Wireless Netw (2007) 13:335–349
DOI 10.1007/s11276-006-7529-7

QoS Mechanisms for the MAC Protocol of IEEE 802.11 WLANs*

José R. Gallardo* · Paúl Medina · Weihua Zhuang

Published online: 3 July 2006

Abstract There are two essential ingredients in order for any telecommunications system to be able to provide Quality-of-Service (QoS) guarantees: connection admission control (CAC) and service differentiation. In wireless local area networks (WLANs), it is essential to carry out these functions at the MAC level. The original version of IEEE 802.11 medium access control (MAC) protocol for WLANs does not include either function. The IEEE 802.11e draft standard includes new features to facilitate and promote the provision of QoS guarantees, but no specific mechanisms are defined in the protocol to avoid over saturating the medium (via CAC) or to decide how to assign the available resources (via service differentiation through scheduling). This paper introduces specific mechanisms for both admission control and service differentiation into the IEEE 802.11 MAC protocol. The main contributions of this work are a novel CAC algorithm for leaky-bucket constrained traffic streams, an original frame scheduling mechanism referred to as DM-SCFQ, and a simulation study of the performance of a WLAN including these features.

Keywords Wireless local area networks · Medium access control · Quality of Service · Connection admission control · Packet scheduling · Leaky-bucket constrained traffic

1. Introduction

IEEE 802.11-based wireless local area networks (WLANs) have been widely deployed in home and business environments. The growing interest in new applications over WLANs, such as voice, video and multimedia streaming, also brings unique and challenging quality-of-service (QoS) requirements. The two main ingredients for any telecommunications system to be able to provide QoS guarantees are connection admission control (CAC) to avoid over saturating the medium, and service differentiation to give each user what it needs and when it needs it.

So far, the popular approaches to provide multimedia users with QoS guarantees, proposed by the Internet Engineering Task Force (IETF), are IntServ [4], and DiffServ [3], along with all the underlying technologies and protocols. These resource-reservation and service-differentiation mechanisms are implemented at the network or Internet Protocol (IP) layer. This means that mobile terminals in a WLAN will not be able to receive the treatment they deserve if there is no service differentiation when they compete for the wireless medium at the medium access control (MAC) level.

The MAC protocol of the original IEEE 802.11 standard contains an optional mechanism for providing real-time applications with a proper QoS service. This mechanism, known as Point Coordination Function (PCF), allows real-time applications to periodically transmit contention-free frames during certain intervals, called Contention-Free Periods (CFPs). The provided guarantee is that at least one frame of one of the registered stations will be sent during

*This work has been partly funded by the Mexican Science and Technology Council (CONACYT) through grant 38833-A.

J.R. Gallardo (✉) · P. Medina
Electronics and Telecommunications Department, CICESE Research Center Km. 107 Carretera Tijuana-Ensenada, Ensenada, Baja California 22860, México
e-mail: jgallard@cicese.mx

W. Zhuang
Department of Electrical and Computer Engineering, University of Waterloo 200 University Avenue West, Waterloo, Ontario N2L 3G1, Canada

each CFP. This is not enough in most situations since not all real-time applications have the same QoS needs in practice. There are some proposed extensions to this MAC protocol under discussion, documented in the IEEE 802.11e draft standard. The new features allow for traffic streams to indicate their characteristics and needs through what is called TSPEC, which in turn can be used by the resource allocator for admission control or the assignment of transmission opportunities (TXOPs). It is also possible for the contention-free periods to start anywhere within the transmission superframe, as needed to satisfy QoS criteria of delay-sensitive applications. However, the important issues of how to locally limit the amount of traffic that competes for the wireless medium and how to decide which backlogged real-time connection to service first and which next remain open.

The main goal of this research work is to identify specific admission control and service differentiation (scheduling) mechanisms so that IEEE 802.11 WLANs can effectively offer QoS guarantees to their users. The remainder of this paper is organized as follows. Section 2 briefly reviews the IEEE 802.11 and 802.11e MAC protocols. Section 3 proposes the Deficit-based Modified Self-Clocked Fair Queueing (DM-SCFQ) algorithm for packet scheduling. Details of the proposed CAC algorithm are presented in Section 4. Performance of the proposed QoS support mechanisms are evaluated in Section 5 via computer simulations, followed by concluding remarks in Section 6.

2. IEEE 802.11 and 802.11e MAC protocols

The IEEE 802.11 MAC protocol [5, 9] defines two transmission modes for data packets: the CSMA/CA-based Distributed Coordination Function (DCF), and the contention-free Point Coordination Function (PCF), where there is a Point Coordinator that controls all transmissions based on a polling mechanism. The point coordinator usually resides in the Access Point (AP). The DCF and PCF modes are time multiplexed in a superframe, which is formed by a PCF contention-free period followed by a DCF contention period. The boundaries between contention-free periods and contention periods are marked by beacons transmitted by the access point.

During a contention period, all stations compete with each other to gain access to the medium. All stations are equally likely to get to transmit a frame since there is no mechanism in the protocol to give some stations more rights than others.

During the contention-free period, on the other hand, the point coordinator in the access point polls terminals in the polling list and accepts their response frames. Each terminal that wants the PCF to schedule its transmissions needs to submit a request to the access point during a contention period. However, the method used to select the stations to be polled does not take into account the stations' needs, and there is no guarantee of a frame delivery within a certain timeframe. So, the MAC protocol defined in the IEEE 802.11 standard cannot fulfill the QoS requirements of multimedia applications.

Since the PCF has limited capability to deliver QoS and has not been widely implemented, the IEEE 802.11 Task Group E has undertaken the definition of a new standard, namely IEEE 802.11e [8, 10, 12], in which a single coordination function is introduced, with three access schemes: Enhanced DCF, Controlled Access, and Controlled Contention. The new coordination function is known as Hybrid Coordination Function (HCF) because there is no clear distinction now between what was previously known as contention period and contention-free period. The HCF introduces a Hybrid Coordinator, which extends the notion of point coordinator in the sense that it can gain access to the channel during the contention periods to create additional contention-free periods as needed to satisfy QoS criteria.

Enhanced DCF is still a contention scheme, but it has now the capacity to give some stations a higher priority than others by manipulating both the time that a station senses the medium to decide if it is free (called AIFS) and the backoff time. The AIFS period is shorter for higher-priority stations allowing them to win access to the medium over the lower-priority ones. Likewise, even though the backoff time is selected randomly, the set of possible values for a higher-priority station includes smaller values than those allowed for a lower-priority one, establishing again an advantage for the former.

Controlled access, on the other hand, is a polling mechanism that takes into account the TSPECs of the different traffic streams when admitting a new session into the polling list and when deciding how to assign transmission opportunities.

Finally, controlled contention is used when the hybrid coordinator decides that it has some extra transmission opportunities available for distribution. It sends a message to a specific set of stations inviting them to place new resource requests.

Even though IEEE 802.11e includes several new features that facilitate and promote the provision of QoS guarantees to the users, there are no specific mechanisms in the protocol to avoid saturating the medium with excessive traffic or for the hybrid coordinator to decide how to assign the available resources. It is here that our proposed schemes find their usefulness.

3. DM-SCFQ scheduling mechanism

Our proposed QoS support schemes take advantage of the new features introduced in the IEEE 802.11e MAC protocol.

The objective is to provide QoS guarantees to those sessions included in the polling list (i.e., registered to receive contention-free transmission services), in the controlled access scheme.

The Generalized Processor Sharing (GPS) algorithm and its practical realization Weighted Fair Queueing (WFQ) are the most important packet scheduling disciplines nowadays due to their unsurpassed fairness and predictability [16,17]. This is the reason why several telecommunications standards have adopted WFQ as part of their QoS mechanisms. However, it is not practical to implement WFQ in a WLAN environment due to the fact that the packet queues and the medium controller are not co-located. In other words, the packets waiting to be serviced are queued in the mobile stations, while the hybrid coordinator is located in the access point. It is necessary for every mobile station to somehow inform the hybrid coordinator of how many packets it has in its queue, and of the arrival time and size of each packet. Fortunately, there are alternatives to WFQ that also work adequately and that do not need as much information. In this research, we propose Deficit-based Modified SCFQ, which combines the ideas embedded in Self-Clocked Fair Queueing (SCFQ) [6], with those in Distributed Deficit Round Robin (DDRR) [18]. DM-SCFQ is similar to DDRR in the sense that it first allows each traffic source to transmit a packet and then it lets it wait long enough to "pay" for the transmission opportunity already given before allowing it to transmit again; i.e., it is based on "deficit". DM-SCFQ is also similar to SCFQ in that it relies on a self-generated virtual time.

