FHCF: A Simple and Efficient Scheduling Scheme for IEEE 802.11e Wireless LAN

Pierre Ansel, Qiang Ni*and Thierry Turletti
Planète Project, INRIA Sophia Antipolis, FRANCE

Abstract

The IEEE 802.11e medium access control (MAC) layer protocol is an emerging standard to support quality of service (QoS) in 802.11 wireless networks. Some recent works show that the 802.11e hybrid coordination function (HCF) can improve significantly the QoS support in 802.11 networks. A simple HCF referenced scheduler has been proposed in the 802.11e which takes into account the QoS requirements of flows and allocates time to stations on the basis of the mean sending rate. As we show in this paper, this HCF referenced scheduling algorithm is only efficient and works well for flows with strict constant bit rate (CBR) characteristics. However, a lot of real-time applications, such as videoconferencing, have some variations in their packet sizes, sending rates or even have variable bit rate (VBR) characteristics. In this paper we propose FHCF, a simple and efficient scheduling algorithm for 802.11e that aims to be fair for both CBR and VBR flows. FHCF uses queue length estimations to tune its time allocation to mobile stations. We present analytical model evaluations and a set of simulations results, and provide performance comparisons with the 802.11e HCF referenced scheduler. Our performance study indicates that FHCF provides good fairness while supporting bandwidth and delay requirements for a large range of network loads.

Keywords: IEEE 802.11e, WLAN, medium access control (MAC), quality of service (QoS)

1 Introduction

IEEE 802.11 wireless LAN (WLAN) [1] has gained a great success for data applications in hotspots, enterprizes, university campuses, hospitals, etc. To share the wireless medium, the 802.11 standard defines two access methods at medium access control (MAC) layer: the mandatory contention-based distributed coordination function (DCF) and the optional point coordination function (PCF).

The explosive growth of multimedia applications in the recent years arose the requirement of Quality of Service (QoS) support such as guaranteed delay, jitter and bandwidth for these applications. However, the original IEEE 802.11 WLAN standard has been mainly designed for data applications and does not provide any QoS support for multimedia applications [2]. To enhance the QoS support of 802.11 WLAN, the IEEE 802.11 standard committee is working on a new standard, called 802.11e [3]. A new medium access method called Hybrid Coordination Function (HCF) has been proposed in the 802.11e draft, which combines a contention-based enhanced DCF access mechanism (called EDCA) and a controlled channel access mechanism (called HCCA) in a single function. Recent performance evaluations of 802.11e HCF [4] show that HCF is more flexible than DCF and PCF and that it can improve the QoS support in 802.11 WLAN. In order to meet the negotiated QoS requirements, the QoS-enhanced AP (QAP) needs to schedule efficiently downlink and uplink frame transmissions. As wireless channel is time-varying and since a lot of multimedia applications have variable bit rate (VBR) characteristics, designing a good HCF scheduling algorithm is a challenging topic. To the best of our knowledge, research issues of 802.11e HCF scheduling algorithm has not yet received much attention. Only a few papers [5, 6] have addressed the problem of 802.11e HCF scheduling algorithm.

*Qiang Ni, the corresponding author. He is now with the Hamilton Institute, National University of Ireland Maynooth (NUIM), Co. Kildare, Ireland. Tel: +353-17086463, Fax: +353-17086269, E-mail: Qiang.Ni@ieee.org.

In order to understand the impact of 802.11e HCF scheduling algorithm on the delay performance, we first derive a mathematical model in this paper, which shows the relationship between polling interval, queue length, and delays. Based on this analytical model, we propose a simple and efficient scheduling algorithm, FHCF, that aims to be fair for different kinds of multimedia flows and compatible with current IEEE 802.11e standard. The performance of the FHCF scheme is evaluated through computer simulations and compared with the performance of the IEEE 802.11e HCF scheme.

The rest of this paper is organized as follows: Section 2 introduces the basic 802.11e HCF scheduling algorithm. Section 3 establishes a mathematical relationship between delay, polling interval and queue length in the context of 802.11e HCF scheduling scheme. This relationship is the basis of the FHCF scheme whose principles are explained in Section 4. Section 5 details the FHCF implementation in the NS-2 simulator and Section 6 compares performance of the FHCF scheduling scheme with the standard HCF scheme. Finally, Section 7 concludes the paper.

2 The 802.11e HCF scheduling algorithm

A simple HCF scheduling algorithm is proposed as a reference design [3] to take into account QoS requirements of different types of traffic. We provide in this section a brief introduction of the 802.11e HCF referenced scheduler, for details about the 802.11e standard and its QoS enhancement mechanisms please refer to our survey paper [2]. In 802.11e HCF, each QoS-enhanced station (QSTA) that requires a strict QoS support is allowed to send QoS requirement packets to the QAP while QAP can allocate the corresponding channel time for different QSTAs according to the requests. Figure 1 shows an example of the new IEEE 802.11e beacon interval, which is composed of alternated modes of contention period (CP) and optional contention-free period (CFP). Contrary to the 802.11 PCF scheme, the 802.11e HCF scheme can operate during both CP and CFP. During the CP, the QAP can start several contention-free bursts, called Controlled Access Periods (CAPs) at any time to control the channel. An important new feature is the concept of transmission opportunity (TXOP), which refers to an instance during which a given QSTA has the right to send packets. Thus a QSTA can initiate multiple transmissions as long as its TXOP has not expired. The aim of introducing TXOP is to limit the time interval during which a QSTA is allowed to transmit frames. Each QSTA can have up to 8 different priority traffic streams (TSs). Basically, each TS first sends a QoS request frame to the QAP containing the mean data rate of the corresponding application, the MAC Service Data Unit (MSDU) size and the maximum required service interval (RSI). Using these QoS requests, the QAP determines first the minimum value of all the RSIs required by the different traffic which apply for HCF scheduling. Then it chooses the highest submultiple value of the 802.11e beacon interval duration (duration between two beacons) as the selected service interval (SI), which is less than the minimum of all the maximum RSIs. Thus, an 802.11e beacon interval is cut into several SIs and QSTAs are polled accordingly during each selected SI. The selected SI refers to the time between the start of successive TXOPs allocated to a QSTA, which is the same for all the QSTAs.

