Tuning the EDCA parameters in WLANs with heterogeneous traffic: A flow-level analysis

C. Cano*, B. Bellalta, A. Sfairopoulou, J. Barceló

NETS Research Group, Dept. of Information and Communication Technologies, Universitat Pompeu Fabra, C/Tànger 122-140, 08018 Barcelona, Spain

**A R T I C L E   I N F O**

Article history:
Received 18 December 2008
Received in revised form 7 October 2009
Accepted 19 February 2010
Available online 4 March 2010
Responsible Editor: L. Lenzini

Keywords:
QoS
WLAN
MAC parameters tuning
EDCA

**A B S T R A C T**

The need to support multimedia services in WLANs has motivated the research on traffic differentiation mechanisms at the MAC layer. The most common approach is the tuning of the different MAC parameters of the heterogeneous traffic profiles in order to provide different channel access probabilities. The benefits of these mechanisms in terms of throughput and transmission delay, as well as their traffic differentiation and QoS capabilities, have been thoroughly studied. However, there are very few results on how the tuning of the MAC parameters impacts on the flow-level metrics, such as blocking probability or average flow duration. In this article, several EDCA-based tuning algorithms have been evaluated by comparing their flow-level response in presence of rigid (e.g. VoIP) and elastic (e.g. P2P) flows. Results show that those algorithms which adapt better to the changing WLAN state (number and type of active flows), and that are designed under multiple objectives, provide significantly higher performance and QoS than static and single-objective configurations.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

During the last years, the use of wireless local area networks (WLANs) as a wireless mechanism to access the Internet has grown considerably. Their low cost and unlicensed use are the main reasons that have motivated their widespread deployment, making feasible their consideration as a complement of cellular networks in order to provide mobile multimedia services.

WLANs are based on a carrier sense multiple access with collision avoidance (CSMA/CA) protocol, which is called distributed coordination function (DCF) [1]. The medium access control (MAC) protocol is the responsible for distributing the channel transmission opportunities, trying to provide a fair access (to give the same channel share) to all active nodes, without even differentiating between the access point (AP) and mobile stations (STAs). This fairness design has several drawbacks, specially remarkable when heterogeneous traffic flows (e.g. VoIP, video and TCP-like traffic) are simultaneously active in the cell, since the network is unable to provide them the required service guarantees (such as a constant bandwidth to real-time flows).

In order to mitigate this problem, the IEEE 802.11e enhanced distributed channel access (EDCA) specification [2] provides traffic differentiation by defining four access categories (ACs) with different MAC parameters leading to different transmission probabilities for each STA/AC. Nevertheless, the traffic differentiation provided by EDCA is static and quite conservative with respect to the best-effort flows, resulting in a low utilization of the transmission resources. This can be improved by adapting the defined AC MAC parameters according to the network state. Several mechanisms – MAC parameter tuning algorithms – have been proposed so far in order to efficiently adjust the MAC parameters with the goal to improve the WLAN performance, thus improving the quality of service (QoS)
provided to real-time flows while the overall network utilization is maximized.

The goal of this work is to compare different MAC parameter tuning algorithms already presented in the literature. The chosen algorithms are the works presented by Serrano et al. [3], Siris et al. [4], Freitag et al. [5] and Bellalta et al. [6]. The DCF and EDCA recommendations are also considered as examples of static algorithms. Although various performance studies of these algorithms can be found, they have been limited to packet-level metrics, such as throughput or delay, or focusing on the impact over the MAC behavior (like the transmission and collision probabilities). However, in this work, a flow level analysis is provided considering an heterogeneous traffic scenario with VoIP and elastic traffic flows. Results show how the different algorithms respond to the WLAN state changes, in terms of number of flows and flow characteristics, providing flow-level metrics (blocking probability, average number of simultaneously active flows, average flow transfer delay, etc.).

The paper is organized as follows: Section 2 describes the IEEE 802.11 and IEEE 802.11e specifications, Section 3 shows the WLAN performance with TCP and VoIP traffic and discusses the validation of the analytical model used in this work. Then, in Section 4 a description of the MAC parameters tuning algorithms considered is provided, while the evaluation method and the results obtained are presented in Section 5. Finally, in the last section, some concluding remarks are given.

2. The DCF and EDCA specifications

The IEEE 802.11e [2] amendment extends the original IEEE 802.11 [1] specification by including a traffic differentiation and call admission control mechanism. There are other innovations, not considered in this work, such as the direct link protocol (DLP) or the different ACK policies, including the Block ACK.

The goal of this section is to provide the reader with the necessary background to understand how the DCF and EDCA work and what are the performance tradeoffs of tuning the MAC parameters in WLANs. With these objectives in mind, the DCF is first introduced and it is used as a basis to explain the EDCA traffic differentiation functions. Additionally, in order to understand how the DCF/EDCA parameters impact on the system performance, several analytical expressions which capture the DCF/EDCA behavior are introduced. Although the specific details of the analytical model are not presented here, the reader can refer to [6–8] for further details. In Section 3 we show how the model helps to predict the tendencies of the WLAN performance with both VoIP and TCP traffic.

2.1. The distributed coordination function

The MAC layer of the IEEE 802.11 defines a way to share the channel by means of a distributed random access mechanism called DCF, basically a CSMA protocol (Fig. 1). Since all the nodes have the same MAC parameters, the medium access is considered fair: all nodes with a packet ready to be transmitted for the first time have the same probability to obtain the right to use the channel.

In the Basic Access mechanism, each time a node receives a packet to be transmitted from the upper layer it first senses the channel in order to determine its status. If the channel is detected free during a DCF inter frame space (DIFS) period the packet is transmitted immediately. Otherwise, the transmission is delayed until the channel is released. Just after that, the node waits a DIFS and a random value, called backoff counter, computed from a uniform distribution in the range $CW(k) = \min(2^k CW_{\text{min}} - 1, CW_{\text{max}} - 1)$, where $k$ is the current packet transmission attempt. The $k$ parameter is initially set to 0 for each packet to transmit and it is increased by one at each failed transmission until a maximum number of retransmissions, called Retry Limit, is reached.

The backoff counter is decreased in one unit each time-slot in which the channel is sensed free, until the countdown reaches zero, moment in which the node starts the packet transmission on the channel. If, during the backoff countdown, the channel is sensed busy due to a successful transmission or a collision, the backoff is frozen until the channel is detected free again (including the required DIFS). Then, the probabilities for a node $i$, which is in backoff, to sense an empty slot ($p_{e,i}$), a successful transmission ($p_{s,i}$) or a collision ($p_{c,i}$) are:

$$p_{e,i} = \prod_{j=1}^{j=i-1} (1 - \tau_j)p_{s,i} = \sum_{z=1}^{2z} \prod_{j=2z}^{j=i} (1 - \tau_j)p_{c,i}$$

$$= 1 - p_{e,i} - p_{c,i} \quad (1)$$

where $\tau_j$ refers to the transmission probability of node $j$. The $j$ and $z$ parameters range from 1 to $n$ (number of active nodes in the cell).

