WTTP : A Scheduling Algorithm for Supporting QoS in IEEE 802.11e*

Invited Paper

Claudio Cicconetti, Luciano Lenzini, Enzo Mingozzi, Giovanni Stea

Dipartimento di Ingegneria dell'Informazione, University of Pisa, Italy
{c.cicconetti|l.lenzini|e.mingozzi|g.stea}@iet.unipi.it

Abstract - **This paper proposes a scheduling algorithm, the Wireless Timed Token Protocol (*WTTP*) for the Access Point of an IEEE 802.11e wireless network, to serve traffic streams with QoS guarantees. *WTTP* provides the established streams with rate-based guarantees. Furthermore, it makes the capacity not used by those streams available to contention-based access. The performance of *WTTP* is compared to that of a reference scheduler defined in the 802.11e draft standard using the *ns2* simulator. The results show that *WTTP* outperforms the *reference* scheduler with VBR traffic, both in terms of resource utilization and maximum number of admitted flows.**

1. Introduction

In recent years Wireless Local Area Networks (WLANs) have become very popular due to the increasing interest of residential and office customers in a ubiquitous service. Within this framework, the IEEE 802.11 has established itself as the worldwide standard in indoor and outdoor wireless LANs [13].

On the other hand, the high level of performance provided by the wired networks is driving users toward an emerging set of applications with Quality of Service requirements, such as phone or videoconference over IP networks, which have variable bit rate traffic and strict delay bounds. Those applications will coexist with legacy TCP-based applications, which have no specific QoS requirements. In order to support applications with QoS requirements the upcoming IEEE 802.11e amendment enhances the IEEE 802.11 MAC with two additional access functions: the Enhanced Distributed Channel Access (EDCA) function, which is based on a distributed control and enables prioritized channel access, and the HCF Controlled Channel Access (HCCA) function, which instead requires centralized scheduling, and allows the applications to negotiate parameterized service guarantees [14].

The IEEE 802.11e draft standard does not specify a mandatory scheduling algorithm; a *reference* scheduler is reported instead for informational purposes. However, the *reference* scheduler is specifically tailored to constant bit rate traffic, which makes it inefficient for servicing variable bit rate traffic, e.g. a multimedia stream [8]. In this paper, we propose a novel scheduling algorithm, namely the *Wireless Timed Token Protocol* (*WTTP*), which explicitly takes into account two different types of traffic: HCCA traffic flows and traffic using contention-based access functions, i.e. EDCA or DCF. More specifically, it provides those flows that have been admitted to use the HCCA function with rate-based guarantees. However, it strives to satisfy their requirements so as to make the capacity temporarily not used by (although reserved for) the HCCA flows fully available for traffic transmitted on a contention basis. The *WTTP* algorithm was implemented in the *ns2* [12] simulator and compared with the *reference* scheduler under different scenarios. The performance results confirm that *WTTP* is more effective in scheduling VBR traffic than the *reference* scheduler, in terms of the maximum number of flows that can be admitted while guaranteeing a specified performance level.

The rest of the paper is organized as follows. Section 2 introduces the IEEE 802.11 standard and the IEEE 802.11e enhancements. Section 3 overviews the related work. In Section 4 the *WTTP* scheduling algorithm is described. Sections 5 and 6 outline the simulation environment. The results are discussed in Section 7 and conclusions drawn in Section 8.

2. Protocol description

This section briefly describes the IEEE 802.11 MAC protocol [13] and the enhancements proposed in the IEEE 802.11e amendment [14].

2.1. IEEE 802.11 MAC

The IEEE 802.11 includes two access mechanisms: the Distributed Coordination Function (DCF) and the Point Coordination Function (PCF). The DCF uses a CSMA/CA access scheme [3]: for a station (STA) to transmit, it shall sense the medium to determine if another STA is transmitting; if the medium is idle, then the transmission may proceed; if the medium is busy, then the STA shall defer until the end of the current transmission. The STA performs a backoff procedure after deferral in order to minimize collisions; the backoff procedure starts after the medium has been sensed idle for a DCF Interframe Space (DIFS). When a unicast frame is correctly received by an STA, the latter transmits an acknowledgment frame after a Short Interframe Space (SIFS), which is shorter than DIFS in order to prevent deferring stations from interrupting ongoing frame exchange sequences.

If the *Point Coordination Function* is used, then the AP alternates a Contention-Free Period (CFP) with a Contention Period (CP). During the CP, the above DCF transfer rules apply. During the CFP the AP polls the stations that are in the CF poll list, using an implementation-dependent algorithm. When polled, a station

*This work has been carried out within the framework of the QUASAR project, funded by the *Italian Ministry of Education, University and Research* (*MIUR*).

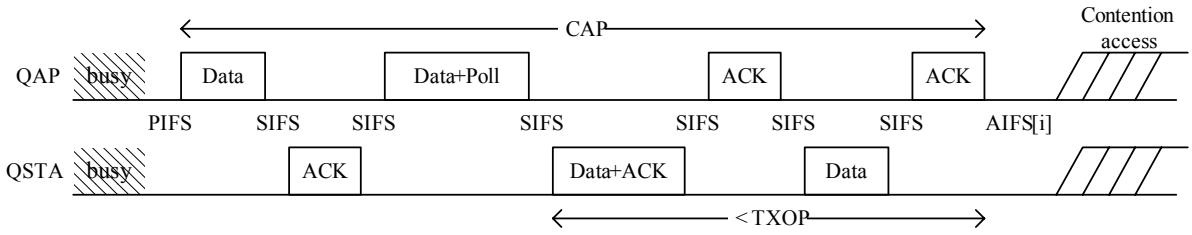

Figure 1. HCCA frame exchange sequence example.

transmits an MSDU. The AP may also use the CFP to transmit frames addressed to associated stations. The first frame of the CFP is transmitted by the AP at regular intervals (Target Beacon Transmission Time, TBTT) after the medium has been sensed idle for a duration greater than a PCF Inter-Frame Space (PIFS), which is chosen such that SIFS < PIFS < DIFS.

