Adaptive Fair Channel Allocation for QoS Enhancement in IEEE 802.11 Wireless LANs

Mohammad Malli, Qiang Ni, Thierry Turletti, Chadi Barakat
Projet Planète, INRIA-Sophia Antipolis, France
E-mail:{mmalli, qni, turletti, cbarakat}@sophia.inria.fr

***Abstract*— The emerging widespread use of real-time multimedia applications over wireless networks makes the support of Quality of Service (QoS) a key problem. In this paper, we focus on QoS support mechanisms for IEEE 802.11 Wireless ad-hoc networks. First, we review limitations of the upcoming IEEE 802.11e Enhanced DCF (EDCF) and other enhanced MAC schemes that have been proposed to support QoS for 802.11 ad-hoc networks. Then, we describe a new scheme called *adaptive fair EDCF* that extends EDCF, by increasing the contention window during deferring periods when the channel is busy, and by using an adaptive fast backoff mechanism when the channel is idle. Our scheme computes an adaptive backoff threshold for each priority level by taking into account the channel load. The new scheme significantly improves the quality of multimedia applications. Moreover, it increases the overall throughput obtained both in medium and high load cases. Simulation results show that our new scheme outperforms EDCF and other enhanced schemes. Finally, we show that the adaptive fair EDCF scheme achieves a high degree of fairness among applications of the same priority level.**

I. Introduction and Motivation

In this paper, we focus on the enhancement of the medium access control (MAC) protocol for 802.11 Wireless LANs (WLANs). Our main objective is to design a scheme that obtains the best performance for multimedia applications whatever is the channel load, and with low complexity. To implement such a scheme, it is necessary to consider service differentiation for flows with different priorities and fairness between flows with the same priority. Before presenting our solution, we first provide a short overview of the 802.11 MAC protocol and the upcoming 802.11e standard.

The IEEE 802.11 MAC layer incorporates two access methods: the basic method called DCF (Distributed Coordination Function) and the optional method called PCF (Point Coordination Function).

The DCF method is based on the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol, which is used within both ad-hoc and infrastructure network configurations. In this protocol, when a node receives a packet to be transmitted, it first listens to the channel to ensure that no other node is transmitting. If the channel has been found idle for an interval of time longer than DCF InterFrame Space (DIFS), the node transmits the packet immediately. Otherwise, it chooses a random backoff time which determines the amount of time the node must wait until it is allowed to transmit the packet. The backoff timer is decremented only during periods in which the channel is idle. When the backoff timer reaches zero, the node transmits the packet. Fairness of DCF has been investigated in [12] and an analysis of several enhanced protocols are presented in [11]. DCF is used to support asynchronous data transmission. This mode performs well under low traffic load. However, DCF suffers from significant throughput degradation and high delay at high load conditions. The waste of bandwidth is caused by the increasing time used for negotiating channel access. Moreover, since DCF does not handle service differentiation, it is unsuitable for real-time applications.

PCF is designed to support time-bounded multimedia services and aims at eliminating the contention during high channel load. However, this protocol is scarcely implemented in 802.11 devices. PCF is known to perform poorly [3], [4], [10] for many reasons such as the access point is not able to poll all the stations during one cycle when the node's transmission duration is variable. Thus, stations that have not been polled must postpone their frames queued for transmission to the next contention free period causing an additional delay penalty.

To deal with these problems, the IEEE 802.11 working group is currently working on an extension to the 802.11 standard called 802.11e. This new proposal aims to introduce QoS support into the 802.11 standard. A new access method is proposed called Hybrid Coordination Function (HCF), which combines the advantages of both DCF and PCF functions. HCF describes some enhanced QoS-specific functions and frame subtypes to allow a uniform set of frame exchange sequences to be used for QoS transfers during both contention periods and contention free periods (when the access point polls the stations to start the transmissions). In this paper, we are only interested in the HCF contention-based channel access, called Enhanced DCF (EDCF) [2]. EDCF can be useful in ad-hoc networks because it does not require the presence of an access point. Each packet from higher layer arrives at the MAC layer with a specific priority value. An 802.11e station can implement four access categories (ACs), where each packet arriving at the MAC layer with a priority is mapped into an AC. Basically, EDCF uses different Arbitration Interframe Spacing ($AIFS[AC]$), Minimum Contention Window value ($CW_{min}[AC]$) and Maximum Contention Window value ($CW_{max}[AC]$) for the contention process to transmit packets belonging to the different ACs instead of single $DIFS$, CW_{min}, and CW_{max} values as in 802.11 DCF. These parameters can be used in order to differentiate the channel access among different priority traffic. $AIFS[AC]$ is determined by:

