.

9.1.4 Les flux croisés

Dans la sous-section précédente, les interférences et les collisions sont dues aux nœuds routeurs, traffiquant le même flux. Notons qe RTS/CTS peut éviter ces collisions et économiser l'énergie des batteries, mais le problème persiste avec les interférences; un nœud peut ne pas bien recevoir les RTS/CTS (pour mettre le NAV à jour), mais continue à causer des interférences au récepteur.

Figure 9.6 montre les cas de deux chemins croisés de deux sauts chacun, qui partagent le nœud/routeur du milieu. Deux flux différents luttent pour passer par le nœud 2. L'opimisation ici est faite sur chacun des flux, en fonction de son propre débit ainsi que le débit de l'autre source.

Si on considère des débits égaux aux deux sources, le point optimal correspond à $\lambda_{1opt} = \lambda_{2opt} = 1/4$. Même sans ordonnancement, juste en utilisant les débits optimaux aux source, les flux passent "fluidement" sur chacun des chemins, avec des délais courts (0.023 au lieu de 1.4 secondes, puisque les files d'attente ne se remplissent pas), de petites gigues et un débit légèrement amélioré par rapport aux $\lambda_1 > \lambda_{1opt}$ ou $\lambda_2 > \lambda_{2opt}$.

9.2 Considérations supplémentaires: l'influence du *backoff* sur le taux de service

Dans la section précédente nous avons négligé le délai dû au *backoff*. Cependant, cette hypothèse dépend typiquement du scénario. Considérons les deux sources S_1 et S_2 de la Fig. 9.7, transmettant avec des débits respectifs λ_1 et λ_2 vers la même destination D située à la portée des deux sources, sans utiliser RTS/CTS. S_1 et S_2 sont hors portée d'interférence l'un de l'autre.

Si on prend l'hypothèse du *backoff* négligeable, le débit de la source S_1 jusqu'à D est ce que D reçoit de S_1 sans entrer en collision avec le flux de S_2, c-à-d

$$\lambda_1 \times (1 - \lambda_2)^2$$

Si les deux sources transmettent à plein débits, i.e. $\lambda_1 = \lambda_2 = \mu_1 = \mu_2 = 1$ le débit utile de bout-en-bout résultant est nul pour les deux flux; les deux sources transmettent continuellement, en causant les collisions continues au routeur qui n'entend rien que des paquets erronnés. Cependant, quand une source ne reçoit pas l'acquitement de la destination, elle double son CW pour éviter les collisions futures. Au bout de plusieurs

incrémentations, le "vide" du *backoff* avec son DIFS pourra contenir la transmission de l'autre source, sans causer des collisions. La destination D commence donc à recevoir un débit complet juste après la période transitoire des premières collisions.

Le CW double de taille après chaque collision ($CW = 2^i - 1; i = 5, .., 10$). La taille moyenne peut donc être écrite sous la forme

$$E[CW] = \sum_{i=5}^{10} (2^i - 1) \times P_i$$

où P_i est la probabilité de collision avec un $CW = 2^i - 1$. P_i est fonction du nombre de nœuds en compétition, du débit de chacun, et de la longueur des paquets. Par suite, tant que CW est petit, P_i est grand, ce qui fait augmenter CW jusqu'à ce qu'il contienne un paquet de l'autre source.

Nous devons noter que dans ce scénario spécifique, le plein débit qui arrive à la destination D pourra provenir d'une seule source qui monopolise le canal. Cette source est très probablement celle qui accède au canal en premier, gardant un petit CW. D'autre part, la deuxième source augmente toujours son CW, avec très peu de chance d'accéder au canal de nouveau.

En conséquence, pour établir un bon modèle de réseaux ad-hoc multi-sauts IEEE 802.11, nous devons prendre le *backoff* en considération lorsqu'on calcul le taux de service.

Dans l'exemple suivant, nous considérons un scénarion à deux sauts (cf. 9.1.2), dans lequel le nœuds routeurs du milieu de chacun des chemin sont à la portée l'un de l'autre, comme dans la Fig. 9.8.

Les deux sources de trafic transmettent à plein débit vers leurs destinations respectives. Notons que les nœuds du milieu peuvent être à portée l'un de l'autre, ou juste interférer l'un avec l'autre. On n'utilise pas RTS/CTS (les résultats sont similaires).

Si on néglige l'incrémentation du CW, notre estimation donne:

$$\mu_3 = \mu_4 = 1/2$$

Cependant le CW a un effet considérable ici également:

Les débits (=1) aux entrées des nœuds 1 et 2 (1/2 aux nœuds 3 et 4) commencent à faires des collisions continuent au niveau des nœuds 3 et 4. Ce qui fait augmenter leur CWs largement, permettant aux paquets de l'une des sources d'être bien reçues. Le premier nœud (parmi 3 et 4) qui échoue à accéder au canal augmente son *backoff*, donnant plus de chances à l'autre qui garde l'accès au canal pour ne longue durée, jusqu'à ce que le premier réussisse à y accéder de nouveau. Ceci résulte en une série de rafales alternées aléatoirement entre les nœuds 3 et 4. Durant chaque rafale, un seul flux est acheminé sur un chemin. Par suite, à court terme, un des flux a un débit de 1/2, l'autre a un débit nul. A long terme, les débits sont partagés également entre les deux flux, ce qui donne une moyenne de 1/4 chacune. La variation des débits est très importante. La même idée s'applique également aux délais; les paquets sont soit rejetés sont acheminés sur des chemins de deux sauts.