Even though it has been shown that the PCF performs better than the DCF with real-time traffic [16], it is nevertheless unable to guarantee that associated stations will have a parameterized QoS. The reasons behind this failure are: (i) the lack of knowledge at the AP of the traffic specifications and requirements; (ii) no mechanism to determine if a station actually has data to send, which leads to a high polling overhead; (iii) the interval between two CFPs is too large and the duration of the CFP is limited; (iv) the TBTT may be delayed due to stations unaware of the PCF procedure; and, finally, (v) no admission control can be performed at the AP [19].

2.2. IEEE 802.11e MAC

The IEEE 802.11e standard, currently in a draft but stable state, describes enhancements to the MAC services and functions aimed at enabling Quality of Service provisioning. Two access mechanisms are defined in addition to DCF and PCF: the Enhanced Distributed Channel Access (EDCA) and the HCF Controlled Channel Access (HCCA). In order to ensure compatibility with legacy devices, the standard allows the coexistence of DCF and PCF with EDCA and HCCA.

The EDCA achieves traffic differentiation in a distributed manner [20]. At each QoS Station (QSTA) four Access Categories (AC) are defined, whose traffic contends for the medium using different access parameters, such as the minimum contention window. In an infrastructure network, the set of parameters for each category is advertised by the QoS Access Point (QAP), which also performs the admission control.

The HCCA is a centralized access mechanism controlled by the Hybrid Coordinator (HC), which resides in the QAP. Each QSTA may have up to eight established Traffic Streams (TS). A TS is characterized by a Traffic Specification (TSPEC) which is negotiated between the QSTA and the QAP. Mandatory fields of the TSPEC include the Mean Data Rate, the Delay Bound, and the Nominal SDU Size. For all established streams the QAP is required to provide a service that is compliant with the negotiated TSPEC under controlled operating conditions.

The QAP enforces the negotiated QoS guarantees by scheduling Controlled Access Phases (CAPs). A CAP is a time interval during which the QAP may either transmit MSDUs belonging to established downlink TSs or poll one or more QSTAs by specifying the maximum duration of the transmission opportunity (TXOP).[1] A QSTA is never allowed to exceed the TXOP limit imposed by the QSTA, including interframe spaces and acknowledgments. If the TS of a polled QSTA is not backlogged, then the QSTA responds with a *Null* frame. Figure 1 shows a sample CAP during which the QAP transmits two data frames and polls the QSTA, which in turn transmits two data frames. It is worth noting that the scheduling of CAPs, i.e. of HCCA traffic streams, also affects the overall capacity left for contention-based access, i.e. EDCA and DCF.

The CAP scheduling procedure is not part of the standard, although a *reference* scheduler is defined in [14]. The *reference* scheduler requires that flows specify the following TSPEC parameters: Mean Data Rate, Nominal SDU Size, Maximum SDU Size and Maximum Service Interval (MSI). The MSI of a given flow is the maximum time that elapses between the start of two consecutive service periods of that flow. The *reference* scheduler produces TDM-like schedules: each TS is periodically allocated a fixed amount of capacity. The period is called Service Interval (SI) and it is the same for all traffic streams. It is computed as the smallest value of the admitted MSIs. The TXOP duration is then set to the time required to transmit all the packets of Nominal SDU Size that arrive at the negotiated Mean Data Rate during the SI. The TXOP is rounded up to contain an integer number of packets of Nominal SDU Size. In order to avoid head-of-line blocking, the actual TXOP value is the maximum between the value obtained by the above procedure and the time to transmit a packet of Maximum SDU Size.

As stated above, the *reference* scheduler produces a periodic schedule. On the other hand, multimedia traffic is inherently of variable bit rate. However, serving VBR traffic with periodic schedules may result in wasting resources, since overprovisioning (with respect to the mean data rate) is required in order to provide delay bounds. This has also been shown in [8], where the authors present a simulation study that highlights the inability of the *reference* scheduler to handle VBR traffic.

3. Related Work

Over the last few years, the scheduling problem in

[1] In order to ensure backward compatibility with PCF, the QAP may periodically generate CFPs as a PC. In this case, the QAP is allowed to schedule CAPs during both contention- and contention-free periods. In this paper we focus on HCCA only, hence we assume that the QAP does not act as a PC.

wireline networks has been extensively studied in the literature [24][25]. Many different algorithms have been proposed, whose performance guarantees can have a statistical or deterministic characterization. Unfortunately, these algorithms, and the related properties, are not directly applicable to the scheduling problem in wireless networks. In [5] the authors provide an exhaustive analysis of the problems and challenges inherent in wireless environments. These include: (i) the capacity of a wireless link is highly variable on both a time-dependent and a location-dependent basis; (ii) the notion of *fairness*, which is well defined in a wireline network [21], is much more blurred in a wireless environment, in which channel characteristics are in fact variable; (iii) even though QoS differentiation and guarantees should be supported, the scheduling algorithms should also deal with the inherent characteristics of the wireless environment (iv) since bandwidth is a scarce resource in wireless networks, the scheduling procedures should strive to maximize the channel utilization; (v) in the case of mobile hosts, scheduling should be concerned with minimizing the energy consumption of served nodes; and, finally, (vi) since the wireless communication systems usually involve the use of low-performance hardware devices, the scheduling algorithm should not be too complex.

Clearly, addressing all the aforementioned issues goes beyond the scope of a single work. Thus, in the rest of the paper, we will only focus on points (iii), (iv), and (vi), i.e. QoS constraints, channel utilization and scheduling complexity, since these are more strictly related to the scheduling problem in the 802.11e HCCA. In this context, a number of different approaches for scheduling CAPs using HCCA have been proposed in the recent past. All of them exploit some optional features of the 802.11e in order to improve performance with respect to the *reference* scheduler. In the rest of this section we briefly describe four representative algorithms, which roughly fit into two classes: real-time scheduling algorithms and algorithms based on queue length estimation.