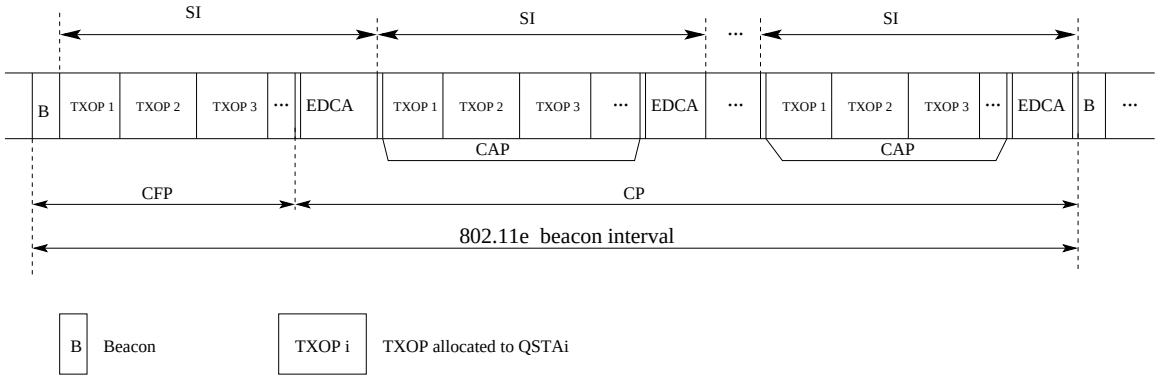

Figure 1: Structure of the 802.11e beacon interval used in the HCF scheduling algorithm [3]

As soon as the selected SI is determined, the QAP evaluates all the TXOPs allocated to the different TSs of the QSTAs which apply for HCCA. The TXOP should correspond to the duration required to transmit all the packets arriving during a SI in a TS queue. Let N_i be the number of packets arriving in the TS queue i for a QSTA during a SI:

$$N_i = \lceil \frac{\rho_i SI}{M_i} \rceil, \tag{1}$$

where ρ_i is the application data rate and M_i the MSDU size for TS queue i. Then the different TXOPs T_i are computed as follows:

$$T_i = N_i \cdot (\frac{M_i}{R} + 2SIFS + ACK), \tag{2}$$

where R is the physical (PHY) layer transmission rate, and ACK is the duration to transmit an acknowledgement packet. This simple HCF scheduling algorithm can be efficient if traffic is strictly CBR. However, when real-time applications such as videoconferencing generate VBR traffic, this scheme may cause the average queue length to increase and possibly packets to be dropped. Even if the mean sending rate of the application is lower than the rate specified in its QoS requirements; peaks of sending rate may not be absorbed by TXOPs allocated according to the QoS requirements. A more flexible scheme that adapts to fluctuating rates is then necessary. This motivated the design of our FHCF scheduling scheme, which is based on the mathematical relationship between parameters and corresponding observations that we obtain in the following section.

3 Relationship between queue length, polling interval, and delays

Let us consider several simplifications to establish an analytical relationship between SI and delay parameters: First, we denote by R_{eff} the effective data throughput corresponding to the actual amount of data packets transmitted per time unit. Note that the PHY layer of 802.11 wireless networks can adapt its transmission mode according to varying conditions of the PHY channel. Variation of PHY layer data rates can be achieved by choosing different modulation and coding schemes. In this work, we do not consider such rate adaptation options in 802.11 [1]. R_{eff} is computed according to the maximum PHY data rate but is less than the maximum PHY data rate because of PHY and MAC layers' overheads. Second, we suppose that data packets are transmitted continuously (without taking packet fragmentation into account). For the delay analysis, we consider the following two cases: 1) The queue is empty at the end of TXOP allocation. This is the ideal case assumption for the IEEE 802.11e HCF scheme, which holds only for CBR-like traffic types; 2) The queue is not empty at the end of TXOP allocation, which is more realistic than the ideal case in real wireless networks.

3.1 Empty queues at the end of the TXOP

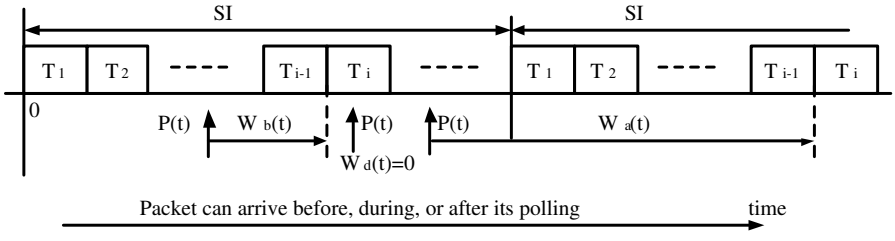

Figure 2: Timing relationships for delay analysis in ideal case (queueing delays not shown)

In this case, the queue of a polled QSTA is assumed to be zero at the end of its TXOP allocation, which is true if traffic is a CBR one. Under such an assumption, all the packets arriving in a SI interval will be serviced either in the same SI or in the next SI, and no later than the next SI. As shown in Figure 2, in

order to predict the delay for a data packet P that arrives in the TS queue with polling order i at time t [1], two different cases should be considered: the packet P can be transmitted in the following TXOP allocation (T_i) of that TS queue, or the packet has to be queued until another TXOP allocation in the next SI if there are some other packets before P in the $TS-i$ queue. In this paper, all 802.11e schedulers do not actively drop a data packet if it is too old in the queue (i.e. the packet delay is larger than the delay bound of that traffic application). Hence, the delay for the data packet can be calculated as the sum of the packet queueing delays (Q) for sending out all the queued packets before P in this SI interval (via the allocated TXOP T_i), and the waiting time delays (W) which is equal to the duration between the packet arrival time and the time the queue is polled. Notice that in current 802.11e MAC scheduling framework, each TS queue can only be polled once each SI interval. Those packets arriving after the polling may not be transmitted in this SI interval and thus has to wait for the next polling interval. Considering that the delay expressions are different if data packets arrive *before*, *during*, or *after* the QSTA is polled by the QAP. We denote respectively by $Q_b(t)$, $Q_d(t)$, and $Q_a(t)$ the packet queueing delays if the packet P arrives before, during and after its polling at time t. Similarly, we denote respectively by W_b, W_d, and W_a the packet waiting time delays while the packet arrives before, during and after its polling at time t. Finally, the delay for the packet P can be expressed as:

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

with:

$$W_b(t)=\sum_{j=1}^{i-1}T_j-t\ , \qquad Q_b(t)=\frac{q_i^0M_i}{R_{eff}}+\frac{\rho_i t}{R_{eff}} \tag{4}$$

$$W_d(t)=0\ , \qquad Q_d(t)=\frac{q_i^0M_i}{R_{eff}}+\frac{\rho_i t}{R_{eff}}-(t-\sum_{j=1}^{i-1}T_j) \tag{5}$$

$$W_a(t)=SI-t+\sum_{j=1}^{i-1}T_j\ , \qquad Q_a(t)=\frac{\rho_i}{R_{eff}}(t-\sum_{j=1}^{i}T_j) \tag{6}$$

where q_i^0 represents the initial number of data packets in the $TS-i$ queue at the beginning of the SI, and we denote

$$\chi_{[i,j]}(t)=\begin{cases} 1 \text{ if } t\in[i,j] \\ 0 \text{ else} \end{cases},$$

$$\chi_{(i,j]}(t)=\begin{cases} 1 \text{ if } t\in(i,j] \\ 0 \text{ else} \end{cases}.$$

By using Equations (4)-(6), Equation (3) can be rewritten as:

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

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

[1] The initial time (time zero) is set at the beginning of the SI.

3.2 Non-empty queues at the end of the TXOP

When traffic is a VBR one, the queue at the end of the TXOP may not be empty as assumed in Section 3.1. In this case, we have to consider q_i^e, the non-zero queue length of the TS at the end of the TXOP T_i.