In the DM-SCFQ scheduling, the same traffic source is polled repeatedly until all the frames corresponding to a single packet are transmitted; the transmission of all the frames corresponding to the same packet counts as a single transmission opportunity. To decide which traffic source will be allowed to transmit next, a service tag is computed for each of the sources at the end of a transmission opportunity. Since, as mentioned above, we cannot have information related to the packets waiting in distant queues (at the mobile stations) without incurring a high overhead cost, we compute the service tag for deciding the transmission of the *i*-th packet coming from source *j* as follows:

$$F_j^i = \frac{E_j^i}{r_j} + F_j^{i-1} \tag{1}$$

where r_j is the share of channel capacity allocated to session *j*, and E_j^i is equal to the length of the previously transmitted packet L_j^i if the most recent poll sent to station *j* resulted in a transmission and the "More Data" bit of the frame is set to 1, or $n \cdot L_j^i$ otherwise. The value of the constant *n* is a design parameter and is used to avoid wasting an excessive amount of bandwidth by polling repeatedly a station that has nothing to transmit for the moment. We used a value of $n =$ 2 in our simulations and the results are satisfactory. E_j^i can be thought of as the *effective* packet length. Equation (1) is equivalent to assuming that all traffic sources are constantly backlogged and we "charge" them for all of the transmission opportunities that they are given, regardless of whether they use them to actually send packets or not. Charging for each transmission opportunity is done *a posteriori*.

DM-SCFQ has two main advantages as compared to WFQ, which are inherited from the scheduling mechanisms it is derived from:

- It is computationally more efficient since it relies on a self-generated virtual time, and
- It does not need to know the arrival time or size of the packet at the head of each queue, since it makes its decisions based on previously transmitted packets.

For comparison purposes, we also present in this paper results related to an implementation of WFQ in a WLAN. As mentioned before, in order to implement WFQ the access point needs to know about the packets waiting in each mobile station's queue. This is done by letting every station inform the access point of how many new packets have arrived to its queue, along with a list of their sizes, every time the station has a chance to send a data frame. To stay updated, the access point polls all stations periodically every *m* superframes, which is again a design parameter that in our case was set to 2. We call this a "cyclic poll". The rest of the time, the access point polls only the winning stations, according to the rules of the relevant scheduling mechanism. This mechanism has the added disadvantage that it needs the modification of the MAC frame structure to include a new field used to exchange the queue-occupancy update information.

4. The proposed CAC algorithm

One of the options included in IEEE 802.11e to specify the traffic characteristics and needs is through leaky-bucket parameters plus an upper bound for the tolerable delay. This option assumes that the traffic is shaped before it enters the WLAN. This option is critical because of the relatively limited capacity of the wireless medium. When traffic complies with this restriction, it is possible to offer deterministic bounds for the delay that the traffic will experience when serviced using a fair scheduling policy, such as GPS [14, 16]. In other words, these results can be used to devise a CAC mechanism for delay-sensitive traffic as long as a fair scheduling algorithm is being used. The output of the CAC mechanism would be a decision as to whether a set of traffic streams can be serviced without violating their QoS guarantees, as well as the fraction of the available bandwidth that has to be allocated to each one of these traffic streams. Resource reservation is implicit in such a CAC mechanism.

Recently, an expression has been found in [2, 14, 20], for the tight delay bounds that characterize GPS systems with leaky-bucket constrained input traffic sources, and CAC algorithms have been proposed based on such bounds. Unfortunately, the previous algorithms do not take full advantage of the available information. In the remainder of this section we propose a CAC algorithm to maximize bandwidth utilization by allocating each session only the portion of the available bandwidth that is strictly needed so as not to exceed a given delay bound. In addition, the proposed CAC algorithm does a straightforward allocation based on systematic calculations, without the need of random search or successive approximations, as is done in [2].

Our algorithm is designed for a single-node system but, as explained in [15], it can be extended to multiple-node systems in many situations of practical interests. For instance, when it is applied at the bottleneck node, or when it is applied at an edge node and the same bandwidth is reserved for each connection at all the nodes included in its end-to-end path.

4.1. Terminology and previous results

A traffic source is said to be leaky-bucket constrained if its arrivals in any given interval $(t_1, t_2]$, denoted by $A(t_1, t_2)$, satisfy the following inequality:

$$A(t_1, t_2) \leq \sigma + \rho \cdot (t_2 - t_1) \tag{2}$$

When the equality holds for $t_1 = 0$ and all values of t_2, the traffic source is called greedy.

A GPS scheduler serving N sessions is characterized by a set of positive real numbers $\{\phi_1, \phi_2, \ldots, \phi_N\}$, which denotes the relative amount of service given to each session in the sense that, if session i is continuously backlogged during the time interval (t_1, t_2), then:

$$\frac{W_i(t_1, t_2)}{W_j(t_1, t_2)} \geq \frac{\phi_i}{\phi_j} \quad j = 1, 2, \ldots, N \tag{3}$$

where $W_i(t_1, t_2)$ is the amount of service given to session i during the time interval (t_1, t_2).

As mentioned above, a tight bound is found in [2, 14], and [20], for the maximum delay that a leaky-bucket constrained session experiences in a GPS server. The relevant bound is tighter than that found in [17], as it explicitly takes into account that the bandwidth unused by some sessions will be redistributed among the backlogged sessions proportionally to their respective weights ϕ_i. It also takes into account information related to the specific time instants at which bandwidth will be released by each session and therefore used by the other sessions that still need it.

In this work, each session will be assumed to be greedy and the incoming traffic of session j will be characterized by ρ_j and σ_j, which are the parameters of its respective leaky-bucket.

Let $\mathcal{L} = \{L(i) \,|\, i = 1, 2, \ldots, N\}$ denote an ordered set of indices meaning that the backlog of session $L(i)$ is cleared i^{th} in order, and let C be the server capacity. The time instant when the backlog of session $L(i)$ is emptied is denoted by $t_{L(i)}$. By definition, let $t_{L(0)} = 0$ be the start of a system busy period. If the buffer of session j is initially empty but starts to build up because its initial allocated bandwidth is less than its traffic arrival rate, then its backlog-clearing time is not $t_j = 0$, but the time when the buffer goes back to being empty.

In addition, let $r_j(t)$ denote the bandwidth (or service rate) effectively allocated to session j at time $t \leq t_j$, including its initial bandwidth $r_j = \phi_j \cdot C$ plus its share of bandwidth released by other sessions. To simplify notation, let r_j^k be the session-j service rate during the time interval $[t_{L(k-1)}, t_{L(k)})$. It is shown in Eq. (5) of [14], that, when $\sum_{j=1}^{N} r_j = C$ (the server is initially saturated), r_j^k satisfies the following equation:

$$r_j^k = coef(k-1) \cdot r_j \tag{4}$$

where

$$coef(k) = \frac{C - \sum\limits_{m=1}^{k} \rho_{L(m)}}{C - \sum\limits_{m=1}^{k} r_{L(m)}}. \tag{5}$$

Equation (4) is valid as long as the buffer of session j is not empty. After its own backlog has been cleared (i.e., $t > t_j$), session j will be serviced at rate ρ_j, which is the traffic arriving rate. Because of this, a necessary condition for the system to be able to properly serve all the sessions after all the backlogs have been cleared is that $\sum_{j=1}^{N} \rho_j \leq C$.

Let τ_j denote the time at which the initial burst of session j has been serviced, i.e., the time when $W_j(0, \tau_j) = \sigma_j$. Let $T_j(t)$ denote the time at which $A_j(0, T_j(t)) = W_j(0, t)$, i.e., the time at which the amount of traffic received was equal to the amount of traffic that has been serviced by time t. If we let $D_j(t)$ denote the delay experienced by the traffic completing service at time t, then $D_j(t) = t - T_j(t)$. It is partly shown in [20], that, in the worst case when the traffic source is greedy, the following equality holds:

$$D_j(t) = \begin{cases} t & \text{for } t \leq \tau_j \\ t - \frac{1}{\rho_j}\left[\int_0^t r_j(\tau) \cdot d\tau - \sigma_j\right] & \text{for } \tau_j < t \leq t_j \\ 0 & \text{for } t > t_j \end{cases} \tag{6}$$

The maximum delay that session j can tolerate is denoted by d_j. The following two sets are also defined:

$$B1 = \left\{ j \,\middle|\, \sigma_j / d_j \geq \rho_j \right\} \tag{7}$$

$$B2 = \left\{ j \,\middle|\, \sigma_j / d_j < \rho_j \right\} \tag{8}$$

In the following, we state the main theorem of [14, 20], and [2]. For a proof, see the references.