$$AIFS[AC] = SIFS + AIFSN[AC] \cdot SlotTime, \quad (1)$$

where $AIFSN[AC]$ is an integer greater than zero. More-

over, the backoff timer is selected from $[1, 1 + CW[AC]]$, instead of $[0, CW]$ as in the DCF. Figure 1 shows the timing diagram of the EDCF channel access. Basically, the smaller $AIFSN[AC]$, $CW_{min}[AC]$, and $CW_{max}[AC]$, the shorter the channel access delay for the corresponding priority, and hence the more capacity share this priority obtains for a given channel load. However, the probability of collisions increases when operating with smaller $CW_{min}[AC]$.

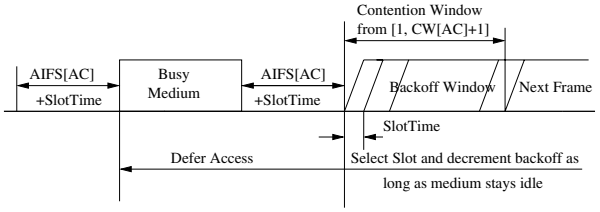

Fig. 1. IEEE 802.11e EDCF channel access

Figure 2 shows the 802.11e MAC with four transmission queues in a station, where each queue behaves as a single enhanced DCF contending entity, i.e., an AC with its own $AIFS[AC]$ and Backoff Timer ($BT[AC]$). When more than one AC within a station have their $BT[AC]$ expire at the same time, the collision is handled in a virtual manner. That is, the highest priority packet among the colliding packets is chosen and transmitted, and the other queues perform a backoff with increased $CW[AC]$ values.

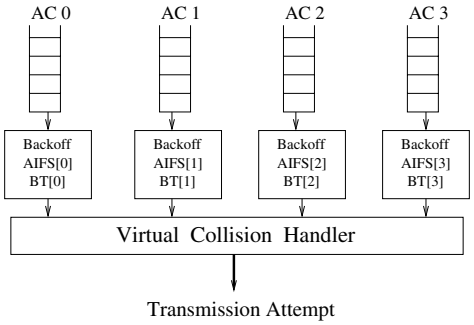

Fig. 2. Four access categories (ACs) for EDCF

As we will observe later in our simulations, EDCF performs poorly when the medium is highly loaded. This is due to the high collision rate and wasted idle slots caused by backoffs in each contention cycle.

The Adaptive EDCF (AEDCF) scheme [5] has recently been proposed to improve the performance of EDCF. This scheme is based on an old version of EDCF (draft 3.1 [1]). It adjusts the $CW[AC]$ size of each traffic class taking into account the channel collision rate, which improves the total goodput of EDCF (draft 3.1) up to 25%. However, when we experimented AEDCF with the new version of 802.11e (draft 4.1 [2]), we found that while AEDCF still improves the total goodput, the performance of background low-priority flows degrades at high load. The reason is that in the 802.11e draft 4.1, $CW_{min}[AC]$ and $CW_{max}[AC]$ values of background traffic are much larger than those of other traffic classes. When the channel is highly congested, the background queue increases its $CW[AC]$ (after a failed transmission) with a multiplicative factor larger than 2 by using AEDCF. Thus, the background traffic will have much larger average $CW[AC]$ size than high-priority traffics with AEDCF. This increases considerably the waiting time of background traffic and impairs the channel utilization.

Recently, the FCR scheme [6] has been proposed to improve the performance of DCF. It uses a fast backoff mechanism with a static backoff threshold. However, as it does not support service differentiation, it cannot provide good performance for multimedia applications. This motivates us to design a new scheme, called *adaptive fair EDCF*, which combines the advantages of service differentiation, fast backoff decrease [6], and an adaptive access scheme (using an adaptive Backoff Threshold). It is based on the latest 802.11e specifications [2], and aims to improve (i) the performance of multimedia applications whatever is the channel load, (ii) the total throughput obtained, and (iii) the fairness between the same priority applications. We implement our scheme in NS [14] and we show with extensive simulations that it outperforms other QoS mechanisms for WLAN.

