An Efficient Scheduling Scheme for IEEE 802.11e

Pierre Ansel, Qiang Ni and Thierry Turletti

PLANETE Project, INRIA Sophia Antipolis, FRANCE
pierre.ansel@polytechnique.org, qni@sophia.inria.fr, turletti@sophia.inria.fr

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, a lot of real-time applications, such as videoconferencing, 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.

1 Introduction

IEEE 802.11 Wireless LAN (WLAN) [1] has gained a great success for data applications in hotspots, enterprises, 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 (EDCF) and a controlled channel access mechanism (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.

In this paper, we propose a simple and efficient scheduling algorithm, FHCF, that aims to be fair for different kinds of multimedia flows and compatible with the 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 explains the principles of the FHCF scheme. Section 4 describes the simulation results to compare our FHCF scheduling scheme to the standard HCF scheme. Finally, Section 5 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. 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 stations.

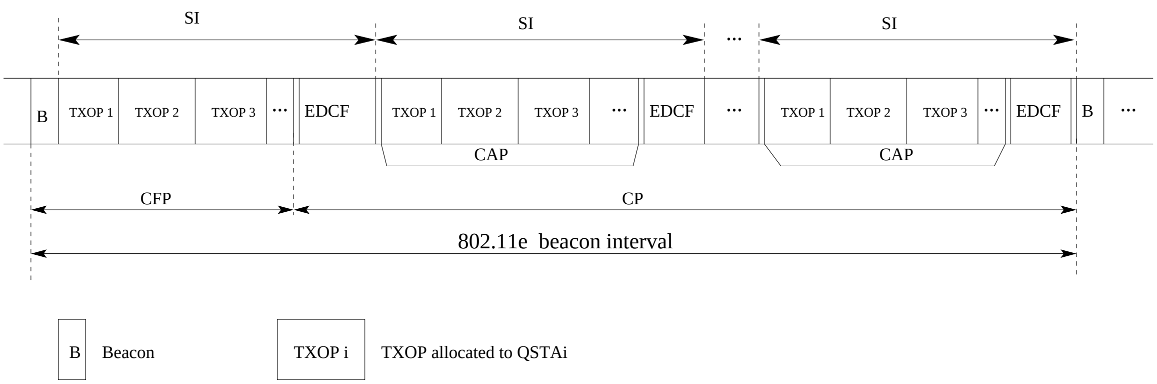

Figure 1. Structure of the 802.11e beacon interval used in the HCF scheduling algorithm [3]

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 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 packet drop. 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.

3 The FHCF Scheme

Basically, our 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 2. 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 since the TXOP is always allocated to a whole QSTA in our scheme.

3.1 QAP scheduler

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

$$q_i^{ideal} = \frac{\rho_i \cdot (SI - \sum_{j=1}^{i} N_j \cdot (\frac{M_j}{R} + 2SIFS + ACK))}{M_i}. \tag{3}$$

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, 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 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:

$$q_i^{est} = \frac{\rho_i(SI - t_i^e)}{M_i} + q_i^e. \tag{4}$$

If the sending rate and packet size of the application change, the above queue length estimation may not be accurate. To solve this problem, the FHCF scheme uses a window of w already known real queue length measurements to adjust the estimation. How to choose the value of w is a tradeoff between complexity and accuracy: with a bigger value of w we can use more previous records for estimation and improve the accuracy, however increases the complexity. In our work, the window size w has been empirically set to 5 through simulation results. Then, during the n-th SI, the QAP computes $|\Delta_i^n|$, the absolute value of the difference between $q_i^{b,real}$ (the real queue length at the beginning of TXOP of the i-th TS) and $q_i^{b,est}$, (the estimation of this queue length): $\Delta_i^n = q_i^{b,real} - q_i^{b,est}$.

Figure 3 shows that for a typical VBR video traffic[1], the sending rate almost follows a Gaussian law and packets have a fluctuating size. Thus Δ_i^n also follows the same Gaussian law with an expected value of 0. By recording a window of w Δ_i^n, and then by adding to $q_i^{b,est}(n)$ a corrective term that accounts for the variability of Δ_i^n, the QAP scheduler is able to improve its estimation of the queue length at the beginning of the next polling of the TS :

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

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 estimated queue length and the ideal case:

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

where $q_i^{est}(n)$ and $q_i^{ideal}(n)$ are given by (4) and (3). 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

[1] This video traffic has been obtained using the VIC videoconferencing tool [7] and the H.261 coding in the QCIF format [8] for a typical "head and shoulder" video sequence, see Section 4.