Theorem 1. *Let d_j^* denote the maximum delay that will be experienced by session j.*

(i) If $r_j(\tau_j) \geq \rho_j$, then $d_j^* = \tau_j$
(ii) Conversely, if $r_j(\tau_j) < \rho_j$, then:

$$d_j^* = D_j\left(t_{L(k)}\right) = t_{L(k)} - \frac{1}{\rho_j}\left[W_j(0, t_{L(k)}) - \sigma_j\right] \tag{9}$$

$r_j(t) < \rho_j$ for all $t < t_{L(k)}$ and where $k \in \{1, 2, \ldots, N\}$ is such that $r_j(t) \geq \rho_j$ for all $t \geq t_{L(k)}$.

In addition to this result, a few lemmas are included in [14], that are used to support their CAC algorithm. Here we enunciate two of those lemmas that are useful for our proposed algorithm. We use the same numbers utilized in the reference to identify the lemmas.

Lemma 1. *If session $j \in B1$ and its maximum-delay requirement is satisfied ($d_j^* \leq d_j$), then:*

(i) The service rate of session j is not less than ρ_j when the last bit of the initial burst is being served, i.e. $r_j(\tau_j) \geq \rho_j$.
(ii) The maximum delay of session j is experienced by the last bit of the initial burst, or equivalently $d_j^* = D_j(\tau_j) = \tau_j$.

Lemma 3. *If $d_j > t_{L(i-1)}$, then the rate r_j needed in order for the last bit of the initial burst to experience a delay equal to d_j is:*

$$r_j = \frac{\sigma_j}{\sum_{k=1}^{i-1}\left(t_{L(k)} - t_{L(k-1)}\right) \cdot coef(k-1) + \left(d_j - t_{L(i-1)}\right) \cdot coef(i-1)} \tag{10}$$

where $coef(k)$ is defined as in Eq. (5).

Corollary 3.1. *If session $j \in B1$ then the rate r_j shown in Eq. (10) is the optimum allocation in the sense that $d_j^* = d_j$.*

From Lemma 3 and Corollary 3.1, we can conclude that, for sessions in $B1$, the maximum allowed delay will not be exceeded as long as τ_j is equal to d_j. Thus the allocated bandwidth for the sessions always has to be computed using Lemma 3. This is already done in [14], and we do not change this part of the algorithm.

4.2. Our new results

This section presents some results that support the main contribution of this research, namely the new CAC algorithm described in next section. All of our results are intended to support the new algorithm for allocating bandwidth to sessions in $B2$. The CAC algorithm defines the minimum initial r_j that has to be allocated to session j so that its maximum delay d_j^* does not exceed a value d_j, a QoS parameter established by the user. This task is equivalent to finding the coefficients $\{\phi_1, \phi_2, \ldots, \phi_N\}$ for the GPS server as $r_j = \phi_j \cdot C$. To achieve the optimum allocation, we need to define criteria to ensure that we do not unnecessarily overprovision the sessions. Thus, we need to take into account the time at which each session frees some bandwidth after emptying its buffer and to keep track of the amount of extra bandwidth that is redistributed among the backlogged sessions. Our algorithm finds the optimum values $\{r_1, r_2, \ldots, r_N\}$ one by one. Each step, indexed by i in our algorithm, is marked by one more session emptying its backlog or, equivalently, reaching its t_j.

Lemma 4. *$D_j(t) \leq t$ for all $t \geq 0$.*

Proof: Notice from Eq. (6) that $D_j(t) = t$ for all $t \leq \tau_j$ and that $D_j(t) = 0 < t$ for all $t > t_j$. Also notice that for $\tau_j < t \leq t_j$:

$$\int_0^t r_j(\tau) \cdot d\tau = W_j(0, t) > \sigma_j \tag{11}$$

Using this result we can conclude from Eq. (6) that $D_j(t) < t$ for $\tau_j < t \leq t_j$ as well. □

Definition. Let t_j^* be the smallest time for which the maximum delay d_j^* is reached for session j, that is

$$t_j^* = \min\left\{t \left| D_j(t) = d_j^*\right.\right\} \cdot \tag{12}$$

The minimum is taken in Eq. (12) to avoid ambiguities, since there can be more than one value of t for which $D_j(t) = d_j^*$.

Theorem 2. *A necessary condition for d_j^* not to be smaller than d_j (bandwidth not to be wasted) is that $t_j^* \geq d_j$.*

Proof: Assume that $t_j^* < d_j$. From Lemma 4, we have $D_j(t_j^*) \leq t_j^*$. By definition, we know that $d_j^* = D_j(t_j^*)$. Putting all together we have:

$$d_j^* = D_j\left(t_j^*\right) \leq t_j^* < d_j \tag{13}$$

This means that, if $t_j^* < d_j$, d_j^* will inevitably be smaller than d_j and as a result bandwidth will be wasted. □

Theorem 3. *Assume that r_j and r_j' are two different values of the initial rate allocated to session j. Assume also that*

$D_j(t)$ and $D'_j(t)$ are the respective values of the delay experienced at time $t \geq 0$. Let τ_j and τ'_j be the times at which the initial burst is serviced corresponding to rates r_j and r'_j, respectively. Finally, let t_j and t'_j be the corresponding backlog clearing times. If $r'_j \geq r_j$ then

(i) $r'_j(t) \geq r_j(t),\ \forall t \leq \min\{t_j, t'_j\}$
(ii) $W'_j(0,t) \geq W_j(0,t),\ \forall t \leq \min\{t_j, t'_j\}$
(iii) $\tau'_j \leq \tau_j$
(iv) $D'_j(t) \leq D_j(t)$ *for all values of* $t \geq 0$.

Proof: Since the rate allocated to session j increases in a multiplicative fashion every time some bandwidth is freed by a user, as shown in Eq. (4), we have $r'_j(t) \geq r_j(t),\ \forall t \leq \min\{t_j, t'_j\}$. This in turn implies that, for the same time interval, $W'_j(0,t) \geq W_j(0,t)$. From here we can easily conclude that $\tau'_j \leq \tau_j$. Using these results and Eq. (6), we have

$$D_j(t) - D'_j(t) = \begin{cases} 0 & \text{for } 0 \leq t \leq \tau'_j \\ \frac{1}{\rho_j}\left[W'_j(0,t) - \sigma_j\right] > 0 & \text{for } \tau'_j < t \leq \tau_j \\ \frac{1}{\rho_j}\left[W'_j(0,t) - W_j(0,t)\right] \geq 0 & \text{for } \tau_j < t \leq t'_j \\ D_j(t) \geq 0 & \text{for } t'_j < t \leq t_j \\ 0 & \text{for } t > t_j \end{cases} \tag{14}$$

which proves that $D'_j(t) \leq D_j(t)$ for all values of $t \geq 0$, as desired. □

Theorem 4. *If $j \notin \{L(k) \mid k = 1, 2, \ldots, i-1\}$, in order for r^i_j to be equal to ρ_j, it is necessary that $r_j = \rho_j / coef(i-1)$.*

Proof: Since the backlog of session j has not been cleared by $t_{L(i-1)}$, we can use Eq. (4) to compute r^i_j. Then we have

$$r^i_j = coef\,(i-1) \cdot r_j = \rho_j \tag{15}$$

Solving for r_j we obtain the desired result. □

The following theorem is similar to Lemma 4 in [14], but it is rephrased to make it more useful for our algorithm.

Theorem 5. *Assume that $j \notin \{L(k) \mid k = 1, 2, \cdots, i-1\}$ and that $d_j \leq t_{L(i-1)}$. If*

$$r_j = \frac{\rho_j \cdot \left(t_{L(i-1)} - d_j\right) + \sigma_j}{\sum_{k=1}^{i-1} \left(t_{L(k)} - t_{L(k-1)}\right) \cdot coef\,(k-1)} \tag{16}$$

then $\tau_j \leq t_{L(i-1)} < t_j$ and $D_j\left(t_{L(i-1)}\right) = d_j$.

Proof: Notice that:

$$W_j\left(0, t_{L(i-1)}\right) = \sum_{k=1}^{i-1} \left(t_{L(k)} - t_{L(k-1)}\right) \cdot r^k_j \tag{17}$$

From Eqs. (4), (16) and (17), we can conclude that

$$W_j\left(0, t_{L(i-1)}\right) = \rho_j \cdot \left(t_{L(i-1)} - d_j\right) + \sigma_j \cdot \tag{18}$$

Since $d_j \leq t_{L(i-1)}$, we can conclude from Eq. (18) that $W_j\left(0, t_{L(i-1)}\right) \geq \sigma_j$, which in turn implies that $\tau_j \leq t_{L(i-1)}$.