Further, different from the case in Section 3.1, packets can be queued later than the next SI interval since all the queues can be nonzero and delays are accumulative. Hence, delay can be discontinuous even during the TXOP duration in this case and it can be expressed as:

$$d_i(t) = (kSI + \sum_{j=1}^{i-1} T_j - t + \frac{q_i^0 M_i}{R_{eff}} + \frac{\rho_i t}{R_{eff}})\chi_{[0,t^d]}(t)$$
$$+ ((k+1)SI - t + \sum_{j=1}^{i-1} T_j + \frac{\rho_i}{R_{eff}}(t - t^d))\chi_{(t^d,SI]}(t),$$

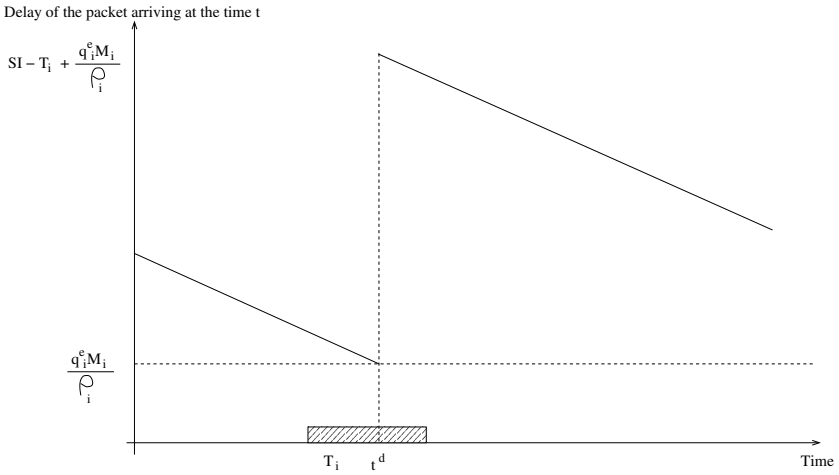

Figure 3: Theoretical delay versus time for the best case

where the value of k depends on *which SI interval* the data packet can be serviced. In the *best case* if the packet can be transmitted no later than the next SI interval, $k = 0$, see Figure 3. In other cases, we have $k > 0$. t^d denotes the time when delay discontinuity happens, and it corresponds to the arriving time of the first packet which can not be transmitted during the current TXOP duration:

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

Suppose that the best case is satisfied ($k = 0$), we can obtain the maximum delay, which is denoted by D_i and is achieved at the time t^d as follows:

$$D_i = \max_t d_i(t) = SI - T_i + \frac{q_i^e M_i}{\rho_i}. \quad (7)$$

Equation (7) shows that the delay is higher than the ideal one in Section 3.1 if some packets remain in the queue at the end of the poll. In Section 3.1, queue is assumed empty at the end of the poll, and thus D_i is almost equal to the SI duration since $T_i << SI$ which means packet delays are bounded by the SI in the ideal case. However, in other cases than ideal case and best case, the 802.11e HCF scheduler will not work well and packet delays will completely uncontrolled if traffic is highly variable. We will validate this observation through computer simulations in Section 6.

Moreover, we can express T_i in order to link the queue size at the beginning of the polling q_i^b with the queue size at the end of the polling q_i^e. Indeed, if $\frac{q_i^b M_i}{R_{eff}}$ represents the time to send the q_i^b packets in the

queue before polling, $\frac{\rho_i}{R_{eff}}T_i$ the time to send the packets arrived in the queue during the TXOP, and $\frac{q_i^e M_i}{R_{eff}}$ the time to send the packets still in the queue at the end of the TXOP, we have the following relation

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

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 (7), the maximum delay in the best case can be expressed as:

$$\begin{aligned} D_i &= SI + \frac{q_i^b M_i}{\rho_i} - \frac{R_{eff}}{\rho_i}T_i \\ &= SI(1 + \sum_{j=1}^{i-1}\frac{T_j}{SI} - \frac{R_{eff}}{\rho_i}\frac{T_i}{SI}) + \frac{q_i^0 M_i}{\rho_i}. \end{aligned} \quad (8)$$

(8) shows that we can control the maximum delay D_i by two different ways: On the one hand, the QAP can reduce the delay by increasing the value of T_i which is allocated to the i-th TS , since the number of packets remaining in the queue at the end of the TXOP (T_i) decreases when the allocated TXOP increases. From (1) and (2), we can note that T_i/SI is independent from the SI duration, so we are able to reduce the maximum delay by reducing the SI duration. However, this increases the number of polls and overheads increase at the same time. On the other hand, we can control the maximum delay if we are able to control the queue length before polling q_i^b, whose value can be measured by the data rate of the TS. When the flow is not a constant bit rate traffic, q_i^b may vary considerably. Actually, this is the main motivation of our FHCF scheme. One advantage of our FHCF scheme is that fairness for flows with the same priority can be achieved without any additional cost because they obtain the same maximum delay which is linked to the selected SI duration.

4 The FHCF Scheme

Basically, the FHCF scheme is composed of two schedulers: the QAP scheduler and the node scheduler. The QAP scheduler estimates the varying queue length for each QSTA before the next SI and compares this value with the ideal queue length, see Figure 4. The QAP scheduler uses a window of previous estimation errors for each TS in each QSTA to adapt the computation of the TXOP allocated to that QSTA. Then, the node scheduler located in each QSTA can redistribute the unused time among its different TSs because the TXOP is not allocated to a particular flow but all flows of a QSTA.

4.1 QAP scheduler

First, the QAP scheduler has to compute the ideal queue length of the TS queue i for each QSTA at the beginning of the next SI:

$$q_i^{ideal} = \frac{\rho_i \cdot (SI - \sum_{j=1}^{i} N_j \cdot (\frac{M_j}{R_{eff}} + 2SIFS + ACK))}{M_i}. \quad (9)$$

In this paper, the ideal queue length refers to the queue size at the beginning of the next SI which was zero at the end of the current TXOP. The ideal queue length evolution assumption is used by the IEEE 802.11e HCF referenced scheduling scheme [3], which is valid only when the sending rate of the application is strictly CBR.

Second, when a QSTA sends a QoS data packet, the QAP uses the QoS control field of the IEEE 802.11e header to record its queue length q_i^e at the end of TXOP. Let t_i^e be the corresponding time at the end of the current TXOP. Note that t_i^e is also recorded by the QAP scheduler. Using these information, the QAP scheduler is able to estimate q_i^{est}, the queue length of the i-th TS at the beginning of the next SI as follows:

$$q_i^{est} = \frac{\rho_i(SI - t_i^e)}{M_i} + q_i^e. \quad (10)$$

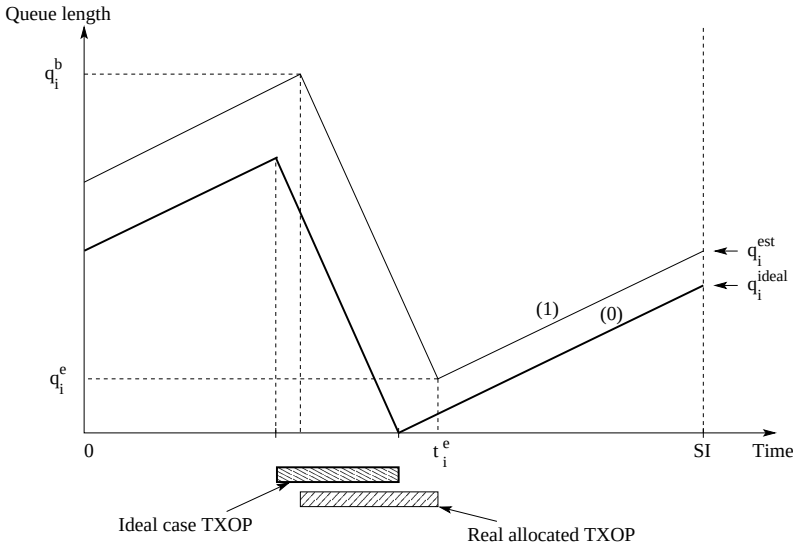

Figure 4: Queue length evolution for a TS: (0). Ideal queue length case; (1). Estimated queue length evolution

