

***FHCF: A Fair Scheduling Scheme for 802.11e  
WLAN***

Pierre Ansel — Qiang Ni — Thierry Turletti

**N° 4883**

July 2003

THÈME 1

 *Rapport  
de recherche*



## FHCF: A Fair Scheduling Scheme for 802.11e WLAN

Pierre Ansel , Qiang Ni , Thierry Turetletti

Thème 1 — Réseaux et systèmes  
Projet Planete

Rapport de recherche n° 4883 — July 2003 — 28 pages

**Abstract:** The IEEE 802.11e Medium Access Control (MAC) is an emerging standard to support Quality of Service (QoS). Some recent works prove that 802.11e Hybrid Coordination Function (HCF) can improve significantly the QoS support in 802.11 networks. A simple HCF scheduler has been proposed in the 802.11e which takes the QoS requirements of flows into account and allocates time to stations on the basis of the mean sending rate. As we show in this paper, this HCF scheduling algorithm is only efficient for flows with strict Constant Bit Rate (CBR) characteristics. However, several real time applications, such as videoconferencing, may sometimes have small variations in their packet sizes, sending rates or even have Variable Bit Rate (VBR) characteristics. In this paper, we propose a new HCF scheduling algorithm, FHCF, that aims to be fair for both CBR and VBR flows. The FHCF scheme uses queue length estimations to tune its time allocation to stations. We present a set of simulations and provide performance comparisons with other schemes. Our performance study indicates that FHCF provides good fairness while supporting bandwidth and delay requirements for a large range of network loads.

**Key-words:** Wireless LAN, IEEE 802.11e, Medium Access Channel (MAC), Quality of Service (QoS)

## FHCF: Un nouveau mécanisme d'ordonnement pour Réseaux Locaux sans Fil 802.11e

**Résumé :** Le standard IEEE 802.11e est un nouveau Contrôle d'Accès au Medium (MAC) élaboré pour répondre aux besoins en Qualité de Service (QoS) des applications sur réseaux locaux sans-fil. De récentes études ont montré que la *Fonction de Coordination Hybride* (HCF), définie dans le standard 802.11e, permet d'améliorer considérablement les performances obtenues par ces applications. Ce standard propose un algorithme d'ordonnement simple, qui exploite la HCF en tenant compte des besoins en QoS des différents flux circulant sur le réseau. L'ordonneur alloue des temps d'accès au medium en fonction des taux moyens d'arrivée de données. Dans ce rapport, nous montrons que cet algorithme n'est efficace que pour des flux dont les caractéristiques sont strictement CBR. Cependant, de nombreuses applications temps-réel telles que la visioconférence, génèrent parfois des flux avec des tailles de paquets ou des débits qui peuvent varier au cours du temps (VBR). Ce rapport présente FHCF, un nouvel algorithme d'ordonnement pour la HCF qui vise à être équitable aussi bien pour des flux CBR que pour des flux VBR. Ce nouveau mécanisme utilise des estimations sur les longueurs des files d'attente pour affiner les allocations de temps d'accès des différentes stations du réseau. Nous présentons un ensemble de simulations pour comparer les performances des différents mécanismes. Cette étude montre que FHCF est capable d'être très équitable envers les différentes stations tout en respectant les besoins en bande passante et en délai de chaque flux pour des charges du réseau très variées.

**Mots-clés :** Réseaux Locaux Sans Fil, IEEE 802.11e, Contrôle d'Accès au Medium (MAC), Qualité de Service (QoS)

## Contents

<b>1</b>	<b>Introduction</b>	<b>6</b>
<b>2</b>	<b>The 802.11e HCF scheduling algorithm</b>	<b>7</b>
<b>3</b>	<b>Relationship between delay and SI</b>	<b>8</b>
3.1	Empty queues at the end of the TXOP . . . . .	8
3.2	Non-empty queues at the end of the TXOP . . . . .	9
<b>4</b>	<b>The FHCF Scheme</b>	<b>11</b>
4.1	A queue length based scheduling scheme at the QAP . . . . .	11
4.1.1	Use of a queue length estimation at the beginning of the SI . . . . .	12
4.1.2	Comparison with the “ideal” queue length distribution . . . . .	12
4.1.3	Estimation of the fluctuations of the number of arriving packets . . . . .	13
4.1.4	Redistribution by the scheduler of the unallocated time . . . . .	16
4.2	The node scheduler . . . . .	17
4.3	FHCF Implementation in NS-2 . . . . .	18
4.3.1	Basis of FHCF scheme . . . . .	18
4.3.2	TSPEC negotiation . . . . .	18
<b>5</b>	<b>Description of simulations</b>	<b>19</b>
5.1	Evaluation of the QAP scheduler . . . . .	19
5.2	Evaluation of the node scheduler . . . . .	20
<b>6</b>	<b>Results</b>	<b>22</b>
6.1	Evaluation of the QAP scheduler . . . . .	22
6.1.1	Comparison of throughputs . . . . .	23
6.1.2	Comparison of the number of dropped packets . . . . .	23
6.1.3	Analysis of latency distributions . . . . .	23
6.2	Evaluation of the node scheduler . . . . .	25
6.2.1	Audio and VBR H.261 video flows . . . . .	25
6.2.2	CBR MPEG4 video flows . . . . .	26
<b>7</b>	<b>Conclusion</b>	<b>27</b>

## List of Figures

1	Structure of the superframe used in the HCF scheduling algorithm proposed in [2] . . . . .	7
2	Theoretical delay versus time . . . . .	10
3	Queue length evolution for a TS ; (0): ideal queue length evolution ; (1): $q_i^{est} > q_i^{ideal}$ ; (2): $q_i^{est} < q_i^{ideal}$ . . . . .	13
4	$q_i^{ideal}$ and $q_i^{est}$ distribution at the beginning of the SI . . . . .	14
5	Distribution of the sending rate of a real VBR video traffic . . . . .	14
6	Diagram of the remaining time usable by our algorithm . . . . .	16
7	Throughput versus time for FHCF, Standard HCF and EDCF . . . . .	22
8	Dropped packets versus time for FHCF, Standard HCF and EDCF . . . . .	22
9	Latency distribution for FHCF, Standard HCF and EDCF . . . . .	22
10	Mean latency for each flow with FHCF, Standard HCF and EDCF . . . . .	22
11	Delays of the considered VBR flow: (1) about 20ms, (2) about 37ms, (3) about 4ms . . . . .	24
12	Delays . . . . .	26
13	Fairness . . . . .	26
14	Total throughput . . . . .	27

**List of Tables**

1	Description of the different traffics . . . . .	19
2	Parameters of the audio traffic . . . . .	20
3	Parameters used for the simulations . . . . .	20
4	The PHY and MAC parameters . . . . .	21
5	Description of the different traffics . . . . .	21
6	Number of drops for each kind of flow with FHCF, Standard HCF and EDCF	23

## 1 Introduction

IEEE 802.11 Wireless LAN (WLAN) [1] is one of the most deployed wireless technologies all over the world. The explosive growth of multimedia applications in the recent years arose the problem of the required Quality of Service (QoS) of these applications such as guaranteed delay, jitter and bandwidth. However, the 802.11 MAC has not been designed in this context and does not support service differentiation.

A new Hybrid Coordination Function (HCF) has been proposed in the upcoming 802.11e draft [2] to enhance the QoS support by combining and extending the mandatory Distributed Coordination Function (DCF) and the optional Point Coordination Function (PCF) of the MAC sub-layer. HCF is composed of two channel access mechanisms: a contention-based channel access and a controlled channel access mechanism. On the one hand, the contention-based channel access, also called Enhanced DCF (EDCF), provides a probabilistic QoS support by using 8 Traffic Categories (TCs) for different types of flows. On the other hand, the HCF Controlled Channel Access (HCCA), which evolved from the PCF, provides a parameterized QoS support. With this mode, a QoS Station (QSTA) has first to send its traffic requirements to the Hybrid Coordinator (HC) which is collocated within the QoS Access Point (QAP). If traffic can be scheduled in the HCCA mode, the QSTA receives a downlink frame notifying the acceptance of the traffic. Then a virtual connection called Traffic Stream (TS) is set up and the QSTA receives a Transmission Opportunity (TXOP) of a certain duration each time it is polled by the QAP.