Let $\sigma, E[T_{s,i}^0]$ and $E[T_{e,i}^0]$ be the duration of an empty slot, the average duration of others successful transmissions and the average duration of others collisions from the point of view of a node $i$ that is in backoff, then the average slot duration is:

$$\gamma_i = p_{e,i}\sigma + p_{s,i}E[T_{s,i}^0] + p_{c,i}E[T_{e,i}^0] \quad (2)$$

which gives an average backoff duration equal to $E[B_i]\gamma_i$, where $E[B_i]$ is the average number of slots selected in a backoff instance and can be computed from the expression given by Tay et al. [9]. Once a node gets the channel, after the backoff period, it can transmit a single message protocol data unit (MPDU) packet. Then, the probability that the STA, sends a packet, $\tau_i$, depends on (a) STA has data to be transmitted ($\rho_i$), (b) its AC parameters and (c) the behavior of other nodes, which results in

$$\tau_i = \frac{\rho_i}{E[B_i] + 1} \quad (3)$$

Additionally, a channel collision occurs if at least two nodes transmit at the same time, i.e., a backoff instance from at least two nodes reach 0 at the same time. Therefore

1 Notice that new arriving packets do not need to backoff if they see their own transmission queue and the channel empty. However, as the traffic load increases, the probability of this case tends to zero.
the probability of collision conditioned to the fact that node $i$ transmits is:

$$p_i = 1 - \prod_{j=1}^{i-1} (1 - \tau_j)$$  \hspace{1cm} (4)

When the data packet is successfully received, the recipient waits for a short inter frame space (SIFS) time and sends a MAC layer ACK to acknowledge the correct reception of the data packet. In case the sender does not receive the ACK frame, it starts the retransmission procedure. After discarding or successfully transmitting a packet, if more packets are ready to be transmitted, the node will start the transmission procedure again. Otherwise, it waits for a new packet from the network layer.

Assume that new packets arrive to the MAC layer following a Poisson process of rate $\lambda$ and that the departure rate of the MAC queue also follows an exponential distribution. The MAC queue is modeled by an $M/M/1/K$ queue with average service time (time elapsed since a packet arrives to the head of the queue since it is released from it) equal to $E[X]$. The service time of a target node $i(E[X_i])$ is computed as shown in Eq. (5).

$$E[X_i] = (M_i - 1)(E[B]_i + E[T_c]) + E[B]_i + E[T_s]$$  \hspace{1cm} (5)

where $M_i$ is the average number of required transmission attempts per packet, $E[B]_i$ is the average slot duration, $E[T_c]$ is the duration of a successful packet transmission and $E[T_s]$ is the average duration of a collision involving node $i$. Therefore, the packet departure rate is simply $\beta = 1/E[X_i]$. The duration of a successful transmission follows Eq. (6).

$$E[T_{s,i}] = T_{data} + SIFS + T_{ack} + DIFS$$  \hspace{1cm} (6)

The reader can refer to [7] for further details on Eq. (6) or the other durations. For example, if different packet lengths are considered, $E[T_{c,i}]$ and $E[T_{s,i}]$ must be computed considering the packet with the maximum length.

### 2.2. The enhanced distributed channel access

The EDCA recommendation provides traffic differentiation by classifying each traffic flow into an AC associated to a MAC transmission queue. Each AC has its own MAC parameters and behaves independently and in parallel with the others queues. The MAC parameters of each AC are: arbitration interframe space ($AIFS$), minimum contention window ($CW_{\text{min}}$), maximum contention window ($CW_{\text{max}}$) and transmission opportunity ($TXOP$). The default values of each parameter are reported in Table 1, where we introduce the $x$ sub-index to refer to the AC $x$ used.

Similar to the DCF, before starting a transmission, the channel must be detected empty during a time called $AIFS_x$. The $AIFS_x$ value is computed from $AIFS_x = DIFS + AIFS_{nx} \cdot \sigma$, where $AIFS_{nx}$ is an integer value specific for each AC. Once the backoff instance is started, the number of backoff slots is computed in the same way than using the DCF but also with different $CW_{\text{min}}$ and $CW_{\text{max}}$ parameters for each AC, resulting in different $E[Bt_{s,i}]$ values. Then, a packet waiting to be transmitted, will wait the selected backoff slots plus the extra slots which compose the $AIFS$ value, at least $(1 + p_{jt,i} \cdot E[Bt_{s,i}])$ times, where $p_{jt,i} \cdot E[Bt_{s,i}]$ is the average number of busy slots that a node in backoff senses before it is able to transmit the packet, with $p_{jt,i}$ as the probability to observe a busy slot. Assuming that only one AC is active in each STA, $p_{jt,i}$ is computed as:

$$p_{jt,i} = 1 - \prod_{j \neq i} (1 - \tau_{j,i})$$  \hspace{1cm} (7)

Then, using EDCA, the transmission probability shown in Eq. (3) can be extended to:

$$\tau_{i,x} \approx \frac{\rho_{l,x}}{AIFS_{nx} + E[Bt_{s,i}] + 1 + (AIFS_{nx} \cdot p_{jt,i} \cdot E[Bt_{s,i}])}$$  \hspace{1cm} (8)

---

2 Several authors [10,11] have shown that the queuing time distribution follows a multi-modal distribution. The assumption taken here, should be seen as a simplification to easily model the queuing behavior.
which incorporates the different AIFS values used for each STA/AC, resulting in a lower transmission probability of those STA/AC with high AIFS values.

Once a node gets the access to the channel, it can transmit up to $N \leq \text{TXOP, MPDU}$ packets. This limit is usually expressed in time units (ms) and corresponds to the consecutive time during which a node can transmit data packets (Fig. 2), but it can also be expressed in number of frames as it is used in this paper.