3.1. Real-time scheduling algorithms

In [10] the authors propose a scheduling algorithm based on the well-known Earliest Due Date (Scheduling based on Estimated Transmission Time, SETT-EDD). SETT-EDD requires that, in addition to the set of parameters required by the *reference* scheduler, flows specify the minimum service interval (mSI), which is the minimum time that must elapse between two consecutive service periods.

During the setup phase, all the traffic streams are aggregated on a per-QSTA basis: the aggregate mean data rate is computed as the sum of the mean data rate values, the aggregate maximum burst size is computed as the sum of the maximum SDU size values, and the aggregate minimum and maximum service intervals are computed as the minimum of the corresponding values. The average TXOP durations of all the QSTAs are computed as in the *reference* scheduler. The aggregate mean data rate and maximum burst size are then used to initialize a token bucket for each QSTA, which is used to estimate the amount of backlogged traffic at the QSTA. At run-time, CAPs are scheduled according to the following algorithm, based on EDD: if a QSTA is served at time t, then it will be served again at a time t' such that $t + mSI \le t' \le t + MSI$.

The SETT-EDD algorithm has been shown to perform better than the *reference* scheduler, especially with video streaming traffic. This improvement is mainly due to the more efficient way of dealing with TSs that have different requirements in terms of Maximum Service Interval. One drawback of this approach is that it exhibits a high online computational complexity: in fact it may require as many as $O(n)$ operations per slot, where n is the number of QSTAs with at least one established TS.

In [17] the authors propose a scheduling algorithm that explicitly considers both periodic and aperiodic traffic, and is aimed at maximizing channel utilization. This is done by (i) concatenating consecutive TXOPs whenever possible, in order to reduce the polling overhead, and (ii) scheduling TXOPs only to QSTAs which perceive that the channel is in a good state. This requires a MAC signaling protocol that notifies the QAP of slow-changing channel state information, such as the one currently under specification in the Task Group k of the IEEE 802.11 committee [15]. More specifically, TSs are classified into *periodic* and *aperiodic*. In a scheduling period, periodic TSs are served first. After that, the remaining service period is filled by granting TXOPs to non-periodic TSs. This is done using a deadline-based approach: a TXOP is served as late as possible without violating its deadline, which is set equal to the Delay Bound advertised in the TSPEC. Since the QAP does not know the arrival time of uplink packets, the worst case is assumed, that is the next packet is enqueued as soon as the previous one has been dispatched. Once this procedure has been completed, all subsequent TXOPs are concatenated, whenever possible, in order to reduce the MAC overhead due to polling and acknowledging. A simulation study shows that the proposed algorithm performs better than the *reference* scheduler in both error-free and erroneous channel conditions, using both periodic and non-periodic traffic.

Even though the aforementioned algorithms have been shown to achieve good performances, it is worth noting that they require a high computational overhead, which, as discussed above, might not be feasible in a wireless environment.

3.2. Algorithms based on queue length estimation

In order to let the QAP know the status of the queues at the QSTAs, the 802.11e defines the following optional feature. QSTAs may piggyback information on the length of their transmission queues using a specific field of the MAC header. The latter only appears in QoS frames, and contains the length (in multiples of 256 bytes) of the transmission queue of the TS to which the packet belongs. This optional feature of the 802.11e is used in the algorithms that follow.

In [1], the information provided by the QSTAs about the status of their own queues is used to feed a control system at the QAP that estimates the actual requirements of the established TSs. The resulting

scheduler is the Feedback Based Dynamic Scheduler (FBDS). FBDS schedules are periodic: on each period T_{CA}, all the TSs are scheduled a TXOP whose duration depends on the requirements estimated by the control system. If the sum of the allocations is greater than the duration of the period, then all the TXOPs are decreased proportionally. The effectiveness of the proposed approach has been tested under several different simulated scenarios.

The piggybacking of queue length information is also exploited in [2], although in a different framework. The scheduling procedure (Fair HCF, FHCF) is similar to that of the *reference* scheduler: the QAP schedules the TXOPs each Service Interval, which is computed as the minimum among the MSI values of the established TSs. However, at the start of each SI, the QAP performs the following operations: (i) it computes the TXOP duration of each TS, and (ii) it tunes the allocation by fairly redistributing the unused SI time, if any, or by fairly decreasing the TXOP duration to the TSs if the sum of the TXOPs exceeds the SI.

The computation of the TXOP duration is based on a comparison between the actual and estimated queue length distributions. To estimate the requirements of the TSs, the QAP uses a window of a fixed number of collected samples, which are combined at the beginning of each SI. Moreover, since the QAP only schedules a QSTA (rather than a specific TS on a QSTA), the authors of [2] also propose that each QSTA runs the same algorithm in order to arbitrate among its TSs, this time using the actual queue lengths as input parameters. FHCF has been shown to be more efficient than the reference scheduler in the presence of mixed CBR and VBR traffic. In fact, it can admit a higher number of TSs with respect to the *reference* scheduler, while still providing the same QoS guarantees. The poorer performance of the *reference* scheduler in terms of the number of admitted VBR TSs is in fact due to the bandwidth overprovisioning that is necessary to obtain feasible delays. Since the FHCF scheduler dynamically adapts the allocation to the actual needs of the VBR TSs, such overprovisioning is not necessary.

Both algorithms inherit from the *reference* scheduler the periodic scheduling of the set of admitted TSs. Therefore, their main contribution consists in a procedure that estimates the actual requirements of the uplink queues. The latter is performed only once for each scheduling period, and hence the computational overhead is much less than that of the real-time schedulers. However, they are based on the optional feature of queue length piggybacking, which may not be available on all devices.

In the following section we describe the main contribution of this paper, i.e. the *WTTP* algorithm. Unlike all the above algorithms, it is not based on periodic scheduling of CAPs. Moreover, it combines scheduling of HCCA TSs with capacity allocation for contention-based access. Thus, our contribution cannot be classified into either of the classes presented above. Nevertheless, *WTTP* is considerably simpler than the real-time scheduling algorithms that we have reviewed. Furthermore, it does not require any optional feature, such as piggybacking.