9.3 Travail future

Afin de rendre ces approches de modélisation plus générales, nous envisageons de travailler sur les points suivants:

- *Des topologies plus complexes:* Dans ce chapitre nous n'avons analysé que des topologies élémentaires afin de pouvoir valider l'approche. Le travail future doit analyser des topologies réelles plus complexes.
- *Des modèles de propagation du signal plus élaborés:* Le canal radio que nous avons simulé est celui de l'espace libre, où l'atténuation du signal est considéré constante avec le temps. D'autres modèles plus proches de la réalité devront être étudiés également.
- *Evaluation de l'économie en énergie de la batterie:* Comme vu dans les paragraphes précédents, quand on réduit le débit aux sources on réduit en conséquence les interférences, les collisions et les retransmissions correspondantes. On peut donc économiser considérablement l'énergie des batteries tout en contrôlant les débits aux sources.
- *L'influence du backoff sur le taux de service:* Dans la Section 9.2, nous avons montré que le *backoff* doit être pris en considération dans plusieurs scénarios. Ceci donne de meilleures estimations pour des topologies plus générales.

9.4 Conclusion

Le contrôle des puissances de transmission réduit les interférences et tend à optimiser les débits disponibles pour chacun des nœuds d'un réseau ad-hoc. D'autre part, les débits aux sources peuvent être controlés afin

de pouvoir optimimiser le débit utile sur un chemin donné ou le débit utile globale du réseau, lui permettant mieux de passer à l'échelle.

Dans ce chapitre nous avons analysé l'applicabilité des propriétés des files d'attente sur les réseaux ad-hoc. Plusieurs topologies élémentaires ont été prises en considération pour pouvoir établir les relations entre les paramètres des files d'attente et ceux d'un réseau ad-hoc multi-sauts IEEE 802.11. Cette approche nous permet d'optimiser les débits et les délais en contrôlant les débits aux sources, en prenant en compte les interférences des flux voisins, les flux transversales et les charges des autres nœuds sur le même chemin. Elle peut être aussi utilisée pour réduire la consommation de l'énergie des batteries, en évitant des transmissions inutiles, qui causent plus de collisions et d'interférences, sans nécéssairement augmenter les débits utiles.

Les simulations et les analyses sont cohérents, ce qui nous montre que l'approche est valide et que nous pourront procéder à l'analyse de topologies plus complexes.

Chapitre 10

Conclusion

L'accès à l'Internet atteint de plus en plus les terminaux sans-fil, comme les PDAs, les téléphones cellulaires etc., enrichissant ainsi notre vie quotidienne de plus d'applications de de facilités. Du travail à la maison en passant par la rue, l'accès sans-fil à l'Internet devient une réalité, et deviendra prochainement un moyen essentiel de communication. Une large bande de standards accompagne cette progression pour supporter les technologies d'accès, nous donnant la liberté de nous déplacer, tout en restant connectés.

Vûs les progrès technologique des DSPs et de la microélectronique en général, les débits des accès sans-fil progressent consiérablement, ouvrant la voie à plus d'applications telles que les courriers électroniques, les navigateurs, l'audio et la vidéo aux terminaux sans-fil. Par suite, les réseaux locaux sans-fil (WLANs), les réseaux personnels et les réseaux ambiants attirent de plus en plus l'attention des chercheurs et des industriels, qui nous préparent sûrement une ère promettante avec des champs d'applications illimités. Comme l'a montré la deuxième génération de téléphones cellulaires, le nombre de terminaux mobiles est en croissance continue et continuera à croître dans le future. La grande variété d'applications dans ces mobiles éxige plusieurs terminaux par utilisateur et une connectivité omniprésente.

Cette croissance en nombre d'utilisateurs, des terminaux sans-fil par utilisateur et du temps de connexion résulte en une charge consdérable sur le canal radio. En effet, ces terminaux fonctionnent dans des bandes de fréquence libres telles que l'ISM et l'U-NII soumises à des règlementations. Les standards de communications sans-fil doivent prendre en considération les limitations en bandes passantes et en puissances d'émissions, tout en assurant une utilisation éfficace du canal radio, malgré la grande charge sur ce dernier et la différence des standards coéxistants.

Les applications temps-réel telles que l'audio et la védéo ont besoin de garanties minimales de qualité de service (QoS) pour fonctionner proprement. Ces contraintes ne peuvent pas être satisfaites avec les protocols *best-effort* actuels, surtout dans un réseau surchargé. En outre, la nature du médium sans-fil pose d'autres défis pour pouvoir assurer la QoS aux applications sans-fil. Le bruit, les interférences, les atténuations etc. sont des éléments de base d'un canal radio, et ne vont pas en même direction des objectifs de la qualité de service.