This paper is organized as follows. The next Section describes our adaptive fair EDCF scheme: the analytical analysis, the scheme algorithm, and the backoff threshold function. Then, Section III presents the simulation results and comparison between *adaptive fair EDCF*, EDCF, AEDCF, and FCR. Finally, the conclusion is presented in Section IV.

II. Adaptive Fair EDCF

A. Analytical Analysis

The main performance impairment of distributed contention-based EDCF scheme comes from packet collisions and wasted idle slots due to backoffs in each contention cycle as shown in Figure 3. Consider the case when the channel

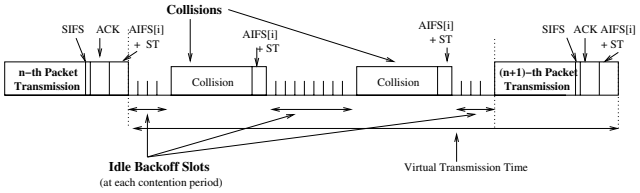

Fig. 3. Basic operations of EDCF

is highly loaded, i.e. all the queues in the different stations have always packets to transmit. Under some ergodicity assumption, the throughput of queue i for one transmission cycle (see Figure 3) can be expressed as:

$$\rho[i] = \frac{\overline{P_{size}}[i]}{\overline{T_x}[i]}, \tag{2}$$

where $\overline{T_x}[i] = \overline{CN} \cdot (\overline{IS}[i] \cdot ST + \overline{PktTx}[i] + AIFS[i] + ST) + \overline{IS}[i] \cdot ST + \overline{PktTx}[i] + SIFS + ACK + AIFS[i] + ST$, is the average virtual transmission time of a packet by the MAC layer. $\overline{P_{size}}[i]$ is the average packet length of flow in queue i, $\overline{PktTx}[i]$ is its transmission time over the wireless

medium, $\overline{CN}$ is the average number of collisions (real and virtual collisions) in a virtual transmission time (or a virtual transmission cycle), $\overline{IS}[i]$ is the average number of idle slots resulting from the queue i's backoff for each contention period, ACK is the acknowledgment's transmission time destined to the queue i after the successful receiving of the n-th Packet, and ST is the SlotTime which depends on the physical layer type.

The above throughput expression shows that the ideal case (in Figure 3) is reached when a successful packet transmission is followed by another successful packet transmission without any collisions or idle time loss, i.e. $\overline{CN} = 0$, and $\overline{IS}[i] = 0$. The maximum throughput of queue i is then:

$$\rho[i] = \frac{\overline{P_{size}}[i]}{\overline{PktTx}[i] + SIFS + ACK + AIFS[i] + ST}. \quad (3)$$

This best medium utilization can be obtained only when EDCF supports a perfect scheduling algorithm among all the queues in the different WLAN's nodes. Ideally, when queue i transmits, the probability that the others transmit should be equal to 0. Thus, only the backoff counter of queue i should expire. Such an ideal MAC function cannot be implemented in practice in a distributed way without the presence of a coordinating point. One way to approximate this perfect scheduling scheme is to use the FCR mechanism proposed for DCF in [6]. The FCR mechanism consists in using a backoff threshold value that separates two backoff states. Assuming that each priority queue acts as a virtual station, we extend the idea of FCR for EDCF by using an adaptive backoff threshold for each priority queue. This allows us to differentiate between the different priorities and to take into account the channel load which is necessary to increase further the total throughput of the medium.

The first backoff stage corresponds to linear decrease as in the standard [2]. During this stage, for each slot time the channel is sensed idle, the backoff timer is decreased by one slot time and the remaining backoff time is compared with the threshold value. When the remaining backoff time reaches the threshold value, the queue starts the second state by reducing the $BT[i]$ exponentially (see section II-B).

Basically, the backoff threshold $Bof\_Th[i]$ (see Figure 4) should increase during low contention periods (when there is a small number of priority queues contending to access the medium) in order to reduce idle time, and it should decrease during high contention periods (when there is a large number of priority queues contending to access the medium) in order to reduce collisions. This is one of the main ideas behind our work, that is adapting the backoff threshold for each class of traffic as a function of the channel load.