Recent performance evaluations of 802.11e HCF [3], [4] show that HCF is more flexible than DCF and PCF and it can improve the QoS support in 802.11 networks. In order to guarantee the strict QoS requirements, the HC needs to schedule efficiently the different transmissions. This paper focuses on scheduling algorithms used for the HCCA mode. As described in more details in the next section, the 802.11e standard [2] proposes a simple HCF scheduling algorithm that is efficient for flows with strict CBR characteristics. However, a lot of applications, such as videoconferencing, have intrinsic VBR characteristics and this scheme becomes unfair because it can not adapt to flows which do not respect exactly their requirements or to flows whose characteristics fluctuate during time.

Another HCF scheduling algorithm has also been proposed in [9]. Actually this NS-2 simulation code implements both an old version of EDCF and an HCF scheduler (802.11e draft 2.0). This HCF scheduler is based on queue length information updated in the QAP each time a QoS data packet is received. A fixed amount of TXOP is allocated to the QSTA which has the highest average weighted queue length. All polled QSTAs receive the same TXOP duration. However, queue lengths of already-polled and non-polled QSTAs may not be updated any more since these QSTAs may not have the opportunity to transmit any packet and consequently to inform the QAP of its current queue length.

In this paper, we propose a new scheduling algorithm that aims to be fair for both CBR and VBR flows. Our scheme has not been designed to provide the lowest delays for all the

flows but rather to guarantee the minimum of all the Service Interval<sup>1</sup> (SI) requirements as maximum delay for all the flows while providing the required bandwidth to each flow.

The rest of this paper is organized as follows: Sections 2 and 3 detail the standard HCF algorithm and the mathematical relationship between scheduling delay, SI and queue length of the TS. Then, in Section 4 explains the principles of the FHCF scheme. Section 5 describes the conducted simulations to compare our FHCF scheme to EDCF and the standard HCF schemes and results are shown in Section 6. Finally, Section 7 concludes the paper.

## 2 The 802.11e HCF scheduling algorithm

A simple scheduling algorithm is proposed in the 802.11e specification [2] to take QoS requirements of traffics into account. Each QSTA that requires the HCCA mode has first to send a QoS request packet to the QoS Access Point (QAP) containing the mean data rate of the application using this TS, the MAC Service Data Unit (MSDU) size and the maximum SI required.

With these QoS requests, the QAP determines first the minimum value of all the SIs required by the different traffics which apply for HCCA. Then it chooses the highest submultiple of the superframe duration (duration between two beacons), which is inferior to the minimum of all the SIs<sup>2</sup>. Thus the superframe is cut into several SIs and QSTAs are polled according to a round-robin polling algorithm during each SI. As shown in Figure 1, after the Contention Free Period (CFP), EDCF periods and Controlled Access Periods (CAPs) are alternating during the Contention Period (CP).

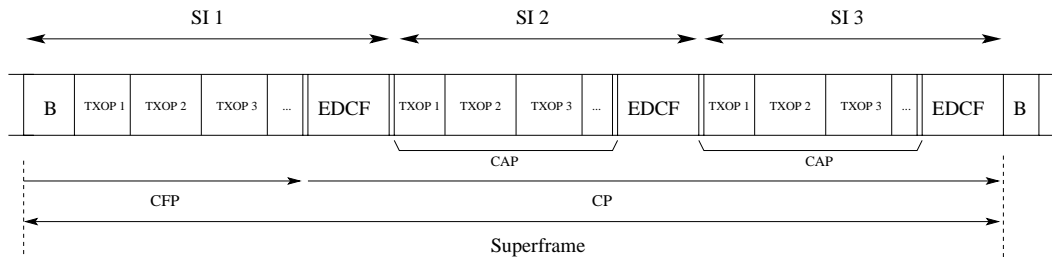


Figure 1: Structure of the superframe used in the HCF scheduling algorithm proposed in [2]

As soon as the SI is determined, the QAP evaluates the different TXOPs allocated to the different TSs of all the QSTAs which applied for HCCA. TXOP corresponds to the time to transmit the number of packets arriving during a SI in the TS queue. The number of

<sup>1</sup>The SI corresponds to the required duration between two polls of the same TS of a QSTA.

<sup>2</sup>For example, with a superframe duration of 500ms and two required SIs of 150ms and 200ms, the QAP chooses 125ms, the highest submultiple of 500 inferior to 150.

packets arriving in the TS queue  $j$  of the QSTA  $i$  during a SI is:

$$N_i^j = \lceil \frac{\rho_i^j * SI}{M_i^j} \rceil, \quad (1)$$

where  $\rho_i^j$  is the application data rate and  $M_i^j$  the MSDU size.

Then the different TXOPs  $T_i^j$  are evaluated:

$$T_i^j = N_i^j \left( \frac{M_i^j}{R} + 2 * SIFS + ACK \right), \quad (2)$$

where  $R$  is the physical transmission rate,  $SIFS$ , the Short Inter Frame Space and  $ACK$ , the time to transmit an acknowledgement packet.

The TXOP allocation scheme ensures that the queue lengths of each TS having been polled, is either constant if  $\frac{\rho_i^j * SI}{M_i^j}$  is an integer, or slowly decreasing.

This simple algorithm can be efficient if traffic respect strictly their QoS requirements. However, some real time applications such as videoconferencing, may generate VBR traffics and this scheme may cause the average queue length to increase. Even if the mean sending rate of such application is lower than the rate specified in its QoS requirements, sometimes, peaks of sending rate will not be absorbed by TXOPs allocated according to the QoS requirements. A more flexible scheme which adapts to fluctuating flows is then needed.

To understand in which extend this algorithm may be improved to guarantee maximum delay and how it can be adapted to traffics with fluctuating packet sizes and sending rates, it is important to understand the relationship between the SI, delays and queue lengths.

### 3 Relationship between delay and SI

Let us consider several simplifications to establish a simple relationship between SI and delay. First, we use an *efficient bandwidth*  $b$  corresponding to the actual amount of data transmitted per time unit. Second, we suppose that data are transmitted continuously (without taking packet cuttings into account). For the delay analysis, we consider the following two cases:

#### 3.1 Empty queues at the end of the TXOP

The delay for the packet  $P$  that arrives in the TS with polling order  $i$  at the time<sup>3</sup>  $t$  is due to the waiting time  $W$  for the next polling and the duration to send all the remaining packets in the queue when  $P$  arrives. The delay expression has different terms which correspond to

<sup>3</sup>Initial time corresponds to the beginning of the SI.

the different periods when the packet arrives: Indexes  $b$ ,  $d$  and  $a$  refer respectively to the periods before, during and after the polling. Then delay can be expressed as:

$$\begin{aligned} d_i(t) &= (W_b(t) + Q_b(t))\chi_{[0; \sum_{j=1}^{i-1} T_j]}(t) \\ &\quad + (W_d(t) + Q_d(t))\chi_{[\sum_{j=1}^{i-1} T_j; \sum_{j=1}^i T_j]}(t) \\ &\quad + (W_a(t) + Q_a(t))\chi_{[\sum_{j=1}^i T_j; SI]}(t) \end{aligned} \quad (3)$$

with:

$$\begin{aligned} W_b(t) &= \sum_{j=1}^{i-1} T_j - t, & Q_b(t) &= \frac{q_i^0 M_i}{b} + \frac{\rho_i t}{b} \\ W_d(t) &= 0, & Q_d(t) &= \frac{q_i^0 M_i}{b} + \frac{\rho_i t}{b} - (t - \sum_{j=1}^{i-1} T_j) \\ W_a(t) &= SI - t + \sum_{j=1}^{i-1} T_j, & Q_a(t) &= \frac{\rho_i}{b} (t - \sum_{j=1}^i T_j) \end{aligned}$$

and

$$\chi_{[a;b]}(t) = \begin{cases} 1 & \text{if } t \in [a; b] \\ 0 & \text{else} \end{cases}$$

is used for the different periods.

Equation (3) can be rewritten as:

$$\begin{aligned} d_i(t) &= \left( \sum_{j=1}^{i-1} T_j - t + \frac{q_i^0 M_i}{b} + \frac{\rho_i t}{b} \right) \chi_{[0; \sum_{j=1}^i T_j]}(t) \\ &\quad + \left( SI - t + \sum_{j=1}^{i-1} T_j + \frac{\rho_i}{b} (t - \sum_{j=1}^i T_j) \right) \chi_{[\sum_{j=1}^i T_j; SI]}(t). \end{aligned}$$

Note that delay is discontinuous at the end of the TXOP since packets arriving after the end of the TXOP have to wait for the next TXOP at the next SI.