The effect of the multiple packet (burst) transmission is a reduction of the time required to transmit $N$ consecutive packets obtained by decreasing the channel contention delay. Therefore, the service time of sending a burst of $N$ messages is smaller than the time to send the same number of messages individually ($X_N < N \cdot X$), with the consequent gain. To analytically model this feature, a service time-dependent queue is used to approximate the departure rate of packets. For the case of saturated flows, they are always transmitting batches of $\text{TXOP}$ packets, resulting in a packet departure rate of $\frac{\text{TXOP}}{X\text{TXOP}}$. Similarly, for the unsaturated sources, the batch size depends on the packets stored at the queue when a transmission starts. As expected, if there are less packets than the maximum bulk size, the scheduled bulk size only considers the current enqueued packets. Then, in general, we approximate the queue departure rate at state $N$ with $\beta(N) = \frac{\text{min}(N, \text{TXOP})}{X\text{TXOP}}$, where the service time, $E[X_i(N)]$, of a bulk of $N$ packets is:

$$E[X_i(N)] = (M_i - 1) (E[B_i] + E[T_{ui}]) + E[B_i] + E[T_{ui}(N)]$$

with $N \leq \text{TXOP}$ and

$$E[T_{ui}(N)] = N \cdot (T_{\text{data}} + SIFS + T_{\text{ack}}) + (N - 1) \cdot SIFS + DIFS$$

The burst transmissions reduce the contention overheads but have also negative effects for the nodes that are currently in backoff, since they must wait longer for a successful transmission to finish. Thus, the successful duration of other transmissions ($E[T_{ui}^b]$) must also consider the burst durations. However, if two nodes collide, the duration involves only the first packet.

### 2.2.1. Modeling two ACs in the access point

In this paper, it has been considered that only the AP can have two simultaneously active ACs, one used by VoIP packets and the other by the elastic traffic. To model this behavior, we approximate the virtual collision handler by a strict priority policy, where the high priority queue always gains the contention against the low one. Then, the system can be working in two situations: (a) the AP transmits high priority packets, with probability $\rho_{hp}$ (the high priority queue utilization) and (b) the AP transmits low priority packets, with probability $1 - \rho_{hp}$. Therefore, all performance metrics are computed averaging the ones obtained in the two working situations.

This approximation avoids the virtual contention between the different ACs of the same node, resulting in optimistic performance results for the VoIP traffic. However, there are very few differences when compared with the real virtual collision handler operation, as the probability that a VoIP packet gains the contention against a best-effort one is significantly higher.

### 3. Assessing the WLAN performance with VoIP and TCP traffic

After introducing the DCF/EDCA protocols and the analytical model used, the purpose of this section is twofold:

1. to show the performance of an EDCA-based WLAN with both VoIP and TCP traffic simultaneously active and
2. to validate and justify the use of the analytical model presented in the previous section.

#### 3.1. WLAN performance with mixed VoIP and TCP traffic

The NS2 (Network Simulator 2) [12] has been used to build and evaluate a single WLAN cell scenario. Two traffic types are considered: (1) a fixed number of persistent uplink and downlink TCP flows (RENO version) and (2) a variable number of G.711-based VoIP calls (which include both uplink and downlink streams) with a constant bit-rate of 64 Kbps/stream. All nodes (STAs and AP) use a

<table>
<thead>
<tr>
<th>AC</th>
<th>AIFS, (packets)</th>
<th>TXOP</th>
<th>CWmin</th>
<th>CWmax</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 (BK)</td>
<td>7</td>
<td>1</td>
<td>CWmin</td>
<td>CWmax</td>
</tr>
<tr>
<td>1 (BE)</td>
<td>3</td>
<td>1</td>
<td>CWmin</td>
<td>CWmax</td>
</tr>
<tr>
<td>2 (VI)</td>
<td>2</td>
<td>4</td>
<td>CWmin/2</td>
<td>CWmin</td>
</tr>
<tr>
<td>3 (VO)</td>
<td>2</td>
<td>2</td>
<td>CWmin/4</td>
<td>CWmin/2</td>
</tr>
</tbody>
</table>

**Table 1** Default EDCA values.

![Fig. 2. EDCA medium access scheme. in this example, station a has more priority than B.](image-url)
single transmission rate equal to 11 Mbps and all queues have a maximum length of $Q = 50$ packets.

The scenario considered is formed by a ranging number of VoIP calls (both downlink and uplink), 2 TCP downlink flows and a variable number of TCP uplink flows: 2 (scenario 2d–2u) and 4 (scenario 2d–4u).

Considering an EDCA-based WLAN using the default EDCA parameters [2], the achieved throughput by the TCP flows and VoIP calls (only the downlink VoIP streams are plotted) is shown in Fig. 3a. The TCP throughput decreases quasi-linearly as the VoIP traffic increases until the VoIP queue at the AP saturates. That means that, the VoIP queue is not able to deliver to the network all incoming VoIP packets, resulting in packet losses. Regarding the delay of the VoIP packets (Fig. 3b), it remains low and constant until the AP is near saturation, point at which, both average delay and jitter increase rapidly. As a conclusion, the maximum capacity of VoIP calls using the default EDCA parameters is equal to 9 with 4 uplink TCP flows, and equal to 10 with only 2 uplink TCP flows for the considered scenario.

The previous results show the successful EDCA traffic differentiation capabilities. However, several questions arise: are the best-effort EDCA parameters too restrictive when there is a small number of VoIP calls? Therefore, can the performance of the TCP traffic be improved? Conversely, at high traffic loads, can the VoIP calls protection be improved in order to increase the VoIP capacity? These questions are partially answered in Fig. 4a and b, which result from modifying the AIFS and TXOP parameters of the best-effort (TCP) queues. Now, the AIFS and TXOP of the best-effort queue are increased to their maximum values (15 and 8.192 ms, respectively). The increase of these two parameters allows to reduce the contention between TCP and VoIP calls (through the AIFS) and the transmission of several TCP packets in the same successful transmission attempt (through the TXOP). As shown in Fig. 4 the previous questions are positively answered.

As a counter effect, the use of longer TXOP for TCP traffic obviously impacts on the delay of the VoIP packets, as it is shown in Fig. 4b. Comparing it with Fig. 3b, both the average VoIP delay and jitter slightly increase. However, due to the longer AIFS, they remain stable until there are 10 active calls, regardless of the number of uplink TCP flows.

The main conclusion of these results is that EDCA is able to provide traffic differentiation, keeping the required QoS for VoIP calls until the AP saturates. However, the EDCA performance for both VoIP and elastic traffic can be improved by a proper tuning of their parameters based on the number and type of active traffic flows, as it is shown through the rest of the paper.

### 3.2. Model validation

An exact match between the results obtained using the analytical model and the NS2 simulations is not possible due to several reasons. The most important reason for that discrepancy is the simplification made regarding the TCP (elastic) flows. The presented analytical model considers each elastic flow as a simple saturated source without any class of congestion control mechanism (elastic flows are always trying to transmit a packet even when the network is overloaded). In those cases where the analytical model has been evaluated against a simulator that considers the elastic flows as saturated sources [6–8], quite accurate results were obtained, thus validating the correctness of the considered assumptions. Another issue of divergence is the fact that the analytical model does not consider the transmission of TCP acknowledgements (that also need to contend for the channel). Finally, the consideration of Poisson arrivals and exponentially distributed service times

![Fig. 3. WLAN performance evaluation (a) throughput and (b) VoIP Delay using the default EDCA MAC parameters (NS2 simulation results).](image-url)
make the analytical model more conservative if compared to the use of more deterministic traffic profiles of the synchronous traffic flows. To illustrate the aforementioned divergence, Fig. 5 shows the VoIP and TCP throughput obtained using the analytical model. Fig. 5a is obtained using the default EDCA parameters while Fig. 5b shows the results using the best-effort AIFS and TXOP maximum values.

