WIRELESS COMMUNICATIONS AND MOBILE COMPUTING
Wirel. Commun. Mob. Comput. (2010)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/wcm.1046

RESEARCH ARTICLE

On centralized schedulers for 802.11e WLANs distribution *versus* grouping of resources allocation

Daniel Camps Mur[1]*, Xavier Pérez-Costa[1], Vladimir Marchenko[2] and Sebastià Sallent Ribes[3]

[1] NEC Europe Laboratories, Germany
[2] RWTH Aachen University, Aachen, Germany
[3] Polytechnic University of Catalonia (UPC), Spain

ABSTRACT

Wireless LAN is becoming a pervasive wireless access technology that can be found in almost any mobile device such as laptops, PDAs, portable game consoles and mobile phones. Each of these groups of devices have a different set of requirements according to their intended use and applications but most of them share two main requirements: QoS support to satisfy applications' demands and power saving functionality to achieve an operating time according to users' expectations. IEEE 802.11e defines two centralized solutions in order to address these problems: Hybrid Coordination Channel Access (HCCA) for QoS and Scheduled Automatic Power Save Delivery (S-APSD) for power saving. The focus of our work in this paper is the analysis and evaluation of a proposed centralized scheduler that makes use of both aforementioned IEEE 802.11e QoS and power saving solutions. Our contributions are as follows: (i) Design and analytical modeling of a proposed centralized scheduler (DRA) that maximizes the minimum distance between the resource allocations with pseudo-polynomial complexity, (ii) Extensive performance evaluation of the QoS and power saving benefits of the *Distribution* proposal (DRA) as compared to a generic *Grouping* one (GRA), and (iii) Evaluation of the complexity and scalability of the proposal to assess its feasibility in practice.

KEYWORDS
distributed allocation; HCCA; power saving; QoS; wireless LAN

***Correspondence**
Daniel Camps Mur, NEC Europe Laboratories, Kurfuersten-Anlage 36, Heidelberg, Germany.
E-mail: camps@neclab.eu

1. INTRODUCTION

The Wireless LAN (WLAN) technology has built in its success on personal computers and laptops to expand into an ever increasing number of mobile devices as mobile phones, PDAs, music players, and portable game consoles. According to technology forecasts this trend is far from having reached the limit of its potential market and for instance the number of Wi-Fi enabled mobile phones is expected to grow from about 141 Million shipments in 2009 to 520 Million by 2014 [1]. However, this success poses new challenges to the WLAN technology in order to satisfactorily fulfill the requirements of an increasingly wider range of devices and users. Within these requirements our work in this paper focuses in two main areas that are key to guarantee users' satisfaction: Quality of Service (QoS) and battery lifetime. QoS support is expected to become more and more important as users start to use WLAN not only for web browsing and file exchanges but also for real-time applications with stringent QoS requirements as voice, video conferencing, and online gaming.

In order to address the need for QoS provisioning, IEEE developed the 802.11e [2] standard. This standard defines the Hybrid Coordination Function (HCF), that includes two different access methods: a contention-based channel access method called the Enhanced Distributed Channel Access (EDCA) and a contention-free channel access method referred to as HCF Controlled Channel Access (HCCA). While EDCA is a distributed scheme that provides prioritized QoS, HCCA is a centralized scheme that allows parameterized QoS provisioning. A thorough overview of the 802.11e QoS enhancements can be found in Reference [3].

Regarding power saving mechanisms to extend battery life, IEEE 802.11 [4] defines a power save mode that allows stations to switch off their radio during inactivity periods in order to save power. IEEE 802.11e defines an enhancement of the 802.11 power save mode, Automatic Power

Save Delivery (APSD), that takes advantage of the QoS mechanisms of 802.11e in order to provide an improved QoS experience. Two modes of operation are available for APSD: Unscheduled and Scheduled. Unscheduled APSD (U-APSD) can be used only by stations accessing the channel using EDCA while Scheduled APSD (S-APSD) can be used with both access mechanisms, EDCA and HCCA. An overview of 802.11 power save mode and U-APSD can be found in Reference [5] and an overview of S-APSD can be found in Reference [6].

While the distributed QoS and power saving mechanisms defined in 802.11e, EDCA for QoS and U-APSD for power saving, are starting to be widely deployed,[†] the centralized schemes, HCCA for QoS and S-APSD for power saving, are expected to cover in the future further needs of devices on QoS guarantees (e.g., hard bounds on delay or jitter) and battery lifetime. Therefore, the focus of our work in this paper is the analysis and evaluation of a proposed resource allocation algorithm for the centralized QoS and power saving capabilities of 802.11e.

The rest of the paper is organized as follows. Section 2 summarizes previous research work in resource allocation algorithms for HCCA and S-APSD. In Section 3 we describe and model analytically our proposed solution for *Distributing* in time as much as possible Resource Allocations (DRA). Section 4 presents a *Grouping* algorithm that will be used for comparison reasons (GRA). Following that, in Section 5 we evaluate the performance of our proposed DRA scheduler as compared to the GRA one in terms of QoS and power consumption for both HCCA and EDCA. Finally Section 6 summarizes our findings and concludes this paper.

2. HCCA AND S-APSD OVERVIEW

The HCF Controlled Channel Access (HCCA) operation is possible in two different ways: the Contention Free Period (CFP), where the Hybrid Coordinator (HC), usually residing in the Access Point (AP), reserves the channel only for HCCA traffic, and the Contention Period (CP) where both EDCA and HCCA traffic can access the channel. In the Contention Period a higher priority is granted to HCCA traffic by allowing the HC to use a shorter inter frame space to access the medium than contention based traffic, the Priority Inter Frame Space (PIFS).

Within the context of HCCA, MAC Service Data Units (MSDUs) are associated to flows and flows are associated to stations. In order to set up a new flow, a station has to submit a Traffic Specification (TSPEC) message to the AP. In the TSPEC message the station describes its traffic flow,[‡] and specifies the delay bound (maximum time before which a MSDU belonging to this flow has to be successfully delivered) and the radio conditions[§] under which this flow is expected to operate.

Two main functions have to be designed in the AP for the purpose of HCCA. The *Admission Control* function that decides if new flows can be accepted or not, and the *Scheduling* function that is in charge of fulfilling the QoS requirements of admitted flows. For the purpose of scheduling, the AP can directly transmit data frames in Downlink or use a special frame, *CF-Poll*, to poll flows in Uplink granting them a certain amount of time, Transmission Opportunity (TXOP), to access the channel. Our focus in this paper is on the HCCA Scheduling function.

Regarding power consumption, S-APSD is a simple but yet powerful extension that allows HCCA stations to switch to a sleep state of low power consumption between successive transmissions. During the initial TSPEC negotiation the AP conveys in the TSPEC response a Schedule Element that contains two main parameters: the *Service Start Time* (SST) and the *Service Interval* (SI). Thus, a S-APSD station awakes every SI after the initial awake time (SST) in order to receive data in Downlink or be polled in Uplink. At every scheduled time a Service Period (SP) starts that only finishes when the AP sends to the station a Data or QoS Null frame with the End of Service Period (EOSP) bit set to 1. A station transmitting different HCCA flows can maintain a Schedule Element for each flow. Figure 1 illustrates the operation of S-APSD over HCCA.

Next, we review some of the main approaches to HCCA scheduling presented in the literature.

2.1. Related work

To the best of the authors' knowledge most of the approaches to HCCA scheduling presented in the literature share the basic idea of polling an admitted flow at regular intervals in order to fulfill the delay and bandwidth requirements defined in a Traffic Specification message (TSPEC) when setting up a flow. In the rest of the paper we will refer to the time interval used by an Hybrid Coordination Fuction (HCF) to poll a station for a flow as the SI of this flow. The most common approach in the literature to schedule HCCA resource allocations is what we defined as *Grouping* approach which was inspired by the example scheduler described in the 802.11e standard [2]. The idea behind this scheduler is to define a basic service interval common to all flows of all stations. This service interval is selected in order to fulfill the most stringent delay requirement among the different flows. Thus, at intervals defined by the basic service interval, the Access Point (AP) starts to sequentially poll all associated flows. The main problem of this approach is that a common service interval for all flows in general

[†] Wi-Fi Alliance certifications already exist for EDCA (WMM) and U-APSD (WMM-PS) [7].

[‡] The Token Bucket model defined in Reference [8] is used for this purpose.

[§] Radio conditions are conveyed with two parameters: (i) Modulation and Coding Scheme (MCS) and (ii) Surplus Bandwidth Allowance (SBA) (which accounts for possible retransmissions).

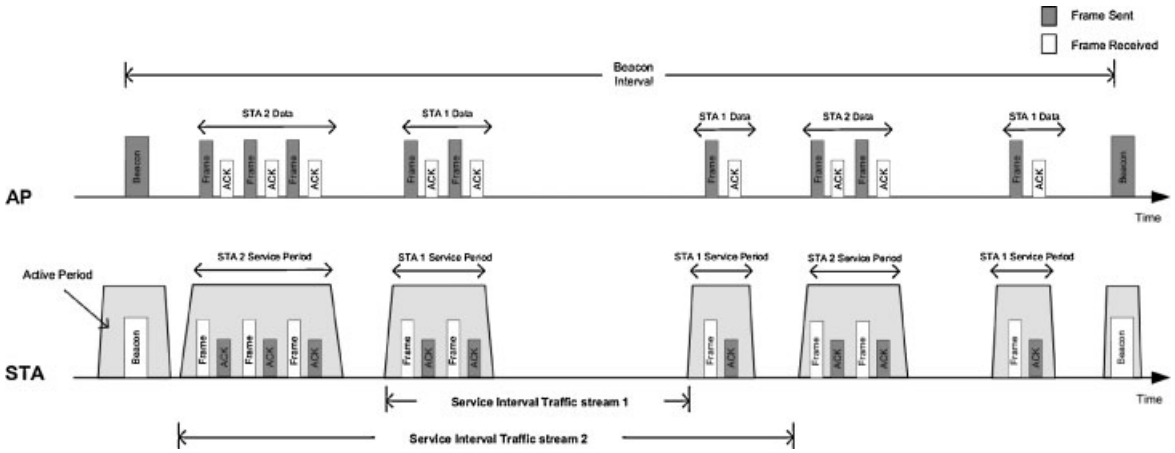


Figure 1. Example of S-APSD over HCCA operation: stations awake at their scheduled times to serve their traffic streams.

cannot be guaranteed to match the packet generation rate of each individual flow. In this case bandwidth may be wasted by unnecessary polling of stations. References [9] and [10] are relevant examples of proposed implementations of this approach.