Ces contraintes font du support de la QoS un sujet abordé par plusieurs groupes de recherche dans le monde. Ce sujet peut être traité sur différents niveaux des couches réseaux. *DiffServ* et *IntServ* de l'IETF proposent des solutions au niveau IP. Cependant, pour les réseaux sans-fil, ces solutions restent sous-optimales si on ne les couplent pas avec un support QoS sur la couche sous-jacente, i.e. la sous-couche MAC.

Les réseaux à permutation de circuits, hérités des réseaux téléphoniques, simplifient le support de QoS et de la séparation entre flux du fait du contrôle centralisé et du contrôle d'admission. Cependant, la permutation de circuits s'avère non convenable à l'Internet, à base de permutation de paquets. Cette dernière se montre plus convenable pour supporter la grande variété d'applications que l'Internet supporte. Sans aucun besoin de signalisation, la permutation de paquets offre actuellement un seul niveau de service, *best-effort*, non convenable pour plusieurs applications "gourmandes".

Dans cette thèse nous orientons notre travail vers le support de la QoS dans les réseaux locaux sans-fil orientés permutation de paquets, e.g. IEEE 802.11. Le travail est divisé en plusieurs parties qui traitent différents aspects de la QoS:

- *Différentiation de service:*
  Dans ce chapitre nous proposons plusieurs mécanismes de différentiation de services pour IEEE 802.11. Tous sont basés sur une simple différentiation des paramètres du niveau MAC: le coefficient d'incrémentation du *backoff*, DIFS, CW_{min} et les tailles maximales des paquets. Nous montrons par simulations comment ces mécanismes fonctionnent avec des flux TCP et UDP. TCP a montré des effets de différentiations réduits à cause de son flux en boucle fermée. Nous proposons des améliorations pour ce dernier dans le

même chapitre. Nous montrons dans la suite l'effet du bruit canal sur ces mécanismes de différentiation. La plupart des résultats de ce chapitre coincident avec l'actuelle proposition de standard IEEE 802.11e.

- *Environnements bruités:*
  IEEE 802.11 utilise des fenêtres de contention pour résoudre l'accès multiple des terminaux au canal. Un terminal double la taille de sa fenêtre de contention à chaque perte de paquet. Cette stratégie diminue les collisions au canal, mais augmente le surcoût des paquets, diminuant ainsi le débit. Cependant, les pertes peuvent également être dues à du bruit sur le canal. L'augmentation de la fenêtre de contention peut alors être très néfaste en terme de performance. Il convient d'augmenter la fenêtre de contention uniquement si la perte a été produite par une collision. Nous proposons une stratégie d'adaptation de la fenêtre de contention qui varie selon l'estimation de la cause de perte des paquets.

- *Environnements congestionnés*
  Eviter l'incrémentation des fenêtres des contentions quand les pertes sont dues au bruit est une opération sensible, puisqu'elle pose un compromis entre gain de temps de *backoff* et des collisions avec les retransmissions correspondantes. Ce fait nous a conduit au compromis inversé; comment éviter les collisions et les retransmissions correspondantes au coût de temps de *backoff*. C'est typiquement le cas des environnements congestionnés après une bonne transmission. Nous analysons deux méthodes de décrémentation de la fenêtre de contention: multiplicative et linéaire. Les deux ont montré un gain considérable en débits et en délais par rapport au standard actuel, qui utilise une remise à une valeur fixe.

- *Estimation de débits et des délais dans les réseaux ad-hoc:*
  Dans un réseau ad-hoc les paquets sont routés suivant des chemins multi-saut. Ainsi le routage est coopératif entre les différents nœuds, et le débit utile moyen disponible à chaque nœud dépend du nombre total des nœuds, des interférences et des collisions. Nous proposons un mécanisme de contrôle de débits aux sources, basé sur l'estimation des débits et des délais, pouvant optimiser les débits utiles ainsi que la consommation d'énergie.

Travail future

Les sujets abordés dans cette thèse peuvent être encore étendus et améliorés. Parmi ces améliorations nous envisageons les points suivants:

- *L'application de DiffServ à la différentiation MAC*: c'est à dire comment faire la correspondance entre les paramètres de *Diffserv* et les paramètres de diffréntiation MAC afin d'optimiser les performances.

- *Modélisation du système* pour la différentiation de services avec des flux TCP, des flux TCP et UDP combinés ainsi que la différentiation par-flux.

- *Distribution des paramètres* de différentiation de services, d'une manière distribuée, en prenant en considération le problème des nœuds cachés.

- *Amélioration du mécanisme pour les environnements bruités* pour obtenir des valeurs précises des CW_{max}. Dans cette thèse nous avons limité notre recherche aux limites supérieure et inférieure de la valeur optimale du CW_{max}.

- *Analyse de l'impact de la décrémentation lente des fenêtres de contention* sur la consommation des batteries.

- *Etablir les relations, quand on utilise la décrémentation lente des fenêtres de contention* entre le gain en débit, le temps de stabilisation, les tailles de paquets, le nombre de nœuds et le paramètre de décrémentation.

- *Etendre l'estimation des débits et des délais* à des topologies plus complexes que celles considérées dans le Chapitre 9. Le modèle de propagation radio utilisé est l'espace libre, où l'atténuation du signal est considérée constante avec le temps. Des modèles plus proches de la réalité doivent être analysés aussi.