Since the sending rate and packet size of the application vary, the above simple method for queue length estimation is not always accurate. To solve this problem, the FHCF scheme proposes to use a window of w already known real queue length measurements (or called history information) to tune the estimation. It means at the n-th SI, the QAP will have at its disposal the w last estimation errors

$$\begin{aligned}
\Delta_i^{n-1} &= q_i^{b,real}(n-1) - q_i^{b,est}(n-1) \\
\Delta_i^{n-2} &= q_i^{b,real}(n-2) - q_i^{b,est}(n-2) \\
&... \\
\Delta_i^{n-w} &= q_i^{b,real}(n-w) - q_i^{b,est}(n-w).
\end{aligned}$$

Then, just before the n-th SI, the QAP estimates $E[|\Delta_i(n)|]$, the expected value of the absolute value of the difference between $q_i^{b,real}(n)$ (the real queue length at the beginning of TXOP of the i-th TS) and the estimation of this queue length, $q_i^{b,est}(n)$ as follows:

$$E[|\Delta_i(n)|] \simeq \frac{\sum_{j=n-w}^{n-1} |\Delta_i(n)|}{w}. \tag{11}$$

In the Appendix, we show that this corrective term is close to the standard deviation of a traffic flow if we suppose that the sending rate of that flow follows a Gaussian distribution. Moreover, from a hardware point of view, it is easier to compute the absolute value of the difference than a estimation of the standard deviation.

Thus, by using Equation (11), the QAP scheduler is able to improve its estimation of the queue length at the beginning of the next polling of the TS queue as follows

$$q_{i,new}^{b,est}(n) = q_i^{b,est}(n) + E[|\Delta_i(n)|].$$

Third, the QAP compares the real queue length estimation to the ideal queue length[2] at the beginning of the next SI. It computes the number of additional packets, DN_i^{est}, which is the difference between the

[2]The ideal queue length was zero at the end of the current TXOP allocation.

estimated queue length and the ideal case:

$$DN_i^{est} = q_{i,new}^{b,est}(n) - q_i^{b,ideal}(n) = q_i^{est}(n) - q_i^{ideal}(n) + E[|\Delta_i(n)|]$$

where $q_i^{est}(n)$ and $q_i^{ideal}(n)$ are given by (10) and (9). Then, the QAP computes the additional required time t_i^{est} (which may be positive or negative) for each TS of each QSTA and reallocates the corresponding TXOP duration according to the estimation of DN_i^{est}:

$$t_i^{est} = DN_i^{est} \cdot (\frac{M_i}{R_{eff}} + 2SIFS + ACK). \tag{12}$$

Then it computes T_P, the sum of all the positive values, T_N, the absolute value of the sum of all the negative values, and T^r, the remaining time of HCCA duration after allocating all the TXOPs computed in one SI using the ideal case. 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 that the additional time t_i^{est} have to be reduced. In order to be fair for all the flows, the scheduler reduces each positive additional time with a percentage β (chosen negative to correspond to a reduction) of t_i^{est}. On the other hand, each negative additional time is increased by the same percentage β of t_i^{est}, where β is expressed as:

$$\beta = -\frac{(T_P - T_N) - T^r}{T_P + T_N}. \tag{13}$$

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

$$t_i^{add} = \begin{cases} (1+\beta)t_i^{est} & \text{if} \quad t_i^{est} \geq 0 \\ (1-\beta)t_i^{est} & \text{if} \quad t_i^{est} < 0. \end{cases} \tag{14}$$

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

Note that our estimator in the *second step* is quite generic and works well for most kinds of traffic types. The key idea of our estimator and corresponding FHCF scheduler is, in order to adapt to the fluctuations of sending rates of different QSTAs, the QAP scheduler tries to allocate the service rate for different QSTAs which is equal to the mean arrival rate plus the expected value of the absolute deviation. More precisely speaking, we tune the TXOP allocation according to the mean sending rate plus this deviation, and then traffic spikes can be absorbed. From Chebyshev's Inequality theorem [13], we know that this estimator represents the average difference from the mean value, and is a measure of how spread out the random variables are. Although the standard deviation estimator could be another choice, it introduces higher computational complexities, and thus we decided to choose this simple estimator, which also provides reasonable accuracy.

How to choose the value of w is a tradeoff between complexity and accuracy: with a larger value of w, a more accurate queue length estimation is normally obtained, but with an increasing complexity. In our work, the window size w has been empirically set to 5 through simulation results.

4.2 Node scheduler

The node scheduler also plays a very important role since it has to redistribute the additional allocated time to the different TSs within a node. It performs almost the same computations as the QAP. Suppose that the number of active TS of a given polled QSTA is p ($1 \leq p \leq 8$). First the node scheduler in this QSTA computes N_i, the number of packets to transmit in the i-th 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 that can be reallocated:

$$T^r = T - \sum_{i=1}^{p} N_i \cdot (\frac{M_i}{R_{eff}} + 2SIFS + ACK).$$

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. Using this information, the node scheduler performs the same computations as the QAP scheduler (see (12), (13) and (14)). The difference is that the coefficient β may be positive if the QSTA has more time than required to send all its packets or may be 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 β is positive.

5 FHCF Implementation in NS-2 Simulator

5.1 Basis of the FHCF scheme

Our NS implementation of 802.11e HCF/FHCF/EDCA [9] is based on Stanford University's early version of NS-2 codes of EDCA/HCF [6]. We have added numerous new features according to the latest IEEE 802.11e standard draft [3]. In Stanford's original NS implementation, 8 kinds of different traffic classes (TC) of EDCA can be both used in HCCA and EDCA mode. However, according to the new 802.11e standard [3], TC queues for EDCA and TS queues for HCCA should be separated in order to prevent packets from EDCA queues to be transmitted through TS queues of HCCA and thus guarantee the delay performance in HCF controlled channel access. In our implementation, if a traffic with priority i is accepted in HCCA, it becomes a Traffic Stream (TS) and all the packets for this class can only be transmitted in HCCA mode. In this way, it is possible to evaluate the performance of the different schedulers only using the HCCA mode. We also implemented the part of queue length estimation in HCCA since Stanford's original implementation is based on the use of old queue length information without any estimations.

Moreover, in their implementation beacon frames could be delayed since the medium was sometimes used at the time the beacon frame was scheduled. We have fixed the problem of beacon frame delay in the simulator 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.

5.2 TSPEC negotiation

As mentioned before, the purpose of the FHCF scheme 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 only use the HCCA mode, not the EDCA mode.

According to the 802.11e standard [3], 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 EDCA 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. Then, 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.

