AU-APSD: Adaptive IEEE 802.11e Unscheduled Automatic Power Save Delivery

Xavier Pérez-Costa
NEC Network Laboratories, Germany
Email: perez@netlab.nec.de

Daniel Camps-Mur
NEC Network Laboratories, Germany
Email: camps@netlab.nec.de

***Abstract*— The integration of the wireless LAN technology in mobile devices such as cellular phones or PDAs has become a user need due to its popularity for providing high speed wireless Internet access at a low cost. Such devices though should meet users' expectations with regard to QoS, e.g., guarantee a reasonable voice quality when VoIP is used, and power saving efficiency, e.g., standby and calling times should be similar to the ones of cellular phones. IEEE 802.11e defines QoS and power saving enhancements that should allow the wireless LAN technology address users' wishes in such specific devices. Our focus is the study of the distributed power saving mechanism of 802.11e, i.e., U-APSD, as compared to the legacy 802.11 power saving mode in order to assess its suitability for solving the challenges of the upcoming mobile devices requirements. Our contributions are as follows. We provide first an overview of the U-APSD functionality. Then, we describe in detail our proposed implementation of the U-APSD mechanism, Adaptive U-APSD (AU-APSD), a generic solution that requires only information available at the MAC layer. Finally, we quantify the performance improvements that are obtained with our proposed AU-APSD implementation as compared to the legacy 802.11 power save mode.**

I. Introduction

The wide adoption of the IEEE 802.11 wireless LAN technology by home and business users due to its capability of providing low cost high speed wireless Internet access is driving a strong trend toward the inclusion of this technology in mobile devices like cellular phones, personal digital assistants (PDAs) or laptops. Several challenges though need to be addressed with respect to QoS and power saving limitations to achieve a seamless integration of Wireless LAN in such devices.

The 802.11 Task Group E has recently completed the design of the 802.11e extension of the standard [1], which introduces QoS and power saving MAC enhancements. 802.11e defines the Hybrid Coordination Function (HCF) to support QoS. Two channel access methods are defined: a contention-based channel access method called the Enhanced Distributed Channel Access (EDCA) and a contention-free channel access referred to as HCF Controlled Channel Access (HCCA). A thorough overview of the 802.11e QoS enhancements can be found in [2].

Regarding the battery usage efficiency, the intrinsic nature of 802.11, which is based in a shared channel access (CSMA/CA), forces wireless stations to continuosly listen to the channel to determine its current status. As a result, a handheld device connected through an 802.11 wireless LAN, will drain its battery after a few hours as opposed to current mobile devices that have a standby battery lifetime up to several days, e.g., cellular phones. Ideally, mobile devices including the wireless LAN technology should achieve a battery consumption similar to current handheld devices in order to meet users' expectations.

The IEEE 802.11 standard provides a power save mode that reduces the time required for a station to listen to the channel. Once every Beacon interval, usually 100ms, the access point (AP) sends a Beacon indicating whether or not a certain station[1] has any data buffered at the AP. Wireless stations wake up to listen to Beacons at a fixed frequency and poll the AP to receive the buffered data by sending power save polls (PS-Polls). Whenever the AP sends data to a station, it indicates whether or not there are more data frames outstanding, using the More Data bit in the data frames, and a station goes to sleep only when it has retrieved all pending data. Although this mechanism significantly alleviates the power consumption problem, a dependency between the data frames MAC downlink delay (AP to station) and the listen interval is introduced. Consequently, some listen interval values can result in downlink delays unacceptable for certain QoS-sensitive applications, e.g., VoIP. Further details about the power save mode operation can be found in [3].

In order to address some of the power saving issues that arise when a specific QoS is desired, the 802.11e standard includes an optional extension of the 802.11 power save mode defined as Automatic Power Save Delivery (APSD). If the AP supports APSD, 802.11e-capable stations can select the method for the delivery of the frames buffered at the AP between standard Power Save Mode or APSD. A main difference between the 802.11 standard power save mode and APSD is that with APSD a station is awake during a Service Period[2] (SP) instead of being awake from the transition to the awake state for receiving a Beacon until the return to the doze state after acknowledging the receipt of the last frame buffered at the AP. Two types of SPs are possible under APSD: unscheduled and scheduled. Unscheduled SPs (U-APSD) are defined only for stations accessing the channel using EDCA

[1] In this paper AP and station refer to what in the 802.11e standard is denoted as QAP and non-AP QSTA respectively.

[2] The definition of a SP will be given in the next section

1-4244-0355-3/06/$20.00 (c) 2006 IEEE

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

while Scheduled SPs (S-APSD) are defined for both access mechanisms, EDCA and HCCA.

New mobile devices incorporating 802.11e functionality are likely to include first the 'distributed' mechanisms of 802.11e, i.e., EDCA and U-APSD, than the centralized ones, i.e., HCCA and S-APSD. This can be seen for instance in the fact that the Wi-Fi^{TM} Alliance [4] has started first the certification of the Wi-Fi^{TM} multimedia extensions (WMM^{TM}) and the WMM Power Save^{TM}, which include EDCA and EDCA plus U-APSD functionality respectively, while the certifications for HCCA and S-APSD are being deferred. Based on that, we focus our study in the analysis of the performance enhancements provided by EDCA in combination with U-APSD since it is the most probable configuration to be implemented in the short-term in 802.11e-capable mobile devices.