In contrast to the *Grouping* solution, another approach called *Earliest Due Date* (EDD), which was first proposed in Reference [11], does not try to avoid collisions between the polling times of different flows but instead uses Earliest Deadline First (EDF) to resolve such collisions. A deadline, related with the service interval and the delay bound, is associated to each flow and is used to decide which flow to poll first in case there is more than one flow pending to be polled at the same point in time. Among these flows the EDD scheduler selects the one with the closest deadline. This solution removes the need to impose a common basic service interval as compared to the grouping approach.

In our previous work [12], we proposed an algorithm that determines the Service Start Time (SST) of each flow in order to distribute in time as uniformly as possible the allocation of resources for the different flows. We refer to this approach as the *Distribution* approach. Like in the EDD scheduler, no requirement needs to be imposed on the service intervals demanded by the different flows. In addition, this scheme minimizes the chances of having more than one HCCA flow pending to be polled at the same point of time. Our *Distribution* approach is related to the concept of *Spectrum Load Smoothing* (SLS) proposed in Reference [13]. SLS allows cognitive radios to support QoS guarantees in a distributed way by coordinating their transmissions across a certain *smoothing period*, which relates to the QoS requirements of each station. Notice though, that spreading in time transmissions to/from a station implementing S-APSD increases its power consumption, because the station has to stay awake until its correspondent transmissions complete. For this reason, we opt to distribute resource allocations by shifting the starting times of the different flows, while trying to complete them as soon as possible once the corresponding station is awake. If stations would tolerate their transmissions to be smoothed over a certain period, our distribution approach and SLS could be applied together to further smooth in time resource allocations. In this paper though, we have not considered this scenario.

As it can be observed, the *Grouping* and *Distribution* approaches pursue opposite objectives. The *Grouping* approach *reserves* the channel for HCCA only once every basic service interval but potentially for a long time since all HCCA flows need to be polled. Instead, in the *Distribution* approach the number of HCCA allocations might be larger but their *individual* duration is in general shorter. The EDD aproach cannot be classified in any of the two groups since the time when each flow makes the initial request together with the service intervals used determines whether HCCA allocations are rather grouped or distributed. Figure 2 provides an example of HCCA allocations distribution according to the *Grouping* and *Distribution* approaches.

We argue that distributing, rather than grouping, the allocation of resources in a wireless medium is a more efficient approach in several aspects. The reason is that establishing hard bounds on the transmission time required to accomodate the QoS of a flow in a WLAN network is a difficult task, because variations in the wireless channel may require to increase the original resources allocated to individual flows by means of using a more robust Modulation and Coding Scheme (MCS) or retransmitting packets. In order to better illustrate the inherent variability in the wireless medium, Table I reports the average and maximum service times required to service every 40 ms a sample video flow[‖] with average and peak rates of 256 kbps and 2 Mbps respectively under different transmission parameters.

Observing the potential range of variation in the resources required to accomodate a single flow, providing deterministic guarantees by dimensioning the system under the worst case situation, i.e., peak rate and slowest MCS, may result into extremely limited system capacities. Instead, we claim

[‖] Real video traces obtained from Reference [14] are used.

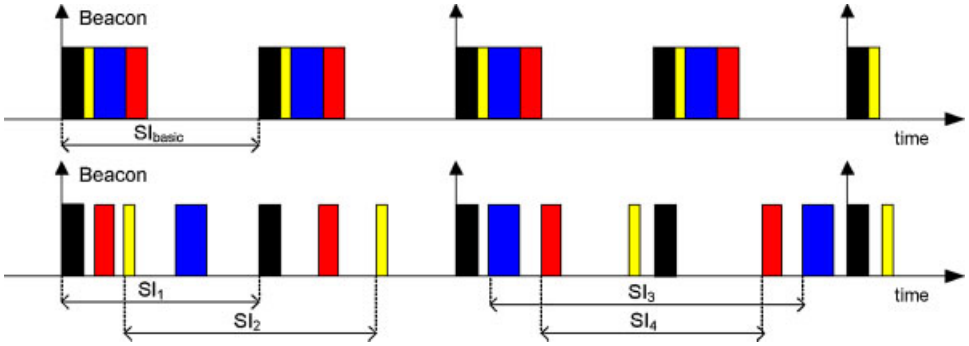


Figure 2. Example of HCCA allocations with a *Grouping* (upper graph) and a *Distribution* (lower graph) approach.

Table I. Required video allocation time for various PHY conditions.

PHY Rate	Average allocation	Peak allocation
54 Mbps	0.22 ms	1.68 ms
24 Mbps	0.47 ms	3.6 ms
6 Mbps	1.83 ms	14.1 ms

that variations in the wireless channel can be better accomodated using a distributed approach, where the scheduling time of different flows are separated as much as possible and hence the probability of a planned allocation having to wait for the completion of a previous allocation is reduced. The previous fact should be indeed benefitial for both QoS and power consumption.

Based on the previous arguments we focus our work in this paper in the design and evaluation of an enhancement of our *Distribution* algorithm proposed in Reference [12] that significantly reduces its computational load and in the evaluation of the performance improvements to be expected as compared to a *Grouping* scheduler.

3. DISTRIBUTED RESOURCE ALLOCATION (DRA)

The problem of scheduling a set of periodic flows, or tasks, has been thoroughly researched in the literature, specially in the field of real time operating systems. In this paper we propose a generic *Distributed Resource Allocation* (DRA) scheduling algorithm that maximizes the minimum distance between the serving times of different flows. The DRA algorithm is based on our previous proposal presented in Reference [12] which required the exploration of the least common multiple (LCM) of the different flows' service intervals and enhances it by: (i) considering the serving time of each flow in the scheduling decision and (ii) achieving a pseudo-polynomial complexity by removing the need of exploring the LCM of the flows' service intervals.

3.1. Defining a way to distribute resource allocations

As discussed in the previous section our goal can intuitively be stated as defining an algorithm that separates as much as possible consecutive resource allocations in the channel. In this section we show that our desired *separation* or *distribution* can be achieved by means of an algorithm that maximizes the minimum distance between allocated flows.

To start off, a better understanding on the concept of distance between periodic flows is needed. Figure 3 depicts the occurences in the channel of two periodic flows with periods, or service intervals in HCCA terminology, SI_1 and SI_2, which have starting times SST_1 and SST_2. The figure also depicts the distance between consecutive allocations of the two flows $d_{2,1}(k)$. Indeed, one can express the distance between an occurrence of flow 2 and the previous occurrence of flow 1 as:

$$d_{2,1}(k) = (d_{2,1}(0) + k\mathrm{SI}_2) \bmod \mathrm{SI}_1 \tag{1}$$

Where $d_{2,1}(0)$ is a reference distance between two resource allocations, $d_{2,1}(0) = \mathrm{SST}_2 - \mathrm{SST}_1$ in Figure 3.

Later in this section, it will be formally shown that the previous sequence, $d_{2,1}(k)$, is indeed periodic and that its elements can be expressed as $d_{2,1}(j) = d_{\min_{2,1}}$

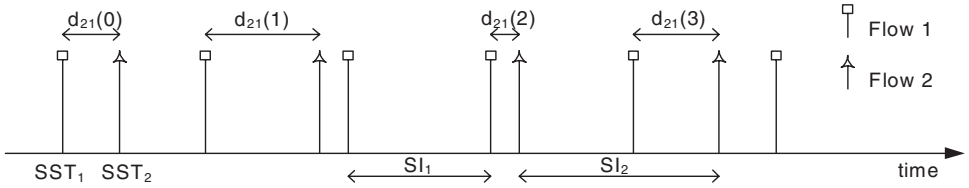


Figure 3. Distances between resource allocations.

$+ j \cdot \gcd(SI_1, SI_2) \quad j \geq 0$, where $d_{\min_{2,1}} = d_{2,1}(0) \bmod \gcd(SI_1, SI_2)$, and gcd stands for greatest common divisor. Thus, an objective way of increasing the separation between the service times of flows 1 and 2 is to increase $d_{\min_{2,1}}$ and/or $\gcd(SI_1, SI_2)$.

Maximizing $\gcd(SI_1, SI_2)$ requires modifying the service intervals used to poll each flow, and is intuitively achieved by setting both service intervals as similar as possible, for instance making them all multiple of a certain value SI_{basic}, being this value as high as possible. However, manipulating the service intervals assigned to each flow in order to maximize the separation between consecutive allocations in the channel can have counter effects. It may turn into wasted bandwidth if the polling interval does not efficiently match the application generation rate, i.e., in the case of voice or video codecs, or it may result in increased delays if adjusting to a multiple of SI_{basic} results in a polling interval above the desired delay bound. Therefore, in this paper we consider that the service interval used to poll each flow will be determined based on a set of different requirements (QoS, bandwidth efficiency, power consumption,etc.), and given the selected service intervals we attempt to separate the allocation of flows as much as possible.

If we consider the service intervals SI_1 and SI_2 as given, the only way to increase the separation between any two allocations of flows 1 and 2 in the previous example is to maximize the minimum distance, $d_{\min_{2,1}}$. Looking at the definition of $d_{\min_{2,1}}$ we can see that although $d_{\min_{2,1}}$ is bounded by $\gcd(SI_1, SI_2)$ it can be maximized by properly setting $d_{2,1}(0)$ which is defined by the starting time of each flow, SST_i. Note that SST_i can be determined by the scheduler and conveyed to the flows in HCCA during the TSPEC negotiation through the Schedule Element.