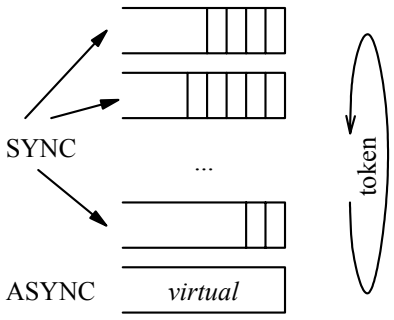

Figure 2. *WTTP* scheduler.

4. Wireless Timed Token Protocol

In this section we describe the Wireless Timed Token Protocol (*WTTP*) scheduler. *WTTP* is based on the Timed Token Protocol (TTP) [11][22], which is a token passing scheme employed as the MAC protocol for ring-based networks, such as the Fiber Distributed Data Interface (FDDI). In TTP there are two types of traffic flows: (i) synchronous flows, with real-time service requirements, and (ii) asynchronous flows, which are served in a best-effort manner. The token circulation is paced based on the Target Token Revolution Time (TTRT), which is a protocol parameter negotiated among all nodes. The synchronous flows are allocated a portion H_i – also called *synchronous capacity* – of the TTRT, that they may use to transmit frames when they receive the token. On the other hand, the asynchronous flows are not reserved any capacity: they may transmit frames only if the token arrives earlier than expected, i.e. before a TTRT time has elapsed from the last token visit. The early arrival of the token usually occurs when the synchronous flows consumed less than the reserved capacity during the previous token revolution.

In this perspective, the 802.11e is very similar to the TTP, since it needs to schedule capacity for two different types of traffic simultaneously, i.e. HCCA TSs and traffic using contention-based access functions. Accordingly, *WTTP* applies the TTP rules for scheduling CAPs so as to provide HCCA TSs with rate-based guarantees, and for computing the amount of capacity which it makes available to traffic transmitted on a contention basis.

Figure 2 shows the *WTTP* system model. Each established TS, both uplink and downlink, is represented by a packet queue (*synchronous* queue). So as each synchronous flow in a TTP ring holds a token for a given time, in *WTTP* the QAP polls each synchronous queue, allowing it to transmit its frames for a *synchronous capacity* proportional to the negotiated Mean Data Rate of the corresponding TS. When scheduling downlink CAPs, the QAP only takes into consideration backlogged downlink queues. When an empty downlink queue receives a packet, then a reference to it is added to the list of synchronous queues. Instead, uplink queues in the system model act as a placeholder for the real queues which reside in the QSTAs. Since the QAP is not aware of the status of uplink queues at the QSTAs, it always considers the uplink queues to be backlogged and therefore it always polls them at each

```
1.  h_sum = 0
2.  foreach synchronous flow i
3.  i.delta += i.H
4.  if ( i.direction == uplink )
5.     poll (i, i.delta)
6.     h = receive ()
7.  else
8.     h = transmit (i, i.delta)
9.  h_sum += h
10. delta = max (0, i.delta - h)
11. if ( TRT - h_sum > 0 )
12.    sleep (TRT - h_sum)
13.    TRT = TTRT
14. else
15.    TRT = TTRT + (TRT - h_sum)
```

Figure 3. WTTP algorithm pseudo-code.

round. If an uplink synchronous queue is not backlogged when polled by the QAP, the QSTA responds with a *Null* frame.

On the other hand, contention-based traffic is represented by a virtual queue (*asynchronous* queue). If the QAP finishes its CAP early, then it lets contention-based traffic take control of the medium for a limited time, so as to maintain a stable polling interval for the TSs. As shown in Figure 2, *WTTP* considers all queues in a round-robin fashion. However, in order to limit the duration of a round, a reference duration TTRT is selected. Synchronous capacities are selected as a fraction of the TTRT. The asynchronous queue, instead, is not reserved a portion of TTRT. However, on each round it is allowed to access the medium until the TTRT has elapsed. Therefore, any capacity temporarily not used by synchronous queues can be exploited by contention-free traffic.

WTTP continuously iterates the procedure in Figure 3, which corresponds to a token revolution. The variable `h_sum` is initialized with 0 (line 1) and stores the elapsed time from the beginning of the current revolution (9). All the synchronous queues are considered in turn (2). Each synchronous queue maintains a deficit counter (3) which takes into account the residual capacity that the queue could not exploit during the previous revolution; this value is non-zero if the head-of-line packet did not fit into the remaining service period capacity (10). This is the same mechanism used by the Deficit Round Robin scheduling algorithm [23], and it ensures fairness with variable length packets. If the queue corresponds to a downlink queue, then the QAP transmits one or more packets up to the value of the deficit counter (8). If the queue corresponds to an uplink queue, then the QAP polls the QSTA with a TXOP limit set to the deficit counter value (5-6), which also includes the time to poll the QSTA. In both cases, the time spent servicing the queue is stored in `h` and added to the duration of the ongoing revolution `h_sum` (9). When the schedule service finishes the QAP computes how early (late) the server is in visiting the asynchronous queue with regard to the expected revolution duration (11). This is achieved by recording in the `TRT` variable the residual time to exhaust the TTRT (11, 13); `TRT` is initialized at the server startup with the value of TTRT. If the token is early, then the contention-based traffic is left the remaining `TRT` (12-13).

Codec	Size	Hd	Pkts/s	Rate
G.711	160	40	50	80
G.723.1	30	40	22	12.3

Table 1. Audio traffic parameters.

Otherwise, if the token is late, the procedure resumes immediately (15).

It may be noted that the name of the `TRT` variable is inherited from the TTP, in which all the nodes keep a counter, named Token Rotation Timer (TRT), to measure the earliness (lateness) of the received token. Each TRT is initialized with TTRT and always counts down while the node is operational. A TRT is refilled with TTRT whenever it reaches zero and whenever an early token is received. Since *WTTP* has only one queue that uses the token status information, i.e. the asynchronous queue, the QAP only keeps one `TRT` variable.