6 Simulation Experiments

In order to evaluate the performance of the QAP scheduler and the node scheduler, two kinds of simulation topologies are used. The first one contains 18 mobile QSTAs and 1 QAP with only one TS per QSTA (see Section 6.1), which is designed to evaluate the performance of the QAP scheduler. The second topology is composed of 6 QSTAs and 1 QAP (see Section 6.2), each one with three different priority TSs to evaluate the performance of the node scheduler. For all the simulations, the destination of all the flows is the QAP (which is node 0 in our case): This allows us to compare fairly end-to-end delays among the different flows.

6.1 Scenario 1

In the first Scenario, six QSTAs send a high priority on-off audio traffic ($64kb/s$), six others QSTAs send a VBR video traffic ($200kb/s$ of average sending rate) with medium priority and the remaining six QSTAs send a CBR MPEG4 [10] video traffic ($3.2Mb/s$) with low priority. Table 1 summarizes the different traffics used for this simulation. We model the audio flow by on-off sources with parameters corresponding to a typical phone conversation [11]. The transport protocol is UDP. Audio flows are mapped to the 6^{th} TS of the MAC layer whereas VBR H.261 and CBR MPEG4 video flows are respectively mapped to the 5^{th} and 4^{th} TS. The different VBR flows have been obtained with the VIC [7] videoconferencing tool using the H.261 coding and QCIF format for typical "head and shoulder" video sequences. We made 6 trace files[3]: 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 sending rate distribution follows a Gaussian law and its mean value 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 $1024bytes$[4]. The RSI request of audio traffic is set to $50ms$ and the RSIs of both VBR and CBR video traffic types are set to $100ms$ and beacon interval is $500ms$. According to the algorithm in 802.11e, the selected SI will be $50ms$. The PHY and MAC layer parameters used in the simulation are summarized in Table 2.

Table 1: Description of the different traffic streams

Node	Application	Arrival period (ms)	Packet size ($bytes$)	Sending rate (kb/s)
$1 \rightarrow 6$	Audio	4.7	160	64
$7 \rightarrow 12$	VBR video	$\simeq 26$	$\simeq 660$	$\simeq 200$
$13 \rightarrow 18$	MPEG4 video	2	800	3200

Table 2: PHY and MAC layer parameters

SIFS	$16\mu s$
DIFS	$34\mu s$
ACK size	$14bytes$
PHY rate	$36Mb/s$
Minimum PHY rate	$6Mb/s$
Slot time	$9\mu s$
CCA time	$4\mu s$
MAC header	$38bytes$
PLCP header length	$4bits$
Preamble length	$20bits$

6.1.1 Comparison of throughput

Throughput curves on Figure 5 show that both FHCF and the 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 HCF provide a constant throughput. As regards audio flows, the throughput is sometimes equal to zero since it matches burst periods and idle periods.

[3]The video trace files are available from [9].
[4]By default, VIC uses a MTU size of $1024bytes$.

6.1.2 Comparison of the number of dropped packets

Table 3 and Figure 6 show that with the standard HCF scheme, no CBR MPEG4 packets are lost[5]. However, 513 VBR packets are lost with the HCF scheme. Please notice that in this work, packets are considered as lost only when reaching the maximum queue length limit and hence dropped by the queue buffers. They are never dropped because the delays are larger than the applications' delay bounds. The queue overflow can occur in the following two cases: First, the peak sending rates of some VBR flows may be much higher than the mean sending rate specified in their requirements, and second, the sending rate is fluctuating heavily during time. This confirms, as explained in Section 2, that HCF has not been designed for this kind of flows but rather for CBR traffic type. 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.

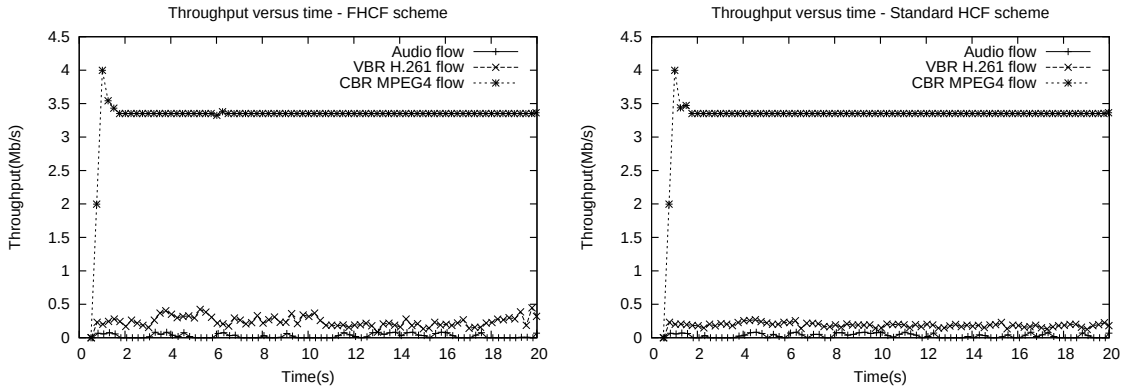

Figure 5: Throughput versus time for FHCF and HCF

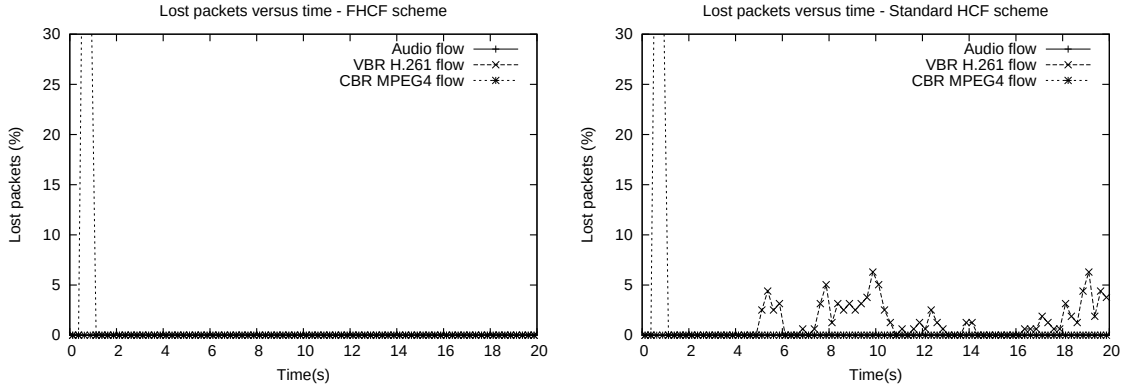

Figure 6: Dropped packets versus time for FHCF and HCF

Table 3: Number of drops for each kind of flow with FHCF and HCF

	Audio	VBR video	CBR MPEG4 video
FHCF	0	0	204
Standard HCF	0	513	204

[5]Some packet drops occur only at the beginning of the simulation because all the flows start at time 0 and the network is congested at the starting time. However, it does not depend on the scheduling/queueing policy and thus we did not consider them as losses.

6.1.3 Analysis of latency distributions

Figure 7 shows that with the FHCF scheme, all the flows have a maximum latency which are equal to the selected SI of the flows (chosen equal to $50ms$), whereas with the HCF scheme, the packet delays of VBR flows are completely uncontrolled. It is because the allocated TXOPs according to the mean rate are too small and thus the delays are very high. 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 sometimes two packets arrive at the same time[6] and the interarrival time is then higher than the mean value. Because packets arrive regularly and the interarrival time is not changing during simulation time, delays of packets are not regularly distributed between 0 and $50ms$.