The previous example is too simplistic in the sense that we consider only two flows in the channel. Consider now the case that N periodic flows are already scheduled and a new flow requests entry to the system that has therefore to be granted a certain starting time SST_i. Clearly, in a general setting it is not possible to select a SST_i for the new flow that maximizes the distances between the polling times of the new flow and all of the already scheduled flows at the same time. In this case the distances between the new flow and each of the already scheduled flows could be weighted to decide on which distances are more important to maximize. However, a compelling argument from Real Time Scheduling Theory (RTST) drives us toward again trying to maximize the minimum of those distances. The problem of scheduling task sets in RTST can be stated in very similar terms to the problem of scheduling flows in HCCA. It is known then from the feasibility tests devised in RTST that if a deadline is associated to each flow and Earliest Deadline First (EDF),[¶] which has been proven optimal under many RTST settings, is used as scheduling discipline, then the *critical instant* [15] occurs when all flows try to access the shared resource at the same time. The critical instant is defined as the release time of a flow for which the response time, and hence the probability of a deadline violation, is maximized. Maximizing minimum distances is therefore an intuitive way of moving away from this worst-case situation trying hence to maximize the number of admitted flows if an admission control for HCCA would be defined similar to the feasibility tests employed in RTST.

Next, we present our DRA algorithm that allocates a start time for a new flow requesting access to the system such that the minimum distance between the service periods of this new flow and the service periods of the already scheduled flows is maximized. Although defined in the context of HCCA, the presented DRA algorithm can be used in any context where scheduling of periodic flows is required.

3.2. The two flows case

To facilitate the understanding of the analysis of our proposal let us first consider a system consisting of two flows. The first flow has already been scheduled and is periodically served according to its Service Start Time (SST) SST_1, Service Interval SI_1 and serving time $TXOP_1$.[#] A second flow requests at a certain point of time, which is considered to be $t = 0$, to be scheduled in the system with service interval SI_2 and serving time $TXOP_2$. The goal of our algorithm is to find the service start time for the requesting flow, SST_2, that maximizes the minimum distance between the serving times of flows 1 and 2. In the rest of the paper we refer to the regular instants where a flow is scheduled to be served as the *release* instants of this flow. Figure 4 represents flows 1 and 2 and $next\_rel\_time(1)$ is a variable defined starting from $t = 0$ that contains the next release time of flow 1.

Let us define the *left* distances between flow 2 and flow 1, $d_{l_{2,1}}(k)$ where $k \in \mathbb{N}$, as the time differences between the $k - th$ release time of flow 2 and the last release time of flow 1. Similarly, the *right* distances, $d_{r_{2,1}}(k)$, are defined as the time differences between a release time of flow 2 and the next release time of flow 1. Additionally let us define the *left* and *right effective* distances, $d_{l\_\mathrm{eff}_{2,1}}(k)$ and $d_{r\_\mathrm{eff}_{2,1}}(k)$, as the difference between the end of the serving time of one flow and the release time of the other flow. All these distances are depicted in Figure 4. In addition, Table II contains a summary of the main variables used throughout this section.

From our definitions it is immediate to derive that $d_{l\_\mathrm{eff}_{2,1}}(k) = d_{l_{2,1}}(k) - TXOP_1$ and $d_{r\_\mathrm{eff}_{2,1}}(k) = d_{r_{2,1}}(k) - TXOP_2$. Therefore, in order to maximize the minimum effective distance between flows 1 and 2, we can focus our analysis on maximizing the minimum *left* and *right* distances.

[¶] Formally, preemptive EDF has to be considered although maximizing the minimum distance is also a common-sense approach in the case of non preemptive EDF [16].

[#] *TXOP* stands for Transmission Opportunity and is the transmission time granted to a polled flow in HCCA.

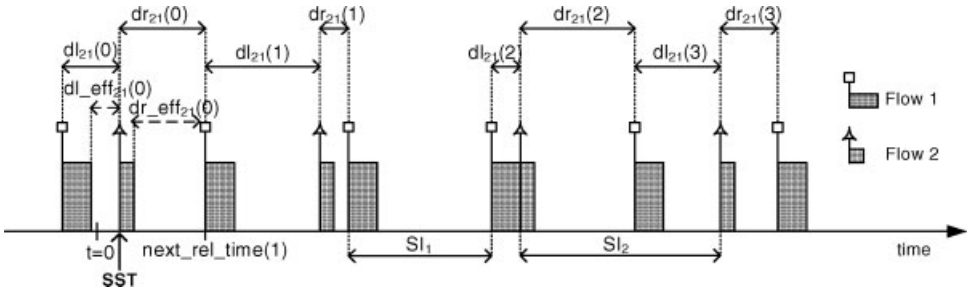


Figure 4. Distances between release times.

Table II. Variables used in the analysis.

SI_i	Service interval used for flow i
$TXOP_i$	Transmission Opportunity used for flow i
$d_{l_{i,j}}(k)$	Sequence of left distances of flows i and j
$d_{r_{i,j}}(k)$	Sequence of right distances of flows i and j
$d_{l\_\text{eff}_{i,j}}(k)$	$d_{l_{i,j}}(k) - TXOP_j$
$d_{r\_\text{eff}_{i,j}}(k)$	$d_{r_{i,j}}(k) - TXOP_i$
$d_{l\_\text{min}_{i,j}}$	Minimum left distance of flows i and j
$d_{r\_\text{min}_{i,j}}$	Minimum right distance of flows i and j
$d_{l\_\text{eff\_min}_{i,j}}$	$d_{l\_\text{min}_{i,j}} - TXOP_j$
$d_{r\_\text{eff\_min}_{i,j}}$	$d_{r\_\text{min}_{i,j}} - TXOP_i$
$\phi_{i,j}$	Reference distance between flows i and j

Our goal is thus to find the SST of flow 2 that maximizes the minimum distance between the release times of flows 1 and 2. Notice for this purpose that $d_{l_{2,1}}(k)$ and $d_{r_{2,1}}(k)$ can be expressed as:

$$\begin{aligned} d_{l_{2,1}}(k) &= (d_{l_{2,1}}(0) + k\text{SI}_2) \bmod \text{SI}_1 \\ d_{r_{2,1}}(k) &= \text{SI}_1 - d_{l_{2,1}}(k) \end{aligned} \quad (2)$$

Where $d_{l_{2,1}}(0)$ is the initial *left* distance taken as reference.

Then, the first step toward our goal is to express the minimum value of $d_{l_{2,1}}(k)$ and $d_{r_{2,1}}(k)$, which we want to maximize, as a function of the SST to be assigned to flow 2. For that purpose notice that $d_{l_{2,1}}(k)$ can be expressed as:

$$d_{l_{2,1}}(k) = d_{l_{2,1}}(0) + k\text{SI}_2 - m\text{SI}_1 \quad (3)$$

Where $k, m \in \mathbb{Z}$. If we now separate the right hand size of the previous equation into terms that can be divided by $d = \gcd(\text{SI}_1, \text{SI}_2)$, where gcd stands for greatest common divisor, we obtain:

$$d_{l_{2,1}}(k) = d_{l_{2,1}}(0) \bmod d + pd + k\text{SI}_2 - m\text{SI}_1 \quad (4)$$

$$d_{l_{2,1}}(k) - d_{l_{2,1}}(0) \bmod d = jd \quad (5)$$

Where $k, m, p, j \in \mathbb{Z}$. Thus, considering that in Equation (2), $d_{l_{2,1}}(k) \leq \text{SI}_1$:

$$d_{l_{2,1}}(k) = d_{l_{2,1}}(0) \bmod d + jd, \quad 0 \leq j < N = \frac{\text{SI}_1}{d} \quad (6)$$

Therefore, the minimum value of $d_{l_{2,1}}(k)$ is $d_{l\_\text{min}_{2,1}} = d_{l_{2,1}}(0) \bmod \gcd(\text{SI}_1, \text{SI}_2)$. In addition, $d_{r\_\text{min}_{2,1}}$ can be obtained in the following way. Notice that $d_{r\_\text{min}_{2,1}} = \text{SI}_1 - d_{l\_\text{max}_{2,1}}$, thus if there are only $N = \frac{\text{SI}_1}{d}$ different values for $d_{l_{2,1}}(k)$, then $d_{l\_\text{max}_{2,1}} = \text{SI}_1 - (\gcd(\text{SI}_1, \text{SI}_2) - d_{l\_\text{min}_{2,1}})$, and $d_{r\_\text{min}_{2,1}} = \gcd(\text{SI}_1, \text{SI}_2) - d_{l\_\text{min}_{2,1}}$.

A simple transformation can now be used to express the minimum *left* and *right* distances as a function of the SST to be assigned to flow 2. If this SST equals $next\_rel\_time(1)$ the minimum *left* distance will be zero, hence:

$$d_{l\_\text{min}_{2,1}} = (\text{SST} - \phi_{2,1}) \bmod \gcd(\text{SI}_1, \text{SI}_2) \quad (7)$$

Where $\phi_{2,1}$ is an initial shift defined as $\phi_{2,1} = next\_rel\_time(1) \bmod \gcd(\text{SI}_1, \text{SI}_2)$.

Having found the expression of the minimum value for the *left* and *right* distances it is immediate to obtain an expression for the minimum *left* and *right effective* distances since $d_{l\_\text{eff\_min}_{2,1}} = d_{l\_\text{min}_{2,1}} - \text{TXOP}_1$ and $d_{r\_\text{eff\_min}_{2,1}} = d_{r\_\text{min}_{2,1}} - \text{TXOP}_2$.

Figure 5 illustrates the minimum values of $d_{l\_\text{eff}_{2,1}}(k)$ and $d_{r\_\text{eff}_{2,1}}(k)$ as a function of the selected SST for given values of TXOP_1 and TXOP_2 ($\text{TXOP}_1 > \text{TXOP}_2$ in the figure). The minimum *left* and *right* effective distances achieve maximum values of $\gcd(\text{SI}_1, \text{SI}_2) - \text{TXOP}_1$ and $\gcd(\text{SI}_1, \text{SI}_2) - \text{TXOP}_2$ respectively, but these maximum values do not occur at the same time.