### 3.2 Non-empty queues at the end of the TXOP

According to the standard HCF algorithm proposed in 802.11e, the queue at the end of the TXOP may not be empty and our first assumption becomes wrong. In this case, we have to consider  $q_i^e$ , the queue length of the TS at the end of the TXOP  $T_i$ .

If we suppose delay discontinuity during the TXOP (see Figure 2), delay can be expressed as:

$$d_i(t) = \left( \sum_{j=1}^{i-1} T_j - t + \frac{q_i^0 M_i}{b} + \frac{\rho_i t}{b} \right) \chi_{[0; t^d]}(t) \\ + (SI - t + \sum_{j=1}^{i-1} T_j + \frac{\rho_i}{b} (t - t^d)) \chi_{]t^d; SI]}(t),$$

where  $t^d$  is the time when delay discontinuity happens.  $t^d$  corresponds to the arriving time of the first packet which can not be transmitted during the current TXOP:

$$t^d = \sum_{j=1}^i T_j - \frac{q_i^e M_i}{\rho_i}.$$

Thus the maximum delay denoted by  $D_i$  is obtained at the time  $t^d$  and is equal to:

$$D_i = \max_t d_i(t) = SI - T_i + \frac{q_i^e M_i}{\rho_i}. \quad (4)$$

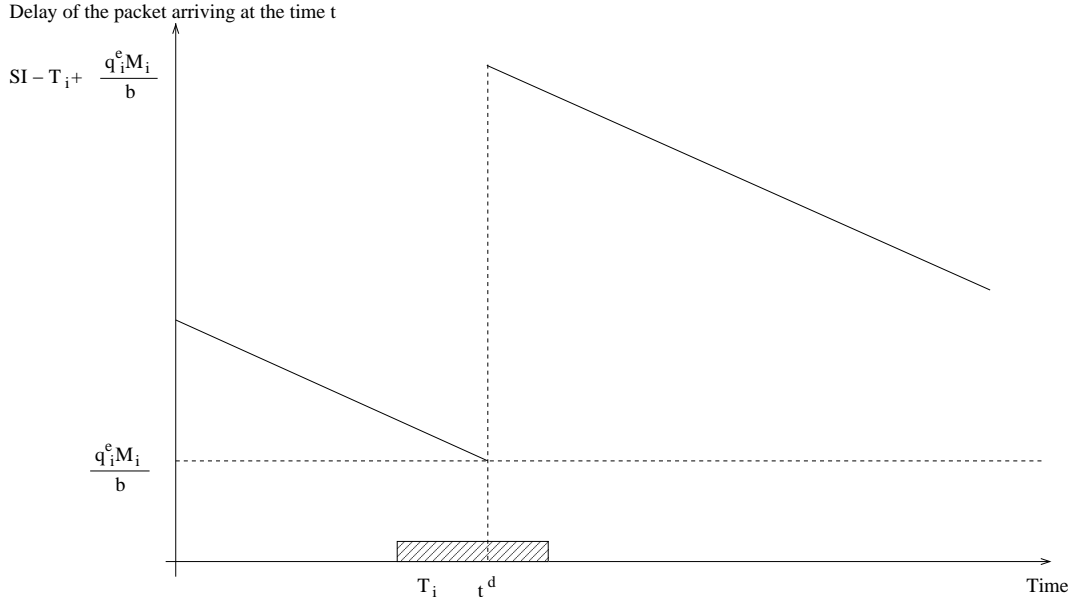


Figure 2: Theoretical delay versus time

Equation (4) shows that delay is higher if some packets remain in the queue at the end of the poll. Note that if we suppose that the queue is empty at the end of the poll,  $D_i$  is almost equal to the SI duration since  $T_i \ll SI$ .

Since the TXOP allocated to the TS with polling order  $i$ , is:

$$T_i = \frac{q_i^b M_i}{b} + \frac{\rho_i}{b} T_i - \frac{q_i^e M_i}{b},$$

with  $q_i^b = q_i^0 + \frac{\rho_i}{M_i} \sum_{j=1}^{i-1} T_j$  the queue length just before polling, using (4), the maximum delay is also equal to:

$$\begin{aligned} D_i &= SI + \frac{q_i^b M_i}{\rho_i} - \frac{b}{\rho_i} T_i \\ &= SI \left( 1 + \sum_{j=1}^{i-1} \frac{T_j}{SI} - \frac{b}{\rho_i} \frac{T_i}{SI} \right) + \frac{q_i^0 M_i}{\rho_i}. \end{aligned} \quad (5)$$

(5) shows that we can reduce the maximum delay  $D_i$  by increasing the TXOP  $T_i$  allocated to the TS  $i$  because the number of packets remaining in the queue at the end of the TXOP decreases when the allocated TXOP increases. From (1) and (2), we can note that  $T_i/SI$  does not depend on the SI duration, so we are able to reduce the maximum delay by reducing the SI duration. However, this increases the utilization rate of the HCCA since the number of SIs, polls and overheads increase in the same time. Furthermore, we can control the maximum delay of packets if we are able to control the queue length before polling  $q_i^b$ , whose value is determined by the data rate of the TS. If we suppose traffic is not CBR traffic,  $q_i^b$  may vary considerably.

The advantages of this scheme are the possibility to control delay of each accepted traffic given that the queue length is controlled before polling. So each traffic should have the same maximum delay linked to the SI duration chosen by the QAP without any additional cost.

## 4 The FHCF Scheme

In this section, we detail the principles of the FHCF scheme that takes queue length information into account to support both CBR and VBR traffics. First we describe the QAP scheduler, which uses a window of previous estimation errors for each TS of each QSTA, to adapt the computation of the TXOP allocated to the QSTA. Then we show how a QSTA can redistribute the unused time among its different TSs through the node scheduler since the TXOP is always allocated to a whole QSTA in our scheme.

### 4.1 A queue length based scheduling scheme at the QAP

If we suppose traffic has not strict CBR characteristics, the standard HCF scheme is not able to provide maximum delay requirements since the average queue length may not be controlled. We propose the following modifications: First, the QAP uses the last queue length information sent by the QSTA during its polling, to estimate queue length at the

beginning of the next SI. Second, we compare this estimation to the “ideal”<sup>4</sup> queue length at the beginning of the next SI to compute a TXOP. Third, the QAP modifies this TXOP by redistributing the remaining time of the CAP (or CFP) to the different QSTAs according to the variations of the different flows (sending rate and/or packet size).

#### 4.1.1 Use of a queue length estimation at the beginning of the SI

When a QSTA sends a QoS data packet, the QAP uses the QoS control field of the 802.11 header to record the queue length of the TS which sends the packet. Reception time of the packet is also recorded. Thus the QAP has at its disposal the couple  $(q_i^e, t_i^e)$  of each TS at the end of each CAP (or CFP).

Using this information, the QAP is able to estimate the queue length of the  $i^{th}$  polled TS  $q_i^{est}$  at the beginning of the next SI according to QoS parameters such as the application data rate and the MSDU size:

$$q_i^{est} = \frac{\rho_i(SI - t_i^e)}{M_i} + q_i^e. \quad (6)$$

#### 4.1.2 Comparison with the “ideal” queue length distribution

Expression (4) shows that the maximum delay will be minimized if we manage to have  $q_i^e = 0$  with the same TXOP allocation as the one proposed in the 802.11e draft [2]. Our algorithm tries to allocate TXOP in order to make each  $q_i^e$  tends to 0 by using the same TXOP allocation basis as the one detailed in Section 2. On this purpose, the QAP computes at the beginning of the SI the ideal queue length distribution according to the polling order and the QoS parameters of each accepted traffic. The ideal queue length can be obtained from:

$$\begin{aligned} q_i^{ideal} &= \frac{\rho_i(SI - \sum_{j=1}^i N_j(\frac{M_j}{R} + 2 \times SIFS + ACK))}{M_i} \\ &\simeq \frac{\rho_i SI}{M_i} (1 - \sum_{j=1}^i \frac{\rho_j}{b}). \end{aligned} \quad (7)$$

Three situations are possible (see diagrams on Figures 3 and 4):

- $q_i^{est} = q_i^{ideal}$ : if we allocate the TXOP  $T_i$  according to the standard scheme, the queue length will be zero at the end of the TXOP and the lowest delays of the packets will be obtained.
- $q_i^{est} > q_i^{ideal}$ : more packets than expected arrived earlier in the queue according to the mean data rate (case 1 in Figure 3).