Regarding the TCP flows it is shown that the TCP throughput is significantly greater than the achieved by the TCP flows using NS2 (Fig. 3a and Fig. 4a). This difference is caused by the simplifications of the analytical model related to the TCP congestion control and TCP acknowledgements. For the VoIP traffic it can be observed that although the analytical model provides worse results for the VoIP traffic compared to the ones obtained with the NS2 simulator, both show similar results in terms of number of calls supported.

As it can be seen, increasing the AIFS and TXOP of the best-effort queue results in a similar performance gain in both, simulation and analytical results. The similar tendencies are found for the elastic traffic as well as for the VoIP traffic, in the downlink and uplink flows. Thus, the results

![Fig. 4. WLAN performance evaluation (a) throughput and (b) VoIP delay using best-effort queue AIFS = 15 and TXOP = 8.192 ms (NS2 simulation results).](image)

![Fig. 5. Analytical WLAN throughput evaluation using (a) the default EDCA MAC parameters and (b) best-effort queue AIFS = 15 and TXOP = 8.192 ms.](image)
presented in this work must be considered as qualitative results only, but valid to understand and evaluate different EDCA parameters tuning algorithms.

There are other analytical models which could also be considered to model simultaneously VoIP and TCP traffic, such as [13] for the DCF and [14] for the DCF/EDCA. They model the presence and impact of TCP acknowledgements by increasing the number of TCP contenders. The same approach (based on modeling the number of contenders through a Discrete Markov Chain) was also followed in [15], where it is shown how the model presented here can be adjusted to capture the TCP behavior in WLANs. Obviously, the use of more detailed analytical models results in more accurate results (compared with real TCP traffic) but at the cost of a higher complexity and (as previously seen) without providing significantly different conclusions.

4. On tuning the MAC parameters

A MAC parameter tuning algorithm is the function \( f(.) \) that selects the future MAC parameters given the current ones and the current state of the WLAN. The function \( f(.) \) refers to each one of the tuning algorithm procedures described in Section 4.1. We consider that the MAC parameters are only changed when a new flow arrives or an already active flow departs. The \( \phi \) parameter represents the change on the system status at \( t+1 \) with respect to the current status at time \( t(\phi(t)) \).

\[
\begin{align*}
[\theta(t+1), \phi(t+1)] &= f(\theta(t), \phi(t) + \phi(t+1)) \tag{11}
\end{align*}
\]

where \( f \) results in a new set of parameters \( \theta(t+1) \), that refers to the following MAC parameters: \( CW_{\text{min}}, CW_{\text{max}}, \text{AIFS} \) and \( \text{TXOP} \), and in a new cell state \( \phi(t+1) \) representing the number of active flows and their characteristics.

A basic classification of the MAC parameters tuning algorithms can be done based on the following considerations:

**Static vs. Adaptive.** Static MAC parameters algorithms define the MAC parameters for all the A\( C \)s and maintain this configuration unchanged. These solutions suffer from inefficiency problems due to the lack of adaptability to the network changes (the EDCA recommendation is the most clear example of a static algorithm). On the contrary, adaptive algorithms [5,6] are capable of selecting the most suitable MAC parameters for each AC or for each station in order to improve the QoS provided to the active flows at any specific moment. These algorithms are the most effective ones as they can adapt the use of the network resources depending on the network status.

**Measurement vs. Model based.** Another classification of the proposals is based on the moment in which the parameters are updated. In measurement-based algorithms [16–18] the computation of the new MAC parameters is made when measured metrics, like the load of the AC queues, the number of contending stations or the collision rate, change. In these algorithms, the time is divided in windows of length \( \Delta t \), where the performance of the network is evaluated. On the other hand, model-based algorithms update the MAC parameters each time a new flow arrives or leaves the system [19,5]. Both approaches can be also merged. An example is the work presented in [20] where a call admission control is combined with a monitoring approach. Each time a new flow arrives, the AP monitors the network status during a certain time in order to decide whether to accept or reject the flow.

**Centralized vs. Distributed.** There is also another type of classification that depends on the location of the MAC parameters computation. It is possible to decide the most suitable MAC parameters in a centralized way (at the AP) [19,21] or in a distributed way (at each one of the stations) [22,23]. Usually, distributed algorithms are measurement based while centralized algorithms support both types of algorithms as the AP has more information of the network state.

The number of MAC parameters that are changed is also another differentiating factor among the various MAC parameters tuning algorithms. Most of the proposals only consider to change the \( CW_{\text{min}} \) and \( CW_{\text{max}} \) parameters [3,4,8,18,24–28], since the \( CW_{\text{min}} \) was assumed to be the only parameter that can be tuned in the DCF. Other works only vary the \( \text{AIFS} \) [29,30] or the \( \text{TXOP} \) [5,23,31] value that is usually adjusted based on the number of flows contend- ing in the same AC. However, there are algorithms that take all the MAC parameters into account in order to make a fine tuning that leads to a more accurate but complex solution [5,6].

Additionally, the algorithms can be iterative or non-iterative. In iterative algorithms [4,6] it is easier to find near optimal solutions as the most suitable configuration can be selected, although most of them cannot work in real-time due to the time required to compute the solution. However, heuristic solutions, able to work in real-time, can be derived from the results obtained by iterative algorithms. Non-iterative algorithms [5,32,3] compute the MAC parameters for a given network configuration based on different metrics (number of flows, load, etc.). These solutions are normally not as accurate as the ones provided by iterative algorithms but they can work in real-time in commercial APs and network cards.

**Nomenclature used.** Two subscripts are used: \( e \) for the elastic flows and \( v \) for the VoIP flows. If none of them is used it is considered the same value for both types of flows. To differentiate between uplink and downlink, the \( u \) and \( d \) subscripts are respectively used. If none of these are specified, the value applies both for uplink and downlink flows. Moreover, the nomenclature considered in the algorithms is:

1. \( \xi \) are the bandwidth requirements of all the defined traffic classes.
2. \( \zeta \) is a boolean parameter that determines if the bandwidth requirements of all the different flows are met.
3. \( \text{EstimateHotspotPerformance} \) is a function that returns the performance of the system (bandwidth, average delay, average packet loss, etc.) for each flow. The performance can be computed by using...
an analytical model or by estimating it from measurements.