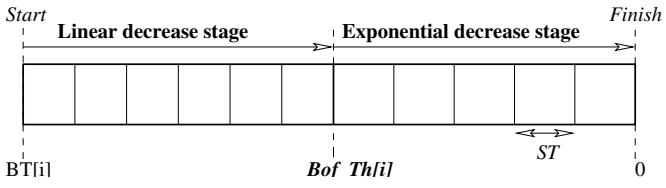

Fig. 4. Backoff Time decrease phases

We observe in our simulations that the quality of multimedia flows with IEEE 802.11e EDCF scheme begins to degrade during moderate channel load to become very bad during high channel load. By using the fast backoff mechanism, the total throughput is higher than with the EDCF, but the performance of multimedia traffic degrades considerably when the number of nodes increases.

To protect the quality of high priority flows without reducing the total throughput, we extend the use of the adapted fast backoff mechanism by a new scheme which consists of increasing the $CW[AC]$ size not only when there is a collision, but also when a queue senses the channel busy during deferring periods (i.e., time when a queue has not expired its $BT[AC]$ yet). This way, it is very likely that the highest priority flow will win the next channel access because of its smaller $CW_{max}[AC]$ value compared to other flows.

B. Scheme Algorithm

The algorithm of our adaptive fair EDCF scheme is described as follows.

1) Backoff Timer Decrease State: All priority queues in the different active stations monitor the medium. If a queue i senses the medium idle for a slot, then it will start decrementing its backoff timer by a slot time as in the IEEE 802.11e specification, i.e., $BT_{new}[i] = BT_{old}[i] - ST$. If a number of consecutive idle slots are detected and the remaining backoff timer value is less or equal than the Backoff Threshold $Bof\_Th[i]$ value, our algorithm will decrease faster (exponentially) the backoff timer as proposed in [6] for DCF:

$$BT_{new}[i] = BT_{old}[i]/2, \quad (4)$$

$$\text{if } BT_{new}[i] < ST, \text{ then } BT_{new}[i] = 0. \quad (5)$$

When the backoff timer reaches zero, the station transmits a packet.

2) Packet Collision State: If a queue notices that its packet transmission has failed possibly due to a packet collision, the queue must react with these modifications: (i) it must double its current $CW[i]$ value by using Equation (6) as in the IEEE 802.11e specification in order to avoid a new collision, (ii) it must update its $BT[i]$ value by using Equation (7), and (iii) it must reduce its $Bof\_Th[i]$ value by using Equation (8) in order to decrease the fast decrease phase (see Figure 4). We explain later the logic behind (8).

$$CW[i] = min(CW_{max}[i], 2 \cdot CW[i]), \quad (6)$$

$$BT[i] = uniform(1, CW[i] + 1) \cdot ST. \quad (7)$$

3) Successful Packet Transmission State: When a queue successfully transmits a packet, it must react with these modifications: (i) it reduce its current $CW[i]$ size to $CW_{min}[i]$ as specified in the IEEE 802.11e specification in order to reduce the idle time, (ii) it must update its $BT[i]$ value by using Equation (7), and (iii) it must increase its $Bof\_Th[i]$ value by using Equation (8) in order to increase the exponential decrease stage (see Figure 4).

4) Deferring State: If a queue is in a deferring state (i.e. waiting the end of a busy period to continue decreasing its backoff timer), whenever it detects the start of a new busy period (which indicates either a collision or a packet transmission in the medium by another station), it will react as if it is in the packet collision state described in Section II-B.2 in contrast with the IEEE 802.11e specification. We propose such a behavior in order to protect multimedia flows transmission and to improve the fairness between the same priority applications especially when the medium is highly congested. Basically, the queues which are in the deferring state double $CW[i]$ when they sense the medium is busy in order to: (i) penalize the low priority queue because it has the largest $CW_{max}[i]$ value while the highest priority queue will gain more transmission opportunities due to its small $CW_{max}[i]$ value, (ii) improve the fairness between the same priority queues by having almost, after the finish of a busy period, the same value of $CW[i]$ equal to $CW_{max}[i]$ and consequently the same transmission opportunity. Our mechanism for adapting the $Bof\_Th[i]$ function (Section II-C) ensures that the protection of multimedia flows performance is accompanied by a total throughput increasing since our scheme implements the fast backoff decrease mechanism.