<sup>4</sup>The ideal queue length corresponds to a null queue length at the end of the TXOP computed according to the 802.11e [2].

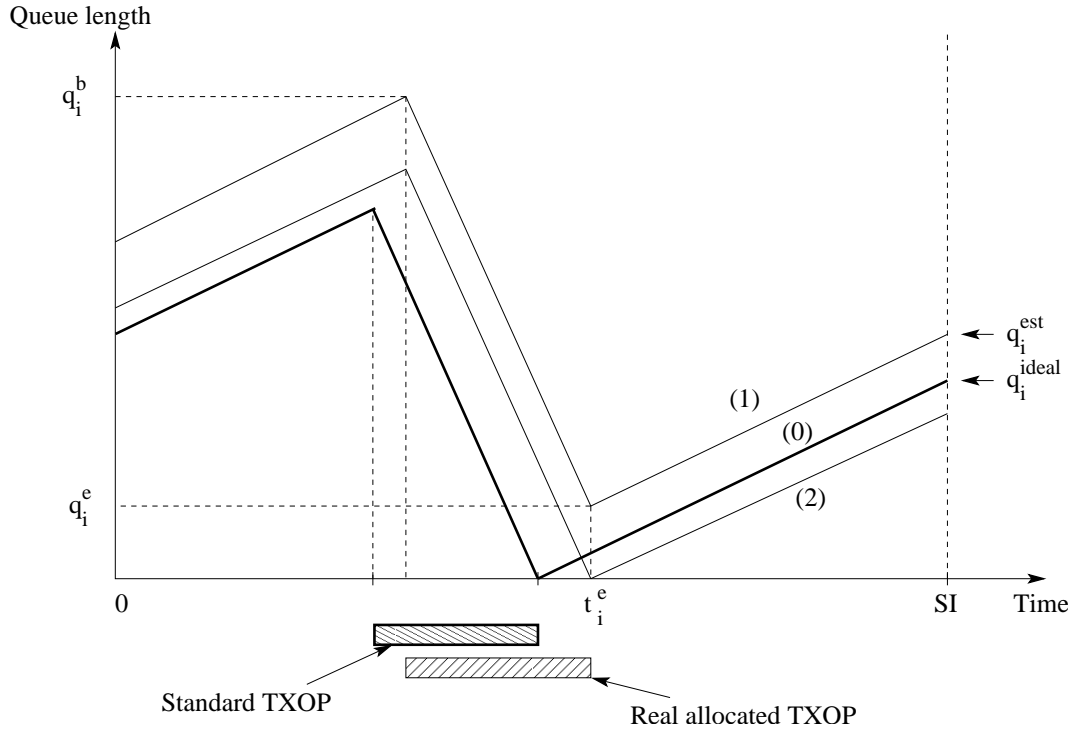


Figure 3: Queue length evolution for a TS ; (0): ideal queue length evolution ; (1):  $q_i^{est} > q_i^{ideal}$  ; (2):  $q_i^{est} < q_i^{ideal}$

- $q_i^{est} < q_i^{ideal}$ : the QSTA has been polled later than expected (case 2 in Figure 3). Then the estimated queue length is lower than the ideal queue length because the real TXOP finishes later than the standard TXOP (see (6)).

#### 4.1.3 Estimation of the fluctuations of the number of arriving packets

Even if the QSTA precises in a QoS request frame<sup>5</sup> its QoS requirements (such as the mean sending rate, the mean packet size and the maximum SI) for a certain type of flow, the real flow may not meet exactly these requirements. In this case, the previous queue length estimation might be wrong due to, for example, sending rate variations.

<sup>5</sup>This frame is called an ADDTS QoS Action frame in [2].

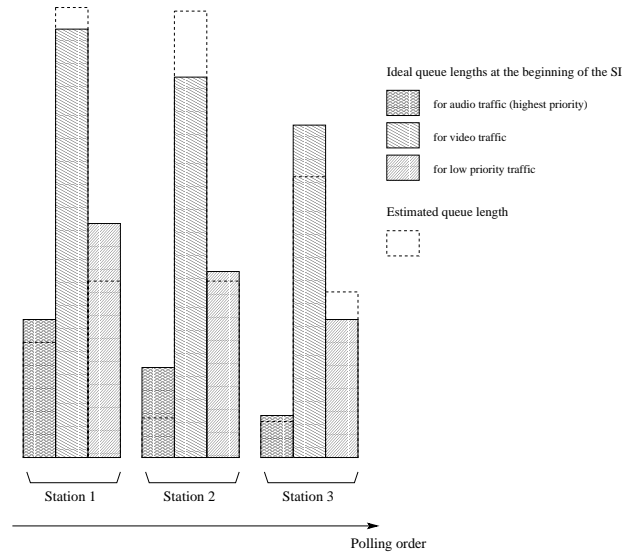


Figure 4:  $q_i^{ideal}$  and  $q_i^{est}$  distribution at the beginning of the SI

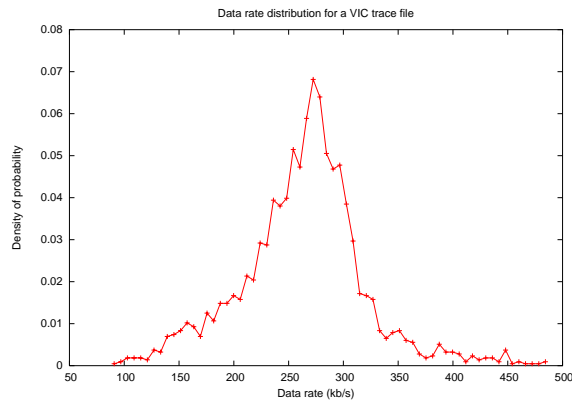


Figure 5: Distribution of the sending rate of a real VBR video traffic

Figure 5 shows that for a typical VBR video traffic<sup>6</sup>, the sending rate almost follows a Gaussian law and packets have a fluctuating size. If we only take the previous queue length

<sup>6</sup>This video traffic has been obtained using the VIC videoconferencing tool and the H.261 coding in the QCIF format for a typical “head and shoulder” video sequence.

estimation into account (see (6)), some packets may not be transmitted at the next TXOP due to a higher sending rate than computed according to the QoS parameters. In this case, latency of these packets is higher than the SI duration and delay can not be controlled. Our scheme uses the queue length information of the TS polled with the polling order  $i$  sent by a QSTA at the beginning of the TXOP. The QAP records at each beginning of TXOP during the SI  $n$  the absolute value  $|\Delta_i^n|$  of the difference between the real queue length ( $q_i^{b,real}$ ) at the beginning of the polling of the TS  $i$  and the estimation of this queue length ( $q_i^{b,est}$ ):

$$\Delta_i^n = q_i^{b,real} - q_i^{b,est}.$$

If the sending rate follows a Gaussian law,  $\Delta_i$  also follows the same kind of law with an expected value of 0. Then, the probability for  $\Delta_i$  to be  $N$  packets is:

$$P_{\Delta_i}(N) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{N^2}{2\sigma^2}\right).$$

To simplify calculations, suppose that data arrive in a continuous bit stream at the TS without being cut into packets and consider  $\delta_i$  the difference between the real amount of data and the estimated amount of data. Like  $\Delta_i$ ,  $\delta_i$  also follows a Gaussian law. The density of probability for  $\delta_i$  is:

$$P_{\delta_i}(y) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{y^2}{2\sigma^2}\right)$$

where  $y$  is the amount of additional data.

Consequently, the density of probability of  $|\delta_i|$  is equal to:

$$P_{|\delta_i|}(y) = \begin{cases} 2P_{\delta_i}(y) & \text{if } y \geq 0 \\ 0 & \text{if } y < 0. \end{cases}$$

The expected value of the variable  $|\delta_i|$  is equal to:

$$E(|\delta_i|) = \int_0^{\infty} \frac{2y}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{y^2}{2\sigma^2}\right) dy = \sqrt{\frac{2}{\pi}} \sigma$$

As expected, the average value of  $|\delta_i|$  is a function of the variance of the distribution of  $\delta_i$ . Like  $|\delta_i|$ , the average value of  $|\Delta_i|$  increases with the variance of the sending rate. Thus, in order to allow the QSTA to transmit more packets due to fluctuating sending rates and packet sizes, the QAP scheduler records a window of  $w$  previous  $|\Delta_i|$ , whose average value is supposed to be close to the expectation of  $|\Delta_i|$ .