On the other hand, by definition a time t will be identified as t_j when the initial burst, plus whatever extra traffic that has arrived since the beginning, has been served. Or equivalently, when $W_j(0,t) = \sigma_j + \rho_j \cdot t$. However, again from Eq. (18), we have that:

$$W_j\left(0, t_{L(i-1)}\right) = \left[\sigma_j + \rho_j \cdot t_{L(i-1)}\right] - \rho_j \cdot d_j \tag{19}$$

which indicates that at $t_{L(i-1)}$ there is still a backlog of size $\rho_j \cdot d_j$, and that t_j has not been reached yet. That is, $t_{L(i-1)} < t_j$.

Now, knowing that $\tau_j \leq t_{L(i-1)} < t_j$, we can substitute Eq. (18) into Eq. (6) and obtain that $D_j\left(t_{L(i-1)}\right) = d_j$. □

Theorem 6. *$D_j(t)$ is a monotonically increasing function for all values of t such that $r_j(t) < \rho_j$.*

Proof: First notice that the condition $r_j(t) < \rho_j$ can only be satisfied for values of t less than t_j, because at t_j the session has to be serviced at a rate higher than its arrival rate (ρ_j) in order to empty its backlog. If we assume that $r_j(t) < \rho_j$ and take the derivative of Eq. (6), we can see that

$$\frac{d\,D_j(t)}{dt} = 1 > 0, \text{ for } t \leq \tau_j \tag{20}$$

and that

$$\frac{d\,D_j(t)}{dt} = 1 - \frac{r_j(t)}{\rho_j} > 0, \text{ for } \tau_j < t < t_j \cdot \tag{21}$$

In both cases the derivative of $D_j(t)$ is positive, which indicates that it is a monotonically increasing function of t. □

The "checkpoints" for a session j are defined as those time instants at which the maximum delay d_j can be achieved (i.e., candidates for t^*_j) [15]. According to Theorem 1, the checkpoints include d_j itself and those $t_{L(k)}$ greater than d_j, $k \in \{1, 2, \ldots, N\}$.

Theorem 7. *If a bandwidth allocation for session j is such that $D_j(t) = d_j$ and $r_j(t^+) < \rho_j$ for a given checkpoint $t \geq d_j$, then the bandwidth allocation that satisfies $D_j(t') = d_j$ for the subsequent checkpoint $t' > t$ is such that $r'_j > r_j$.*

Proof: Without loss of generality, assume that $t' = t_{L(i)}$, for some value of $i \in \{1, 2, \ldots, N\}$. From Lemma 3 if $t = d_j$, or from Theorem 5 if $t > d_j$, we have

$$W_j(0,t) = \rho_j \cdot \left(t - d_j\right) + \sigma_j \tag{22}$$

where

$$W_j(0,t) = r_j \cdot \left[\sum_{k=1}^{i-1} \left(t_{L(k)} - t_{L(k-1)}\right) \cdot coef(k-1) + \left(t - t_{L(i-1)}\right) \cdot coef(i-1)\right] \tag{23}$$

Under the assumption that $r_j\left(t^+\right) < \rho_j$, we have

$$W_j(0,t) + r_j(t^+) \cdot \left(t_{L(i)} - t\right) < \rho_j \cdot (t - d_j) + \sigma_j + \rho_j \cdot \left(t_{L(i)} - t\right). \tag{24}$$

But, $r_j\left(t^+\right) = r_j \cdot coef(i-1)$, which implies that

$$r_j \cdot \left[\sum_{k=1}^{i} \left(t_{L(k)} - t_{L(k-1)}\right) \cdot coef(k-1)\right] < \rho_j\left(t_{L(i)} - d_j\right) + \sigma_j \tag{25}$$

or equivalently that

$$r_j < \frac{\rho_j\left(t_{L(i)} - d_j\right) + \sigma_j}{\sum_{k=1}^{i} \left(t_{L(k)} - t_{L(k-1)}\right) \cdot coef(k-1)} = r_j' \tag{26}$$

The last equality is obtained using Theorem 5. □

In the following, we discuss how the preceding theorems can be applied to achieve the optimum bandwidth allocation.

The key of the CAC algorithm is to try to achieve $D_j(t) = d_j$ at each checkpoint t by allocating the proper amount of bandwidth as indicated by Lemma 3 or by Theorem 5, respectively, depending on whether t is equal to d_j or not.

At step i, from Theorem 2, if $d_j > t_{L(i-1)}$, the only viable candidate for the location of t_j^* is d_j. If we allocate enough bandwidth so that $\tau_j = d_j$ (and therefore $D_j(d_j) = d_j$) and if the allocated bandwidth is such that $r_j(d_j^+) = r_j^i \geq \rho_j$, we have found the optimum bandwidth allocation for this session, unless there are sessions that free bandwidth before d_j (in which case we can allocate less bandwidth and still satisfy the delay requirement). On the other hand, if the allocated bandwidth is such that $r_j(d_j^+) < \rho_j$ (only possible for sessions in $B2$, according to Lemma 1), the delay will continue to increase, as indicated by Theorem 6, with the maximum value being greater than d_j. Therefore, we need to allocate more bandwidth to this session, but we do not have sufficient information yet to know how much more.

For session j, if there exists a checkpoint $t_{L(i-1)}$ greater than d_j, it is because the allocation calculated at the previous checkpoint $t_{L(i-2)}$ (either equal to or larger than d_j) was such that $r_j(t^+) < \rho_j$. From Theorem 7, we can see that the new bandwidth calculated at this step is greater than the previous one. Again we have to check if this new allocation is such that $r_j\left(t_{L(i-1)}^+\right) \geq \rho_j$. If so, we have found the optimum allocation for this session because there is no more bandwidth freed by any sessions before this checkpoint. On the other hand, if the calculated bandwidth is such that $r_j\left(t_{L(i-1)}^+\right) < \rho_j$, we do not have the optimum bandwidth allocation and should keep trying at the subsequent checkpoints, if there are any.

If all the finite checkpoints have been tried and the optimality condition is not met, using Theorem 4, session j will be allocated an amount of bandwidth such that $r_j(t^+) = \rho_j$, where t is equal to the largest finite checkpoint. This bandwidth allocation satisfies the delay requirement since, according to Theorem 1, the maximum delay happens at t and, from Theorem 3, this maximum delay is less than d_j because a smaller amount of bandwidth would make $D_j(t) = d_j$.

There are two reasons to postpone the decision on the optimum bandwidth allocation for a given session: either we do not yet have enough information to decide what the optimum allocation is (as explained above), or we know how much bandwidth should be allocated but there is no sufficient bandwidth available unless some sessions free a portion of their own bandwidth. In the latter case, we postpone our decision, hoping for other sessions to free enough bandwidth for use by the relevant session.

If a decision cannot be made for session j at a specific step i, for either of the two reasons, the session will be moved temporarily into another set (referred to as $N1$ or $N2$ depending on whether the original set was $B1$ or $B2$), indicating that this session is not a suitable candidate for $L(i)$.

Once we find the optimum bandwidth allocation for session j, it is removed from its original set ($B1$ or $B2$) and is put into another set denoted by P.

At the end of step i, we select $t_{L(i)}$ as the smallest among the potential backlog-clearing time values computed at this step (denoted by $t_{j,i}$), corresponding to sessions in $B1$, $B2$ and P. As derived in [14], the potential backlog-clearing times are given by

$$t_{j,i} = \frac{r_j^i \cdot t_{L(i-1)} + \sigma_j - W_j\left(t_{L(i-1)}\right)}{r_j^i - \rho_j}. \tag{27}$$

The selected session, defined above as $L(i) \in \mathcal{L}$, is then removed from its current set and placed into a set denoted by H. If there is a draw (a tie) between two or more sessions because their corresponding values $t_{j,i}$ coincide, then all such sessions are selected as the new elements of $\mathcal{L}$ because their allocated bandwidth will not change after this point.

However, it is possible that, after going through $B1$, $B2$ and P, there is no candidate for $L(i)$ because all the decisions related to sessions in $B1$ and $B2$ were postponed. When this situation is encountered, if $N1$ is not empty, it means that there is not enough bandwidth to serve all the sessions; otherwise we are forced to allocate bandwidth to all the sessions in $B2$ such that $r_j(t^+) = \rho_j$, where t is equal to the largest finite checkpoint.

As an interesting remark, recall that Eq. (4) is only valid when the server is initially saturated, i.e. when $\sum_{j=1}^{N} r_j = C$. If we find at the end of the algorithm that this condition is not satisfied, it is proposed to repeat the process with a reduced value of C until the relevant condition is approximately met [14]. This approach, although not necessary for finding a suitable set of weights, can be useful to find out roughly the minimum value of the channel capacity $C_{\min}$ that would be enough to serve the relevant set of sessions. It is also important to point out that a byproduct of this algorithm is the possibility to calculate the minimum buffer requirements for each session, given by

$$B_j^* = A_j\left(t_j^*\right) - W_j\left(t_j^*\right) = \left(\sigma_j + \rho_j \cdot t_j^*\right) - \int_0^{t_j^*} r_j(z) \cdot dz. \quad (28)$$

This expression can be easily evaluated after knowing the values of r_j and t_j^*.