Figure 8 represents the mean latency for each QSTA (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 3) 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:

$$\frac{1}{2}(SI - T_i + 2\frac{q_i^e M_i}{R_{eff}}), \tag{15}$$

is equal to $25ms$ ($T_i << SI$).

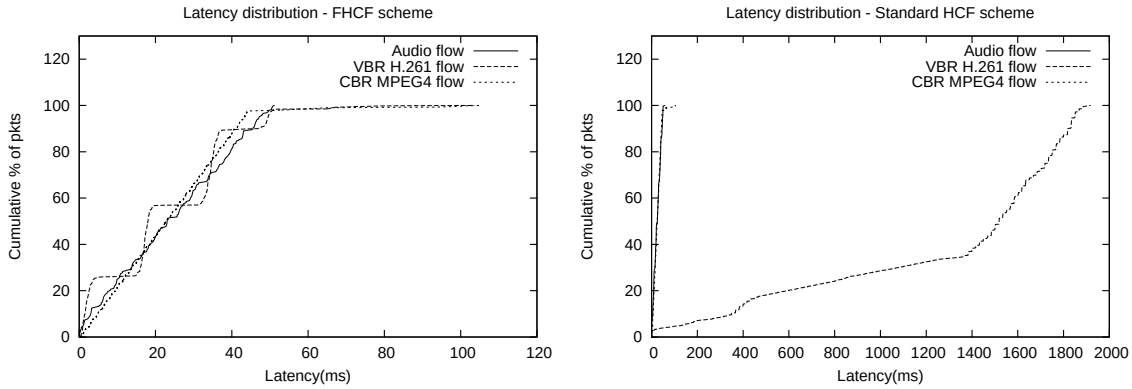

Figure 7: Latency distribution for FHCF and HCF

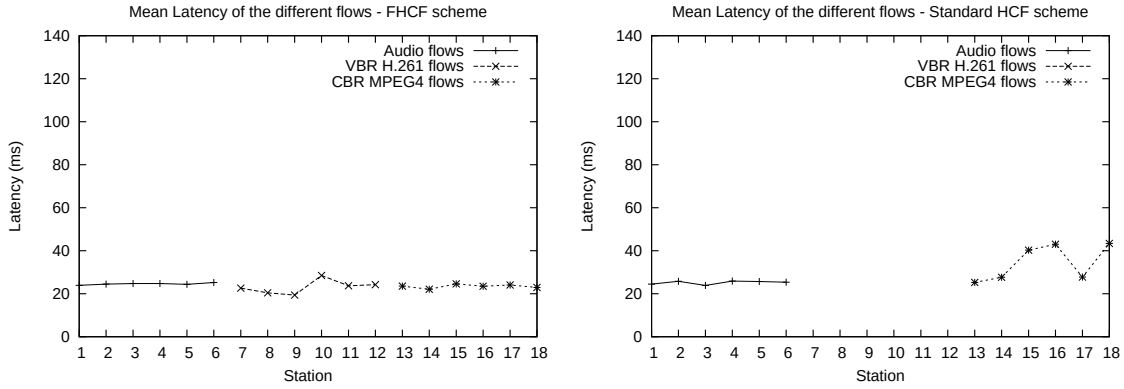

Figure 8: Mean latency for each flow with FHCF and HCF

[6]This difference is due to VIC fragmentation.

As regards the standard HCF scheme, the delays of the VBR flow are completely uncontrolled (see Figure 7) because the queue lengths are increasing during time. In this case, TXOPs are not long enough to absorb remaining bursts and the queue sizes are endless filling up because queues with large capacities are used. Note that the standard HCF scheme may be efficient if TXOPs are allocated 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.

6.2 Scenario 2

In Scenario 2 (see Table 4), each QSTA sends three audio, VBR H.261 and CBR MPEG4 video flows simultaneously through three different MAC-layer priority classes. This topology aims at evaluating the behaviors of the different TSs in the same QSTA and with the same priority TS in different QSTAs. The HCCA load has been changed by increasing the packet size of the CBR MPEG 4 traffic from $600bytes$ ($2.4Mb/s$) to $1000bytes$ ($4Mb/s$) using a $100bytes$ increment and keeping the same inter-arrival period of $2ms$. This is a time-consuming way to increase load because CBR video packets need more time to be transmitted, while another way to increase the network load is to increase the number of QSTAs. We chose the first way to increase the traffic load.

To compare fairness in terms of delay between the same kinds of traffics for the different schemes, we use Jain's fairness index [12]:

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

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.

Figures 9 and 10 show respectively the mean delays and the fairness of several types of flows obtained with the various schemes for different loads of the network (see Table 4).

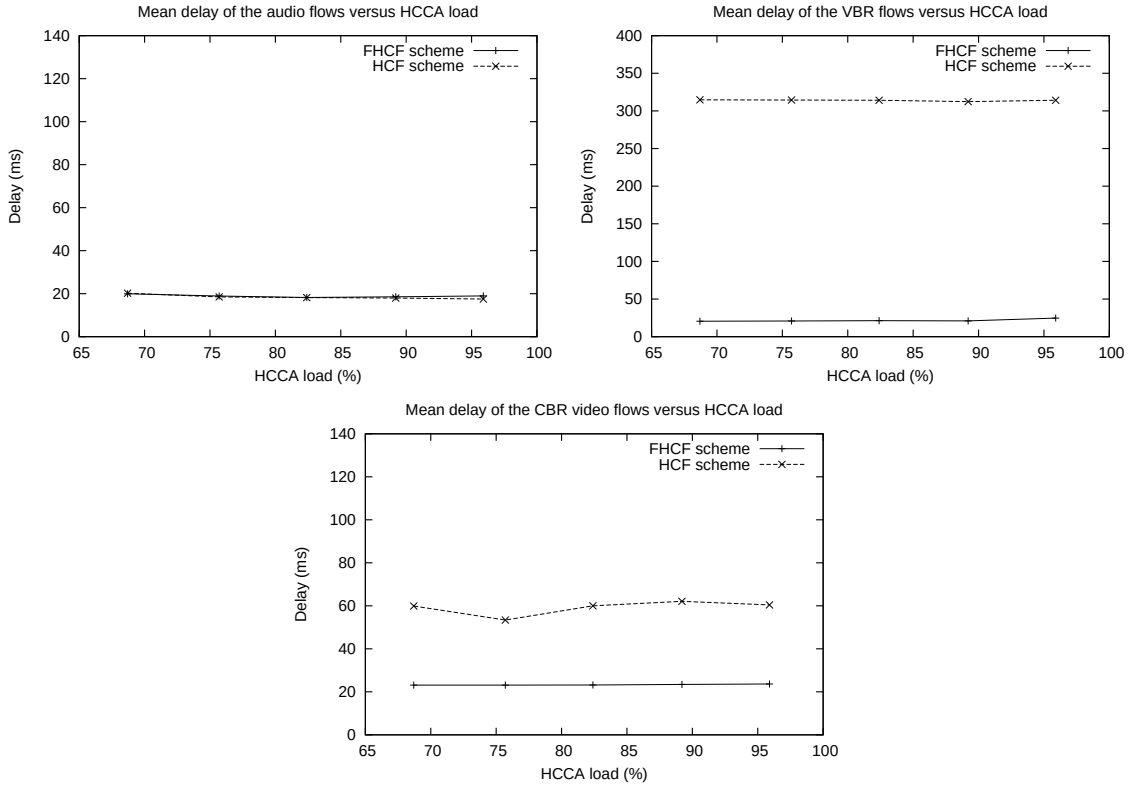

Figure 9: Mean delays versus load

Table 4: Description of different traffic streams

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	$600 \rightarrow 1000$	$2400 \rightarrow 4000$

6.2.1 Audio and VBR H.261 video flows

Figure 9 shows that with FHCF, delay curves are almost horizontal lines which means that delays do not strongly depend on the network load. For the same reason as in Scenario 1, the delays of VBR flows with the standard HCF scheme are very high (the mean delays for the VBR flows are almost $300ms$).