### 4.1. MAC parameter tuning algorithms

In this section, the algorithms considered in this work are presented. Only two ACs have been considered, the voice (v) and best-effort (e) queues. The evaluated tuning algorithms have been modified in order to compute in a centralized manner the MAC parameters for each one of the AC queues (\( CW_{\text{min,v}} \), \( CW_{\text{max,v}} \), \( TXOP_{\text{v}} \) and \( AIFS_{\text{v}} \)). Thus, all of them can be categorized as model-based and centralized tuning algorithms.

The two first algorithms considered are representatives of the several solutions proposed so far based on the tuning of binary exponential backoff (BEB) parameters, while the other two are more sophisticated algorithms that consider the tuning of more than one MAC parameter:

- **\( AC_w \) (Adaptive \( CW_{\text{min}} \))**: It is based on the solution proposed by Serrano et al. [3] and is the simplest algorithm considered. It uses a fixed configuration for all the parameters except for the contention window of non-sensitive flows.
- **\( IC_w \) (Iterative \( CW_{\text{min}} \))**: This algorithm is based on the one presented by Siris et al. in [4] where an iterative search of the \( CW_{\text{min}} \) MAC parameter is provided.
- **\( AC_{\text{w,TS}} \) (Adaptation of \( CW_{\text{min}} \) and \( TXOP \))**: This algorithm is a modification of the one provided by Freitag et al. in [5] where a non-iterative algorithm to adjust the \( TXOP \) and the \( CW_{\text{min}} \) is provided.
- **\( IC_{\text{w,TS}}A \) (Iterative \( CW_{\text{min}} \), \( TXOP \) and \( AIFS \) search)**: It is the iterative algorithm presented by Bellalta et al. in [6] that, apart from considering the \( CW_{\text{min}} \) and \( TXOP \) parameters, it additionally takes into account the \( AIFS \) parameter.

All the algorithms have also been compared with the regular DCF and the EDCA recommendation. Table 2 shows the fixed MAC parameters and the range of the variable ones for each configuration and algorithm.

It is assumed that the MAC parameter tuning algorithms work jointly with a CAC (Call Admission Control) also specified by the IEEE 802.11e [2]. When a new flow arrives to the system, it sends an ADD traffic stream (ADDTS) request to the CAC including its own TSPEC (Traffic flow SPECifications [2]). Upon the reception of an ADDTS message, the CAC, based on the TSPEC information and the network state, decides if the new flow can be accepted by tuning the MAC parameters (following the algorithm specified) if necessary. Only if the bandwidth requirements of all the active flows are met the new flow is accepted. Otherwise, the flow is rejected and the MAC parameters are reset to the previous values. Once a flow leaves the system, it is assumed that it also notifies the CAC by sending a DELeete traffic stream (DELTs) [2], in this case the CAC also computes the new working MAC parameters following the specified algorithm.

#### 4.1.1. Algorithm 1. Adaptive \( CW_{\text{min}} – AC_w \)

The first algorithm is based on the work presented by Serrano et al. [3] where a quasi static configuration for the EDCA parameters is proposed under the goal to protect the VoIP flows from the elastic ones. They propose a conservative configuration, which gives a high priority to access the channel to the VoIP packets, setting the \( TXOP_v \) to 10 packets, \( AIFS_v = 2 \) (standard value) and \( CW_{\text{min,v}} = CW_{\text{max,v}} = 32 \) slots (see Table 2). Conversely, the algorithm is very restrictive with the elastic flows, setting the \( TXOP_e \) equal to 1 packet and, specially, with \( AIFS_e \) set to 15 slots. The \( CW_{\text{min,e}} \) and \( CW_{\text{max,e}} \) are variable and depend on the number of elastic contending stations (\( n_e \)), as can be seen in Algorithm 1, Lines 5–6. Notice that there are not any differences between the uplink and downlink configurations.

**Algorithm 1. Adaptive \( CW_{\text{min}} – AC_w \)**

1: Initialize the MAC parameters
2: loop
3: Wait for a new flow request
4: Store \( (CW_{\text{min,e}}, CW_{\text{max,e}}) \)
5: Set \( CW_{\text{min,e}} = \min(1024, n_e \times 16) \)
6: Set \( CW_{\text{max,e}} = \min(1024, 32 \times CW_{\text{min,e}}) \)
7: \( \zeta = \text{EstimateHotspotPerformance} \) \((\phi, \theta, \xi)\)
8: if \( (\zeta = 1) \) then
9: Accept incoming request
10: else
11: Reject incoming request.
12: Restore \( (CW_{\text{min,e}}, CW_{\text{max,e}}) \)
13: end if
14: end loop

This configuration allows a certain grade of adaptation to the Hotspot state. In presence of VoIP flows, the high \( TXOP_v \) and \( AIFS_v \) values provide a higher protection for the VoIP flows regarding the elastic ones. Moreover, the variable tuning of the BEB parameters of the elastic flows provides a low contention against the VoIP flows as the number of elastic flows increases.

#### 4.1.2. Algorithm 2. Iterative \( CW_{\text{min,e}} \) search – \( IC_w \)

This algorithm is based on the algorithm presented by Siris et al. [4]. The algorithm searches iteratively for an optimal configuration of the \( CW_{\text{min,e}} \) parameter (the other parameters take the EDCA default values) with the goal to maximize the Hotspot throughput.

Additionally, it has been adapted to the case where elastic and VoIP flows are simultaneously present in the system.

The pseudo-code is shown in Algorithm 2. The \( CW_{\text{min,e}} \) adaptation (Algorithm 2, Line 8) is based on a simple iterative increment of the contention window parameter, governed by the \( \Delta_SF \) (step increments) and \( SF \) (initial value) parameters. The values used are set by the authors in order to consider all the range of possible values for the \( CW_{\text{min,e}} \) at fine-grained steps. The algorithm is executed until the \( \zeta \) (bandwidth requirements) of sensitive flows is achieved or until the maximum aggregated throughput of the elastic

---

\( \Delta_SF \) The original algorithm measures the system status in temporal windows to adjust the MAC parameters.
flows $S_e$ is reached. In any case the maximum $CW_{min,e}$ value is 1024. The range and the value of the other EDCA parameters are shown in Table 2.