We describe our CAC algorithm in more detail in the following section using pseudo-code, where BW stands for bandwidth.

4.3. Our CAC algorithm

Parameters needed: ρ_j, σ_j, d_j for all sessions $j \in \{1, 2, \ldots, N\}$ and C
Variables: i, $L(i)$, $t_{L(i)}$, $r_{L(i)}$, $r_{aux}(j)$, C_{rem}, set(j), forced

```
0) If (sum of all $\rho_j$'s, $j \in \{1,2,\ldots,N\}$, is greater than $C$)
   {
      Display "Impossible to serve all sessions"; Stop algorithm;
   }
1) Separate sessions into $B1$ and $B2$;
   Define $N1$ and $N2$ initially as empty sets (sessions for which decision is postponed for a subsequent step);
   Define $P$ initially as an empty set (sessions for which BW has been assigned but $t_j$ has not been computed yet);
   Define $H$ initially as an empty set (sessions for which BW has been assigned and $t_j$ has been computed);
   Define $H0$ initially as an empty set (sessions with the minimum $t_{j,i}$ at the end of step $i$);
   Make $i = 1$, $L(0) = 0$, $t_{L(0)} = 0$, $C_{rem} = C$, $forced = $ FALSE;
2) For all sessions in $B1$:
   If ($t_{L(i-1)} < d_j$)
   {
      $r_{aux}(j) =$ Lemma 3($i, j$); // We want to make $\tau_j = d_j$
      If ($r_{aux}(j) \le C_{rem}$) Calculate $t_{j,i}$;
      Else set($j$) = $N1$; // Hoping for other sessions to free enough BW
   }
   Else // If ($t_{L(i-1)} \ge d_j$)
   {
      Display "Impossible to serve all sessions";
      Stop algorithm; // Session went into $N1$ hoping for BW to be freed by other sessions, but it did not happen
   }
3) For all sessions in $B2$:
   If ($t_{L(i-1)} < d_j$)
   {
      $r_{aux}(j) =$ Lemma3($i, j$); // We want to make $\tau_j = d_j$
      If ($r_j^i \ge \rho_j$)
      {
      If ($r_{aux}(j) \le C_{rem}$) Calculate $t_{j,i}$;
      Else If (!forced) set($j$) = $N2$; // Hoping for other sessions to free enough bandwidth
      Else
      {
       Display "Impossible to serve all sessions";
       Stop algorithm; // No session will free BW after this point
       }
     }
   }
   Else // If ($t_{L(i-1)} \ge d_j$)
   {
     $r_{aux}(j) =$ Theorem5($i, j$); //We want $D_j(t_{L(i-1)}) = d_j$
     If ($r_j^i \ge \rho_j$)
     {
       If ($r_{aux}(j) \le C_{rem}$) set($j$) = $P$; // Found optimum allocation
       Else
       {
         Display "Impossible to serve all sessions";
         Stop algorithm; // BW freed after this point will not help
       }
     }
   }
   If ($r_j^i < \rho_j$)
   {
     If (!forced) set($j$) = $N2$; //We have to wait to find optimum
     Else // If (forced)
```

```
        {
          $r_{aux}(j) = $ Theorem4$(i, j)$; // We want to make $r_j^i = \rho_j$
          If ($r_{aux}(j) \le C_{rem}$) $t_{j,i} = $ INFINITY;
          Else
          {
            Display "Impossible to serve all sessions";
            Stop algorithm; // No session will free BW after
            this point
          }
        }
      }
4)  For all sessions in $P$: Calculate $t_{j,i}$;
5)  If (($B1 \cup B2 \cup P$) is not empty)
    {
      Find $t_{\min} = \min_{j \in (B1 \cup B2 \cup P)} t_{j,i}$; // It could be INFINITY
      Move all sessions with $t_{j,i} = t_{\min}$ into $H0$;
      Move all sessions in ($B1 \cup B2$) with $d_j \le t_{\min}$ into $P$;
    }
    Else // If ((B1 ∪ B2 ∪ P) is empty)
    {
      If ($N1$ is not empty)
      {
        Display "Impossible to serve all sessions";
        Stop algorithm;
      }
      Else // If ($N1$ is empty)
      {
        forced = TRUE;
        Return sessions in $N2$ into $B2$
        Repeat from step 3.
      }
    }
6)  For all sessions in $H0$:
    {
      $C_{rem} = C_{rem} - r_{aux}(j)$;
      If ($C_{rem} \ge 0$)
      {
        $L(i) = j$;   $t_{L(i)} = t_{j,i}$;   $r_{L(i)} = r_{aux}(j)$;   set$(j) = H$;
        $i = i + 1$;
      }
      Else // If ($C_{rem} < 0$)
      {
        Display "Impossible to serve all sessions";
        Stop algorithm;
      }
    }
7)  If ($i < N$)
    {
      Return sessions in $N1$ into $B1$ and sessions in $N2$ into
      $B2$
      forced = FALSE;
      Repeat from step 2.
    }
    Else // If ($i == N$)
    {
      Display "All sessions can be served simultaneously";
      Display all the values $r_j$, $j \in \{1, 2, \ldots, N\}$
      Stop algorithm;
    }
```

4.4. Comparison of our algorithm to previously proposed ones

If we compare our algorithm to those presented in [2], the main difference is that the previously proposed algorithms are not based on systematic calculations, but rely instead either on random search techniques or on successive approximations to find a solution, which can be very inefficient or even ineffective, as explained in [15].

If, on the other hand, we compare our algorithm to that presented in [14], we can easily see the following differences.

- Our algorithm takes advantage of draws (ties) when it comes to finding the smallest backlog-clearing time at each step, in the sense that we can find several optimum allocation values in a single step. The algorithm in [14], does not check for draws, which causes the unnecessary repetition of many calculations;
- For sessions in $B2$, when $d_j \le t_{L(i-1)}$ the algorithm in [14] uses a result similar to our Theorem 5 to allocate a rate such that $D_j\left(t_{L(i-1)}\right) = d_j$. To ensure that this is the maximum delay that the session will experience, the algorithm verifies if that allocation is such that $r_j^{i-1} < \rho_j \le r_j^i$. If not, the algorithm proceeds to allocate a rate such that either $r_j^{i-1} = \rho_j$ or $r_j^i = \rho_j$, depending on whether it is the first inequality what is not satisfied or the second. The first option represents a potential under-provision of bandwidth because the rate is reduced from the value corresponding to $D_j\left(t_{L(i-1)}\right) = d_j$. The second option represents a potential over-provision of bandwidth because $D_j\left(t_{L(i-1)}\right)$ may have become the maximum delay experienced by session j with $D_j\left(t_{L(i-1)}\right) < d_j$ and $r_j^i = \rho_j$.
- To summarize, the algorithm proposed in [14], can over-provide bandwidth to some sessions, resulting in inefficient use of the resources. More importantly, it can also under-provide bandwidth, leading to QoS violations of the sessions involved.

Finally, the algorithm presented in [15] is equivalent to the algorithm presented here. It is important to mention, however, that our algorithm was developed independently as part of an M.Sc. thesis project [13]. Even though both algorithms rely on completely different calculations and comparisons, we can see their equivalence by noticing that the algorithm in [15] allocates weights derived from Proposition 5, which in turn is based on Eqs. (9) and (10) to compute what they call

$\phi_j^-(t)$ and $\phi_j^+(t)$, where t is one of the so-called checkpoints. Assigning $\phi_j = \phi_j^-(t)$ causes that $D_j(t) = d_j$ (equivalent to Lemma 3 in this work when t is equal to d_j, or to Theorem 5 when t is equal to $t_{L(i-1)} > d_j$). In addition, that algorithm makes sure that the allocated weights are such that $\phi_j \geq \phi_j^+(t)$, which implies that $r_j(t^+) \geq \rho_j$ (similar to using our Theorem 4). A detailed look at step B4 of that algorithm and at steps 2 and 3 of our algorithm reveals that both algorithms are based on the same principles.