A similar approach, i.e. applying the TTP rules to packet scheduling in a centralized manner, has been used by the Timed Token Service Discipline (TTSD) [18], proposed as an algorithm to schedule packets at a single output wired link for QoS and non-QoS flows. In [18] the authors show that (i) the synchronous flows are guaranteed a minimum rate and (ii) the asynchronous flows fairly share the bandwidth not reserved for (or temporarily not used by) the synchronous flows. These properties are expected to hold also for *WTTP*, though further investigation is needed to formally prove this assertion.

5. Traffic models

This section describes the traffic models used in the performance analysis. We consider two types of QoS traffic: audio and video. Data traffic is also considered. Stations with data traffic operate in asymptotic conditions, i.e. they always have a frame to transmit. The packet length of data traffic is constant and equal to 1500 bytes.

5.1. Audio

An audio stream is characterized by the encoding scheme and the packet arrival pattern. We simulate two different audio encoding schemes (G.711, G.723.1) whose parameters are shown in Table 1, where Hd is the cumulative size in bytes of IP/UDP/RTP headers, and Rate is the gross transmission rate in *Kbps* [7].

Audio traffic can be modeled as an ON/OFF source: during the ON periods (also called *talkspurt* periods) the traffic is CBR with parameters that depend on the encoding scheme; during the OFF periods (also called *silence* periods) no packets are generated.

We simulate the following ON/OFF sources:

- **cbr:** (limit case) the talkspurt period never ends;
- **traditional**: talkspurt and silence periods are distributed according to exponential distributions with mean 1.004 and 1.587, respectively [4].

5.2. Video

We simulate the video traffic according to pre-encoded MPEG4 trace files [9]. MPEG4 encoders produce video streams with frames of variable size at fixed intervals. The “Gupta” and “Reisslein” video streams (see Table 2) have been encoded using a frame rate of 30 *fps*, which corresponds to a frame interarrival

Trace	I *ms*	MR *Kbps*	PR *Mbps*	MDR *Kbps*
Fitzek	40	58	0.69	175
Gupta	33	70	0.84	154
Reisslein	33	72	1.0	182

Table 2. Video traces used in simulation.

Parameter	Value
SIFS (μs)	10
PIFS (μs)	30
DIFS (μs)	50
SlotTime (μs)	20
PHY header (μs)	192
Data rate (*Mb/s*)	11
Basic rate (*Mb/s*)	1

Table 3. Simulation parameters.

time of 33 *ms*, while the "Fitzek" (see Table 2) frame rate is 25 *fps*, corresponding to a frame interarrival time of 40 *ms*.

Table 2 reports the frame interarrival time (I), the Mean Rate (MR) and the Peak Rate (PR) of each video stream. The table also reports the additional Mean Data Rate (MDR) parameter to be used for the TSPEC negotiation. The latter is determined as the smallest rate such that the 95th percentile of the access delay is smaller than the frame interarrival time, assuming that the video stream is transmitted by a HCCA TS which is granted a TXOP of fixed duration at exactly the same rate as the frame rate.

6. Performance metrics

This section introduces the metrics used to evaluate the performance of the scheduling algorithms under consideration.

As far as HCCA is concerned, we consider the *access delay*. The latter is the time elapsed from the packet reaching the MAC layer to that packet being successfully acknowledged. We collect the Cumulative Distribution Function (CDF) of the access delay. Furthermore, we define the *Null rate* as the number of *Null* messages per second. The latter only applies to uplink flows and it is a direct measure of the overhead due to the specific polling scheme that is used by the corresponding scheduling algorithm.

With regard to contention-based access, we measure the throughput of a station that transmits data traffic in asymptotic conditions.

7. Performance analysis

In all our simulation scenarios, a variable number of stations contend to access the medium according to DCF and HCCA. The physical layer parameters are those specified by the High Rate Direct Sequence Spread Spectrum (HR-DSSS), also known as 802.11b, and are reported in Table 3. Although data integrity is a key requirement for data transmission, we assume that the wireless channel is error-free. Hence, MAC level fragmentation and multirate support are disabled. This assumption allows us to focus specifically on the system's performance in ideal conditions. Channel errors must obviously be taken into account in order to have a more realistic system model, but the analysis of their influence on the scheduling algorithms is left for future

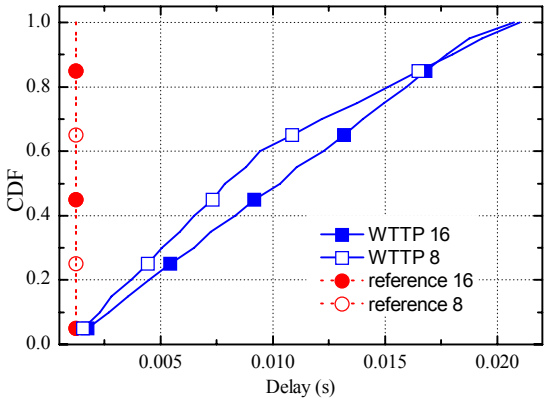

Figure 4. Scenario 1 access delay – CBR.

work. Furthermore, we assume that all nodes can directly communicate with each other. Therefore, the hidden node problem and packet capture are not taken into consideration, and the RTS/CTS protection mechanism is disabled.

We implemented the *reference* and the *WTTP* schedulers in the *ns-2* network simulator [12], using the HCCA implementation framework described in [6]. In all the simulation runs, we estimated the 95% confidence interval for each performance measure. However, in the curves reported below, the confidence intervals are not drawn whenever negligible.

Since we are comparing *WTTP* to the *reference* scheduler, we stick to the rule of setting the TTRT to the smallest Maximum Service Interval of the established traffic streams (i.e. the SI of the *reference* scheduler) and the synchronous capacities H_i to a proportional share of the TTRT, according to their Mean Data Rate. An optimal choice of TTRT and H_i requires further investigation and is left for future work.

We compare the *reference* and the *WTTP* algorithms under three different scenarios, as described below.

7.1. Scenario 1: audio uplink only

To compare the overhead of polling STAs in the *reference* and *WTTP* schedulers, we assess the performance of the system with a variable number of uplink G.711 audio flows, ranging from 1 to 16, the latter being the maximum number of such flows that the *reference* scheduler can support. The experiments were repeated using audio flows encoded according to the G.723 scheme. Since the results exactly match those related to G.711, they are not reported here.