- *Evaluation du gain en énergie* quand on optimise les débits aux sources d'un réseau ad-hoc. Quand on réduit le nombre de paquets mis sur le canal, on réduit aussi les collsions et les retransmission, réduisant ainsi l'énergie cosommée des batteries.

Bibliography

[1] IEEE 802.11 WG, *ANSI/IEEE Std 802.11:Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC) Enhancements for Quality of Service (QoS) IEEE 802.11/D2.0*, IEEE, 2001.

[2] João L. Sobrinho and Krishnakumar A. S., "Quality-of-Service in Ad Hoc Carrier Sense Multiple Access Wireless Networks," *IEEE Journal on selected areas in communications*, August 1999.

[3] Xue Yang and Nitin H. Vaidya, "Priority scheduling in wireless ad-hoc networks," in *Proceedings of MobiHoc*, Lausanne, Switzerland, June 2002.

[4] Michael Barry, Andrew T. Campbell, and Andras Veres, "Distributed control algorithms for service differentiation in wireless packet networks," in *Proceedings of IEEE Infocom*, Anchorage - Alaska, April 2001.

[5] Charles A. Eldering, Mouhamadou Lamine Sylla, and Jeffrey A. Eisenach, "Is there a Moore's law for bandwidth ?.," *IEEE Communications magazine*, October 1999.

[6] LAN/MAN Standards Committee, *ANSI/IEEE Std 802.11:Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Computer Society, 1999.

[7] ETSI TS 101 761-1, *Broadband Radio Access Networks (BRAN); HIPERLAN Type 2;*, ETSI, 2000.

[8] Jennifer Bray, Charles F. Sturman, and Joe Mendolia, *Bluetooth 1.1 Connect without cables.*, Prentice Hall, December 2001.

[9] Ajay Chandra V. Gummalla and John. O. Limb, *Wireless medium access control protocol*, IEEE Communications surveys, http://www.comsoc.org/pubs/surveys, second quarter 2000.

[10] L. Kleinrock and F. A. Tobagi, "Packet Switching in Radio Channels: Part 2: The Hidden Terminal Problem in Carrier Sense Multiple-Access and the Busy-Tone Solution.," *IEEE Trans. on Comm. COM-23*, 1975.

[11] Vaduvur Bharghavan, "A new protocol for medium access in wireless packet networks," Tech. Rep., University of Illinois, Urbana-Champaign, 1996.

[12] Songwu Lu, Vaduvur Bharghavan, and Rayadurgam. Srikant, "Fair scheduling in wireless packet networks.," in *Proceedings of ACM SIGCOMM, Cannes, France*, 1997.

[13] N. Abramson, *The ALOHA system, computer communication networks.*, Prentice Hall, 1973.

[14] N. Abramson, "The ALOHA system: Another alternative for computer communications.," in *Proceedings of AFIPS*, 1970.

[15] Lawrence G.. Roberts, "Aloha packet system with and without slots and capture.," *ACM Sigcomm computer communication review*, 1972.

[16] L. Kleinrock and F. A. Tobagi, "Packet Switching in Radio Channels: Part 1: CSMA Modes and Their Throughput Delay Characteristics.," *IEEE Trans. on Comm. COM-23*, 1975.

[17] LAN/MAN Standards Committee, *IEEE Std 802.3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications.*, IEEE Computer Society, 2000.

[18] R.M. Metcalfe and D.R. Boggs, "Ethernet: Distributed packet switching for local computer networks.," *Communications of the ACM*, 1976.

[19] Cheng-shong Wu and Victor O.K.. Li, "Receiver-initiated busy tone multiple access in packet radio networks.," in *Proceedings of ACM Sigcomm*, 1988.

[20] P. Karn, "MACA: A new channel access method for packet radio.," in *ARRL/CRRL Amateur radio 9th computer networking conference*, 1990.

[21] K. Biba, "A hybrid wireless MAC protocol supporting asynchronous and synchronous MSDU delivery services.," in *IEEE 802.11 working group paper 802.11/91-92*, 1992.

[22] Vaduvur Bharghavan, Alan Demers, Scott Shenker, and Lixia Zhang, "MACAW: A media acess protocol for wireless LANs.," in *Proceedings of ACM Sigcomm*, 1994.

[23] Chane L. Fullmer and J. J. Garcia-Luna-Aceves, "Floor acquisition multiple access for packet radio networks," in *Proceedings of ACM Sigcomm*, 1995.

[24] Chane L. Fullmer and J. J. Garcia-Luna-Aceves, "Solutions to hidden terminal problems in wireless networks," in *Proceedings of ACM Sigcomm*, 1997.

[25] W. Diepstraten, G. Ennis, and P. Berlnager, "DFWMAC: distributed foundation wireless nedium access control," Tech. Rep., IEEE Document P802.11-93/190, 1993.

[26] Brian Crow, Indra Widjaja, Jeong Geun Kim, and Prescott T. Sakai, "Performance of IEEE 802.11 wireless local area networks," in *Proceedings of SPIE*, 1996.