5. Performance evaluation

In this section, we analyze the applicability of the specific QoS mechanisms that we propose for an IEEE 802.11 WLAN. According to the IEEE 802.11e draft standard, in order for a session to be established between delay-sensitive stations, an *ADDTS request* frame has to be received by the HC located in the AP. This frame has to include a TSPEC, which in our case consists of the parameters ρ, σ, and d. If it is an upstream (only the transmitter is located in the relevant BSS) or bidirectional communication (both the transmitter and receiver are located in the same BSS), the *ADDTS request* frame has to be sent by the transmitting station. If it is a downstream communication (only the receiver is located in the relevant BSS), it has to be the receiving station which initiates the service request. We assume that there is a higher-layer mechanism for the receiving station to learn about the need to establish the session and about the corresponding TSPEC parameters.

When an upstream or downstream session is requested, the hybrid coordinator will divide the maximum allowed delay d by two, assuming that there will be a comparable delay at the distant side of the network.

When a bidirectional session is to be established, which is not explicitly included in the draft standard, the transmitter will make a request as in the case of an upstream session. The hybrid coordinator will identify bidirectional sessions whose receivers are located within the same BSS. The parameters σ and ρ are multiplied by two by the hybrid coordinator to take into account the fact that the wireless medium will be used twice by each frame.

The weights produced by the CAC algorithm corresponding to bidirectional sessions are divided by two, to indicate the packet scheduling mechanism that each frame will use the channel twice (equivalent to multiplying the length of each packet by two).

The variability of the channel characteristics, including the effective transmission rate or channel capacity C, is always a source of concern in a wireless environment. Any CAC algorithm needs to know the channel capacity in order to decide if it is possible to serve a set of clients, and ours is no exception. We estimated the average channel capacity empirically (via simulations) assuming that there are no transmission errors and that the only factor affecting it is the overhead introduced by the packet-scheduling portion of the MAC protocol. We found out that the channel capacity is reduced by about 12% for WFQ and by about 5% for DM-SCFQ. To take into account that the channel effective rate may be varying, it can be monitored in real time using the method described in [11], and compared each time with the minimum capacity needed $C_{\min}$, which can be estimated using a repetition of the CAC algorithm, as described at the end of Section 4.B. If at any time the channel capacity goes below this threshold, a warning can be issued to all the sessions in the polling list.

Table 1 Traffic characterization

TRAFFIC TYPE	σ (bits)	ρ (bps)	d (sec)	r_j (bps)
Voice	2,025	27,000	0.2	46,484.82
Video-conference	33,000	165,000	0.4	269,135.14
Video (Jurassic Park)	247,000	960,000	0.4	1,235,000.00
WWW	45,000	64,000	2.0	104,391.81

It is also possible that a station already included in the polling list decides to decrease (or increase) its transmission rate because of a degradation (or improvement) of the wireless medium. The hybrid coordinator has to keep track of these changes and, when detected, it runs again the CAC algorithm with the TSPEC values modified accordingly for the stations that intend to change their transmission rates. The parameters are changed to incorporate that a station with a reduced transmission rate occupies the channel for a longer period in order to transmit the same information, and vice versa. If the outcome of the CAC algorithm indicates that it

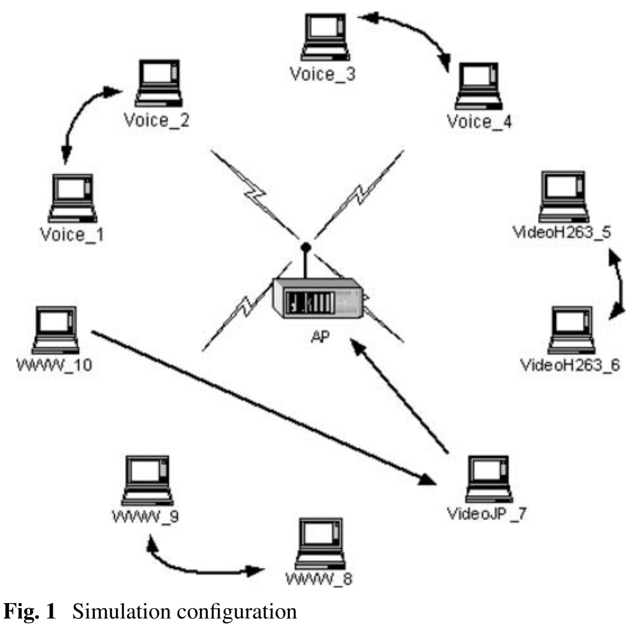

Fig. 1 Simulation configuration

is no longer possible to serve all sessions, a *DELTS request* frame will be sent to the stations that intend to change their transmission rates.

Another important task to be performed by the hybrid coordinator is to identify long inactivity periods of specific sessions in order to drop them from the polling list, thus avoiding the waste of reserved resources. The length of the maximum inactivity period can be negotiated when the session establishment is requested.

5.1. Simulation model

Our simulation scenario is an infrastructure network with a physical layer compatible with the IEEE 802.11b standard running at 5.5 Mbps.

Three different types of traffic models are used to simulate terminals using video, voice and WWW services. For video traffic, we use trace-driven traffic generators, one corresponding to the movie Jurassic Park encoded in MPEG1, and the other corresponding to videoconference encoded in H.263 [19]. For voice traffic, we use a two-state Markov chain, as described in [7], in which the voice traffic has been compressed according to the ITU-T G.729 standard. Finally, for WWW traffic, we use the model described in [1], in which each terminal generates traffic equivalent to that produced by 20 users browsing the web. All applications run on TCP/UDP-IP. Each of these traffic sources is characterized with its corresponding token-bucket parameters (ρ, σ), as described in [21]. Table 1 gives these parameters, along with the initial rate obtained from our CAC algorithm, which is the minimum necessary to satisfy the QoS requirements of each traffic source.

In all of the simulations, we include 4 voice stations and 2 videoconference stations communicating in pairs. We also include a video source (Jurassic Park) sending its traffic to a distant receiver through the AP. The number of WWW traffic sources varies, depending on the target network traffic load and on the scheduling mechanism (which influences the

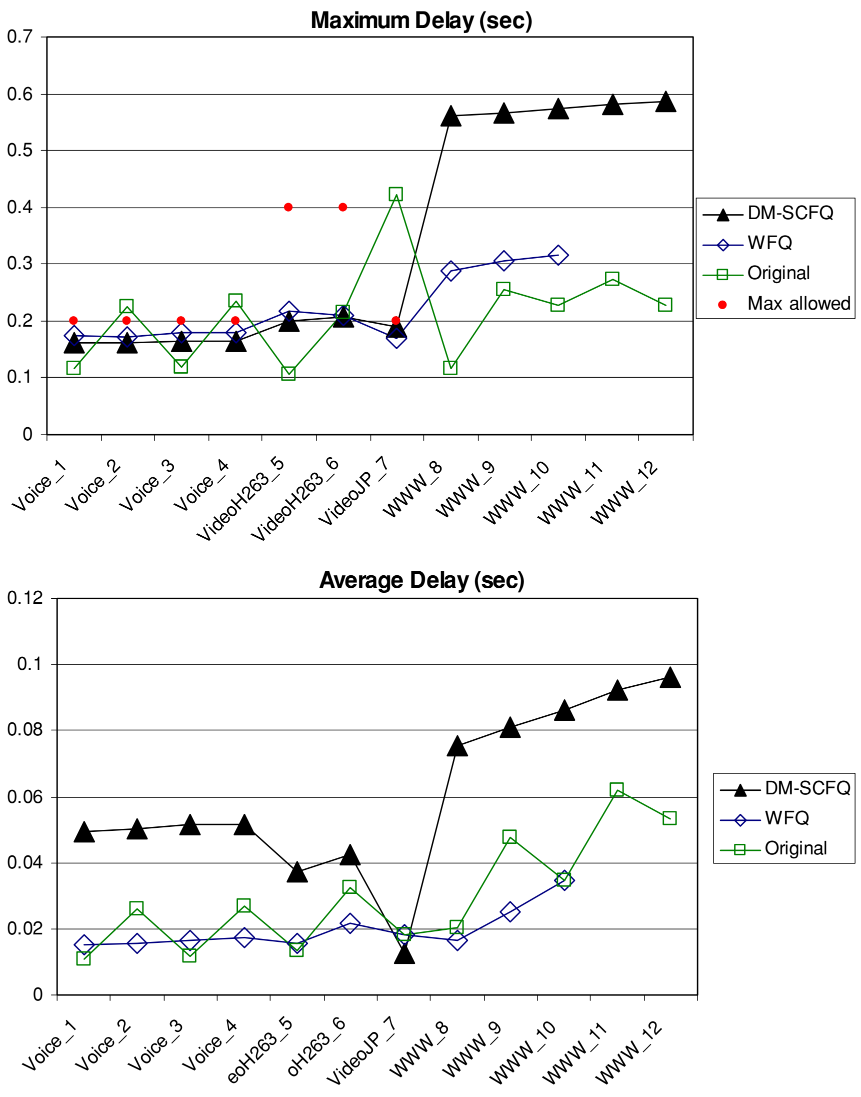

Fig. 2 Maximum and average delay comparison when only sessions accepted by the CAC algorithm are active