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

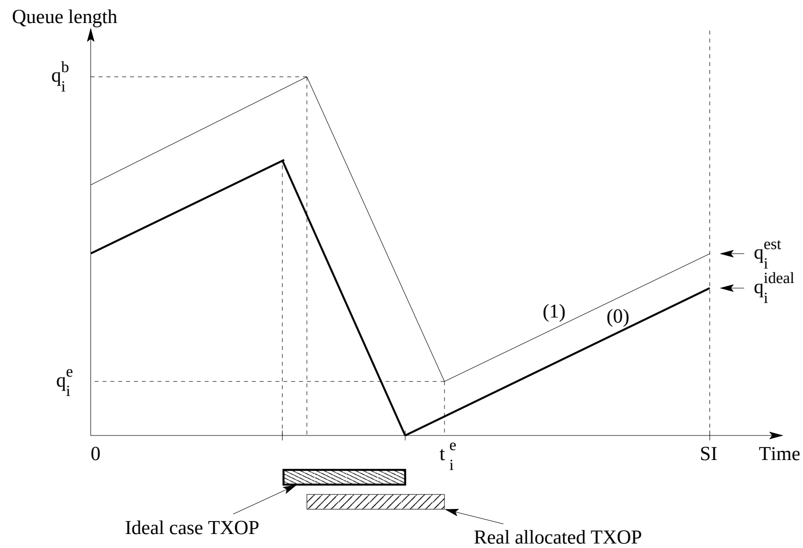

Figure 2. Queue length evolution for a TS: (0). Ideal queue length case; (1). Estimated queue length evolution

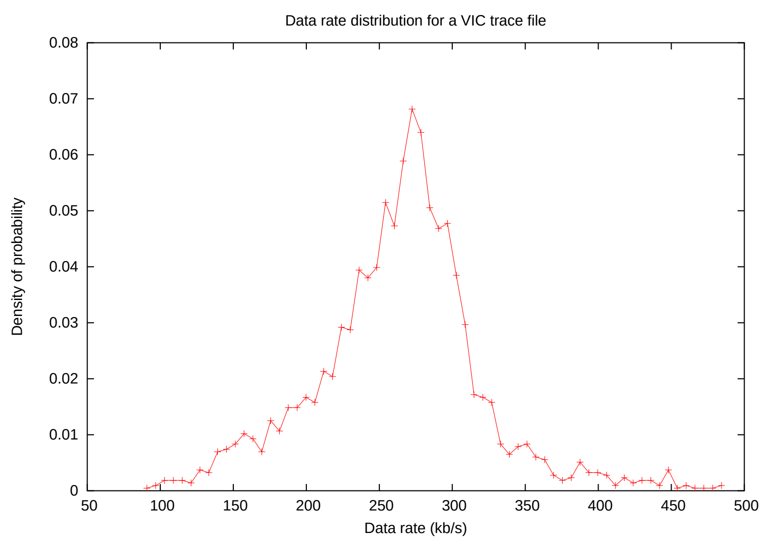

Figure 3. Distribution of the sending rate of a real VBR video traffic

TXOP duration according to the estimation of DN_i^{est}:

$$t_i^{est} = DN_i^{est} \cdot (\frac{M_i}{R} + 2SIFS + ACK). \tag{5}$$

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 all 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:

$$\beta = -\frac{(T_P - T_N) - T^r}{T_P + T_N}. \tag{6}$$

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

$$t_i^{add} = \begin{cases} (1+\beta)t_i^{est} & \text{if} \quad t_i^{est} \geq 0 \\ (1-\beta)t_i^{est} & \text{if} \quad t_i^{est} < 0. \end{cases} \tag{7}$$

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.

3.2 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:

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

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 (5), (6) and (7)). The difference is that the coefficient β 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 β is positive.

4 Simulation Experiments

We have implemented the FHCF scheme in the ns-2 simulator, our source code is available at [9]. In order to evaluate the performance of both QAP scheduler and node scheduler, two kinds of simulation topologies are used. The first one contains 18 mobile QSTAs and 1 QAP with only one TS per station (see Section 4.1), which is designed to evaluate the performance of the QAP scheduler. The second topology is composed of 6 QSTAs and 1 QAP (see Section 4.2), 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.