[27] Brian P. Crow, Indra Widjaja, Jeong Geun Kim, and Prescott T. Sakai, "IEEE 802.11 wireless local area networks," *IEEE Communication magazine*, September 1997.

[28] F. Calì, M. Conti, and E. Gregori, "IEEE 802.11 wireless LAN: Capacity Analysis and protocol enhancement," in *Proceedings of IEEE Infocom*, 1998.

[29] Harshal S. Chhaya and Sanjay Gupta, "Performance modeling of asynchronous data transfer methods of IEEE 802.11 MAC protocol," *Wireless networks*, vol. 3, 1997.

[30] Harshal S. Chhaya and Sanjay Gupta, "Throughput and fairness properties of asynchronous data transfer methods in the IEEE 802.11 MAC protocol," in *Proceedings of the 6th intl. conference on personal, indoor and mobile radio communications, PIMRC*, October 1996.

[31] Y.C. Tay and K.C. Chua, "A capacity analysis for the IEEE 802.11 MAC protocol.," *Wireless networks*, 2001.

[32] Giuseppe Bianchi, "IEEE 802.11 saturation throughput analysis," *IEEE communications letters*, vol. 2, 1998.

[33] Giuseppe. Binachi, "Throughput evaluation of the IEEE 802.11 distributed coordination function," in *Proceedings of the 5th intl. workshop on mobile multimedia communications, MoMuc*, October 1998.

[34] Giuseppe Binachi, Luigi Fratta, and Matteo. Oliveri, "Performance evaluation and enhancement of the CSMA/CA MAC protocol for 802.11 wireless LANs," in *Proceedings of the 6th intl. conference on personal, indoor and mobile radio communications, PIMRC*, October 1996.

[35] Giuseppe Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE journal on selected areas in communications*, vol. 18, 2000.

[36] Vladimir. Vishnevsky and Andrey. Lyakhov, "IEEE 802.11 wireless LAN: saturation throughput analysis with seizing effect consideration.," *Cluster Computing*, 2002.

[37] Vladimir. Vishnevsky and Andrey. Lyakhov, "Comparative study of 802.11 DCF and its modification in teh presence of noise.," *Submitted to SPECT*, 2002.

[38] Can Emre Koksal, Hisham Kassab, and Hari. Balakrishnan, "An analysis of short-term fairness in wireless media access protocols," in *Proceedings of ACM Sigmetrics*, 2000.

[39] Larry Taylor, *HIPERLAN white paper*, June 1999, http://www.hiperlan.com/.

[40] Philippe Jacquet, Pascale Minet, Paul Muhlethaler, and Nicolas. Rivierre, "Priority and collision detection with active signaling: The channel access mechanism of HIPERLAN.," *Wireless Personal Communications*, 1997.

[41] Philippe Jacquet, Pascale Minet, Paul Muhlethaler, and Nicolas. Rivierre, "Data transfer in HIPERLAN," *Wireless Personal Communications*, 1997.

[42] Philippe Jacquet and Paul. Muhlethaler, "Simulation of high performance radio LAN type 1 with ATM traffics," *Wireless Personal Communications*, 2000.

[43] G. Wu, K. Mukumoto, and A. Fukuda, "An integrated voice and data transmission system with idle signal multiple access - dynamic analysis.," *IEEE Transactions on communications*, 1993.

[44] G. et al. Wu, "An R-ISMA integrated voice/data wireless information system with different packet generation rates," in *Proceedings of IEEE ICC*, 1996.

[45] F. Watanabe, G. Wu, and H. Sasaoka, "Performance evaluation of reserved idle signal multiple access with collision resolution," in *Proceedings of the 3rd international workshop on mobile multimedia communications MoMuC*, 1996.

[46] F. Watanabe, G. Wu, and H. Sasaoka, "Stable throughput of reserved idle signal mutliple access with collision resolution," in *ICICE Transaction on communications*, 1997.

[47] G. Wu, K. Mukumoto, and A. Fukuda, "Performance evaluation of reserved idle signal multiple-access scheme for wireless communication networks," in *Transactions on vehicular technology*, 1994.

[48] K.C. Chen and C.H. Lee, "RAP - A novel medium access protocol for wireless data networks," in *Proceedings of IEEE Globecom*, 1993.

[49] J.J. Lai, Y.W. Lai, and S.J. Lee, "A medium access control for wireless networks," in *Proceedings of IEEE ICC*, 1998.

[50] H. Chou, C. Lee, and K. Chen, "Group randomly addressed polling with reservation for wireless integrated service networks," in *Proceedings of IEEE PIMRC*, 1995.

[51] Meng-Che Li and Kwang-Cheng. Chen, "GRAPO - Optimized group randomly addressed polling for wireless data networks," *International journal of wireless information networks*, 1995.

[52] N. Amitay, "Distributed switching and control with fast resource assignment/handoff for personal communications systems," *IEEE JSAC*, 1993.

[53] N. Amitay, "Resource auction multiple access (RAMA): efficient method for fast resource assignment for decentralized wireless PCS," *Electronic letters*, 1992.

[54] Pinheiro A.L.A. and J.R.B. de Marca, "A fair deterministic packet access protocol: F-RAMA (Fair resource assignment multiple access," *Electronic letters*, 1996.