Considering again Figure 5, the overall minimum distance, i.e., $\min\{d_{l\_\text{eff\_min}_{2,1}}, d_{r\_\text{eff\_min}_{2,1}}\}$, is a periodic set of triangular shapes of unitary slope. This minimum distance function has period $T = \gcd(\text{SI}_1, \text{SI}_2)$ and presents discontinuities at points $\phi_{2,1} + k \cdot \gcd(\text{SI}_1, \text{SI}_2)$, $k \in \mathbb{N}$, which we refer hereafter as *critical points*. Thus, the values of SST that

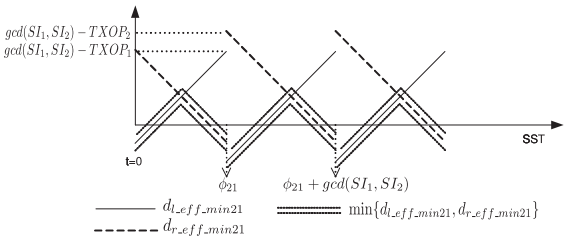


Figure 5. Minimum *left* and *right* effective distances.

maximize the minimum distances between any two release times of flow 1 and flow 2 are:

$$\mathrm{SST}_{OPT} = \phi_{2,1} + k \cdot \frac{\gcd(\mathrm{SI}_1, \mathrm{SI}_2)}{2} + \\ + \frac{\mathrm{TXOP}_1 - \mathrm{TXOP}_2}{2} \quad (8)$$

Where k is an odd integer, $k = \{\ldots, -1, 1, 3, 5, 7, \ldots\}$.

3.3. The *N* flows case

Based on the 2 flows case analysis, we extend our results now to the N flows case which consists in N periodic flows already scheduled in a system with service intervals SI_i and serving times TXOP_i, where $i = 1 \ldots N$, and a new flow arriving at time $t = 0$ which requires to be scheduled with service interval SI_{N+1} and serving time TXOP_{N+1}. As in the previous case, our goal is to find the initial release time for the new flow, SST_{N+1}, that maximizes the minimum effective distance between this flow and the already scheduled flows.

Defining as before the *left* and *right* distances of the new flow, $N + 1$, with each of the already scheduled flows, i where $i = 1 \ldots N$, we can divide the N-flows problem into N different 2-flow problems:

$$d_{l\_\min_{N+1,i}} = (\mathrm{SST} - \phi_{N+1,i}) \bmod \gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1})$$
$$d_{r\_\min_{N+1,i}} = \gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1}) - d_{l\_\min_{N+1,i}}$$

With $\phi_{N+1,i} = next\_rel\_time(i) \bmod \gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1})$.

Similarly, the *left* and *right* effective minimum distances can be defined as $d_{l\_\mathrm{eff}\_\min_{N+1,i}} = d_{l\_\min_{N+1,i}} - \mathrm{TXOP}_i$ and $d_{r\_\mathrm{eff}\_\min_{N+1,i}} = d_{r\_\min_{N+1,i}} - \mathrm{TXOP}_{N+1}$. Figure 6 represents the minimum *left* and *right* effective distances in a system where two flows were already scheduled and a new flow has requested access. In the figure $d_{l\_\mathrm{eff}\_\min_{N+1,i}}$ and $d_{r\_\mathrm{eff}\_\min_{N+1,i}}$ are not depicted separately but instead we directly plot $\min\{d_{l\_\mathrm{eff}\_\min_{N+1,i}}, d_{r\_\mathrm{eff}\_\min_{N+1,i}}\}$ for each scheduled flow. Notice in the figure that $\mathrm{TXOP}_1 > \mathrm{TXOP}_3 = \mathrm{TXOP}_2$.

Notice that in case of having N flows, our goal is to find SST_{N+1} that maximizes $\min\{d_{l\_\mathrm{eff}\_\min_{N+1,i}}, d_{r\_\mathrm{eff}\_\min_{N+1,i}}\}$, which we refer to as the *absolute* minimum distance.

Thus, realizing that the absolute minimum distance is also a periodic set of triangular shapes of unitary slope, a simple algorithm that finds the SST_{N+1} that maximizes this distance can be implemented in the following way:

(1) Compute $\gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1})$ for all the N already scheduled flows, $0 < i \leq N$.
(2) Compute the period of the absolute minimum distance, $T' = \mathrm{lcm}(\gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1}), \ldots, \gcd(\mathrm{SI}_N, \mathrm{SI}_{N+1}))$.
(3) For each already scheduled flow generate all critical points, $\phi_{N+1,i} + k \cdot \gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1})$, contained in T'.
(4) Define a sorted list L containing all critical points.
(5) Define a function F that operating on list L obtains the SST that maximizes the minimum effective distance.

In order to find the optimum SST, the function F needs to go through all the critical points contained in L and obtain the maximum of the triangular shape between each two consecutive points. Hence, the function F needs to examine in each critical point the value of $\min\{d_{l\_\mathrm{eff}\_\min_{N+1,i}}, d_{r\_\mathrm{eff}\_\min_{N+1,i}}\}$ for all N already scheduled flows and find out the initial value of the triangular shape. Our implementation of function F has a worst-case time complexity bounded by $O(M(N + 1))$, where M is the maximum size of list L and N is the number of already scheduled flows. For the sake of space and clarity of the explanation our implementation of function F is not included here.**

A pseudo-code implementation of our algorithm to distribute resource allocations is shown in Algorithm 1. In the rest of the paper we refer to this algorithm as the *DRA* algorithm.

It is interesting to discuss how the DRA algorithm behaves in an overloaded situation, since as the number of flows increases, or flows with high $TXOP$s are considered, overlapping between flows may be unavoidable. This situation though does not impose any constraint on the behavior of DRA. If overlapping occurs the effective minimum distance will become negative, as illustrated in Figure 6, and DRA will simply select the starting service time that results in the minimum amount of overlapping. However, in order to limit the amount of overlapping such that no harm is done on the QoS requirements of a specific flow, an admission control module complementing DRA would be needed.

Finally, to illustrate the distribution achieved under a max–min distance criteria Figure 7 depicts two example allocations. The upper part of the figure shows the resulting distribution when scheduling three flows with a period of 40 ms, where flow i enters the system before flow j if $i < j$. The lower part of the figure shows the resulting allocation when scheduling, again in the specified order, three flows of periods 40, 60, and 80 ms. It is worth noticing that in a general setting where each flow has a different period, the resulting distribution depends on the order of arrival of

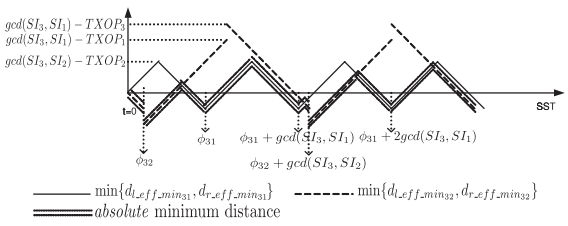


Figure 6. Minimum *left* and *right* distances with 3 flows in the system.

** A pseudo-code description of function F and an analysis of its complexity can be found at: http://147.83.113.45/wireless/F_function_description.pdf or it can be made available from the authors upon request.

Algorithm 1 Distributed resource allocation algorithm (DRA)

–Variables definition

$SI_i \leftarrow$ Service interval of flow i (obtained from the TSPEC).

$next\_rel\_time(i) \leftarrow$ Next release time of flow i.

$list\_critical\_points \leftarrow$ List that contains the critical points of the absolute minimum distance function.

–Routine for scheduling a new flow

1: **for** $i = 1$ to N **do**
2: $partial\_gcd(i) \leftarrow gcd(SI_{N+1}, SI_i)$
3: $\phi_{N+1,i} \leftarrow next\_rel\_time(i) \bmod gcd(SI_{N+1}, SI_i)$
4: **end for**
5: $T' \leftarrow lcm(partial\_gcd(1), \ldots, partial\_gcd(N))$
6: $\phi_{min} = min\{\phi_{N+1,1}, \ldots, \phi_{N+1,N}\}$
7: $j \leftarrow 0$
8: **for** $i = 1$ to N **do**
9: $k \leftarrow 0$
10: **while** $\phi_{N+1,i} + k \cdot partial\_gcd(i) \leq \phi_{min} + T'$ **do**
11: $list\_critical\_points(j) \leftarrow \phi_{N+1,i} + k \cdot partial\_gcd(i)$
12: $k \leftarrow k + 1$, $j \leftarrow j + 1$
13: **end while**
14: **end for**
15: $L \leftarrow$ **Quicksort** $list\_critical\_points$ in increasing order
16: $SST_{OPT} \leftarrow F(L)$.

the flows. Indeed, under HCCA it is possible for the AP to reschedule the allocation times of previously scheduled flows when a new flow wants to enter the system by sending a new Schedule Element to each existent flow in the system, hence paying a price in increased signaling and complexity. In this paper though we do not explore the problem of finding, given a set of N flows to be scheduled, the scheduling order that results in the maximum separation under DRA.

3.4. Complexity of DRA

In this section we analyze the complexity of our proposed DRA algorithm in order to provide an upper bound on the computational load to be expected when implementing it in practice.

Looking at Algorithm 1 the worst case time complexity of DRA can be analysed in the following way. When a new flow has to be scheduled, DRA first computes $\gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1}), \forall i = 1 \ldots N$, having a complexity of $O(N \log \mathrm{SI}_{\max})$, where $\mathrm{SI}_{\max} = \max_i \mathrm{SI}_i$ and N is the number of flows already scheduled in the system. After that, computing T' requires again a complexity of $O(N \log \mathrm{SI}_{\max})$ and computing $\phi_{\min}$ requires a complexity of $O(N)$. After these initial operations the algorithm constructs the list containing the critical points of the absolute minimum distance function, *list_critical_points*, which takes complexity $O(M)$, being M the number of elements in this list. Once *list_critical_points* is populated, it can be sorted with a worst case[††] complexity of $O(M \log M)$. Finally, the function F computes for each critical point the value of the absolute minimum distance,[‡‡] taking a worst case complexity of $O(M(N + 1))$.