Using this mechanism, the QAP is able to estimate by excess the queue length  $Q_i^{est}(n)$  at the beginning of the next polling of the TS with the linear estimation  $q_i^{b,est}(n)$  of the queue length at the beginning of the polling:

$$Q_i^{est}(n) = q_i^{b,est}(n) + \frac{\sum_{j=n-w}^{n-1} |\Delta_i^j|}{w}.$$

Then the difference between the estimation and the ideal queue length before polling  $q_i^{b,ideal}$  becomes:

$$q_i^{b,est}(n) + \frac{\sum_{j=n-w}^{n-1} |\Delta_i^j|}{w} - q_i^{b,ideal}(n).$$

Since  $q_i^{b,est}(n)$  and  $q_i^{b,ideal}(n)$  depend on the time of the beginning of the TS  $i$  polling, and also depend on all the previous real TXOPs, the use of  $q_i^{est}(n)$  (see (6)) and  $q_i^{ideal}$  (see (7)) is sufficient to estimate the amount of additional packets in the TS. Indeed, the estimation of the queue length and the ideal queue length evolve linearly during time (see (6), (7)) and then, we obtain:

$$q_i^{b,est}(n) - q_i^{b,ideal}(n) = q_i^{est}(n) - q_i^{ideal}(n).$$

Finally, our scheme bases its time reallocation on the estimation of the number of additional packets  $\Delta_i^{est}$ :

$$\Delta_i^{est} = q_i^{est}(n) - q_i^{ideal}(n) + \frac{\sum_{j=n-w}^{n-1} |\Delta_i^j|}{w}$$

where  $q_i^{est}(n)$  and  $q_i^{ideal}(n)$  are given by (6) and (7).

#### 4.1.4 Redistribution by the scheduler of the unallocated time

Our algorithm consists in redistributing a part of the remaining time  $T^r$  to each QSTA in order to absorb some possible peak sending rates:  $T^r = T_{CAP} - \sum_{i=1}^{K \times p} T_i$ , where  $T_{CAP}$  is the maximum bound of the CAP duration within a SI,  $K$ , the number of QSTAs,  $p$ , the number of TSs in each QSTA and  $T_i$ , the TXOP allocated to the  $i^{th}$  TS to be polled ( $T_i$  may be 0 if traffic has not been accepted or if this TS does not require TXOP to transmit its packets).

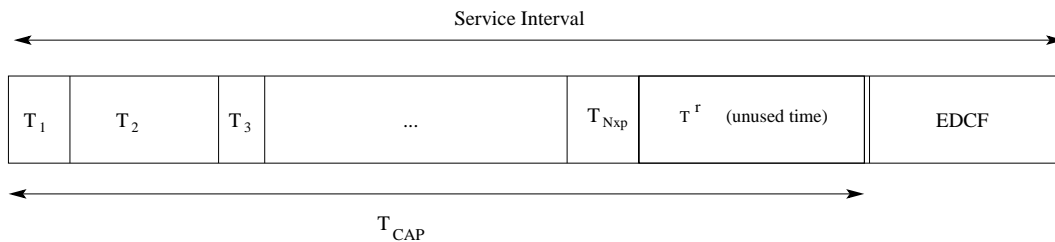


Figure 6: Diagram of the remaining time usable by our algorithm

First, the algorithm computes the additional required time  $t_i^{est}$  (which may be positive or negative) for each TS of each QSTA according to the estimation of  $\Delta_i^{est}$ :

$$t_i^{est} = \Delta_i^{est} \left( \frac{M_i}{R} + 2 * SIFS + ACK \right). \quad (8)$$

Second, it computes separately  $T_P$ , the sum of all the positive values, and  $T_N$ , the opposite of the sum of all the negative values. If  $T_P - T_N > T^r$ , it means that the scheduler is not able to allocate all the time it expected according to the estimations, and all of these additional times  $t_i^{est}$  need to be reduced. On the one hand, the scheduler reduces each positive additional time fairly with a percentage  $\beta$  (chosen negative to correspond to a reduction) of this additional time. On the other hand, all the negative additional times are increased by the same percentage  $\beta$  of the negative additional time.

Consequently,  $\beta$  is the solution of:

$$(1 + \beta)T_P - (1 - \beta)T_N = T^r$$

and can be expressed as:

$$\beta = -\frac{(T_P - T_N) - T^r}{T_P + T_N}. \quad (9)$$

Finally, the *effective additional time*  $t_i^{add}$  allocated to the TS  $i$  is:

$$t_i^{add} = \begin{cases} (1 + \beta)t_i^{est} & \text{if } t_i^{est} \geq 0 \\ (1 - \beta)t_i^{est} & \text{if } t_i^{est} < 0. \end{cases} \quad (10)$$

When it is time for the QAP to poll a QSTA, the QAP computes the sum of all the normal TXOPs and the *effective additional times* allocated to the different TSs of the QSTA.

## 4.2 The node scheduler

The node scheduler also plays a very important role since it has to redistribute the additional allocated time to the different TSs. It performs almost the same computations as the QAP. First it computes  $N_j$ , the number of packets to transmit in each TS, and the time required to transmit a packet according to its QoS requirements (packet size, data rate). Second, according to its allocated TXOP  $T$  and the number of packets the QSTA should transmit from each TS, it evaluates the remaining time  $T^r$  it can reallocate:

$$T^r = T - \sum_{j=1}^p N_j \left( \frac{M_j}{R} + 2 * SIFS + ACK \right).$$

Since each QSTA knows exactly its own TS queue sizes at the beginning of the polling, it is able to estimate more precisely its queue sizes at the end of the TXOP and consequently the required additional time per TS. With this information, the QSTA performs the same computations as the QAP (see (8), (9) and (10)). The difference is that the coefficient  $\beta$  may be positive if the QSTA has more time than required to send all its packets or negative if, on the contrary, the remaining time is less than required. Thus, just after the CF-Poll reception, the QSTA can redistribute additional time to its different TSs with the option to add more time to each TS if  $\beta$  is positive.

## 4.3 FHCF Implementation in NS-2

### 4.3.1 Basis of FHCF scheme

Our implementation [11] of FHCF is based on Stanford's NS-2 implementation of EDCF/HCF [9]. In this implementation, 8 TCs of different priorities have been assigned different EDCF parameters such as CWmin, CWmax depending on the priority of the TC. These TCs are also used in HCCA mode. According to the 802.11e standard [2], TC and TS queues should be separated in order to prevent EDCF from sending packets from TS queues. In our implementation, if a traffic from a priority  $i$  is accepted in HCCA, the TC  $i$  becomes a Traffic Stream (TS) and packets can only be sent in HCCA. By this way, it is possible to evaluate the performance of the different schedulers using only CFPs and CAPs.

As regards the HCCA mode, we extended it to respect QoS traffic requirements because Stanford's implementation is based on the use of old queue length information without any additional estimation.

Moreover, in their implementation, beacon frames could be delayed since the medium was sometimes used at the time the beacon was scheduled. We fix the problem of beacon delay in our code by adding a beacon timer in every QSTA, to check before each transmission if there is enough time to transmit packets without disturbing the beacon sending.

### 4.3.2 TSPEC negotiation

As mentioned before, the purpose of FHCF algorithm is to improve the HCF scheduling algorithm by adapting it to fluctuating flows and by providing fairness. In this optic, we only consider HCCA traffic in the remainder of the paper. If a QSTA wants to send a certain flow, it will use the HCCA.

According to the 802.11e standard [2], a QSTA has to send a QoS request frame to the QAP whenever it wants to transmit packets of a certain TS in HCCA. In our implementation, QSTAs first start to transmit packets in EDCF in order that the QAP records source and priority of traffics. Thus, the first packet sent from a certain TC plays the role of a QoS request frame and the QAP considers this packet as a QoS request frame. If this flow is accepted, the QAP schedules it and the QSTA is informed at the reception of the next CF-Poll which contains both the TXOP duration and an 8 bit integer indicating the accepted traffics. If the  $j^{th}$  bit is one, it means that the TC  $j$  is taken into account by the scheduler to plan packets transmission during the HCCA mode.