4.1 Scenario 1

In Scenario 1, 6 QSTAs send a high priority on-off audio traffic ($64kb/s$), another 6 QSTAs send a VBR video traffic ($200kb/s$ of average sending rate) with medium priority and 6 QSTAs send a CBR MPEG4 [10] video traffic ($3.2Mb/s$) with low priority. Table 1 summarizes the different traffic 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 PHY and MAC layer parameters used in the simulation are summarized in Table 2.

Figure 4 shows that with the FHCF scheme, all the flows have a maximum latency which corresponds to the selected SI of the different flows (chosen equal to $50ms$) whereas for standard HCF scheme, 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

[3] The video trace files are available from [9].

[4] By default, VIC uses a MTU size of $1024bytes$.

Table 1. 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
$7 \rightarrow 12$	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 Badwidth	$6Mb/s$
SlotTime	$9\mu s$
CCA Time	$4\mu s$
MAC header	$38bytes$
PLCP header length	$4bits$
Preamble length	$20bits$

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[5] 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 $50ms$. As regards the standard HCF scheme, the delays of the VBR flow are completely uncontrolled (see Figure 4) because the queue lengths are increasing during time. 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.

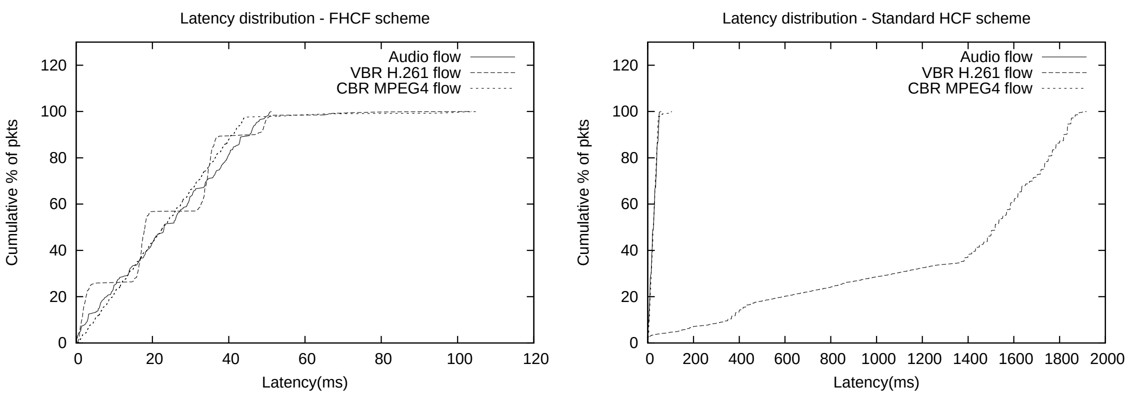

Figure 4. Latency distribution for FHCF and standard HCF

Figure 5 shows the fairness comparison between FHCF and HCF schemes when the HCCA load rate is modified by increasing the sending rate of the CBR MPEG4 traffic. In order to compare fairness for different schemes between the same kinds of traffic, 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 both VBR and CBR video flows, FHCF is much fairer than 802.11e HCF since our scheduler can estimate the varying queue length and allocate the TXOP fairly between the different flows.

4.2 Scenario 2

In Scenario 2 (see Table 3), 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 $600bytes$

[5] This is due to VIC fragmentation.

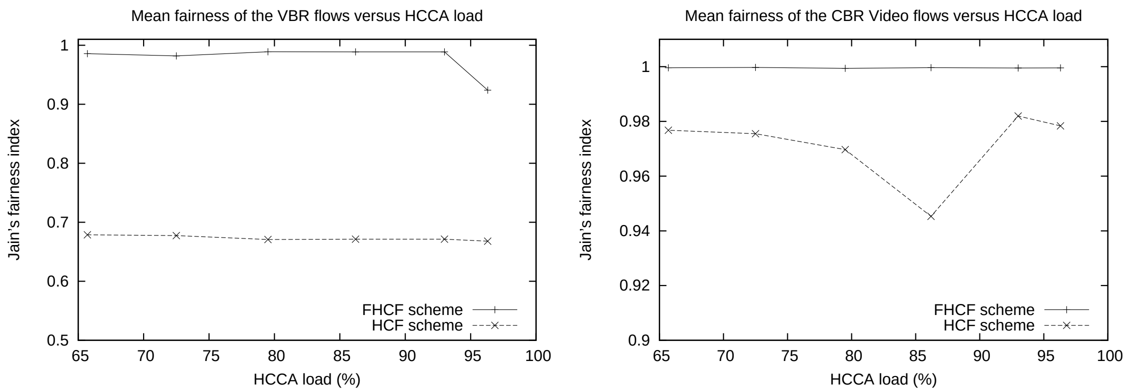

Figure 5. Fairness versus load for FHCF and standard HCF

$(2.4Mb/s)$ 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 choose the first way to increase the traffic load.