Therefore, a key parameter to evaluate the complexity of the DRA algorithm is the size M of *list_critical_points*. In order to obtain an upper bound on M an upper bound on the period of the absolute minimum distance function (T') is needed:

$$T' = \mathrm{lcm}(\gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1}), \ldots, \gcd(\mathrm{SI}_N, \mathrm{SI}_{N+1})) \leq \mathrm{SI}_{N+1}$$

Hence, the maximum size of *list_critical_points* can be upper bounded as:

$$M = \sum_{i=1}^{N} \frac{T'}{\gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1})} \leq \leq \mathrm{SI}_{N+1} \sum_{i=1}^{N} \frac{1}{\gcd(\mathrm{SI}_i, \mathrm{SI}_{N+1})} \quad (9)$$

Therefore, $M \leq \mathrm{SI}_{N+1} N$. Based on the previous analysis we can express the overall worst case time complexity of DRA as bounded by $O(\mathrm{SI}_{N+1} N(N + \log \mathrm{SI}_{N+1} N) + 2N \log \mathrm{SI}_{\max})$. Notice that formally DRA has a pseudo-polynomial complexity because SI_{N+1} and $\mathrm{SI}_{\max}$ are not necessarily polynomial in the size of the problem description. In a practical setting though, the typical service intervals used in wireless networks result in a reduced complexity of DRA, as it will be shown next.

Notice that the algorithm proposed in Reference [12], which is based on the exploration of the aggregated period of the different flows, has a worst case time complexity bounded by $O(\mathrm{lcm}(\mathrm{SI}_i, \ldots, \mathrm{SI}_{N+1}) \frac{N}{precision})$ where lcm stands for *least common multiple* and $0 \leq precision \leq \mathrm{SI}_{N+1}$ is a parameter used to explore the space of possible starting times. Instead, DRA breaks the dependency on the lcm of the different flows' service intervals. An alternative approach to DRA is the *Dissimilar Offset Assignment* (DOA) algorithm proposed in Reference [17] in the context of Real Time Operating Systems, which, like DRA, assigns offsets to different flows in order to maximize their relative distance. This algorithm also breaks the dependency on the lcm of the flows' service intervals achieving a worst time complexity of $O(N^2(\log \mathrm{SI}_{\max} + \log N^2))$. Next, we experimentally evaluate the performance of these algorithms against the one of DRA both in terms of time complexity and achieved flow separation.

In our evaluation we consider that each flow has a service interval randomly selected between 10 and 100 ms, and a

[††] Notice that in practice we use QuickSort which has a worse worst-case performance, $O(M^2)$, but provides the best performance in average.

[‡‡] Computing $d_{\mathrm{eff\_min}_{i,j}}$ takes linear complexity since it only involves a modulo operation.

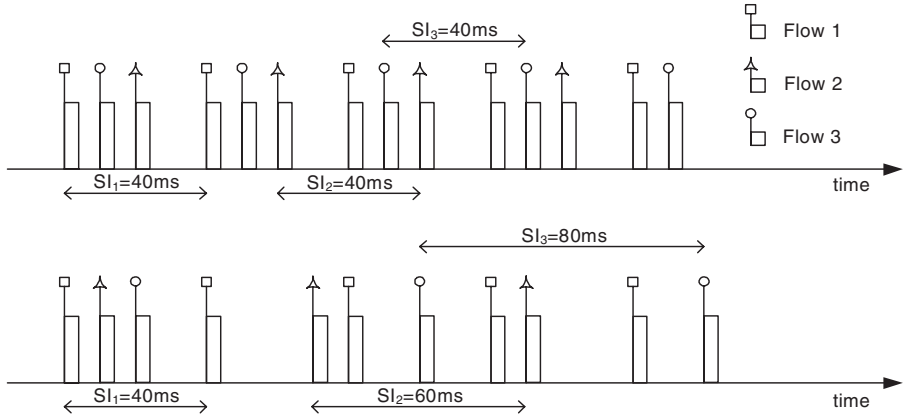
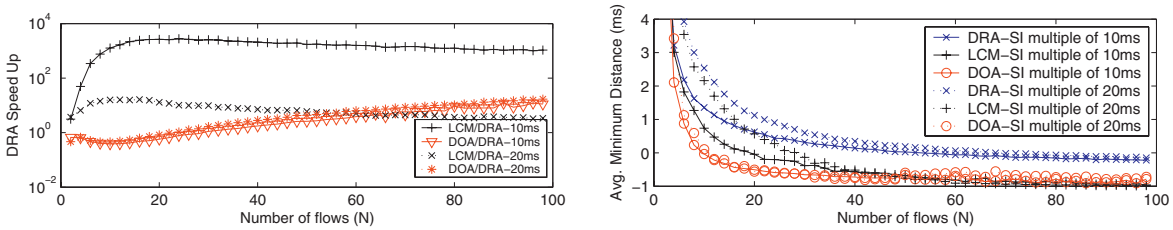


Figure 7. Resulting distributions with a max–min criteria.



(a) Relative computational complexity

(b) Average Minimum Distance

Figure 8. Performance of DRA: (a) relative computational complexity, and (b) average minimum distance.

serving time randomly selected between 0 and 1 ms.[§§] In addition, we consider a first setting where all flows have a service interval multiple of 10 ms and a second one where all flows have a service interval multiple of 20 ms. Figure 8 depicts the results of our evaluation. More than 500 independent scheduling instances were used to obtain the presented average values; however, we have not depicted the 95% confidence intervals in the graphs because they were too small to add significant information.

Figure 8(a) illustrates the ratio between the computational time required by the algorithms under study and DRA, in order to schedule a new flow when N flows are already present in the system. It is clear in the figure that, as predicted by the analysis, the complexity of an LCM based approach explodes when the flows' service intervals are multiple of 10 ms. Instead, DRA and DOA exhibit a stable and similar complexity regardless of the flows' service intervals and the number of scheduled flows. Finally, Figure 8(b) illustrates the average minimum left distance experienced with the three algorithms under study. As observed in the figure the ability to separate flows increases for all algorithms when all flows are multiple of 20 ms, instead of 10 ms, and DRA shows to always achieve the highest flow separation. Based on these computational load and scalability results we conclude that in general the DRA algorithm would be a feasible solution in practice. In addition, given that DRA is the algorithm providing the best trade-off between flow separation and time complexity, in the rest of this paper we consider DRA as the algorithm representative of the *Distribution* approach.

4. GROUPING OF RESOURCE ALLOCATIONS (GRA)

In order to compare our proposed *Distributed Resource Allocation* algorithm (DRA) with a *Grouping* one, we designed an enhanced *Grouping of Resource Allocations* algorithm (GRA) suited for stations with strict power consumption requirements operating in S-APSD. The presented scheduler is based on the common idea of defining a basic service interval, SI_{basic}, and scheduling the HCCA transmissions consecutively, see Figure 2 for an example. However, we introduce two differences with respect to the reference scheduler described in the 802.11e standard [2]: (i) we do not force all flows to be polled every basic service interval, but allow flows to be polled at multiples of the basic service interval, and (ii) we do not allow any flow to be polled prior to its nominal polling time. This last constraint is appropriate in the case of S-APSD where a station awakes to be polled at its nominal polling interval.

[§§]This service time is arbitrarily chosen for the purpose of the experiment. Notice that the values of the service time of the different flows have no impact on the complexity of the algorithm.

A pseudo-code implementation of our proposed grouping scheduler is shown in Algorithm 2. This algorithm is run in the Hybrid Coordinator every time a new HCCA flow is admitted and returns the Service Start Time (SST) and the Service Interval (SI) for the new flow.

Algorithm 2 Grouping resource allocations algorithm (GRA)

–*Variables definition*

$SI_{\text{basic}} \leftarrow$ Basic service interval.
$next\_SST \leftarrow$ Variable indicating the next available slot.
$D, r, S \leftarrow$ Delay bound, Mean Data Rate and Average MSDU size specified in the TSPEC.
$Tx\_time \leftarrow$ Transmission time of an MSDU of size S plus overhead.

–*Routine for scheduling a new flow*

1: $SI \leftarrow \left\lfloor \frac{D}{SI_{\text{basic}}} \right\rfloor SI_{\text{basic}}$
2: **if** $next\_SST$ not initialized **then**
3: $\quad next\_SST \leftarrow current\_time$
4: **end if**
5: $SST \leftarrow next\_SST + \left\lceil \frac{current\_time - next\_SST}{SI} \right\rceil SI$
6: $next\_SST \leftarrow next\_SST + \left\lceil \frac{SI \times r}{S} \right\rceil Tx\_time$

The algorithm starts by determining the service interval that will be used to poll the new flow. For this purpose we make use of the delay bound, D, specified by the station in the TSPEC message. Specifically, the flow will be polled at the maximum service interval below D that is a multiple of the basic service interval SI_{basic}. Once SI has been determined, the algorithm computes the SST for the new flow. A reference variable, $next_$SST, contains the next available polling slot. Note that $next_$SST does not refer to the current time but to a reference time that is determined by the first flow admitted in the system. The actual SST granted to the requesting station corresponds to the next nominal polling time given $next_$SST and SI. Finally, the $next_$SST variable is updated for the next requesting flow.

Our proposed grouping scheduler allocates for the current flow a transmission slot computed based on the expected average TXOP, which depends on the mean data rate, nominal MSDU size, assigned polling interval and experienced radio conditions. Different allocation polices, like a peak rate police, could also be used to compute the transmission slot, although a price would be paid in the maximum number of admitted flows. The admission control modules for both DRA and GRA schedulers are left out of the scope of this paper.