C. Backoff Threshold Function

We propose the following function to determine the *backoff threshold*:

$$Bof\_Th[i] = \frac{CW_{max}[i] - CW[i]}{CW_{max}[i] - CW_{min}[i]} \cdot \frac{BT[i]}{CW[i]} \cdot CW_{min}[i] \cdot ST, \quad (8)$$

This function is adaptive and it supports service differentiation. It adapts to the channel load by including the parameter $CW[i]$, while it differentiates between the different priority flows by including the parameters $CW_{min}[i]$ and $CW_{max}[i]$. The logic behind this adaptation is illustrated by these two conditions: (i) when medium load decreases and the queue decrements its $CW[i]$ value (after a successful transmission), it must extend the exponential decrease stage (see Figure 4), by increasing its $Bof\_Th[i]$ parameter, in order to reduce the idle time (ii) when medium load increases and the queue increments its $CW[i]$ value (after a collision or a busy period detection), it must reduce the exponential decrease stage (see Figure 4), by decreasing its $Bof\_Th[i]$ parameter, in order to avoid a new collision.

Another issue to take into account for defining $Bof\_Th[i]$ function is that it must be scaled to $BT[i]$ value and not to $CW[i]$ value (see Equation (7)). The main benefit of this scaling appears when $CW[i]$ is increased to a big value and a small $BT[i]$ value is chosen randomly. In this case the $Bof\_Th[i]$ must follow $BT[i]$ and have a small value. We choose to compute the $Bof\_Th[i]$ first by only using the $CW[i]$, then we scale it by the ratio $BT[i]/CW[i]$ to account for $BT[i]$. Keeping all this in mind, we explain next the $Bof\_Th[i]$ function written in Equation (8).

First, we draw a linear function (see Figure 5) which joins the two points $A(CW[i] = CW_{min}[i], Bof\_Th[i] = CW_{min}[i])$ and $B(CW[i] = CW_{max}[i], Bof\_Th[i] = 0)$. The intuition behind choosing A is that we want to have only fast $BT[i]$ decrease in low load scenarios (no linear decrease phase). Let us evaluate the profit of choosing the point A in the scenario where we have only one user's application transmitting (one queue). In this scenario, the queue almost has successful transmissions (if no transmission errors) which results in $CW[i] = CW_{min}[i]$. The queue does not need to wait a lot of time before transmitting a packet because there is no other contending queues; so it is more efficient that the queue decreases its $BT[i]$ in an exponential way to reduce the idle time. Besides, we choose the point B to let the queue's transmission be friendly during congestion periods by making the backoff timer decrease all time linearly. Thus when $CW[i] = CW_{max}[i]$, the queue will not have an exponential decrease stage in order to wait more time before transmitting a packet and consequently to reduce the number of collisions.

Finally, the ratio $BT[i]/CW[i]$ is included in the Equation (8) as a factor which is multiplied by the linear function of the segment $[AB]$ (drawn in Figure 5). The reason is to scale the value of the $Bof\_Th[i]$ taken from the linear function (see Figure 5) in order to account for $BT[i]$, which can take any value between 1 and $CW[i] + 1$ (Equation (7)). The absence of this factor in the Equation (8) can have a bad impact on the performance when the gap between $BT[i]$ and $CW[i]$ increases. Basically, when the value taken from the linear function is equivalent to $n\%$ compared to $CW[i]$ value, $Bof\_Th[i]$ must have a value which is equivalent to $n\%$ compared to $BT[i]$ value.

As we can observe, this scheme is of low complexity. Knowing the current $CW[i]$ value for a queue is enough to update its $Bof\_Th[i]$ value when necessary.

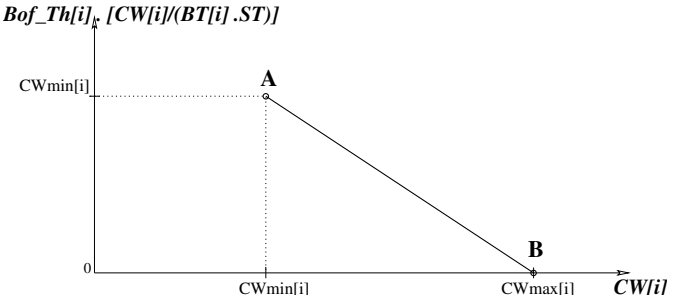

Fig. 5. Backoff Threshold Function

III. Simulation Results

We implemented our scheme in NS-2[1] and experimented it with different ad-hoc network topologies to evaluate the performance of multimedia flows with different channel loads, see Table I. In all cases, each node sends three different flows (audio, video, and data) to a common receiver as shown in Figure 6. The physical data rate is set to 36 Mb/s. For all the scenarios considered, we set the EDCF flows parameters (based on draft 4.1 [2]) for the three ACs as shown in