As shown in Figure 10, Jain's fairness index between audio flows obtained with the HCF scheme and the FHCF scheme, is very high. The reason is that they both allocate TXOPs by excess to these audio flows. Concerning the VBR flows, FHCF is always fairer than HCF.

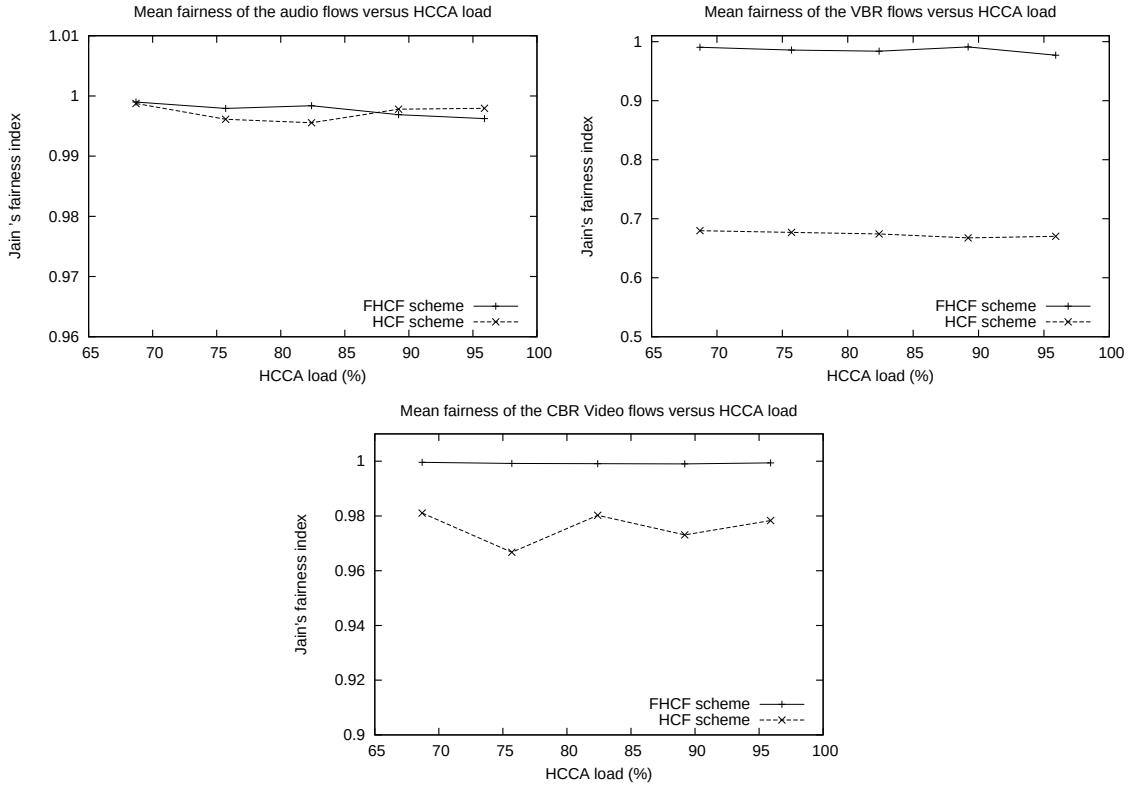

Figure 10: Fairness versus load

6.2.2 CBR MPEG4 video flows

In our simulations, CBR flows are the most responsible for the network load. Figure 9 shows that the mean delays of both FHCF and standard HCF schemes are not affected by the traffic load, while the delay of FHCF is smaller than that of HCF. As seen in Figure 10, we observe that FHCF is fair between the different

CBR flows on a large range of loads since node schedulers succeed in redistributing time among the different TSs up to a very high network load (96%). On the other hand, with HCF fairness performance is poor because both schedulers are not able to absorb traffic fluctuations.

Figure 11 shows that the total throughput increases linearly both with standard HCF and FHCF schemes, while the total throughput is almost the same for the two schemes.

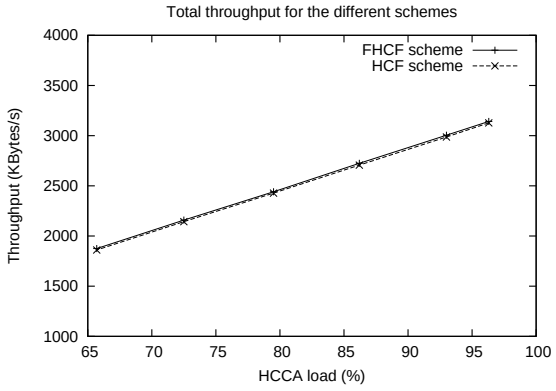

Figure 11: Total throughput

7 Conclusion

In this paper, we have derived an analytical model which explains the relationship between the 802.11e HCF polling interval, queue length, and delays. Based on it, we have proposed the FHCF scheduling scheme for the upcoming 802.11e MAC layer standard, with the aim of supporting fluctuating rates and/or packet sizes with QoS requirements. To allocate TXOPs, FHCF uses the mean sending rate of VBR applications plus estimated absolute deviations instead of the maximum sending rate usable for the standard HCF scheme. In this way, FHCF provides better QoS performance if a mean sending rate is used by the HCF scheduler, or it saves around 14% to 37% of the time allocated to the VBR flows if a maximum sending rate is used by the scheduler. Consequently, more traffic flows can be transmitted with good quality in HCCA. Moreover, the FHCF scheme is shown to achieve a higher degree of fairness among different multimedia flows than the 802.11e HCF scheme. Future work includes the design of adaptive and robust scheduling algorithms for error-prone IEEE 802.11e wireless channels and robust admission control mechanisms for 802.11e wireless networks.

Acknowledgements

The authors would like to thank Dr. Chadi Barakat (INRIA), Prof. Sunghyun Choi (Seoul National University, Korea), Dr. Matthew Sherman (AT&T Shannon Laboratory, USA), the guest editors, and other anonymous reviewers for providing valuable comments to improve the quality of the paper. This work has been supported by the French ministry of industry in the context of the national project RNRT-VTHD++.

References

[1] IEEE 802.11 WG: IEEE Std 802.11-1999, Part 11: Wireless LAN MAC and physical layer specifications. Reference number ISO/IEC 8802-11:1999 (E), (1999).