Figure 4 shows the CDF of the audio packet access delay for both scheduling algorithms, in the case of CBR traffic and for two different numbers of established audio flows, i.e. 8 and 16, respectively. As expected, the *reference* scheduler performs much better than *WTTP* when the traffic is CBR. In fact, when the *reference* scheduler is used, the access delay is constant and lower than the packet interarrival time. This is because the packets are served periodically and consistently with their CBR arrival pattern, hence the access delay is fixed and equal to the offset between the service and the arrival start times, respectively. On the other hand, when *WTTP* is used, the CDF increases almost linearly up to a value nearly equal to the packet interarrival time, i.e. the access delay is distributed

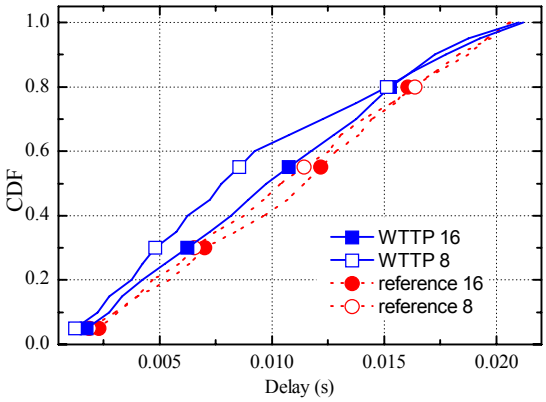

Figure 5. Scenario 1 access delay – ON/OFF.

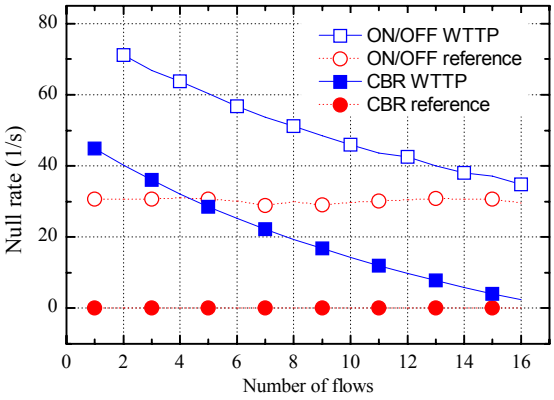

Figure 6. Scenario 1 null rate.

almost uniformly between zero and the packet interarrival time.

Figure 5 shows the CDF of the audio packet access delay for both scheduling algorithms, when the traffic is ON/OFF, and again for the two cases of 8 and 16 audio flows, respectively. In both cases, *WTTP* performs better than the *reference* scheduler. In fact, the *WTTP* CDF always lies to the left of the *reference* scheduler CDF, and the gap increases when the number of audio flows decreases. The main reason for this result is that, due to the inherent cycle time properties of *WTTP*, the latter polls the QSTAs at a higher rate than the *reference* scheduler, thus reducing the time an audio packet has to wait for the next TXOP.

This is confirmed by Figure 6, which shows the *Null rate* as a function of the number of flows for each scheduler. The average rate of the *Null* messages is almost constant when the *reference* scheduler is used. In particular, if the arrival pattern is CBR then the above rate is zero because the QAP exactly polls the QSTA once for each packet arrival: in this case the *reference* scheduler is optimum. On the other hand, if the audio source is ON/OFF, then the rate of the *Null* messages is greater than zero and it depends on the talkspurt-silence duty cycle. In both the CBR and the ON/OFF cases, the *Null rate* with *WTTP* is always greater than the corresponding value with the *reference* scheduler. Furthermore, it is not constant, but it decreases with the increasing number of audio flows.

In conclusion, this scenario shows that, as expected, when CBR traffic is considered the *reference* scheduler performs better than *WTTP* in terms of access delay.

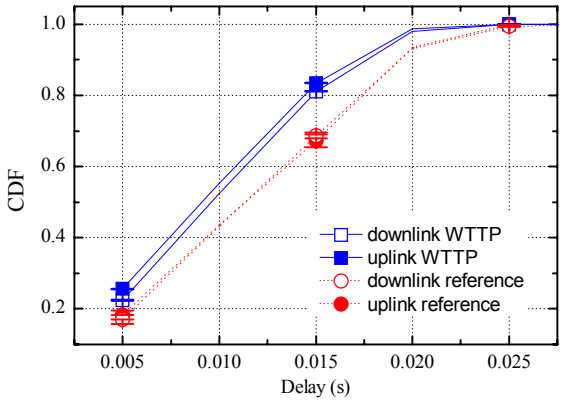

Figure 7. Scenario 2 access delay.

However, in the case of ON/OFF traffic, the opposite is true, since audio traffic always experiences a smaller delay with *WTTP* than with the *reference* scheduler.

7.2. Scenario 2: audio and data mix

To evaluate the amount of capacity provided by both algorithms to contention-based traffic in presence of QoS-guaranteed flows, we devised a scenario in which there is a mix of audio and data traffic. More specifically, there are ten stations with data traffic in asymptotic conditions, and a variable number of bi-directional G.711 audio flows, ranging from 2 to 10. The latter is the maximum number of flows that the *reference* scheduler can support. As in Scenario 1, the performance results with G.723 audio flows are comparable and therefore they are not reported.

Figure 7 shows the CDF of the access delay of uplink and downlink audio flows with ON/OFF traffic when their number is set to 10. As in Scenario 1, the CDF curves increase almost linearly irrespectively of the scheduling algorithm, up to a value nearly equal to the packet interarrival time. However, unlike the previous scenario, in this case a non negligible fraction of traffic experiences a delay greater than the packet interarrival time. This is because the CAP start times may be delayed when contention-based traffic is present. In fact, contention-based frames may (i) collide with each other shortly before a CAP is scheduled or (ii) collide with the first frame that the QAP transmits to start a CAP. However, Figure 7 also shows that the *WTTP* distributions, both uplink and downlink, always lie to the left of the corresponding ones of the *reference* scheduler. Furthermore, the amount of traffic experiencing a delay exceeding the packet interarrival time is about 0.02, whereas the corresponding value for the *reference* scheduler is about 0.08. The reason is the same as explained in the previous scenario, i.e. *WTTP* polls audio flows at a higher rate than the *reference* scheduler.