[1]Our ns simulation code is available on *http://www-sop.inria.fr/planete/software/.*

Table II. Each simulation is run for 15 seconds and the results shown are averages over 5 simulations with different flow starting times. Fairness curves are averaged over 20 simulations, and are computed for each priority flow. Figure 7

number of nodes	4	6	8	10	12	14	16
load (%)	19	31	44	55	68	80	100

TABLE I

LOAD PERCENTAGES IN DIFFERENT TOPOLOGY CASES

	Audio	Video	Background
Transport	UDP	UDP	UDP
Priority	3	2	0
CW_{min}	7	15	31
CW_{max}	15	31	1023
AIFSN	1	1	2
Packet Size	160 bytes	1280 bytes	1500 bytes
Packet Interval	20 ms	10 ms	12.5 ms
Flow Rate	8 Kbytes/s	128 Kbytes/s	120 Kbytes/s

TABLE II

USED EDCF PARAMETERS FOR THE THREE TCS

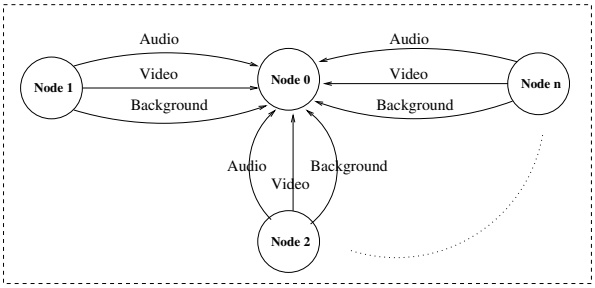

Fig. 6. Simulation Topology

shows that *adaptive fair EDCF* provides significantly more total throughput compared to EDCF and AEDCF, mainly in high load situations (33% total goodput gain when the channel is fully loaded). Also, we observe in Figure 8 that the new scheme protects video flow throughput for all channel loads, while the video flow throughput deteriorates in moderate and high load cases with EDCF (see Figure 9), AEDCF [5] (see Figure 10), and FCR [6] (see Figure 11). Moreover, we can see from Figure 10 that AEDCF impairs the throughput of background traffic more than the other schemes because the AEDCF background queue waits a lot of time before transmitting a packet due to the fact that it increases its $CW[i]$ in AEDCF (after a transmission failed) by a multiplicative factor larger than 2 while the other schemes (EDCF, FCR, and adaptive fair EDCF) use always a multiplicative factor equal to 2. Figures 12 and 13 show the cumulative fractions for packets delay respectively in *adaptive fair EDCF* and EDCF cases when the percentage of the channel load is 80%. In this case of high load, we observe that with the adaptive fair EDCF scheme (see Figure 12), 90% of packets obtain approximately maximum delays of $1.5ms$ for audio, $4ms$ for video, and $1.7s$ for background traffic, while with EDCF, about 90% of packets obtain $11ms$ for audio, $700ms$ for video, and $7s$ for background traffic (see Figure 13), which is much higher that with our proposed scheme. As shown in Figure 14, our scheme provides higher channel utilization than EDCF when channel load is higher than 60%. Our gain on medium utilization is up to 34% when load is 100%. Moreover, the adaptive fair EDCF scheme allows to reduce considerably the number of collisions in both moderate and high load states as shown in Figure 15.