5. PERFORMANCE EVALUATION

In this section we compare the performance of the *Distributed* (DRA) and *Grouping* (GRA) schedulers presented in this paper. The goals of this study are: (i) Evaluate whether the additional complexity required by DRA is justified by relevant performance improvements and (ii) Gain a deep insight on the reasons for the performance differences due to the DRA and GRA schedulers both on HCCA and EDCA stations. We consider specially relevant a scenario where both centralized and contention based schemes coexist in order to understand how HCCA, if eventually deployed, can affect the performance of the large number of devices already in the market implementing EDCA.

The analysis has been performed via simulations. We extended the 802.11 libraries provided by OPNET [18] to include the power saving extensions defined in 802.11 and 802.11e, EDCA, HCCA and our designed DRA and GRA scheduling algorithms.

We consider an infrastructure WLAN where all the stations use the 802.11g physical layer transmitting at 54 Mbps and the Access Point (AP) generates Beacon frames every 100 ms.

Four different kinds of stations are considered in our evaluation that constitute a basic *Cluster* for our experiments:

- Medium Access (MA): HCCA. Power Saving (PS): S-APSD. Application (App): Bidirectional Voice calls using G.711 (64 Kbps) with a packetization interval of 20 ms.
- MA: HCCA. PS: S-APSD. App: Bidirectional Video conference calls emulated by means of a VBR stream [14] at 30 fps with an average rate of 1 Mbps and a peak rate of 9 Mbps.
- MA: EDCA (AC_VO). PS: U-APSD. App: Bidirectional Voice calls using G.711 (64 Kbps) with a packetization interval of 20 ms.
- MA: EDCA (AC_BE). PS: U-APSD. App: FTP downloads of a 50 MB file. We consider TCP New Reno and a RTT of 20 ms between the AP and the server storing the file. The AP has a Power Save buffer with a size of 100 packets. The TCP advertised window is configured such that it does not limit the growth of the TCP congestion window.

U-APSD is configured differently for Voice and FTP stations. Voice stations running U-APSD generate trigger frames when an uplink data frame is available, or generate a QoS Null frame after 20 ms without sending an uplink frame in order to retrieve the downlink data buffered in the AP. FTP U-APSD stations generate a trigger frame upon receiving a Beacon frame indicating that there are downlink frames buffered for them. The interested reader is referred to Reference [5] for more details on this U-APSD implementation.

In order to decide the TXOP to be granted to a flow in HCCA every time it is polled we use the description of each flow included in the TSPEC message, a similar method is used in Reference [19].

The following EDCA settings have been used at the stations and AP in our evaluation:[‖‖]

[‖‖] Parenthesis indicate that a different value is used at the AP.

Table III. EDCA configuration for the different ACs.

EDCA	AIFS	CWmin	CWmax	TXOP length
AC_VO	2(1)	3	7	1.5 ms
AC_BE	3	15	1023(63)	0 ms

Table IV. Current consumption levels of a popular PCMCIA card.

Cisco Aironet^{TM}	Sleep	Listen	Rx	Tx
Current (mA)	15	203	327	539

Regarding power consumption, we defined a model based on a popular WLAN card/chipset [20]. This model consists of four basic states: Sleep, Listen, Reception, and Transmission. We obtain the power consumed by the stations by computing the amount of time spent during an active session in each state and then applying the current consumed in each state per unit of time. The current consumption values used are shown in Table IV.[¶¶]

Based on this simulation setup, in order to compare the performance of the GRA and DRA schedulers we performed an experiment that consisted in increasing the number of clusters until a point where the WLAN network cannot meet the QoS demands of the Voice and Video stations. In addition, we study the effect of varying the service interval used to serve HCCA traffic by defining two different configurations. In the first configuration, referred as $\langle 20, 40\rangle$ configuration, HCCA Voice stations are polled every 20 ms and HCCA Video stations every 40 ms. In the second configuration, referred as $\langle 40, 80\rangle$ configuration, HCCA Voice stations are polled every 40 ms and HCCA Video stations every 80 ms. Each point in the following graphs has been obtained considering at least 15 independent simulation runs of 300 s. Average values are plotted with the correspondent confidence intervals at 95%, and cumulative distribution functions are obtained considering together the values obtained across all simulation runs.

In the following, we present our major findings based on these experiments which are divided for clarity reasons in three subsections: (i) Performance of HCCA stations, (ii) Performance of EDCA Voice stations, and (iii) Performance of EDCA FTP stations.

5.1. Performance of HCCA stations

The upper part of Figure 9(a) shows the delay value at 99% of the cumulative distribution function (cdf99) experienced by the Voice connections over HCCA for the two configurations under study: $\langle 20, 40\rangle$ and $\langle 40, 80\rangle$. Due to space reasons only downlink (AP $\rightarrow$ STA) results are shown although similar results were observed in uplink (notice that there is no contention in the uplink for HCCA). We can see in the figure how when congestion increases (>5 clusters) DRA can still maintain a delay close to the configured deadline, while GRA results in deadline violations of around 10 ms. The reason for the improved behavior of DRA is that the probability of a Voice connection being delayed by a big Video connection (VBR) is lower since DRA separates HCCA flows as much as possible.

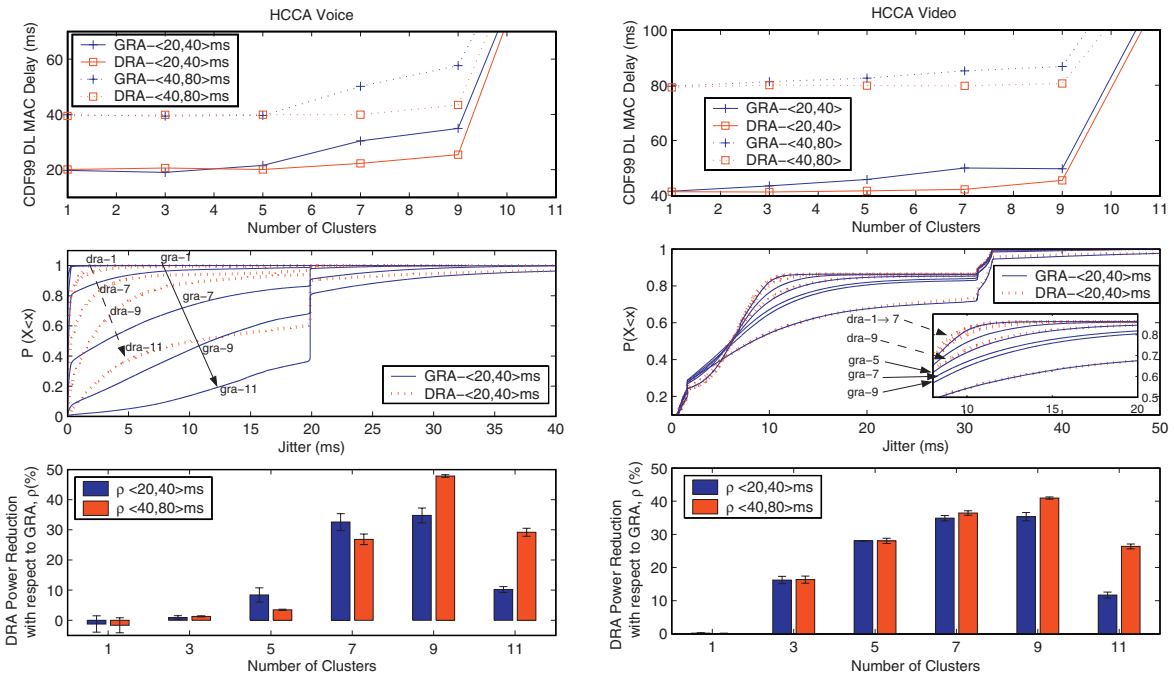
The middle part of Figure 9(a) illustrates the cdf of the instantaneous packet delay variation or jitter[***] experienced by the HCCA Voice connections, when the DRA and GRA schedulers are used. For the sake of clarity we only include the results for the $\langle 20, 40\rangle$ configuration; the results for the $\langle 40, 80\rangle$ configuration did not significantly differ from the presented ones. In addition, we include a cdf curve for each simulated scenario (number of clusters from 1 to 11), and we add a label next to each curve to identify the simulated scenario. Notice that since in the $\langle 20, 40\rangle$ configuration the polling rate of a Voice flow equals the flow's packet generation rate, the theoretical jitter should be zero. However, when the number of flows increases, and due to the variability in the service times of Video flows, the jitter experienced by a Voice flow increases substantially. It is significant though that due to a higher separation between service times, the jitter of a Voice stream under DRA is kept lower than under GRA. In addition, the presented cdf curves show a step at 20 ms which corresponds to the Voice packet generation interval. This step appears when due to an excessive delay several Voice packets are downloaded in a single service period and is significantly higher in the case of GRA.

The lower part of Figure 9(a) plots the relation between the average power consumed by the Voice stations running S-APSD under the DRA and GRA schedulers. This ratio is computed as $\rho = \frac{\text{Power}_{\text{GRA}} - \text{Power}_{\text{DRA}}}{\text{Power}_{\text{GRA}}}$ and can be understood as the average power reduction obtained with DRA. The results show that the higher separation between flows obtained with DRA results in up to a 35–45% power reduction for each configuration. The reason is that by separating the different flows we minimize the waiting time of a station since the moment it awakes until it is served. With a detailed look at the graph we can also see that the difference between GRA and DRA is maximum just before the network saturates.

Regarding the Video conference connections, Figure 9(b) shows the performance obtained by DRA and GRA, where we can see that smaller but still significant improvements than for Voice are obtained for Video in terms of delay, jitter and power consumption. The reason for the smaller differences observed in the case of Video, specially in terms of delay and jitter, is the *bigger* size of the Video flows. In this case, the amount of traffic transmitted by the flow itself is the major contributor to the delay or jitter, rather than the time the flow has to wait to be served once it awakes, which

[¶¶] For the sleep mode we used the value of a previous model of a Cisco PCMCIA card (Cisco Aironet 350) since no information was available for the referenced one.

[***] $jitter(k) = |\text{Delay}(k+1) - \text{Delay}(k)|$.