Figure 8 shows the throughput of a data station against the number of audio flows. When the audio source is CBR, then the throughput with the *reference* scheduler is higher than that with *WTTP*. This is because, as shown in the previous scenario, *WTTP* polls the audio flows at a higher rate than the packet arrival rate, and hence there is some unnecessary overhead due to *Null* message transmission. However, the *Null rate* is zero for the *reference* scheduler, and hence more ca-

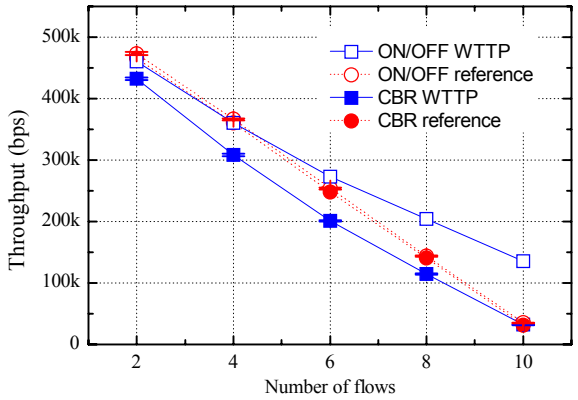

Figure 8. Scenario 2 throughput.

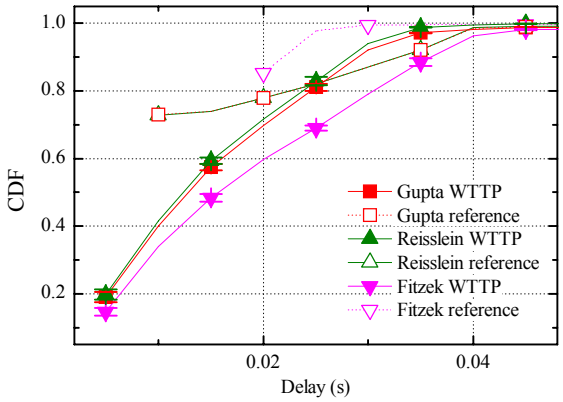

Figure 9. Scenario 3 access delay.

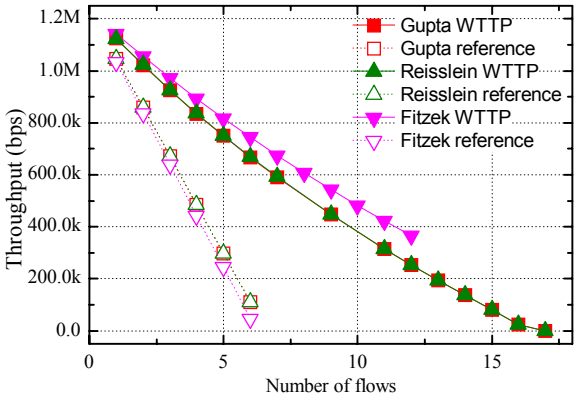

Figure 10. Scenario 3 throughput.

pacity is available for data traffic. Since the *Null rate* decreases when the number of audio flows increases, the gap decreases when the number of flows increases.

On the other hand, if the audio source is ON/OFF, the opposite is true, i.e. the throughput with *WTTP* is higher than that with the *reference* scheduler. In fact, in this last case, both the *reference* scheduler and *WTTP* experience a *Null rate* greater than zero, since they poll audio flows during their silence periods. However, the overhead due to polling an idle uplink TS is higher with the *reference* scheduler. In the latter case, when the QAP receives a *Null* message, the channel remains idle until the next TXOP or the end of the CAP, whereas *WTTP* immediately serves the next TS. The higher the number of audio flows, the higher the overall *Null rate*, hence, the gap between the curves increases.

In conclusion, this scenario confirms that *WTTP* improves the performance of the audio flows in terms of access delay, even in the presence of data traffic. Furthermore, *WTTP* provides the contention-based traffic with a higher amount of capacity at higher loads.

7.3. Scenario 3: Video and data mix

In this last scenario, we evaluate the performance with VBR traffic. The number of stations with data traffic in asymptotic conditions is set to 5; the number of bi-directional video flows ranges from 1 to the maximum number of admitted flows. As an admission control rule, we require the 95th percentile of the access delay to be smaller than the frame interarrival time.

Table 4 reports the maximum number of flows admitted by the *reference* and *WTTP* algorithms for each pre-encoded video trace. As shown in the table, the number of flows that can be served by *WTTP* is much higher than that of the *reference* scheduler, which is therefore less efficient from an admission control limit point of view. The poorer performance of the *reference* scheduler with VBR traffic is due to the fact that by necessity it wastes the overprovisioned capacity. In fact, when a VBR stream does not entirely exploit its allocated TXOP, the channel remains idle until either the start of the subsequent TXOP or the end of the current CAP. In such conditions, on the other hand, *WTTP* immediately serves the next TS or lets the contention-based traffic take control of the channel. Therefore, *WTTP* can actually exploit statistical multiplexing of VBR traffic.

Trace	*reference*	*WTTP*
Fitzek	6	12
Gupta	6	16
Reisslein	6	16

Table 4. Scenario 3: Number of flows admitted.

Figure 9 shows the CDF of the access delay of a video flow when the number of established video flows is the maximum allowed by the *reference* scheduler. With VBR traffic the behavior of both schedulers is dependent on the video trace: the "Fitzek" *WTTP* curve lies entirely below the *reference* one; instead, the "Gupta" and "Reisslein" *reference* curves have heavier tails than the *WTTP* ones. In all cases, the CDF values of *WTTP* are more uniformly distributed than the *reference* values; this is due to the round-robin nature of *WTTP*.

Figure 10 reports the throughput of the contention-based traffic against the number of established video flows for each video trace under consideration. In all cases, the throughput achieved with *WTTP* is considerably higher than that with the *reference* scheduler. The reason is again that *WTTP* can make the unused capacity available for contention-based access.