[55] Zhensheng Zhang and Anthony S. Acompara, "Performance of a modified polling strategy for broadband wireless LANs in a Harsh fading environment," in *Proceedings of IEEE Globecom*, 1991.

[56] R.J. Haines and A.H. Aghvami, "Indoor radio environment considerations in selecting a media access control protocol for wideband radio data communications," in *Proceedings of IEEE ICC*, 1994.

[57] A.S. Acampora and S.V. Krishnamurthy, "A new adaptive MAC layer protocol for wireless ATM networks in Harsh fading and interference environements.," in *Proceedings of ICUPC*, 1997.

[58] D.J. Goodman and A. Saleh, "Packet reservation multiple access for local wireless communications," *IEEE Transaction on communications*, 1989.

[59] W.C. Wong and D.J. Goodman, "Integrated data and speech transmission using packet reservation multiple access," in *Proceedings of IEEE ICC*, 1993.

[60] P. Narasimhan and R.D. Yates, "A new protocol for the integration ofvoice and data over PRMA," *IEEE JSAC*, 1995.

[61] J.M. DeVile, "A reservation-based multiple access scheme for future universal mobile telecommunications system," in *Proceedings of IEE conference on mobile and personal communications*, 1990.

[62] M. Aciardi, F. Davoli, and C. Nobile, "Independent stations algorithm for the maximization of one-step throughput in a multiple access channel," *IEEE Transactions on communications*, 1988.

[63] R. Bolla, F. Davoli, and C. Nobile, "A RRA-ISA multiple access protocol with and without simple priority schemes for real-time and data traffic in wireless cellular systems," *Mobile networks and applications*, 1997.

[64] Giuseppe et al. Bianchi, "C-PRMA: A centralized packet reservation multiple access for local wireless communications," *IEEE Transaction on vehicular technology*, vol. 46, 1997.

[65] M.J. Karol, Z. Liu, and K.Y. Eng, "An efficient demand-assignment multiple access protocol for wireless packet (ATM) networks.," *Wireless networks,ACM-press, Baltzer science publishers.*, 1995.

[66] Jo et al.. Mikkonen, "The MAGIC WAND - Functional overview," *IEEE JSAC*, 1998.

[67] Bob O'Hara and Al Petrick, *IEEE 802.11 handbook. A designer's companion.*, IEEE Press, 1999.

[68] Charles Perkins, *Mobile IP, Design Principles and practices*, Addison-Wesley, 1998.

[69] Charles Perkins, *IP Mobility Support for IPv4*, 2002, IETF Network Working Group, RFC 3220.

[70] D. Johnson and C. Perkins, *Mobility Support in IPv6*, 2002, IETF Mobile IP Working Group Internet Draft.

[71] F. A. Tobagi and V. Bruce Hunt, "Performance Analysis of Carrier Sense Multiple Access with Collision Detection.," *Computer Networks 4*, 1980.

[72] J. Weinmiller, H. Woesner, JP Ebert, and A. Wolisz, "Analyzing the RTS/CTS mechanism in the DFWMAC media access protocol for wireless LANs," in *IFIP TC6*, 1995.

[73] Stefan Mangold, Sunghyun Choi, Peter May, Ole Klein, Guido Hiertz, and Lothar Stibor, "IEEE 802.11e wireless LAN for Quality of Service," in *Proceedings of European Wireless*, Florence - Italy, February 2002.

[74] Tero Ojanperä and Ramjee Prasad, *Wideband CDMA for third generation mobile communications.*, Artech House Publishers, 1998.

[75] Andrew J. Viterbi, *CDMA, Principles of spread spectrum communication*, Addison-Wesley, first edition, 1995.

[76] Fluhrer, Scott, Itsik Mantin, and Adi Shamir, "Weaknesses in the Key Scheduling Algorithm of RC4.," in *Eighth Annual Workshop on Selected Areas in Cryptography*, 2001.

[77] Mantin and Shamir, "A Practical Attack on Broadcast RC4," in *FSE*, 2001.

[78] Martin Johnsson, *HiperLAN-2, the broadband radio transmission technology operating in the 5GHz frequency band*, 1999, HiperLAN-2 Global forum.

[79] Brent A. Miller, Chatschik Bisdikian, and Anders Edlund, *Bluetooth revealed.*, Prentice Hall, 2000.

[80] Chatschik. Bisdikian, "An overview of the Bluetooth wireless technology.," *IEEE Communications magazine*, December 2001.

[81] J.L. Massey, *On the optimality of SAFER+ diffusion*, http://csrc.nist.gov/encryption/aes/round1/conf2/papers/massey.pdf.

[82] Angela Doufexi, Simon Armour, Peter Karlsson, Andrew Nix, and David Bull, *A compaison of HiperLAN/2 adn IEEE 802.11a.*

[83] Imad Aad and Claude Castelluccia, "Differentiation mechanisms for IEEE 802.11," in *Proceedings of IEEE Infocom*, Anchorage - Alaska, April 2001.

[84] Imad. Aad and Claude. Castelluccia, "Introducing service differentiation into IEEE 802.11," in *ISCC, Antibes - France*, July 2000.