[2] Ni Q., Romdhani L., and Turletti T.: A survey of QoS enhancements for IEEE 802.11 wireless LAN. Wiley Journal of Wireless Communications and Mobile Computing (JWCMC), John Wiley & Sons Publisher, **4 (5)** (2004) 547–566.

[3] IEEE 802.11 WG: IEEE 802.11e/D4.1, Wireless MAC and physical layer specifications: MAC enhancements for QoS. (February 2003).

[4] Mangold S., Choi S., May P., et al.: IEEE 802.11e wireless LAN for Quality of Service. Proceeding of European Wireless, **1** (2002) 32–39.

[5] Grilo A., Macedo M., and Nunes M: A scheduling algorithm for QoS support in IEEE 802.11e networks. IEEE Communication Magazine, **10** (2003) 36–43.

[6] Garg P., Doshi R., Greene R., et al.: Using IEEE 802.11e MAC for QoS over wireless. IEEE IPCCC, (2003).

[7] McCanne S., Jacobson V.: VIC: a flexible framework for packet video. ACM Multimedia, (1995).

[8] ITU-T Recommendation H.261: Video codec for audiovisual services at $p \times 64$ kb/s. (1993).

[9] Ansel P., Ni Q., and Turletti T.: FHCF: A fair scheduling scheme for 802.11e WLAN", INRIA Research Report No 4883, July 2003. Implementation and NS simulation codes available from *"http://www-sop.inria.fr/planete/qni/fhcf/"*.

[10] ISO/IEC JTC1/SC29/WG11: MPEG4 coding of audio visual objects: visual. (1998).

[11] Soni P. M., Chockalingam A.: Performance analysis of UDP with energy efficient link layer on Markov fading channels. IEEE Transactions on Wireless Communications, **1** (2002).

[12] Jain R.: The art of computer systems performance analysis. John Wiley & Sons publisher, (1991).

[13] Jacod J., Protter P.: Probability essentials. Springer publisher, (2003).

Appendix

The key idea of our estimator (see Equation 11) is that, in order to allow the QSTAs to adapt to the fluctuations of sending rates, the QAP scheduler tries to estimate the standard deviation by using the expected value of estimation errors.

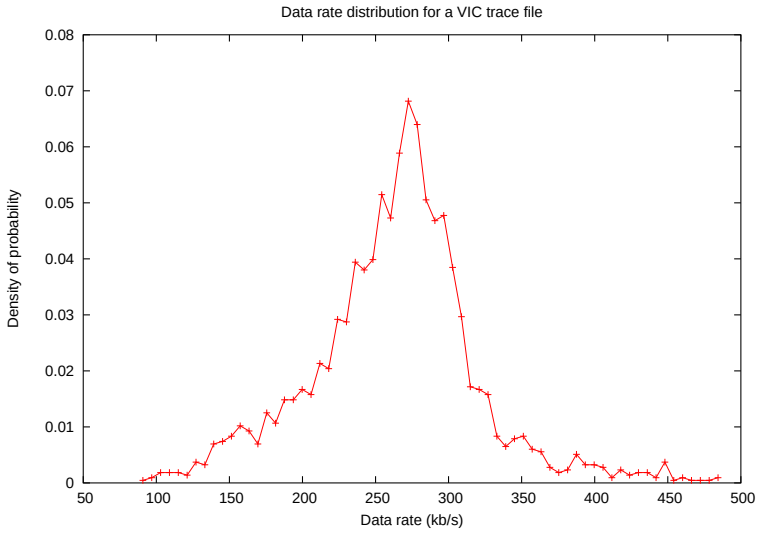

Figure 12: Distribution of the sending rate of a real VBR video traffic

Figure 12 shows that for a typical videoconferencing VBR video traffic, the sending rate almost follows a Gaussian distribution and packets have a fluctuating size. Thus, $\Delta_i(n)$ also follows the same Gaussian distribution with an expected value of 0. Then, the probability for Δ_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 δ_i the difference between the real amount of data and the estimated amount of data. Like Δ_i, δ_i also follows a Gaussian law. The density of probability for δ_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 additionnal 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:

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

Thus this corrective term is close to the standard deviation in the context of a Gaussian distribution of the sending rates($\sqrt{\frac{2}{\pi}}$ times the standard deviation). While the standard deviation estimation might provide tighter envelop estimates, it is more complicated and introduces higher computation complexity, and thus we decide to choose this simple estimator (Equation 11), which also provides reasonable accuracy.

However, this estimator is quite generic and works well for most kinds of traffic types.

Pierre Ansel received a multidisciplinary in-depth scientific training in different fields such as Pure and Applied Mathematics, Physics, Mechanics, Computer Science and Economics from the Ecole Polytechnique, Palaiseau, France. Then, he joined the Ecole Nationale Superieure des Telecommunications, Paris, France in 2005 where he went further into electronics, databases, computer network security and high speed networks. He received a multidisciplinary master of sciences degree and an additional master of sciences degree in telecommunications in 2005. He did a summer internship in 2003 in INRIA, Sophia Antipolis, France where he worked on the Quality of Service in 802.11 networks at Planete Group, France. Then in 2004, he joined France Telecom R&D, Issy-les-Moulineaux, France to work on Intranet Security issues. He designed a WiFi security supervision architecture based on WiFi Intrusion Detection Sensors. He is currently a French civil servant and belongs to the French Telecommunications Corps..

E-mail: pierre.ansel@polytechnique.org

Qiang Ni received the BEng., M.S. and Ph.D. degrees from Hua Zhong University of Science and Technology (HUST), Wuhan City, China in 1993, 1996 and 1999 respectively. From 1999 to 2001, he was a post-doctoral research fellow in the multimedia and wireless communication laboratory, HUST, China. He visited the wireless and networking group of Microsoft Research Asia Lab during the year of 2000. In 2001, he joined INRIA, France, where he was a research staff member at the Plante group. He is currently a Senior Researcher at the Hamilton Institute, National University of Ireland, Maynooth. Since 2002, he has been active as a voting member for the IEEE 802.11 wireless LAN standard working group. He serves as Technical Program Committee (TPC) member/session chair for many wireless, communications and networking conferences such as the IEEE GlobeCom 2005, WirelessCom 2005, IEEE Sarnoff Symposium 2005, ICC 2004, WiOPT 2005/04, and VTC 2003, etc. His current research interests include communication protocol design and performance analysis for

wireless networks, cross-layer optimizations, vertical handover and mobility management in mobile wireless networks, and adaptive multimedia transmission over hybrid wired/wireless networks. He has published over 40 international journal/conference papers, book chapters, and standard drafts in wireless communications, networking and multimedia fields. He is a member of IEEE.

E-mail: Qiang.Ni@ieee.org

Thierry Turletti received the M.S. (1990) and the Ph.D. (1995) degrees in computer science both from the University of Nice - Sophia Antipolis, France. He has done his PhD studies in the RODEO group at INRIA Sophia Antipolis. During the year 1995-96, he was a postdoctoral fellow in the Telemedia, Networks and Systems group at LCS, MIT. He is currently a research scientist at the Planète group at INRIA Sophia Antipolis. His research interests include multimedia applications, congestion control and wireless networking. Dr. Turletti currently serves on the Editorial Board of *Wireless Communications and Mobile Computing*.

E-mail: Thierry.Turletti@sophia.inria.fr