Algorithm 2. Iterative $CW_{min,e}$ search – ICw

1: Initialize the MAC parameters
2: Initialize algorithm parameters:
   $W = 320$, $\Delta S = 0.1$, $SF = \Delta S$
3: Initialize algorithm variables
   $\zeta_{iter-1} = 0$, $S_e_{iter-1} = 0$
4: loop
5:   Wait for a new flow request
6:   Store ($CW_{min,e}$)
7:   while ($\Delta S < W/100$) do
8:     $CW_{min,e} = W \times SF$
9:     $[\zeta_{iter}, S_e_{iter}] = \text{EstimateHotspotPerformance}(\phi, 0, \zeta)$
10:    if ($\zeta_{iter-1} = 1$) then
11:     Restore last parameter configuration:
12:        $\zeta(\text{iter} - 1)$
13:     break
14:     end if
15:     SF = SF $+ \Delta S$
16:     $\zeta_{iter-1} = \zeta_{iter}$
17:     $S_e_{iter-1} = S_e_{iter}$
18: end while
19: end loop
20: if ($\zeta_{iter} = 1$) then
21:     Accept incoming request.
22: else
23:     Reject incoming request
24: end if
25: Restore ($CW_{min,e}$)
26: end loop

When there are only elastic flows, an increment of the $CW_{min,e}$ parameter reduces the channel collisions, resulting in some cases in a higher throughput. With respect to the VoIP calls, a high $CW_{min,e}$ increases the protection of rigid flows since the channel contention is reduced.

4.1.3. Algorithm 3. Adaptation of $CW_{min}$ and TXOP – ACwTx

In this algorithm the $CW_{min}$ and TXOP parameters of all traffic classes are adjusted depending on the throughput required and the number of stations of each class ($n$). This algorithm is based on the one presented by Freitag et al. in [5] and it is one of the most complete solutions to tune the MAC parameters as it considers almost all of them. In the implementation used in this work, the main difference is that the bandwidth required by each flow is used instead of the instantaneous traffic load of the queues, as they propose a measurement based (continuous monitoring of the transmission queue) algorithm.

The procedure is shown in Algorithm 3. It has two clear parts: (a) the TXOP adjustment and (b) the $CW_{min}$ adjustment. First, the TXOP of the class with lowest bandwidth ($B_{min}$) requirements is set to 1 and the TXOP of the class with highest bandwidth requirements ($B_{min}$) is set depending on the relation between the bandwidth requirements of the two classes ($B_{min}/B_{min}$) (Algorithm 3, Line 8). Additionally, the TXOP in the downlink (TXOP$_d$) value is also increased by the number of active flows in the downlink in order to mitigate the unfairness problem between uplink and downlink flows (Algorithm 3, Lines 10 and 11). The $CW_{min}$ parameter is set depending on the number of contending stations of each class. Each class adjusts the $CW_{min}$ value independently of the others (Algorithm 3, Lines 12–22). Basically, the $CW_{min,a}$ of class $a$ is doubled if the number of stations in that class exceeds the current $CW_{min,a}$ value or it is divided by two if the number of stations is smaller than half of the current $CW_{min}$. However, the initial $CW_{min,b}$ of class $b$, with lower priority than class $a$ is always set to the current $CW_{min,a}$ value.

In Algorithm 3 a detailed specification of the procedure when only two classes (best-effort and voice) are considered is shown. The default parameters and the range values of the variable ones are summarized in Table 2.

Algorithm 3. Adaptation of $CW_{min}$ and TXOP – ACwTx

1: Initialize the MAC parameters
2: TXOP$_{e,u} = 1$
3: TXOP$_{v,u} = 1$
4: loop
5: Wait for a new flow request
6: Store (TXOP$_{e,u}$, TXOP$_{v,u}$, $CW_{min,e}$, $CW_{min,v}$)
7: if ($n_v > 0$) then
8:     TXOP$_{v,u} = [B_v/B_{min}]$
9: end if
10: TXOP$_{v,d} = TXOP_{v,u} \times n_v$
11: TXOP$_{e,d} = TXOP_{e,u} \times n_e$
12: if ($n_v > CW_{min,v}$) then
13:     $CW_{min,v} = 2 \times CW_{min,v}$
14: else if ($n_v < CW_{min,v}$) then
15:     $CW_{min,v} = CW_{min,v}/2$
16: end if
17: $CW_{min,e} = CW_{min,v}$
18: if ($n_e > CW_{min,e}$) then
19:     $CW_{min,e} = 2 \times CW_{min,e}$
20: else if ($n_e < CW_{min,e}/2$) then
21:     $CW_{min,e} = CW_{min,e}/2$
22: end if
23: $[\zeta] = \text{EstimateHotspotPerformance}(\phi, 0, \zeta)$
24: if ($\zeta = 1$) then
25:     Accept Flow
26: else
27:     Reject Flow
28: Restore (TXOP$_{e,u}$, TXOP$_{v,u}$, $CW_{min,e}$, $CW_{min,v}$)
29: end if
30: end loop

This algorithm is able to adjust the TXOP value proportionally to the bandwidth requirements of each class in relation with the others, which will result in a more fair channel occupation. Furthermore, it controls the collision
probability by tuning the \( CW_{\text{min}} \) value based on the number of contending stations of each class.

4.1.4. Algorithm 4. Iterative \( CW_{\text{min}} \), TXOP and AIFS search – \( ICW_{\text{TXA}} \)

This algorithm, that was presented by Bellalta et al. in [6], is based on a set of heuristics observed from the optimal EDCA parameters selected during a max–min optimization process. It was designed to increase the performance of the elastic flows, specially when the number of VoIP flows is low, by initializing the EDCA parameters in order to maximize the elastic throughput. Once the number of VoIP flows increases the algorithm tunes the MAC parameters trying to increase their protection, assuring that their bandwidth requirements are achieved. To perform this operation, the algorithm searches iteratively the most suitable parameters given the WLAN status.

Regarding the considered EDCA parameters, in the case of the VoIP flows the only parameter tuned is the TXOP, while for the elastic flows the three EDCA parameters are considered \( (CW_{\text{min},e}, TXOP_{e} \) and \( AI\text{FS}N_{e} \). All the EDCA parameters considered are initialized with the goal to increase the elastic throughput \( (TXOP_{e} = 1, TXOP_{u} = 10, AI\text{FS}N_{e} = 3, CW_{\text{min},e} = 32) \). Then, they are sequentially changed (only one change at each iteration) as shown in Algorithm 4 (see Table 2 for the parameter range) to increase the VoIP protection. In order to mitigate the unfairness problem the following settings are considered: (i) the AI\text{FS}N value is decreased in one unit for the downlink, (ii) the TXOP in the downlink is set to \( n_{e,d} \) times the TXOP in the uplink \( (\text{Algorithm 4, Line 20}) \), where \( n_{e,d} \) is the number of downlink elastic active flows, and (iii) the \( CW_{\text{min},u} \) is adjusted depending on the number of elastic active flows in the uplink \( (n_{e,u}) \) \( (\text{Algorithm 4, Line 5}) \).