Figures 6 and 7 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 3).

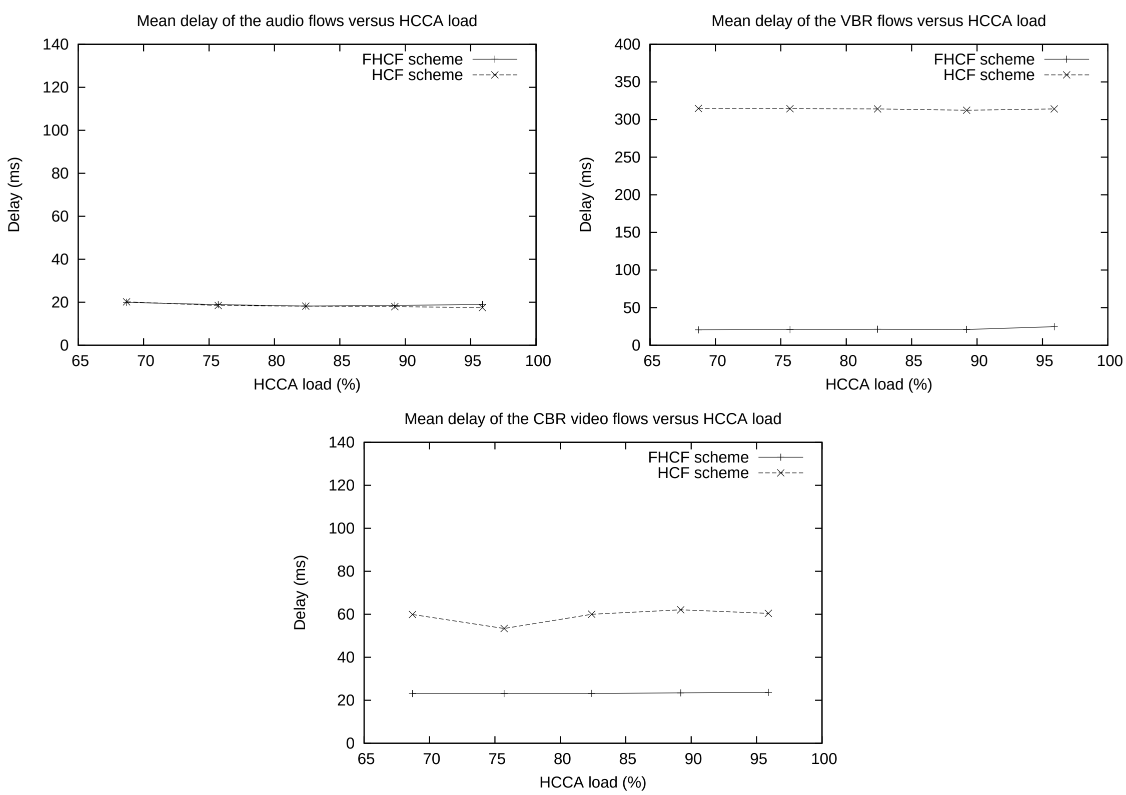

Figure 6. Mean delays versus load

Audio and VBR H.261 video flows. Figure 6 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