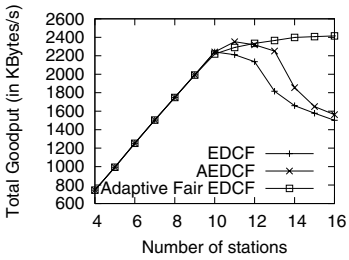

Fig. 7. EDCF, AEDCF, and Adaptive Fair EDCF total goodput

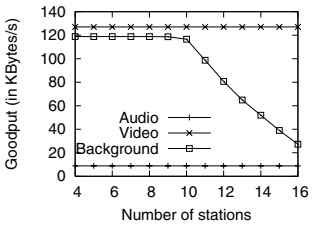

Fig. 8. Adaptive Fair EDCF

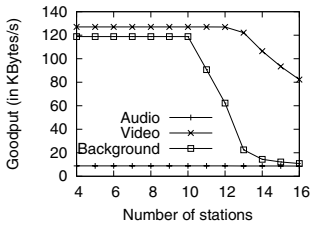

Fig. 9. EDCF

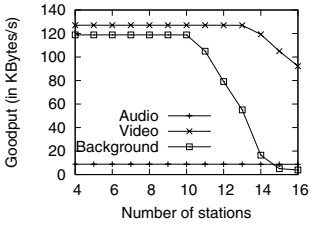

Fig. 10. AEDCF

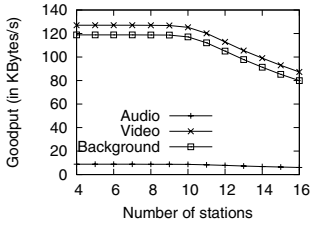

Fig. 11. FCR

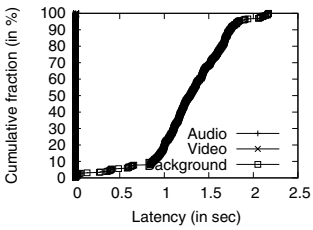

Fig. 12. Adaptive Fair EDCF

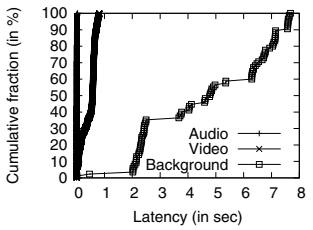

Fig. 13. EDCF

We now evaluate the fairness of our scheme between flows of the same priority over our simulation time of 15 seconds, which is relatively a short time interval. For this purpose, we calculate the fairness index defined by Jain [13] for each scheme and for each run, then we average over all runs. This

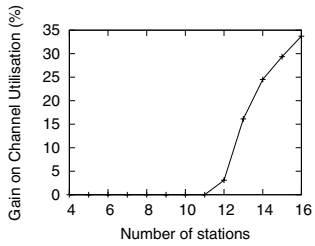

Fig. 14. Channel utilization gain

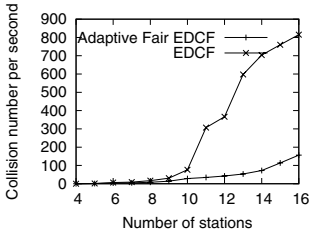

Fig. 15. Collision rate

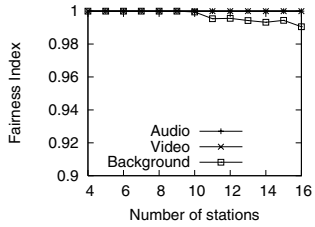

Fig. 16. Adaptive Fair EDCF

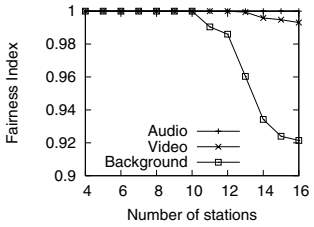

Fig. 17. EDCF

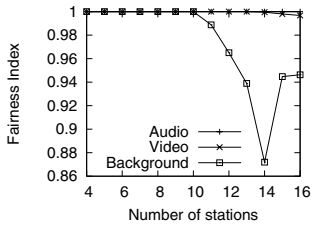

Fig. 18. AEDCF

fairness index FI is defined as:

$$FI = \frac{(\sum_{i=1}^{n} T_i)^2}{n \cdot \sum_{i=1}^{n} (T_i)^2}, \quad (9)$$

where n is the number of the same priority flows, and T_i is the throughput of flow i. We recall that $FI \leq 1$, and it is equal to 1 if all T_i are equal, which corresponds to the highest degree of fairness between the different users. As shown in Figures 16, 17, and 18, our scheme is always fairer than EDCF and AEDCF especially when the channel is highly loaded. The main reason comes from the fact that *adaptive fair EDCF* doubles $CW[i]$ every time the channel is sensed busy, i.e. whereas the queue is in the collision state (see section II-B.2) or in the deferring state (see section II-B.4). In the high load case, this behavior contributes to the fact that the $CW[i]$ values of all the queues of all the stations reach rapidly their maximum values. This provide better fairness between different users since the queues of the different users will be transmitting almost all the time at the same contention window. This is not the case when using EDCF and AEDCF because these two schemes double the $CW[i]$ values only in a collision state, which results in a slow increase in the $CW[i]$ values and hence in a large variability in the $CW[i]$ values among users. The large variability in the $CW[i]$ values leads to a problem of fairness at short time scales of the order of the 15 seconds we are using for our simulations.