(a) HCCA Voice Performance (b) HCCA Videoconference Performance

Figure 9. Performance of HCCA stations: (a) HCCA voice performance, and (b) HCCA videoconference performance.

is one of the main differences between the DRA and GRA schedulers.

5.2. Performance of EDCA Voice stations

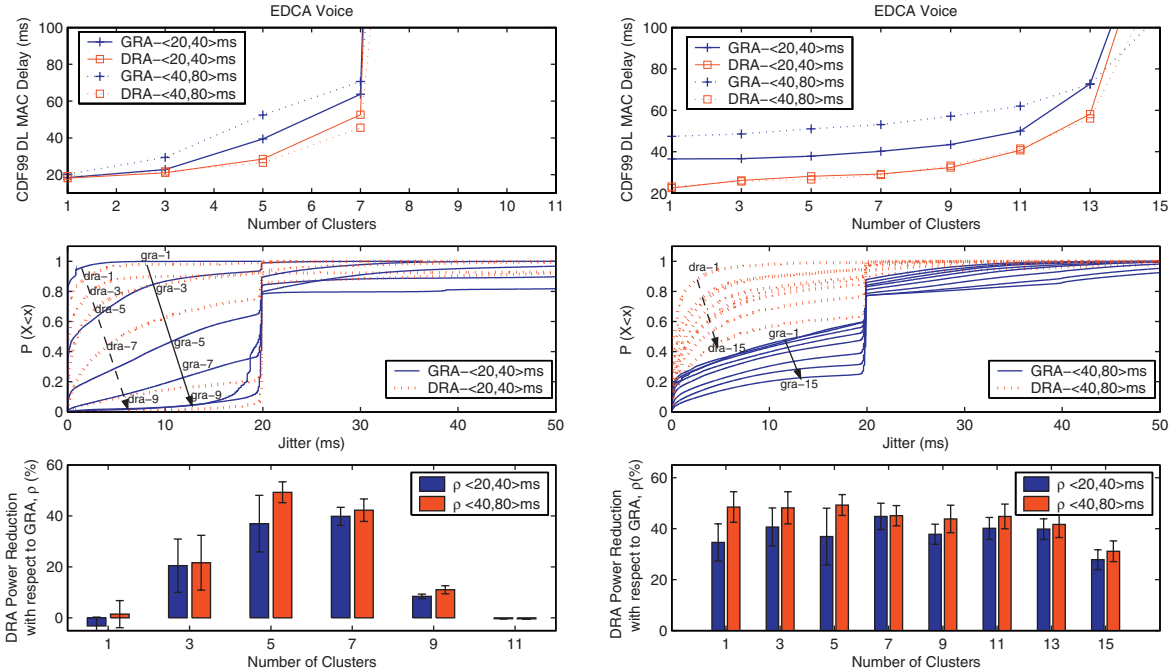
The upper part of Figure 10(a) depicts the downlink delay results experienced by the EDCA Voice stations in our experiment. A smaller delay and jitter is experienced by these stations when DRA is used, instead of GRA, for HCCA. In addition, looking at power consumption in the lower part of Figure 10(a) we observe that up to a 49% power consumption reduction can be achieved with DRA with respect to GRA.

In order to better understand the impact of GRA and DRA over EDCA Voice stations we perform a modification in our previous experiment. Instead of increasing at the same time all the stations belonging to our basic cluster, we fix the number of HCCA Voice and Video stations to 10 (5 stations of each), consuming approximately a 50% of the available bandwidth, and increase only the number of EDCA stations (Voice and FTP). The results of this modified experiment with fixed HCCA load are shown in Figure 10(b).

The upper part of Figure 10(b) shows the delay experienced by the EDCA Voice stations in our modified experiment. Interestingly, even when the EDCA load in the network is very small, e.g., one station, a bigger worst case delay is experienced by the EDCA Voice stations when GRA is used for HCCA instead of DRA. Indeed, the delay experienced by the EDCA Voice stations shows a strong dependency on the service interval used for HCCA in the case of GRA but not in the case of DRA. The reasons for the observed behavior are the following. In case of using GRA, which places consecutively all HCCA transmissions, and having allocated approximately 50% of the bandwidth for HCCA, an EDCA transmission may have to wait up to 1/2 of the GRA basic service interval, which is 20 ms in the $\langle 20, 40 \rangle$ configuration and 40 ms in the $\langle 40, 80 \rangle$ configuration, before accessing the medium. Thus, when the service interval for HCCA increases, the contiguous HCCA allocation under GRA also increases and so does the worst case delay experienced by EDCA Voice stations. On the other hand, when DRA is used, the EDCA Voice stations see a medium without long allocations for HCCA, which results in a lower worst case delay and a smaller dependency on the HCCA service interval.

The previous effect has also a significant impact on jitter as observed in Figure 10(b). In this case, in order to illustrate the underlying dynamics we present the results of the $\langle 40, 80 \rangle$ configuration, where we can see how blocking the channel for long periods as in GRA, results in backlogged EDCA Voice packets and a bigger jitter. This effect is minimized with DRA.

Finally, the lower part of Figure 10(b) depicts a consistent power consumption reduction with DRA between 35% and 45% until the network starts to saturate and slightly higher in the $\langle 40, 80 \rangle$ configuration. We have observed that an important factor affecting the power consumed by EDCA



(a) Increasing cluster experiment (b) Fixed HCCA load experiment

Figure 10. Performance of EDCA Voice stations; (a) Increasing cluster experiment; (b) Fixed HCCA load experiment.

Voice stations is the increased number of retransmissions experienced under the GRA scheduler. The reason is that the long HCCA allocations created by GRA synchronize the EDCA Voice stations trying to access the medium creating regions of a higher collision probability.

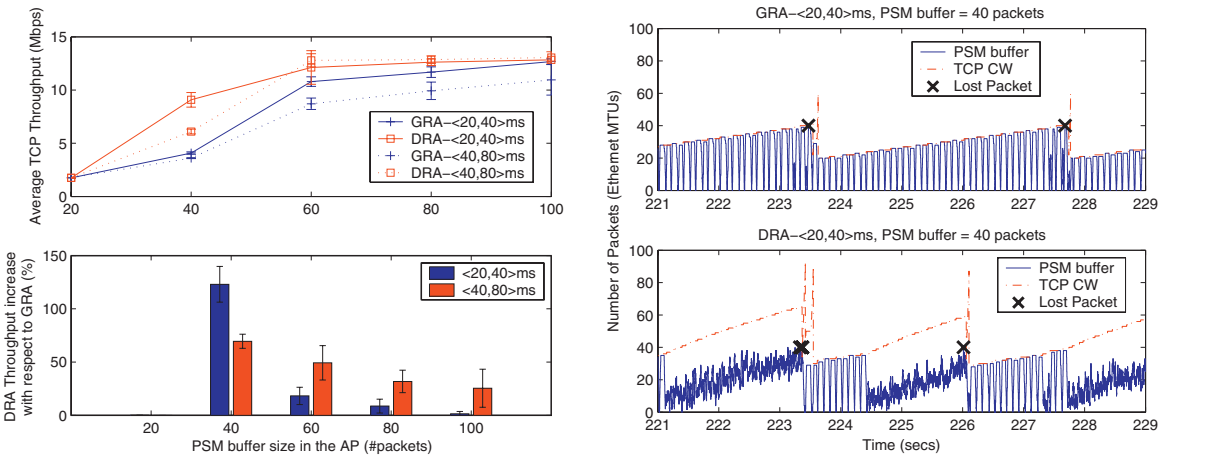
Notice that the long HCCA allocations created under GRA could be reduced by computing the transmission slots in GRA considering a worst case transmission rate for the HCCA stations. Thus, if HCCA stations would actually transmit at higher rates, *holes* between consecutive GRA allocations would be created that could be used by EDCA stations. A price though would be paid in the number of simultaneous flows that can operate under HCCA. Instead, DRA by its principle of operation reduces the access delay of EDCA stations to the channel even in the case that HCCA stations operate at their worst-case transmission rate.

5.3. Performance of EDCA FTP stations

No significant differences in the performance of EDCA FTP connections were observed in our basic experiment when the DRA or GRA schedulers were used. Therefore, in order to gain a deeper understanding on the behavior of TCP with the two HCCA schedulers under study, we perform the following simplified experiment. We consider 5 HCCA Video conference and 5 HCCA Voice stations and a single EDCA station performing a FTP download and modify the size of the Power Save (PS) buffer at the AP. The RTT between the AP and the FTP server is kept at 20 ms.

Figure 11(a) shows the average throughput experienced by the EDCA FTP connection when DRA and GRA are used to schedule the HCCA traffic considering the $\langle 20, 40\rangle$ and $\langle 40, 80\rangle$ configurations. In addition, in order to better understand the dynamics depicted in Figure 11(a), Figure 11(b) depicts for GRA and DRA a sample trace containing the number of packets in the PS buffer, the TCP Congestion Window and the lost TCP packets when the size of the PS buffer in the AP is 40 packets.

Looking at the number of packets in the PS buffer for the GRA trace we can see two clear periodic behaviors superimposed. First, a slow saw-tooth pattern due to New Reno's congestion avoidance. Second and more interesting, we observe how the PS buffer empties and fills up periodically. The reason for this second behavior is the power saving method being employed by the WLAN station. The station awakes to receive the Beacon frame, downloads the buffered frames in the AP, generates the correspondent TCP ACKs in uplink and goes back to sleep before the next window of TCP packets arrives at the AP. This phenomena, first pointed out in Reference [21], increases the effective RTT perceived by the TCP sender and reduces the experienced TCP throughput. However, a different behavior can be observed in the equivalent trace for DRA. Here, after cumulating certain number of packets in the queue, the next window of TCP packets arrives before the station can go back to sleep forcing the station to remain awake. This results in a reduction of the experienced RTT and in an increased throughput. The reason for the previous phenomena is related with the performance inversion effect



(a) FTP connection performance

(b) TCP instantaneous behavior with GRA and DRA schedulers

Figure 11. Performance of EDCA FTP stations: (a) FTP connection performance; (b) TCP instantaneous behavior with GRA and DRA schedulers.

described in Reference [21] and is explained as follows. Under GRA, TCP traffic is transmitted immediately after a long HCCA allocation. This means that the medium is free for EDCA for a long time until the next HCCA allocation, and so the exchange of TCP packets between the AP and the EDCA FTP station can complete before the new window of TCP packets arrive at the AP, which results in the station going to sleep and the RTT of the TCP connection increasing due to the Beacon interval. Instead, under the DRA scheduler the EDCA TCP transmissions are more often interrupted by HCCA transmissions. Thus, the transmission of the TCP buffered data takes longer to complete, which results in a new window of TCP data arriving at the AP before the station goes back to sleep, and TCP observing a reduced RTT. The previous effect is specially relevant in the $\langle 20, 40 \rangle$ configuration.