In conclusion, this scenario shows that *WTTP* is more efficient than the *reference* scheduler with video and data traffic mixture, in terms of: (i) maximum number of admitted video flows (ii) capacity allocated to contention-based traffic.

8. Conclusions and future work

In this work we devised a scheduling algorithm, based on the Timed Token Protocol (TTP), to be im-

plemented at the QAP of an IEEE 802.11e network to schedule flows with rate-based guarantees and allocate the unused capacity to contention-based traffic. Furthermore, we carried out a simulative analysis to compare the above scheduling algorithm with the *reference* scheduler proposed in the IEEE 802.11e standard.

This analysis highlighted that *WTTP* is more efficient than the *reference* scheduler when the HCCA traffic is VBR, in terms of (i) number of flows that the QAP may admit without significant performance degradation, and (ii) capacity allocated to the contention-based traffic. When the traffic is CBR there are cases when the *reference* scheduler is better than *WTTP* in terms of capacity allocated to the contention-based traffic.

This work can be extended in many directions, including, for example: analysis of the formal properties of the *WTTP* algorithm; adaptation of the system model to non-ideal channel conditions; and integration of queue estimation algorithms into the scheduling mechanism in order to reduce the MAC overhead.

REFERENCES

[1] A. Annese, G. Boggia, P. Camarda, L. A. Grieco, and L. Mascolo. Providing delay guarantees in IEEE 802.11e networks. *IEEE Semiannual Vehicular Technology Conference*, May 2004.

[2] P. Ansel, Qiang Ni, and T. Turletti. An efficient scheduling scheme for IEEE 802.11e. *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks*, March 2004.

[3] D. Bertsekas and R. Gallager. Data Networks. Prentice-Hall, 1987.

[4] P.T. Brady. A model for generating on-off speech patterns in two-way conversation. *Bell System Technical Journal*, vol. 48, pp. 2445–2472, September 1969.

[5] Y. Cao, O. K. Li. Scheduling Algorithms in Broad-Band Wireless Networks. *Proceedings of the IEEE*, vol. 89, no. 1, pp. 76-87, January 2001.

[6] C. Cicconetti, L. Lenzini, E. Mingozzi, and G. Stea. A software architecture for simulating IEEE 802.11e HCCA. *Internet Performance, Simulation, Monitoring and Measurement*, March 2005.

[7] Cisco Press. Traffic Analysis for Voice over IP, November 2001.

[8] J. Cowling and S. Selvakennedy. A detailed investigation of the IEEE 802.11e HCF reference scheduler for VBR traffic. *International Conference On Computer Communications And Networks*, October 2004.

[9] Frank H.P. Fitzek and Martin Reisslein. MPEG4 and H.263 Video Traces for Network Performance Evaluation. *IEEE Network*, vol. 15, no. 6, pp. 40–54., November/December 2001.

[10] A. Grilo, M. Macedo, and M. Nunes. A scheduling algorithm for QoS support in IEEE 802.11e networks. *IEEE Wireless Communications*, vol. 10, no. 3, pp. 36 -43, June 2003.

[11] R. M. Grow. A timed-token protocol for local area networks. *Proc. Electro/82, Token Access Protocols*, Electronic Conventions, Inc., May 1982.

[12] http://www.isi.edu/nsnam/ns/.

[13] IEEE Computer Society LAN MAN Standards Committee. IEEE 802.11: Wireless LAN Medium Access Control and Physical Layer Specifications, August 1999.

[14] IEEE Computer Society LAN MAN Standards Committee. IEEE 802.11: Wireless LAN Medium Access Control and Physical Layer Specifications. Medium Access Control (MAC) Quality of Service (QoS) Enhancements, D13.0, January 2005.

[15] IEEE Computer Society LAN MAN Standards Committee. IEEE 802.11: Wireless LAN Medium Access Control and Physical Layer Specifications. Medium Access Control (MAC) Radio Resource Measurement, D1.6, February 2005.

[16] A. Köpsel, J.-P. Ebert, and A. Wolisz. A performance comparison of point and distributed coordination function of an IEEE 802.11 WLAN in the presence of real-time requirements. *International Workshop on Mobile Multimedia Communications*, October 2000.

[17] T. Korakis and L. Tassiulas. Providing quality of service guarantees in wireless LANs compliant with 802.11e. *Computer Networks*, vol. 47, no. 2, pp. 239–255, February 2005.

[18] L. Lenzini, E. Mingozzi, and G. Stea. Design and performance analysis of the generalized timed token service discipline. *IEEE Transactions on Computers*, vol. 53, pp. 879–891, July 2004.

[19] A. Lindgren, A. Almiquist, and O. Schelen. Quality of Service Schemes for IEEE 802.11 Wireless LANs - An Evaluation. *Mobile Networks and Applications*, vol. 8, no. 3, pp. 223–235, June 2003.

[20] S. Mangold, S. Choi, P. May, O. Klein, G. Hiertz, and L. Stibor. IEEE 802.11e wireless LAN for quality of service. *Proc. European Wireless*, vol. 1, pp. 32–39, February 2002.

[21] 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. 2, no.2, pp. 137-150, April 1994.

[22] K. C. Sevcik and M. J. Johnson. Cycle Time Properties of the FDDI Token Ring Protocol. *IEEE Transactions on Software Eng.*, vol. 13, no. 3, pp. 376-385, March 1987.

[23] M. Shreedhar and G. Varghese. Efficient fair queuing using deficit round-robin. *IEEE/ACM Transactions on Networking*, vol. 4, no. 3, pp. 375–385, June 1996.

[24] D. Stiliadis and A. Varma. Latency-Rate Servers: A General Model for Analysis of Traffic Scheduling Algorithms. *IEEE/ACM Transactions on Networking*, vol. 6, pp. 675-689, October 1998.

[25] H. Zhang. Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks. *Proceedings of the IEEE*, vol. 83, no. 10, pp. 1374-1396, October 1995.