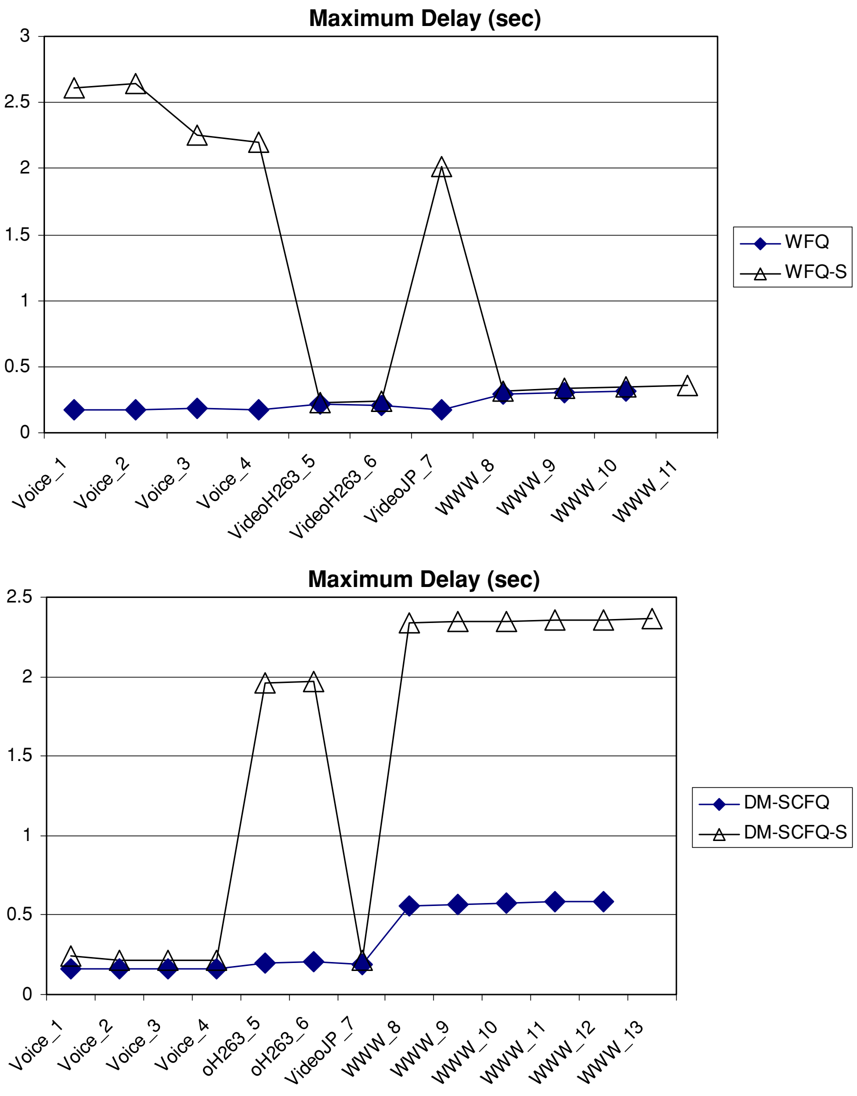

Fig. 3 Maximum delay comparison when more sessions are active than those accepted by the CAC algorithm (saturation)

effective transmission rate). Figure 1 illustrates the simulation configuration.

We use OPNET as our platform to simulate and evaluate our proposed schemes, as well as to compare it to the widely known WFQ scheduling mechanism. The simulation time in all cases is 2 minutes with a warm-up period of 1 minute.

We compare the performance of HCF-CA alone to that of HCF-CA when it is combined with the QoS mechanisms proposed in this paper. Our emphasis is on the delay experienced by packets from the moment they are generated to the moment they arrive at their final destination after having used the wireless medium.

5.2. Numerical results

The following three experiments are carried out:

- Experiment I – only sessions accepted by the CAC algorithm are allowed to be active;
- Experiment II – one WWW extra station is allowed to transmit in addition to those accepted by the CAC algorithm. We denote this case with a letter S to indicate that the system is saturated;
- Experiment III – one of the WWW stations that was accepted by the CAC algorithm in the first setup (station WWW_10) transmits ten times as much traffic as it specified in its request. We denote this case with a letter M to indicate that a misbehaving station is present.

Figure 2 shows a comparison of the maximum and average delay achieved in Experiment I. With the original MAC protocol, the maximum delay experienced by the traffic coming from the different sources is rather random, as no differentiation is exercised. The maximum acceptable delay is exceeded for several sessions. Contrary to this, the maximum delay achieved for both WFQ and DM-SCFQ is below the maximum allowed delay in all cases. It can also be seen that the maximum delay achieved by the source transmitting the

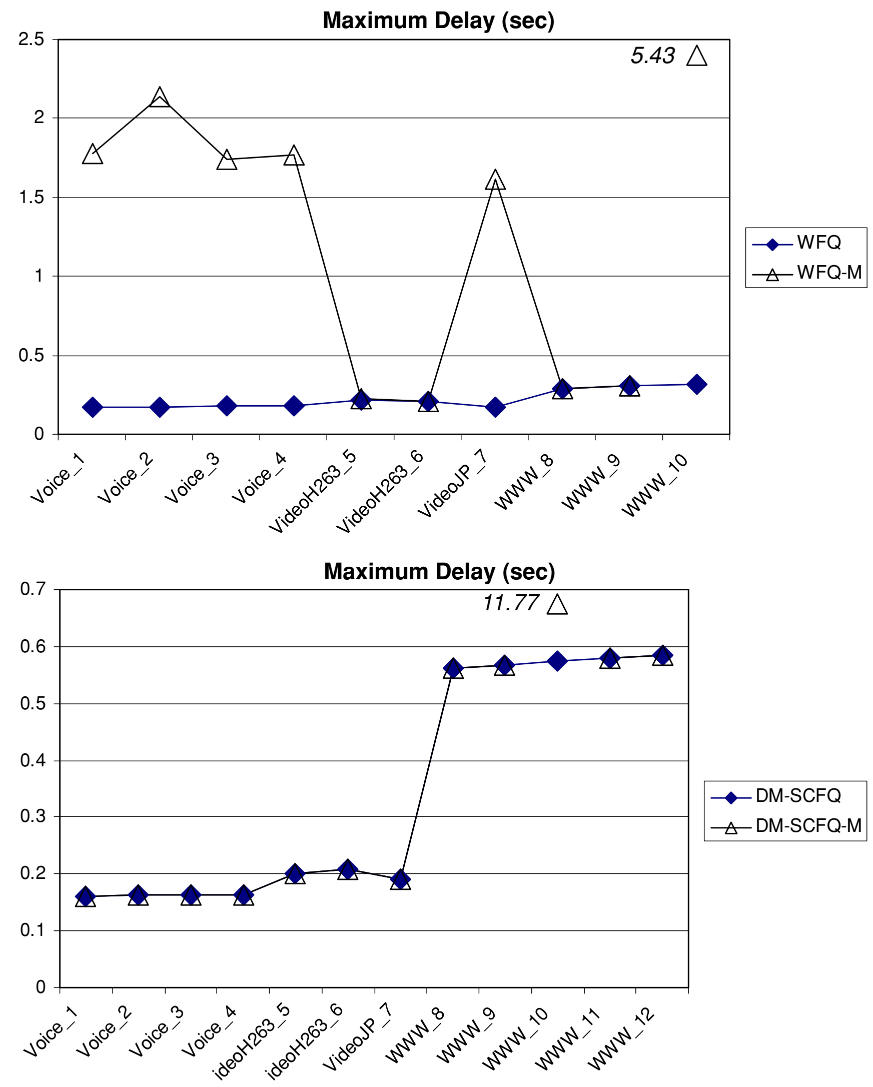

Fig. 4 Maximum delay comparison when one session is sending more traffic than it specified in its request (misbehavior)

Jurassic Park movie (source 7) is smaller for WFQ than it is for DM-SCFQ, while the maximum delay achieved by all the other delay-sensitive applications is larger for WFQ than it is for DM-SCFQ. This is due to the fact that the more opportunities a session has to transmit the more updated the hybrid coordinator remains regarding the condition of its buffer. Since source 7 is the one that transmits with the highest data rate, it has a chance to inform more often the hybrid coordinator of its newly arrived packets, which means that it does not have to wait for the cyclic poll as often as the other stations, thus keeping its delays smaller.