As regards QoS parameters of the TSPEC negotiation, the nominal MSDU size, the mean sending rate and the maximum acceptable SI are determined statically at the beginning of the simulation for each among the 8 TCs of different priorities. This does not change the spirit of the scheduler which can also be adapted to a real TSPEC negotiation with different QoS requirements for TSs of the same priority.

## 5 Description of simulations

Basically, our scheme consists of 2 different schedulers: the QAP scheduler which distributes TXOPs to the different QSTAs and the node scheduler which redistributes its allocated TXOP to the different TSs accepted in HCCA.

In this section, we compare the behavior of EDCF, the standard HCF and the FHCF schemes for different kinds of traffic.

We use two different topologies to evaluate the performance of these schemes. The first one contains 19 nodes with only one traffic per node and is used to evaluate the performance of the QAP scheduler. The second topology uses 6 nodes, each one with traffics from different priorities and is used to evaluate the performance of the node schedulers. For all the simulations, the destination of all the flows is the QAP: This allows us to compare fairly end-to-end delays among the different flows.

### 5.1 Evaluation of the QAP scheduler

With the first considered topology, 6 QSTAs send a high priority CBR on-off audio traffic ( $64kb/s$ ), 6 QSTAs send a VBR video traffic ( $200kb/s$  of average sending rate) with medium priority and 6 QSTAs send a CBR MPEG4 [7] video traffic ( $3.2Mb/s$ ) with low priority. Table 1 summarizes the different traffics used for this simulation and Table 2 details the audio flow characteristics chosen in order that on-off periods correspond to a typical phone conversation [12]. In each QSTA, we use only one of the 8 TSs. The transport protocol used in each node is UDP. Audio flows are mapped to the 6<sup>th</sup> TS of the MAC layer whereas VBR H.261 and CBR MPEG4 video flows are respectively mapped to the 5<sup>th</sup> and 4<sup>th</sup> TS. The different VBR flows have been obtained with the VIC [8] videoconferencing tool using the H.261 coding and QCIF format for typical “head and shoulder” video sequences. We made 6 trace files available from [11]: the mean sending rate was close to  $200kb/s$  with a mean packet size of  $660bytes$  and a mean interarrival time of  $26ms$ . A simple analysis of the trace files shows that the mean sending rate belongs to a window of  $80kb/s$  around the mean value of  $200kb/s$  and the mean packet size between  $600$  and  $700bytes$ . Packet sizes of these flows belong to a large range of values between  $20$  and  $1024bytes$ <sup>7</sup>.

Table 1: Description of the different traffics

Node	Application	Arrival period ( <i>ms</i> )	Packet size ( <i>bytes</i> )	Sending rate ( <i>kb/s</i> )
1 → 6	Audio	4.7	160	64
7 → 12	VBR video	≈ 26	≈ 660	≈ 200
13 → 18	MPEG4 video	2	800	3200

<sup>7</sup>By default, VIC uses a MTU size of  $1024bytes$ .

Table 2: Parameters of the audio traffic

Maximum application rate	Mean burst time	Mean idle time
$64kb/s$	$400ms$	$600ms$

For simulations with FHCF and standard HCF, we set the duration parameters to the values summarized in Table 3 where the HCCA duration limit corresponds to the sum of the first CFP following the beacon frame and all the CAPs within a superframe (duration between 2 beacon frames). However, the HCCA duration limit is a maximal bound and the CFP (or each CAP) may prematurely terminate if the QAP sends a CF-END packet. The HCCA utilization rate corresponds to the amount of HCCA used for the different flows (TXOPs and overheads) which have been accepted in this mode. In all the simulations considered here, all the flows have been accepted in HCCA, which means that we do not consider the case of any rejected flow using EDCF. This is in order to compare the real HCCA/EDCF performance of the different schemes.

Table 3: Parameters used for the simulations

Scheme	Beacon period	HCCA duration limit	HCCA utilization rate
FHCF	$500ms$	$490ms$	82.4%
Standard HCF	$500ms$	$490ms$	82.4%
EDCF	$500ms$	$1ms$	0%

PHY and MAC parameters are summarized in Table 4. For the simulation of EDCF, we used 3 TCs: TC 6 for audio flows, TC 5 for VBR video flows and TC 4 for CBR MPEG4 flows but the parameters settings of CWmin and CWmax only differentiate audio and video flows. The mapping of VBR video and CBR video traffics to different TCs with the same EDCF parameters, allows us to compare EDCF behavior with high load (CBR MPEG4) and low load (VBR H.261) video traffics. EDCF parameters have been chosen in order to allocate higher priority to audio traffics, than to video traffics and finally to any background traffics. The fact to consider background traffic in our simulations does not bring any additional interesting information since the EDCF parameters for the low priority TC are chosen in order to have a large contention window (with CWmin higher than all the CWmaxs of high priority TCs).

## 5.2 Evaluation of the node scheduler

To evaluate the node scheduler algorithm, we use the second topology with 7 QSTAs including the QAP with 6 QSTAs sending, each of them, both audio, VBR H.261 and CBR MPEG4 video traffics with the same features than before (see Table 5). This topology

Table 4: The PHY and MAC parameters

PHY parameters		MAC parameters	
SIFS	16 $\mu$ s	CWmin (audio)	7
DIFS	34 $\mu$ s	CWmax (audio)	15
ACK size	14bytes	CWmin (video)	15
PHY rate	36Mb/s	CWmax (video)	31
Minimum Badwidth	6Mb/s		
SlotTime	9 $\mu$ s		
CCA Time	4 $\mu$ s		
MAC header	38bytes		
PLCP header length	4bits		
Preamble length	20bits		

aims at evaluating the behavior of the different schemes in terms of fairness between the same kinds of flows and in terms of respect of QoS requirements for different HCCA loads. HCCA load has been modified by increasing the packet size of the CBR MPEG 4 traffic from 600bytes (2.4Mb/s) to 1000bytes (4Mb/s) using a 100byte increment and keeping the same interarrival period of 2ms. This is a time-consuming way to increase load because CBR video packets need more time to be transmitted. Another way to increase the network load is to increase the number of QSTAs in order to increase the number of collisions in EDCF.

To compare fairness in terms of delay between the same kinds of traffics for the different schemes, we use Jain's fairness index [13]:

$$J = \frac{(\sum_{i=1}^n d_i)^2}{n \sum_{i=1}^n d_i^2}$$

where  $d_i$  is the mean delay of the flow  $i$  and  $n$  is the number of flows for which the fairness index is computed.

Table 5: Description of the different traffics

Node	Application	Arrival period (ms)	Packet size (bytes)	Sending rate (kb/s)
1 $\rightarrow$ 6	Audio	4.7	160	64
1 $\rightarrow$ 6	VBR video	$\simeq$ 26	$\simeq$ 660	$\simeq$ 200
1 $\rightarrow$ 6	CBR video	2	600 $\rightarrow$ 1000	2400 $\rightarrow$ 4000