[85] Imad Aad and Claude Castelluccia, "Remarks on per-flow differentiation in IEEE 802.11," in *European Wireless*, Florence, February 2002.

[86] Van Jacobson, "Congestion avoidance and control," in *Proceedings of ACM Sigcomm*, August 1988, pp. 314–329.

[87] J. Postel, *User datagram protocol*, Request For Comments 768.

[88] "Network Simulator, http://www.isi.edu/nsnam/ns/," .

[89] George Xylomenos and George C. Polyzos, "TCP and UDP performance over a wireless LAN.," in *Proceedings of Infocom*, 1999.

[90] Eitan Altman, Kostya Avrachenkov, and Chadi Barakat, "A Stochastic Model of TCP/IP with Stationary Random Losses," in *Proceedings of ACM Sigcomm*, Stockholm, Sweden, August 2000.

[91] K. Nichols, S. Blake, F. Baker, and D. Black, *Definition of the differentiated services field in the IPv4 and IPv6 headers*, Request For Comments 2474.

[92] Torsten Braun, Claude Castelluccia, Günter Stattenberger, and Imad. Aad, "An analysis of the DiffServ approach in mobile environments," Tech. Rep., INRIA, April 1999, http://www.inrialpes.fr/planete/people/MobiQoS/paper2.ps.

[93] Seoung-Bum Lee, Gahng-Seop Ahn, Xiaowei Zhang, and Andrew T. Campbell, "INSIGNA, An IP-based quality of service framework for mobile ad-hoc netwoks," *Journal of parallel and distributed computation - Special issue on wireless and movile computing and communications*, vol. 60, 2000.

[94] George Xylomenos and George C. Polyzos, "Link layer support for quality of servcie on wireless internet links," *IEEE Personal communications*, October 1999.

[95] Norival R. Figueira and Joseph Pasquale, "Providing quality of service for wireless links: wireless/wired networks," *IEEE Personal communications*, October 1999.

[96] Ibrahima Niang, Bachar Zouari, Hossam Afifi, and Dominique Seret, "Amélioration de schémas de QoS dans les rseaux sans fil 802.11," in *Proceedings of CFIP*, 2002.

[97] Chunhung Richard Lin and Mario. Gerla, "Real-time support in multihop wireless networks.," *Wireless networks*, 1999.

[98] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E.. Knightly, "Distributed multi-hop scheduling and medium access with delay and throughput constraints.," in *Proceedings of MOBICOM*, 2001.

[99] A. Muir and J.. Garcia-Luna-Aceves, "Supporting real-time multimedia traffic in a wireless LAN.," in *Proceedings of SPIE, Multimedia Computing and Networking, San José, CA*, 1997.

[100] Claude Chaudet and Isabelle Guérin-Lassous, "BRuIT: bandwidth reservation under interferences influence," in *Proceedings of European wireless*, 2002.

[101] Claude Chaudet, "Qualité de service et réseaux ad-hoc - un état de l'art -," in *MS3G, Services liés à la mobilité et réseaux mobiles de 3ème génération*, Lyon, France, December 2001.

[102] R. Garcés and J.. Garcia-Luna-Aceves, "Collision avoidance and resolution multiple access with transmission groups.," in *Proceedings of IEEE Infocom, Kobe, Japan.*, 1997.

[103] M Gerla and P. Joahnsson, "Bluetooth: Technology, Application, Performance," in *ACM Mobicom tutorial*, July 2001.

[104] Thyagarajan Nandagopal, Tae-Eun Kim, Xia Gao, and Vaduvur Bharghavan, "Achieving MAC layer fairness in wireless packet networks.," in *Proceedings of Mobicom*, 2000.

[105] Timucin Ozugur, Mahmoud Naghshineh, Kermani Parviz, Michael Olsen, Babak Rezvani, and John Copeland, "Balanced media acess methods for wireless networks.," in *Proceedings of Mobicom*, 1998.

[106] Mo Jeonghoon, J. La Richard, Anantharam Venkat, and C. Walrand Jean, "Analysis and comparison of TCP Reno and Vegas.," in *Proceedings of Infocom*, 1999.

[107] D. Chiu and R. Jain, "Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks.," in *Computer Networks and ISDN Systems 17*, 1989.

[108] Charles L. Phillips and Royce D. Harbor, *Feedback control systems*, Prentice-Hall, 1988.

[109] Piyush Gupta and P.R. Kumar, "The Capacity of Wireless Networks," *IEEE Transactions on Information Theory*, March 2000.

[110] Jinyang Li, Charles Blake, Douglas S.J. De Couto, Hu Imm Lee, and Robert Morris, "Capacity of Ad Hoc Wireless Networks," in *Proceedings of ACM Mobicom*, Rome - Italy, July 2001.

[111] Matthias Grossglauser and David Tse, "Mobility Increases the Capacity of Ad-hoc Wireless Networks," in *Proceedings of IEEE Infocom*, Anchorage - Alaska, April 2001.