Figure 3 shows a comparison of the maximum delay achieved in Experiment II, in which there is an extra active session in addition to those accepted by the CAC algorithm. It is expected that the maximum allowed delay will be exceeded because the system is saturated. The goal of these simulations, however, is to show how sensitive the scheduling mechanisms are to the extra load. It can be seen that the added delay for DM-SCFQ is proportional to the maximum delay that a session can tolerate, while it is quite unpredictable for WFQ. This again is a consequence of the random delay introduced in WFQ between the generation of a new burst of packets and the time when the hybrid coordinator is informed of its existence.

Figure 4 shows the results obtained in Experiment III. It is clear that WFQ is very sensitive to this type of behavior due to the fact that not only is the misbehaving source generating extra traffic, but it is also taking advantage of each transmission to inform the hybrid coordinator more often of its new packets. DM-SCFQ, on the other hand, is very robust against this type of behavior and only the misbehaving station (WWW_10) gets an excessive delay.

6. Conclusions

This work proposes and analyzes specific admission control and service differentiation mechanisms that can be used to enhance the IEEE 802.11 MAC protocol in order to effectively provide QoS guarantees. Our proposal includes a novel traffic scheduling mechanism that takes into account the limitations typical of a wireless environment, as well as a CAC algorithm based on recent results related to tight delay bounds for systems with leaky-bucket constrained input traffic serviced using a fair scheduling policy.

The simulation study indicates that our proposed mechanisms have acceptable performance. In comparison, our proposed schemes have much better performance in terms of fairness and robustness than the well-known WFQ scheduling algorithm.

References

1. P. Barford and M.E. Crovella, "Generating representative web workloads for network and server performance evaluation," ACM SIGMETRICS, Madison WI, pp. 151–160 (July 1998).
2. P. Barta, F. Németh, R. Szabó and J. Bíró, "Call admission control in generalized processor sharing schedulers with tight deterministic delay bounds," *Computer Communications*, Vol. 26, No. 2 (Feb. 2003) pp. 65–78.
3. S. Blake, D. Black and M. Carlson, Internet Engineering Task Force, RFC 2475, *An Architecture for Differentiated Services* (December 1998).
4. R. Braden, D. Clark and S. Shenker, Internet Engineering Task Force, RFC 1633, *Integrated Services in the Internet Architecture: An Overview* (June 1994).
5. M.S. Gast, *802.11 Wireless Networks: The Definitive Guide*, Sebastopol, CA: O'Reilly, (2002).
6. S.J. Golestani, "A self-clocked fair queuing scheme for broadband applications," *Proc. 1994 IEEE INFOCOM*, Toronto, Canada, pp. 636–646, (12–16 June 1994).
7. D.J. Goodman and S.X. Wei, "Efficiency of Packet Reservation Multiple Access," *IEEE Transactions on Vehicular Technology*, Vol. 40, No. 2 (Feb. 1991) pp. 170–176.
8. A. Grilo and M. Nunes, "Performance evaluation of IEEE 802.11e," *Proc. 13th International Symposium on Personal, Indoor and Mobile Radio Communications* (PIMRC 2002), Lisbon, Portugal (September 15–18, 2002).
9. IEEE, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, ANSI/IEEE Standard 802.11 (1999 Edition).
10. IEEE, "Wireless medium access control (MAC) and physical layer (PHY) specifications: Medium access control (MAC) enhancements for quality of service (QoS)," *Draft Supplement to ANSI/IEEE Std 802.11, 1999 Edition*, Std 802.11e/D3.1 (July 2002).
11. M. Kazantzidis, M. Gerla and S.-J. Lee, "Permissible throughput network feedback for adaptive multimedia in AODV MANETs," *Proc. 2001 IEEE International Conference on Communications (ICC 2001)*, Helsinki, Finland, Vol. 5, No. 11–14, pp. 1352–1356, (11–14 June 2001).
12. S. Mangold, S. Choi, P. May, O. Klein, G. Hiertz and L. Stibor, "IEEE 802.11e wireless LAN for quality of service," invited paper, European Wireless 2002 (EW 2002), Florence, Italy (February 2002).
13. P. Medina, "Introduction of quality of service mechanisms in the medium access protocol of IEEE 802.11 wireless local area networks." *M. Sc. Thesis*, CICESE Research Center, (2004).
14. D. Nandita, J. Kuri and H.S. Jamadagni, *Optimal Call Admission Control in Generalized Processor Sharing (GPS) Schedulers*, IEEE INFOCOM 2001, Anchorage, AK, April 22–26, (2001). Also in Technical Report TR-00-02, Center for Electronic Design and Technology, Indian Institute of Science.
15. A. Panagakis., N. Dukkipati, I. Stavrakakis and J. Kuri, "Optimal call admission control on a single link with a GPS scheduler", *IEEE/ACM Transactions on Networking*, Vol. 12, No. 5, pp. 865–878, (Oct. 2004).
16. A.K. Parekh and R.G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single-node case," *IEEE/ACM Transactions on Networking*, Vol. 1, No. 3 (1993), pp. 344–357.
17. A.K. Parekh and R.G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the multiple-node case," *IEEE/ACM Transactions on Networking*, Vol. 2, No. 2 (1994), pp. 137–150.
18. R.S. Ranasinghe, L.L.H. Andrew and D. Everitt, *Distributed Contention-Free Traffic Scheduling in IEEE 802.11 Multimedia Networks*, 10th IEEE Workshop on Local and Metropolitan Area Networks. Selected Papers, D. Skellern, A. Guha and F. Neri, (Editors). Piscataway, NJ: IEEE, pp. 18–28 (2001).
19. O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems," *Proc. 20th Annual Conference on Local Computer Networks*, Minneapolis, MN, pp 397–406 (1995).
20. R. Szabó, P. Barta, F. Németh and J. Bíró, Worst-case deterministic delay bounds for arbitrary weighted generalized processor sharing schedulers, Proc. Intl. Conference on Broadband Communications, High Performance Networking, and Performance of Communication Networks, IFIP-TC6/European Commission (NETWORKING 2000), Paris, France (May 14–19, 2000).
21. D.E. Wrege, E.W. Knightly, H. Zhang and J. Liebeherr, Deterministic Delay Bounds for VBR Video in Packet-Switching Networks: Fundamental Limits and Practical Trade-Offs, *IEEE/ACM Transactions on Networking*, Vol. 4, No. 4 (1996), pp. 352–362.

José R. Gallardo received the B.Sc. degree in Physics and Mathematics from the National Polytechnic Institute in Mexico City, the M.Sc. degree in Electrical Engineering from CICESE Research and Graduate Education Center in Ensenada, Mexico, and the D.Sc. degree in Electrical Engineering from the George Washington University, Washington, DC. From 1997 to 2000 he worked as a Research Associate at the Advanced Communications Engineering Centre of the University of Western Ontario, London, Ontario, Canada. From May to December 2000, he worked as a Postdoctoral Fellow at the Broadband Wireless and Internetworking Research Laboratory of the University of Ottawa. Since December 2000, Dr. Gallardo has been with the Electronics and Telecommunications Department of CICESE Research Center, where he is a full professor. His main areas of interest are traffic modeling, traffic control, as well as simulation and performance evaluation of broadband communications networks, with recent emphasis on wireless local area networks (WLANs) and wireless sensor networks (WSNs).

Paúl Medina received the B.Eng. degree from the Sonora Institute of Technology, Obregon, Mexico, and the M.Sc. degree from CICESE Research and Graduate Education Center, Ensenada, Mexico, both in Electrical Engineering. From July to September 2005, he worked as a Research Associate at the Broadband Wireless and Internetworking Research Laboratory of the University of Ottawa, Canada. Mr. Medina is currently with CENI^{2}T, Ensenada, Mexico, working as a lead engineer in projects related to routing and access control in wireless sensor networks, as well as IP telephony over wireless LANs.

Weihua Zhuang received the B.Eng. and M.Eng. degrees from Dalian Maritime University, Liaoning, China, and the Ph.D. degree from the University of New Brunswick, Canada, all in electrical engineering. Since October 1993, she has been with the Department of Electrical and Computer Engineering, University of Waterloo, ON, Canada, where she is a full professor. She is a co-author of the textbook *Wireless Communications and Networking* (Prentice Hall, 2003). Dr. Zhuang received the Outstanding Performance Award in 2005 from the University of Waterloo, and the Premier's Research Excellence Award in 2001 from the Ontario Government. She is an Editor/Associate Editor of *IEEE Transactions on Wireless Communications, IEEE Transactions on Vehicular Technology, EURASIP Journal on Wireless Communications and Networking,* and *International Journal of Sensor Networks.* Her current research interests include multimedia wireless communications, wireless networks, and radio positioning.