Based on the previous effect, the dynamics exhibited in Figure 11(a) can be easily explained. When the buffering in the AP is small, the EDCA FTP station always goes back to sleep, in both DRA and GRA, before the new window of TCP packets arrives at the AP resulting in a lower throughput. In addition, when the buffering is large, in both DRA and GRA, the TCP congestion window can grow big enough, and the EDCA FTP station stays awake during the lifetime of the TCP connection resulting in an increased throughput. However, it is in the intermediate zone that DRA can provide a significant throughput gain due to the reasons previously stated. Note that a situation of limited buffering is specially relevant for mobile devices that usually support small TCP advertised windows.

With respect to power consumption, no significant differences were observed between DRA and GRA. The reason is that the total power consumed by the EDCA FTP station in this scenario mostly depends on the size of the file being downloaded which is the same in both cases.

6. SUMMARY AND CONCLUSIONS

In this paper we have proposed and analyzed an efficient and scalable scheduler (DRA) for the centralized IEEE 802.11e QoS (HCCA) and power saving (S-APSD) solutions that, unlike most of the proposals existent in the literature, distributes in the channel as much as possible the resource allocations of the different admitted flows. The performance of our *distribution* proposal, DRA, has been compared to a *grouping* one, GRA, and the results indicate that significant improvements are to be expected.

Specifically, the main conclusions that can be drawn from our study are threefold: (i) Distributing in the channel HCCA allocations benefits the isolation of the different flows resulting in significant power savings for stations running S-APSD, up to 45% in our scenarios, (ii) QoS stringent applications running EDCA and U-APSD, e.g., Voice, also benefit from the distribution of HCCA allocations by experiencing reduced worst case delays and increased power savings, up to 45% in our experiments, and (iii) Distributing HCCA transmissions in the channel results in a more uniform medium from the perspective of TCP connections in power saving running over EDCA, which significantly improves throughput, depending on the amount of buffering available in the AP.

Finally, we would like to notice that the DRA scheduler introduced in this paper could be extended in several ways which we leave as future work. First, it would be interesting to assess the impact of DRA in the admission region of a network with QoS and power consumption guarantees. Appropriate admission control algorithms should be designed for that purpose. Second, algorithms could be devised that, by appropriately pre-sorting the flows to be processed by DRA, improve even further the achieved flow separation. Finally, DRA could be extended to allow a set of overlapping co-channel APs to pro-

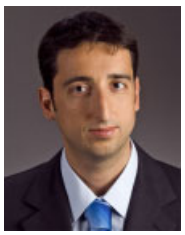
vide QoS and power saving guarantees in a distributed way.

REFERENCES

1. Press Release. *Wi-Fi is now a must-have for mobile phones; User affinity to drive annual shipments to 300 million in 2011*, April 1, 2009.
2. IEEE 802.11 WG. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. Amendment 8: medium access control (MAC) quality of service enhancements. *IEEE Standard 802.11e*, November 2005.
3. Mangold S, Choi S, Hiertz GR, Klein O, Walke B. Analysis of IEEE 802.11e for QoS support in wireless LANs. *IEEE Wireless Communications* 2003; **10**(6): 40–50.
4. IEEE 802.11 WG. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Standard 802.11*, June 1999.
5. Perez-Costa X, Camps-Mur D, Vidal A. On distributed power saving mechanisms of wireless LANs 802.11e U-APSD vs 802.11 power save mode. *Computer Network* 2007; **51**(9): 2326–2344.
6. Pérez-Costa X, Camps-Mur D. IEEE 802.11E QoS and power saving features overview and analysis of combined performance, *IEEE Wireless Communications* 2010; **17**(4): 88–96.
7. Available at: http://www.wi-fi.org
8. Shenker S, Partridge C, Guerin R. Specification of guaranteed quality of service. *RFC 2212, Internet Engineering Task Force*, September 1997.
9. Ansel P, Ni Q, Turletti T. *FHCF: a simple and efficient scheduling scheme for IEEE 802.11e*. Springer/Kluwer Journal on Mobile Networks and Applications (MONET) 2006; **11**(3): 391–403.
10. Zhao Q, Tsang D. An equal-spacing-based design for QoS guarantee in IEEE 802.11e HCCA wireless networks. *IEEE Transactions on Mobile Computing* 2008; **7**(12): 1474–1490.
11. Grilo A, Macedo M, Nunes M. A scheduling algorithm for QoS support in IEEE 802.11e networks. *IEEE Wireless Communications* 2003; **10**(3): 36–43.
12. Pérez-Costa X, Camps-Mur D, Palau J, Rebolleda D, Akbarzadeh S. Overlapping aware scheduled automatic power save delivery algorithm. *European Wireless Conference*, Paris, April 2007.
13. Berlemann L. Distributed quality-of-service support in cognitive radio networks. *Ph.D. Thesis*, February 2006.
14. Fitzek FHP, Reisslein M. MPEG-4 and H.263 video traces for network performance evaluation. *IEEE Network* 2001; **15**(6): 40–54.
15. Sha, Abdelzaher T, Arzen KE *et al*. Real time scheduling theory: a historical perspective. *Real-Time Systems* 2004; **28**(2–3): 101–155.
16. Georges L, Mhlethaler P, Rivierre N. A few results on non-preemptive real time scheduling. *INRIA Research Report nRR3926*, 2000.
17. Goossens J. Scheduling of offset free systems. *Real-Time Systems* 2003; **24**(2): 239–258.
18. OPNET Technologies. Available at: http://www.opnet.com
19. Higuchi Y, Foronda A, Ohta C, Yoshimoto M, Okada Y. Delay guarantee and service interval optimization for HCCA in IEEE 802.11e WLANs. *Wireless Communications and Networking Conference*, (WCNC, 2007) 2007: 2080–2085.
20. Cisco Aironet 802.11a/b/g Wireless CardBus Adapter. Available at: http://www.cisco.com/
21. Krashinsky R, Balakrishnan H. Minimizing energy for wireless web access with bounded slowdown. In *proceedings of the eighth Annual International Conference on Mobile Computing and Networking (MOBICOM)*, September (Atlanta, Georgia, USA, September 23–28, 2002). MobiCom '02. ACM, New York, NY, 119–130.

AUTHORS' BIOGRAPHIES

Daniel Camps Mur is a researcher in the Mobile and Wireless Networks Group at NEC Network Laboratories in Heidelberg, Germany. He is currently pursuing a Ph.D. at the telematics department of the Polytechnical University of Catalonia (UPC), where he obtained an M.Sc. degree in 2004. His current research interests include MAC QoS and power saving protocols for WLAN and WiMAX networks. His master thesis work received the 'Mobile Internet and 3G Mobile Solutions' award from the Spanish Association of Telecommunication Engineers (COIT). In the field of network simulations he also received the OPNET's Significant Technical Challenge Solved Award.



Xavier Pérez-Costa is a Chief Researcher at NEC Laboratories Europe in Heidelberg, Germany, where he is currently the responsible of managing several projects related to wireless networks. In the wireless LAN area, he leads a project contributing to 3G/WiFi mobile phones evolution and received NECs R&D Award for his work on N900iL, NECs first 3G/WiFi phone. In the WiMAX area he manages a team researching on NECs WiMAX products future enhancements. In the Future Wireless Internet area, he is responsible for a team

contributing to the EU-FP7 project Flexible Architecture for Virtualizable Wireless Future Internet Access (FLAVIA). He has participated in IEEE standardization working groups as well as their corresponding certification bodies, and has been included in the major contributor lists of IEEE 802.11e and 802.11v. He has served in the program committee of several conferences, including IEEE ICC, WCNC, and INFOCOM, and holds over 10 patents. He received both his M.Sc. and Ph.D. degrees in Telecommunications from the Polytechnic University of Catalonia (UPC) and was the recipient of the national award for the best Ph.D. thesis on multimedia convergence in telecommunications from the Official Association of Telecommunication Engineers (COIT).

Vladimir Marchenko graduated from RWTH Aachen University in 2007 with Diploma degree in Computer Engineering and main focus on Wireless Communications. During his studies he made internships at Siemens Corporate Research, Princeton, NJ, USA and NEC Laboratories, Heidelberg, Germany. After the studies he was employed as Applications Engineer for ZigBee at MeshNetics and after company acquisition works at the same position for Atmel Corp.

Sebastià Sallent Ribes obtained his M.Sc. degree in Telecom Engineering in 1979, and the Ph.D. degree in Telecommunication Engineering from the Catalonia Polytechnical University (UPC) in Barcelona in 1987. Since 1979 to 1985 he was engaged at PHILIP. His current research interests are in Broadband and optical access networks (MAC protocols, QoS, traffic modelling). He also was engaged in several projects with EU (AT-DMA, @DAN, COST, Teleregions, Euro-NGI, Euro-NF (NoE), Phosphorous, Federica,..), ESA (European Space Agency), private R+D and several research projects funded by the Spanish Government related to broadband and access networks. Author and co-author of more than 100 publications, supervisor of several proceedings, and member and reviewer of the IEEE. He headed the Telematic Engineering Department at UPC. Nowadays he heads the i2CAT Foundation, Internet and Digital Innovation in Catalonia.