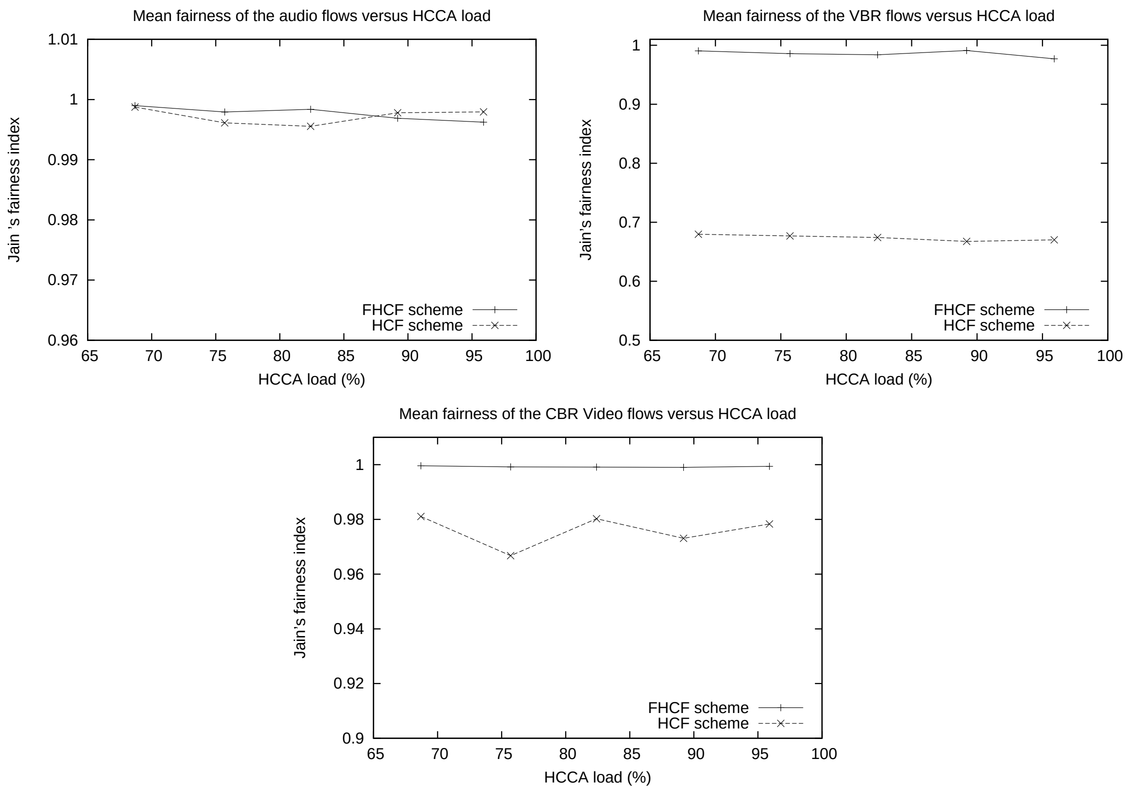

Figure 7. Fairness versus load

Table 3. 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	$600 \rightarrow 1000$	$2400 \rightarrow 4000$

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 7, 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.

CBR MPEG4 video flows. In our simulations, CBR flows are the most responsible for the network load. Figure 6 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 7, 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%). However with HCF, fairness performance is poor since both schedulers are not able to absorb traffic fluctuations.

Figure 8 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.

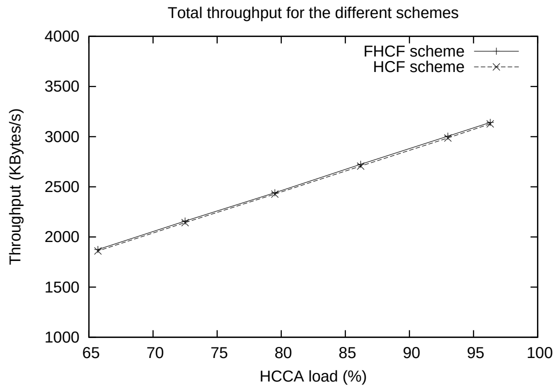

Figure 8. Total throughput

5 Conclusion

We have designed and evaluated a new MAC scheduling scheme, the FHCF, for the upcoming 802.11e 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 instead of the maximum sending rate usable for the standard HCF scheme. In this way, 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. Moreover, the FHCF scheme is shown to achieve a higher degree of fairness among different multimedia flows. Future work will include designing adaptive and robust scheduling algorithms for error-free/error-prone IEEE 802.11e wireless channels based on signal processing and time-series theories.

Acknowledgements

The authors would like to thank Dr. Chadi Barakat (INRIA), Prof. Sunghyun Choi (Seoul National University, Korea) and Dr. Matthew Sherman (AT&T Shannon Laboratory, USA) for providing valuable comments. 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. Journal of Wireless and Mobile Computing, John Wiley, **4** (2004) 1–20 (to appear).
3. IEEE 802.11 WG: IEEE 802.11e/D4.1, Wireless MAC and physical layer specifications: MAC enhancements for QoS. (2002).
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 simulations 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, (1991).