IV. Conclusion

In this paper, we described a new scheme for QoS enhancement for IEEE 802.11 wireless ad-hoc networks. We extend the basic EDCF scheme by using an adaptive fast backoff mechanism to improve the total throughput along with a window doubling mechanism at busy periods to protect further high priority flows and improve the fairness between those of the same priority especially in the scenarios where the channel is highly congested. Our scheme adapts the stations aggressiveness during accessing the medium according to its load by using an adaptive fast backoff decrease mechanism. It protects further the transmission of multimedia flows and the fairness between those of the same priority by increasing the current contention window size whenever the queue detects the channel busy in both the transmission failure state and the deferring state. The simulation results we obtained show that *adaptive fair EDCF* provides good multimedia flow performance in all channel loads as well as a higher total throughput than EDCF. Besides, it provides a higher degree of fairness than EDCF and AEDCF between the different flows of the same priority.

References

[1] IEEE WG, *Draft Supplement to Standard for Telecommunications and Information Exchange between Systems-LAN/MAN Specific Requirements- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC)*, Enhancements for Quality of Service (QoS), 802.11e Draft 3.1, May, 2002.

[2] IEEE WG, *Draft Supplement to Standard for Telecommunications and Information Exchange between Systems-LAN/MAN Specific Requirements- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC)*, Enhancements for Quality of Service (QoS), 802.11e Draft 4.1, February, 2003.

[3] S. Mangold, S. Choi, P. May, O. Klein, G. Hiertz, and L. Stibor, *IEEE 802.11e Wireless LAN for Quality of Service*, In Proc. of European Wireless (EW2002), Florence, Italy, February 2002.

[4] Q. Ni, L. Romdhani, and T. Turletti,, *A Survey of QoS Enhancements for IEEE 802.11 Wireless LAN*, to appear in Journal of Wireless Communications and Mobile Computing, John Wiley, vol.4, pp. 1-20, 2004.

[5] L. Romdhani, Q. Ni, and T. Turletti, *Adaptive EDCF: Enhanced Service Differentiation for IEEE 802.11 Wireless Ad Hoc Networks*, IEEE WCNC'03 (Wireless Communications and Networking Conference), New Orleans, Louisiana, March 16-20, 2003.

[6] Y. Kwon, Y. Fang, and H. Latchman, *A Novel MAC Protocol with Fast Collision Resolution for Wireless LANs*, IEEE Infocom 2003, 2003.

[7] S. Choi, J. del Padro, and S. Shankar, and S. Mangold, *IEEE 802.11e Contention-Based Channel Access (EDCF) Performance Evaluation*, January 2002.

[8] P. Garg, R. Doshi, R. Greene, M. Baker, M. Malek, and X. Cheng, *Using IEEE 802.11e MAC for QoS over Wireless*, IPCCC'03, 2003.

[9] I. Aad and C. Castelluccia, *Differentiation mechanisms for IEEE 802.11*, IEEE Infocom 2001, April 2001.

[10] A. Lindgren, A. Almquist, and O. Schelen, *Quality of Servise Schemes for IEEE 802.11 Wireless LANs*, In Proc. of IEEE LCN 2001, November 2001.

[11] K.C Chen, *Medium access control of wireless LANs for mobile computing*, IEEE Network, vol. 8, no. 5,pp. 50-63, september 1994.

[12] H.S. Chahalaya and S. Gupa, *Throughput and fairness proprietes of asynchronous data transfer methods in the IEEE 802.11 MAC protocol*, Personal Indoor Mobile and Radio communication conference, pp. 613-617,1995.

[13] R. Jain, A. Durresi, and G. Babic, *Throughput Fairness Index: An Explanation,*ATM Forum/99-0045, Feb. 1999.

[14] S. McCanne and S. Floyd, *Ucb/lbnl/vint network simulator (ns) version 2.1b6,* http://www-mash.cs.berkeley.edu/ns/, 2000.

[15] Y. Xiao, *Enhanced DCF of IEEE 802.11e to support QoS*, May 2003.