## 6 Results

### 6.1 Evaluation of the QAP scheduler

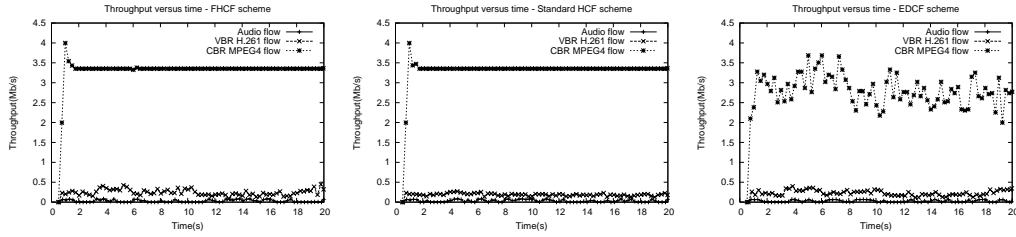


Figure 7: Throughput versus time for FHCF, Standard HCF and EDCF

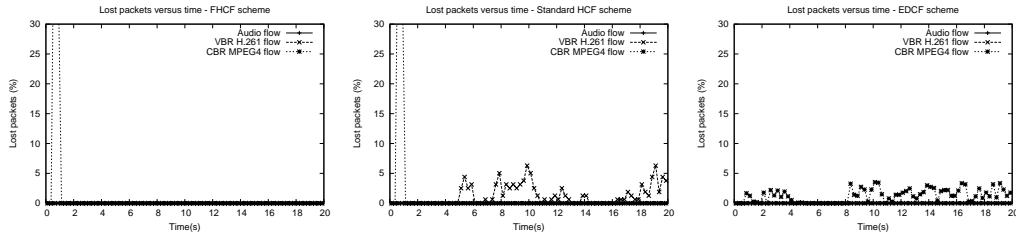


Figure 8: Dropped packets versus time for FHCF, Standard HCF and EDCF

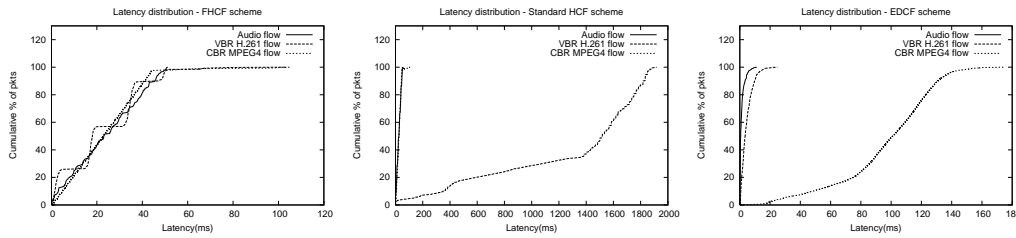


Figure 9: Latency distribution for FHCF, Standard HCF and EDCF

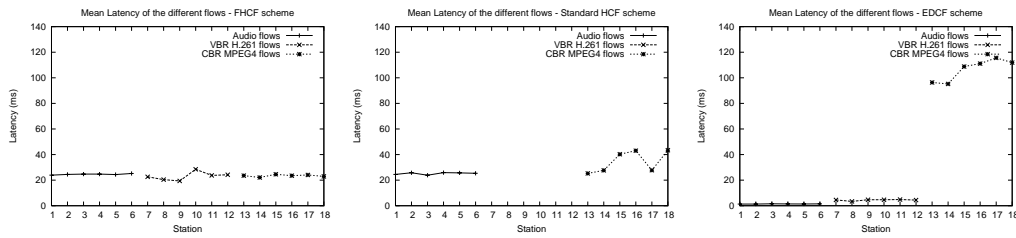


Figure 10: Mean latency for each flow with FHCF, Standard HCF and EDCF

In order to simplify curves and to compare the different flows, only one curve is shown for each kind of flows on all the figures.

### 6.1.1 Comparison of throughputs

Throughput curves on Figure 7 show that both FHCF and Standard HCF schemes succeed in providing the required throughputs for all the flows. For VBR H.261 flows, throughput is more fluctuating with FHCF than with standard HCF scheme since FHCF is able to adapt all the allocated TXOPs according to the queue length evolutions. For CBR MPEG4 flows, both FHCF and standard HCF provide a constant throughput whereas EDCF provides a more fluctuating throughput due to collisions and the use of a probabilistic channel access. As regards audio flow, throughput is sometimes equal to zero since it matches burst periods and idle periods.

### 6.1.2 Comparison of the number of dropped packets

Table 6 and Figure 8 show that EDCF drops a lot of MPEG4 packets because of the high load of the 6 CBR MPEG4 flows ( $3.2Mb/s$  each). Indeed these flows introduce a lot of collisions with the EDCF scheme (11.4% of the simulation duration is in collisions) and the throughput of CBR MPEG4 flows is then reduced (Figure 7 shows that the throughput granted to the CBR flows is lower in EDCF than in the other schemes). With the standard HCF, there are no CBR MPEG4 packets lost<sup>8</sup> but 513 VBR packets are lost. There are two possible reasons: First, some VBR flows may have a higher sending rate than the mean sending rate specified in their requirements, and second the sending rate is also fluctuating during time. This confirms, as explained in Section 2, that standard HCF has not been designed for this kind of flows but rather for CBR traffics. On the contrary, FHCF succeeds in having no dropped packets since the TXOPs allocated to the different QSTAs are adapted to the real queue lengths on the basis of their nominal QoS requirements.

Table 6: Number of drops for each kind of flow with FHCF, Standard HCF and EDCF

	Audio	VBR video	CBR MP4 video
FHCF	0	0	204
Standard HCF	0	513	204
EDCF	1	3	8443

### 6.1.3 Analysis of latency distributions

For the FHCF scheme, Figure 9 shows that all the flows have a maximum latency which corresponds to the SI requirements of the different flows (chosen at  $50ms$ ) whereas for

<sup>8</sup>Except at the beginning of the simulation because all the flows start at time 0 and the network is congested at the starting time.

other schemes, some flows may not have their QoS requirements met. We observe on the same figure that the latency distribution curve of the VBR flow has a stair shape. If we analyze the trace file of the VBR flow represented on the different curves, we note that the interarrival time of packets is not  $26ms$  (see Table 1) but precisely  $33ms$  since two packets arrive sometimes at the same time<sup>9</sup> and the real interarrival time is then higher than the mean value. Because packets arrive regularly and the real interarrival time is not changing during time, delays of packets are not regularly distributed between 0 and  $50ms$ . On the whole, with a SI of  $50ms$  and an interarrival time of  $33ms$ , Figure 11 shows that delays take only three values (provided that queue lengths are almost always empty). However, the steps of the VBR curve obtained with FHCF on Figure 9 are not completely vertical since 33 is not an exact submultiple of 100 and the three values of delay are derivating slowly during time.

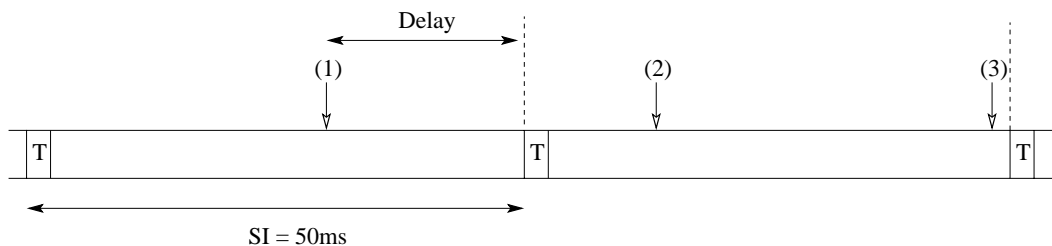


Figure 11: Delays of the considered VBR flow: (1) about  $20ms$ , (2) about  $37ms$ , (3) about  $4ms$

Figure 10 represents the mean latency for each station (sending only one traffic). We observe that for the FHCF scheme, all the flows have a mean latency almost equal to  $25ms$ . This means (see Figure 2) that the queues at the end of the TXOPs allocated to each QSTA are almost always empty ( $q_i^e \simeq 0$ ). Effectively, the mean latency on a SI, which can be expressed as:

$$\frac{1}{2}(SI - T_i + 2\frac{q_i^e M_i}{b}), \quad (11)$$

is equal to  $25ms$  ( $T_i \ll SI$ ). However, for the VBR flows, the mean latency is not exactly the same. The unfairness comes from the effect summarized in Figure 11. Indeed, the three different arrival times of the packets in the SI may be different from a flow to the other (shifted in time as shown in Figure 11).

As regards the standard HCF scheme, the delays of the VBR flow are completely uncontrolled (see Figure 9) because the queue lengths are increasing during time (that is why, according to (11), their mean latency is high) and we did not choose to represent them on Figure 10. Note that the standard HCF scheme may be efficient if TXOPs are allocated

<sup>9</sup>This is due to VIC fragmentation.

according to the maximum sending rate of the VBR flows but, in this case, fewer flows than with FHCF can be accepted in HCCA. In our example of VBR flows, the gain with FHCF is between 14% and 37% depending on the flow. We can also observe on Figure 10 that, sometimes, even for CBR traffic, the standard HCF scheme is not fair: It may happen for some flows (see (1)) that

$$\lceil \frac{\rho_i^j * SI}{M_i^j} \rceil = \frac{\rho_i^j * SI}{M_i^j}.$$

In this case, the standard HCF scheme does not allocate any additional time (see (2)) to the flows. That is exactly what happens in our simulation: the interarrival time of CBR packets is  $2ms$  and exactly 25 packets arrive in the TSs during the SI ( $50ms$ ). Consequently, the final queue length, at the time QSTAs with CBR traffics finish to be polled, is not decreasing during time and the lowest delays (about  $25ms$ ) are not supported. On the contrary, for audio flows, this scheme is relatively fair since TXOPs are computed on the basis of the peak sending rates and queues are always empty at the end of the polling.