List of acronyms

ACF	Association Control Function
ACH	Access Feedback Channel
ACK	ACKnowledgment
ACL	Asynchronous ConnectionLess
ADSL	Asymmetric Digital Subscriber Line
AIFS	Arbitration Inter-Frame Spacing
AP	Access Point
ARQ	Automatic Repeat reQuest
ASCH	Association Control Channel
AT	ATtention
ATIM	Announcement Traffic Indication Message
ATM	Asynchronous Transfer Mode
BB	Black Burst
BCH	Broadcast CHannel
BEB	Binary Exponential Backoff
BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
BS	Base Station
BSA	Basic Service Area
BSS	Basic Service Set
BTMA	Busy Tone Multiple Access
BTPS	Busy Tone Priority Scheduling
CA	Collision avoidance
CBR	Constant Bit Rate
CCK	Complementary Code Keying
CD	Collision Detection
CDMA	Code Division Multiple Access
CFP	Contention Free Period
CL	Convergence Layer
CP	Contention Period
CRC	Cyclic Redundancy Check
CSMA	Carrier Sense Multiple Access
CTS	Clear To Send
CW	Contention Window
DCC	DLC user Connection Control
DCCH	Dedicated Control CHannel
DCF	Distributed Coordination Function
DFWMAC	Distributed Foundation Wireless Medium Access Control
DFS	Dynamic Frequency Selection

DIFS	DCF IFS
DL	DownLink
DLC	Data Link Control
DMT	Discrete MultiTone
DQRUMA	Distributed-Queuing Request Update Multiple Access
DS	Distribution System
DSP	Digital Signal Processor
DSSS	Direct Sequence Spread Spectrum
DTMP	Disposable Token MAC Protocol
EC	Error Control
EDCF	Enhanced Distributed Coordination Function
EDD	Earliest Due Date
EM	ElectroMagnetic
ESS	Extended Service System
ETSI	European Telecommunications Standards Institute
EY-NPMA	Elimination Yield - NPMA
FCH	Frame Control Channel
FDD	Frequency Division Duplex
FEC	Forward Error Correction
FFT	Fast Fourier Transform
FHSS	Frequency Hopping Spread Spectrum
FSK	Frequency Shift Keying
GAMA	Group Allocation Multiple Access
GFSK	Gaussian FSK
GRAP	Group RAP
GRAPO	GRAP Optimized
HC	Hybrid Coordinator
HCF	Hybrid Coordination Function
HCI	Host Controller Interface
HiperLAN	High Performance European Radio LAN
HMAC	Hashing Message Authentication Code
IBSS	Independent BSS
ID	IDentification
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IFFT	Inverse FFT
IFS	Inter Frame Spacing
IP	Internet Protocol
IR	Infra-Red
IrMC	Infrared Mobile Communications
ISDN	Integrate Services Digital Network
ISI	Inter-Symbol Interference
ISM	Industry, Scientific and Medical
ISMA	Idle Sense Multiple Access
ISO	International Organization for Standardization
ISP	Internet Service Provider
IV	Initialization Vector
L2CAP	Logical Link Control and Adaptation Protocol
LAN	Local Area Network
LCCH	Link Control CHannel
LCH	Long Transport Channel
LLC	Logical Link Control
LMP	Link Management Protocol
LOS	Line Of Sight
LPI	Low Probability of Interception

MAC	Medium Access Control
MACA	Multiple Access with Collision Avoidance
MACAW	MACA for Wireless networks
MH	Mobile Host
MT	Mobile Terminal
NACK	Negative ACKnowledgement
NAV	Network Allocation Vector
NPMA	Non-preemptive Priority Multiple Access
NS	Network Simulator
OBEX	OBject EXchange
OFDM	Orthogonal Frequency Division Multiplexing
OSI	Open System Interconnection
PAN	Personal Area Network
PCF	Point Coordination Function
PDA	Personal Digital Assistant
PDU	Packet Data Unit
PER	Packet Error Rate
PF	Persistence factor
PHY	PHYsical layer
PIFS	Polling IFS
PIN	Personal Identification Number
PKI	Public Key Infrastructure
PPP	Point to Point Protocol
PRMA	Packet Reservation Multiple Access
PRNG	Pseudo-Random Number Generator
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service
QPSK	Quadrature Phase Shift Keying
R-ISMA	Reservation ISMA
RAP	Randomly Addressed Polling
RAMA	Resource Auction Multiple Access
RCH	Random access CHannel
RI-BTMA	receiver Initiated BTMA
RLC	Radio Link Control
RR	Random Range
RRA	Random Reservation Access
RRA-ISA	RRA - Independent Stations Algorithm
RRC	Radio Resource Control
RSA	Rivest, Shamir and Adleman
RTS	Request To Send
RTT	Round Trip Time
S-Aloha	Slotted Aloha
SBCH	Slow Broadcast CHannel
SCH	Short transport CHannel
SCO	Synchronous Connection Oriented
SDP	Service Discovery Protocol
SIFS	Short IFS
TC	Traffic Category
TCP	Transport Control Protocol
TCS	Telephony Control Signaling
TDD	Time Division Duplex
TDMA	Time Division Multiple Access

UDCH	User Data CHannel
UDP	User Datagram Protocol
UL	UpLink
U-NII	Unlicensed National Information Infrastructure
VMAC	Virtual MAC
WEP	Wired Equivalent Privacy
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network
WT	Wireless Terminal
XOR	eXclusive OR

Index