**Algorithm 4. Iterative \( CW_{\text{min}}, \) TXOP and AIFS search – \( ICW_{\text{TXA}} \)**

1: Initialize the MAC Parameters
2: loop
3: Wait for a new flow request
4: Store \( (CW_{\text{min},e}, TXOP_{e}, AI\text{FS}N_{e}, TXOP_{u}) \)
5: \( CW_{\text{min},u} = n_{e,u} \times 32 \)
6: while \( (\zeta = 0 \& \& \& TXOP_{e,u} < 1) \) do
7: \( TXOP_{v,u} = TXOP_{v,u} + 1 \)
8: \( TXOP_{v,d} = TXOP_{v,u} \times n_{e} \)
9: \( [\zeta] = \text{EstimateHotspotPerformance} (\phi, \theta, \zeta) \)
10: if \( (\zeta = 1) \) then
11: break
12: end if
13: \( AI\text{FS}N_{v,u} = AI\text{FS}N_{v,u} - 1 \)
14: \( AI\text{FS}N_{v,d} = AI\text{FS}N_{v,u} - 1 \)
15: \( [\zeta] = \text{EstimateHotspotPerformance} (\phi, \theta, \zeta) \)
16: if \( (\zeta = 1) \) then
17: break
18: end if
19: \( TXOP_{v,u} = 1 \)
20: \( TXOP_{v,d} = TXOP_{v,u} \times n_{e,d} \)
21: \( [\zeta] = \text{EstimateHotspotPerformance} (\phi, \theta, \zeta) \)
22: if \( (\zeta = 1) \) then
23: break
24: end if
25: end while
26: while \( (CW_{\text{min},e,d} < 1024) \) do
27: \( CW_{\text{min},e,d} = 2 \times CW_{\text{min},e,d} \)
28: \( [\zeta] = \text{EstimateHotspotPerformance} (\phi, \theta, \zeta) \)
29: if \( (\zeta = 1) \) then
30: break
31: end if
32: end while
33: if \( (\zeta = 1) \) then
34: Accept Flow
35: else
36: Reject Flow
37: Restore \( (CW_{\text{min},e}, TXOP_{v,e}, AI\text{FS}N_{e}, TXOP_{e}) \)
38: end if
39: end loop

This algorithm tries to tune almost all the EDCA parameters in order to find the most suitable configuration which provides the bandwidth requirements of the VoIP flows but also maximizes the throughput achieved by elastic flows. Notice that the algorithm does not need to start the search from the beginning each time a new flow arrives or leaves the system, the initial step can be the last working configuration in order to reduce the computational effort.

5. Results

A Hotspot scenario with a single AP providing service (e.g., access to Internet) to a group of stations (STAs) has been considered (see Fig. 6). STAs and the AP use the DSSS PHY specifications in the 2.4 GHz band [1]. The general system parameters are reported in Table 3. Ideal channel conditions are assumed, i.e., no packet is lost due to propagation impairments or hidden terminal problems. Furthermore, it has been considered that all nodes transmit at the same data rate (11 Mbps), which remains constant during all the simulation time.

Each active STA carries a single traffic flow, which can be the uplink part of a VoIP call (rigid flow, unsaturated) or caused by a file transfer from, for example, a P2P application (elastic flow, saturated). The VoIP and elastic generated packets are mapped to the voice (v) and best-effort (e) queues, respectively. A STA becomes active when a new flow starts (arrives). The flow arrival rates of both VoIP and elastic flows follow a Poisson process and both the duration of VoIP calls and the total information to transfer for the elastic flows are exponentially distributed. Moreover, notice that the duration of elastic flows depends on the instantaneous bandwidth assigned (which changes dynamically with the system status). In Table 4 the flow characteristics of each traffic type are shown.

5.1. Flow-level EDCA-based WLAN simulator

A simulator based on the component oriented simulation toolkit (COST) libraries [33] has been used to imple-
ment and evaluate the described Hotspot scenario. The simulator implements both the flow-level (arrivals, flow durations and departures) and the packet-level (packet transmissions, collision, backoff algorithm, etc.) behavior of a WLAN cell. However, to reduce the high computation time required for each simulation due to the different time scales of the packet-level ($l$) and the flow-level (seconds), the packet-level performance is estimated using the MAC model presented in Section 2. In Fig. 7 the structure of the simulator is shown. It has the following main blocks: the traffic flow generators, where each instance is assigned to a single STA, the call admission control, which manages both the new flow requests and the selection of the most suitable EDCA parameters using the EDCA MAC parameter tuning block. Results provided by the simulator include blocking probability, average number of active VoIP calls, average bandwidth used by each elastic flow, average elastic flow duration, etc. All these metrics are obtained by sampling the simulator variables periodically (each 0.2 s) or by post-processing the collected data depending on the metric measured.

5.2. Effect of increasing the number of elastic STAs

In this first scenario, there are $n_v = 15$ VoIP STAs and $n_{e,d} = 4$ elastic downlink STAs while the number of elastic uplink STAs ($n_{e,u}$) ranges from 0 to 20. It is important to notice that these values refer to the number of STAs holding each flow type, regardless of their activity, which depends on the flow characteristics (flow arrivals) and the system response (accepted_blocked, bandwidth assigned, etc.). The goal of this scenario is to evaluate the performance of the different algorithms when the number of stations carrying elastic flows increases. As a reference, based on the simulation parameters shown in Table 4, the expected number of simultaneously active VoIP calls is around 7.

In Fig. 8a, the blocking probability of VoIP flows is shown. Using the regular DCF the blocking probability increases rapidly to 1. A noticeable improvement is found if the EDCA is used, but even with $n_{e,u} = 2$ elastic uplink flows the blocking probability exceeds the 1% that is the typical maximum value recommended for a cellular
Similar results are obtained with the $AC_w$ and $IC_w$ algorithms that are both based on the BEB adaptation. The $AC_w$ performs slightly better mainly due to the difference of the $AIFSN$ of the elastic flows that is set to 15 while in the $IC_w$ is set to 3. This difference allows the $AC_w$ algorithm to increase the protection for the VoIP calls with respect to $IC_w$. The $AC_wT_x$ solution works even worse compared with the previous two for the scenario considered. The $AC_wT_x$ solution sets the $CW_{min}$ of the elastic flows to a fixed value equal to 32 (as the number of elastic flows is smaller than this value) while in the previous algorithms the value of the $CW_{min}$ of elastic flows increases, either proportionally to the number of flows ($AC_w$) or because the requirements of the VoIP calls cannot be achieved ($IC_w$). Finally, the $IC_wT_xA_i$ algorithm achieves a higher VoIP protection providing a lower blocking probability in all the cases, the maximum VoIP blocking probability obtained in this scenario is 1.16% when the number of P2P elastic flows is equal to 12. Fig. 8b shows the average number of active VoIP calls showing, as expected, a direct relation with the blocking probability.