For EDCF, we can observe that the audio flow has a very low latency (under  $5ms$ ). On the contrary, the video flows (mapped to TCs which have the same EDCF parameters) have different behaviors: the VBR H.261 flows have very low latency compared to the CBR MPEG4 flows (up to  $180ms$ ). EDCF has then a different behavior depending on the load of the TCs. Note that if we use the same TC for both CBR MPEG4 flows and for VBR H.261 flows, performance in terms of throughput and delays is worse. On the contrary, FHCF scheme is less load-dependant in the presence of traffics which do not fluctuate too much and exhibits the same behavior for the different flows. However, if traffics fluctuate a lot around their nominal QoS requirements, the FHCF scheduler may not have enough time to absorb the different fluctuations of sending rates.

## 6.2 Evaluation of the node scheduler

We use a second set of simulations to evaluate the FHCF node scheduler and to show the behavior of the different schemes when the load increases (by increasing the sending rates of the CBR MPEG4 traffics). The node scheduler of the standard HCF implements the same algorithm as the QAP scheduler. It computes at the beginning of the TXOP the different numbers of packets each TS has the right to send according to the QoS requirements.

Figures 12 and 13 show respectively the mean delays and the fairness of the different types of flows obtained with the various schemes for several loads of the network (see Table 5).

### 6.2.1 Audio and VBR H.261 video flows

Figure 12 shows that with FHCF, delay curves are almost horizontal lines which means that delays do not strongly depend on the network load. However, EDCF succeeds in providing very low delays since audio flows and VBR flows are low rate flows compared to the CBR flows. For the same reasons as for the previous simulations, the delays of the VBR flows

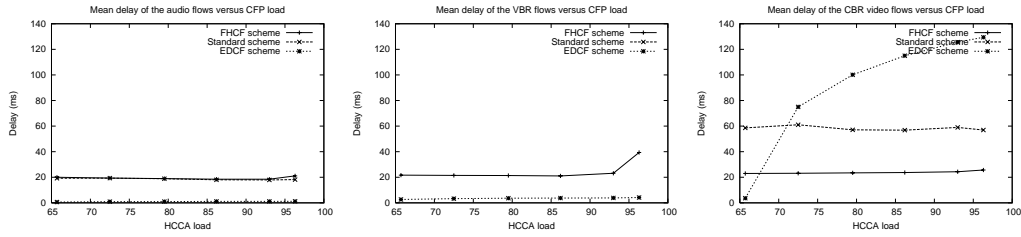


Figure 12: Delays

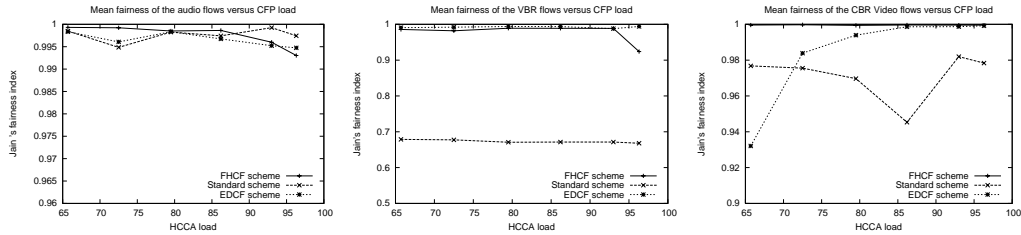


Figure 13: Fairness

obtained with the standard HCF scheme are too high to be represented on the figure (the mean delays for the VBR flows are almost  $300ms$ ).

As shown in Figure 13, Jain's fairness index between audio flows obtained with the different schemes, is very high. For FHCF and standard HCF schemes, the reason is that they both allocate TXOPs by excess to these flows. EDCF is also very fair between the audio flows since there are few collisions between them due to low audio sending rates and important periods of silence (OFF). For the VBR flows, EDCF is fairer than FHCF for the same reason than explained on Figure 11. Note that if we use VBR flows with a non-constant interarrival time, fairness between them would be improved.

### 6.2.2 CBR MPEG4 video flows

In our simulations, CBR flows are the most responsible for the network load. Figure 12 shows that the performance of EDCF in terms of delays are reduced if the load of the network increases while the VBR flows are not affected by the load. However, both VBR and CBR flows are mapped to TCs which have the same EDCF parameters. Consequently, CBR flows suffer from many collisions (11% of the simulation duration) which degrade the performance of EDCF. On the contrary, FHCF and standard HCF schemes seem not to be affected by the load. We can also observe that FHCF is still very fair between the different CBR flows on a large range of loads. Hence node schedulers succeed in redistributing time among the

different TSs up to a very high network load (96%). However, with FHCF, fairness decreases when the load reaches very high values (from 90%) since both schedulers are not able to absorb traffic fluctuations anymore.

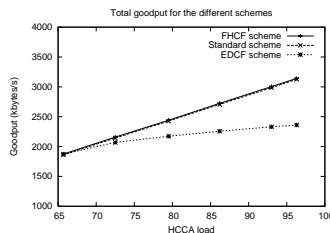


Figure 14: Total throughput

Figure 14 shows that the throughput increases linearly with standard HCF and FHCF schemes whereas EDCF seems to reach a bound due to collisions.

## 7 Conclusion

We have designed and evaluated a new MAC scheduling scheme for 802.11e HCF with the aim of supporting fluctuating rates and/or packet sizes while remaining fair for the different types of flows with QoS requirements. Even in high load (96%), the FHCF scheme is able to control delays and respect QoS requirements for the different flows.

FHCF is shown to be up to 34% more efficient than EDCF by removing collisions when the network is high-loaded. Moreover, to allocate TXOPs, FHCF uses the mean sending rate of VBR applications instead of the maximum sending rate usable for the standard scheme. By this way, with our simulations, FHCF allows to save between 14% and 37% of the time allocated to the VBR flows according to the standard HCF based on maximum sending rates. Consequently, more flows can be accepted in HCCA.

## References

- [1] IEEE Std 802.11-1999, Part 11: Wireless LAN MAC and Physical Layer specifications, Reference number ISO/IEC 8802-11:1999(E), IEEE Std 802.11, 1999.
- [2] IEEE 802.11e/D4.1, Draft Supplement to Part 11: Wireless MAC and Physical Layer specifications: MAC Enhancements for QoS, February 2003.
- [3] A. Grilo, M. Nunes, "Performance Evaluation of IEEE 802.11e", PIMRC 2002.
- [4] S. Mangold, et al., "IEEE 802.11e Wireless LAN for Quality of Service", European Wireless, February 2002.

- 
- [5] Q. Ni, L. Romdhani, T. Turetli and I. Aad, “QoS Issues and Enhancements for IEEE 802.11 Wireless LAN ”, INRIA Research Report, RR-4612, 2002.
- [6] “Video codec for audiovisual services at  $p \times 64$  kb/s”, *ITU-T Recommendation H.261*, 1993.
- [7] “MPEG4 Coding of audio visual objects: Visual”, *ISO/IEC JTC1/SC29/WG11*, March 1998.
- [8] S. McCanne and V. Jacobson, “vic: a flexible framework for packet video”, *ACM Multimedia*, 1995.
- [9] “HCF/EDCF NS-2 Stanford implementation”, see <http://nondot.org/radoshi/cs444n/>.
- [10] “The Network Simulator (ns2)”, see “<http://www.isi.edu/nsnam/ns>”.
- [11] FHCF implementation and simulations available from “<http://www-sop.inria.fr/planete/qni/fhcf/>”.
- [12] P.M. Soni and A. Chockalingam, “Performance Analysis of UDP With Energy Efficient Link Layer on Markov Fading Channels”, *IEEE Transactions on Wireless Communications*, Vol. 1, No. 4, October 2002
- [13] R. Jain, “The Art of Computer Systems Performance Analysis”, John Wiley & Sons, 1991.



---

Unité de recherche INRIA Sophia Antipolis  
2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes  
4, rue Jacques Monod - 91893 ORSAY Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique  
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

---

Éditeur  
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)  
<http://www.inria.fr>  
ISSN 0249-6399