The aggregated elastic throughput is shown in Fig. 9. The DCF achieves higher throughput at the expense of rejecting more VoIP calls. Comparable results are obtained among the others, EDCA, $AC_w$, $IC_w$ and $AC_wT_x$ algorithms in which the throughput increases as more VoIP calls are blocked. However, the $IC_wT_xA_i$ algorithm provides higher throughput as it adapts better to the network conditions, maximizing the throughput of the elastic flows but maintaining a good blocking probability for the VoIP calls. The highest throughput is obtained when the number of elastic flows is low since it is not required to give higher prioritization to the VoIP calls.

To evaluate the uplink/downlink fairness, Fig. 10a and b show the bandwidth achieved by elastic uplink and downlink flows. The DCF maximizes the elastic uplink throughput while the elastic downlink becomes more affected by the unfairness problem. With the EDCA, $AC_w$ and $IC_w$ defined parameters the results are quite similar as the only difference applied is to decrease in one unit the $AIFSN$ in the downlink. The $AC_wT_x$ and the $IC_wT_xA_i$ algorithms allow to mitigate the downlink unfairness by adjusting the $TXOP$ depending on the number of flows in each direction apart from decreasing the $AIFSN$. Once again, the $IC_wT_xA_i$ algorithm obtains a noticeable increase in the downlink throughput by increasing the $CW_{min}$ of the elastic uplink flows based on the number of elastic uplink active flows. Fig. 10c and d show the

### Table 4

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Elastic</th>
<th>VoIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application/Codec</td>
<td>File sharing G.711</td>
<td></td>
</tr>
<tr>
<td>Required bandwidth</td>
<td>&gt;10 Kbps</td>
<td>80 Kbps</td>
</tr>
<tr>
<td>Packet Length</td>
<td>1500 bytes</td>
<td>200 bytes</td>
</tr>
<tr>
<td>Av. flow length</td>
<td>10 Mbytes</td>
<td>240 s</td>
</tr>
<tr>
<td>Av. Inter-arrival time</td>
<td>100 s</td>
<td>300 s</td>
</tr>
</tbody>
</table>

Fig. 7. Flow-level simulator architecture.

Fig. 8. (a) Blocking probability of VoIP flows (b) average VoIP active calls.
duration of the elastic uplink and downlink flows. It can be seen how the duration depends on the total bandwidth achieved by the elastic flows (i.e., less bandwidth leads to higher flow duration), but also depends on the number of the elastic flows since the total bandwidth is shared among them. The $ACw$ and the $ICwTxAi$, thanks to the downlink parameters adaptation, minimize the duration of the flows in the downlink, which leads to a lower congestion in the system. Therefore, in order to mitigate the downlink unfairness additional mechanisms, apart from decreasing by one unit the $AIFSN$, should be considered in order to increase the downlink throughput to values similar to the ones obtained for the elastic uplink throughput.

5.3. Effect of increasing the VoIP call interarrival time

In the second scenario, there are $n_v = 15$ VoIP STAs, $n_{s.d} = 4$ elastic downlink STAs and $n_{s.u} = 15$ elastic uplink STAs. The VoIP offered load is changed by varying the call interarrival time.

---

**Fig. 9. Aggregated elastic bandwidth.**

**Fig. 10.** Throughput of elastic (a) uplink flows (b) downlink flows and average elastic (c) uplink and (d) downlink flow duration.
interarrival time, which ranges from 200 to 1000 s. The goal of this scenario is to evaluate the flexibility of the tuning algorithms to increase the elastic throughput as the VoIP traffic is reduced.

The results obtained for the VoIP flows are shown in Fig. 11. It can be seen that by changing the MAC parameters of the AC queues an important improvement can be achieved: while the blocking probability obtained with DCF is near 1, the other algorithms reduce this value considerably (Fig. 11a). As seen in the first scenario, the ICwTxAi algorithm allows to maintain a blocking probability of VoIP flows below the 1% threshold for interarrival values greater than 300 s. The number of average active calls (Fig. 11b), as expected, decreases as the interarrival time of the VoIP calls increases.

Fig. 12 depicts the aggregated bandwidth of elastic flows. DCF provides the highest elastic bandwidth as a high number (nearly all) of the VoIP calls are blocked. On the opposite, the ICwTxAi also provides the highest elastic throughput with respect to the other algorithms but maintaining the lowest VoIP blocking probability. It shows a good adaptation when the VoIP traffic load decreases, allowing the elastic flows to increase the bandwidth used. Notice that, when the load of the VoIP traffic is low, the throughput achieved by algorithm ICwTxAi is similar to the one obtained using DCF but with very low VoIP blocking probability. Observe also the upwards trend of the ICwTxAi algorithm that adapts better to the network conditions than the others. The rest of the algorithms cannot maximize the elastic throughput as the MAC parameters of these flows are only tuned to give a higher priority to the VoIP calls, leading to an unnecessary waste of available resources that can be used by the elastic traffic.

Fig. 13a and b show the uplink and downlink bandwidth achieved by elastic flows. The unfairness problem that clearly appears in the DCF case is reduced by using the ACwTx and ICwTxAi algorithms, while the algorithms that only apply a change in the AIFS for the downlink (ICw and ACw) cannot cope with this problem. In Fig. 13c and d the duration of the elastic uplink and downlink flows is shown, which is directly related to the elastic downlink throughput obtained.

6. Concluding remarks

Adaptively tuning the MAC parameters allows a dynamic traffic differentiation and fairness mitigation, increasing the priority (and thus the protection) of real-time flows. At the same time, it can be used to maximize the aggregated throughput achieved in the network by the different flows, including the elastic non-priority ones.

Four different algorithms have been evaluated in a flow-level approach. Simulation results show that all the algorithms considered increase the network performance when compared with the DCF and EDCA specifications. However, the most effective algorithms are those which
adapt better to the changes on the WLAN state (defined as the instantaneous mix of active flows) and thus, are able to find the most suitable allocation of the channel resources in each situation.

A basic issue pointed out is the importance of the MAC tuning algorithm design, which should take into consideration that the system state is dynamic and therefore, it must try to work always at optimal points, regardless of the random changes in the WLAN traffic. This can be achieved by tuning the EDCA parameters of all traffic types, specially focusing on those associated with the regulation of the best-effort (elastic) traffic.

References

[8] B. Bellalta, M. Meo, M. Oliver, A BEB-based admission control for VoIP calls in WLAN with coexisting elastic TCP flows, ITC/IEEE NEW2AN06 (LNCS), St. Pterersburg, Russia, June 2006.