In the area of providing QoS in a wireless LAN a lot of research has been done during the last several years, see for example [2], [5]. With respect to the 802.11 power save mode, the infrastructure mode has been studied for instance in [6],[7] where the main focus was to improve the performance for web-like traffic. Regarding U-APSD, in [8] an early specification of U-APSD (802.11e draft 6.0) was studied where the U-APSD power saving performance was compared to the 802.11 power save mode one in a scenario where all users generate and receive constant bit rate voice traffic. In our previous work [9] we studied the effect of using the standard power save mode in conjunction with the 802.11e QoS mechanisms, in [10] we proposed an adaptive algorithm to control the delay introduced by the 802.11 power save mode for applications requiring QoS guarantees, and in [11] we studied the performance improvements obtained when U-APSD is used, configured in a static way, as compared to standard power save mode. The static U-APSD solution limits the MAC downlink delay introduced by U-APSD to a certain specified value depending on the Access Category (AC). The proposal though relies in the assumption that either these limits will be provided by higher layers to the MAC layer or that a common value valid for all types of traffic using the same Access Category can be determined. The paper at hand extends our previous results by proposing a generic way to configure U-APSD that requires only knowledge available at the MAC layer.

The rest of the paper is structured as follows. In Section II an overview of the U-APSD functionality is given. Section III describes our proposed *adaptive* implementation of U-APSD, *AU-APSD*. The performance results obtained by using the proposed AU-APSD implementation as compared to 802.11 power save mode are provided in Section IV. Finally, Section V summarizes the results and concludes the paper.

II. UNSCHEDULED APSD (U-APSD)

Unscheduled Automatic Power Save Delivery (U-APSD) is the APSD method defined in 802.11e to improve the QoS provided to stations accessing the channel using the EDCA mechanism as compared to legacy power save mode. The main idea behind the U-APSD design is the usage of data frames sent in the uplink by stations (STA $\rightarrow$ AP) as indications (*triggers*) of the instants when the power saving stations are awake and then take advantage of it for delivering any data frames that were buffered at the AP while the stations were in doze mode. Because of its specific functionality this method is specially suited for bi-directional traffic streams even though it provides alternative methods for its usage in other cases. In the following we describe in detail the U-APSD functionality assuming a basic knowledge of the 802.11 standard power save mode and of the EDCA mechanism of 802.11e. Please see [2] and [9] for an overview of 802.11e and standard 802.11 power save mode respectively.

A. Functionality Description

As previously mentioned, the main difference between the power saving method defined in the 802.11 standard and APSD is that with APSD a station is awake during a Service Period (SP) instead of being awake from the transition to the awake state for receiving a Beacon until the return to the doze state after acknowledging the receipt of the last frame buffered at the AP.

An *unscheduled SP* begins when the AP receives a *trigger frame* from a station and ends when the station receives a QoS Data or QoS Null frame indicating the end of the service period (EOSP). QoS Null frames are the substitutes in U-APSD of PS-Polls, and allow a station to request the delivery of the frames buffered at the AP even if a station has no data frame to transmit in the uplink. This enables the usage of U-APSD by applications which do not generate uplink traffic often enough to meet the QoS application requirements.

Each AC at the stations can be configured separately to be delivery/trigger-enabled respectively. When a station has an AC configured as *delivery-enabled*, the AP is allowed to use EDCA to deliver traffic from the AC to a STA during an unscheduled SP triggered by a station. When a station AC is *trigger-enabled*, frames of subtype QoS Data and QoS Null from the station, that map to the AC, trigger an unscheduled SP if one is not in progress.

During a SP one or more data frames of delivery-enabled ACs might be delivered by the AP to a station up to the number of frames indicated in the maximum service period length following the rules of an acquired transmission opportunity. The maximum service period length is a field contained in the QoS Info field filled by the station at association. In each directed data or management frame associated with delivery-enabled ACs sent to a station, the More Data (MD) bit indicates that more frames are buffered for the delivery-enabled ACs. For all frames except for the final frame of the SP, the EOSP subfield of the QoS control field of the QoS data frame shall be set to 0 to indicate the continuation of the SP.

In order to guarantee backward compatibility of legacy stations that do not support APSD, the procedure of the AP to assemble the traffic indicator map (TIM) has been modified in such a way that if at least one of the ACs is non delivery-enabled it indicates the buffer status *only* of the ACs non delivery-enabled. Note that for this case it means that the Beacon will not indicate whether frames of ACs delivery-enabled are buffered. Only in the case that all ACs are delivery-enabled the TIM indicates the buffer status of delivery-enabled ACs.

B. U-APSD advantages with respect to 802.11 power save mode

Three main advantages are introduced with U-APSD with respect to 802.11 power save mode.

- The first advantage is the improvement of the QoS of the ACs with U-APSD enabled since, because of the possibility of generating triggers at any point of time, the delay introduced by U-APSD can be limited based on the application QoS needs instead of depending on the listen interval configuration of the 802.11 power save mode.
- The second advantage is the reduction of the overhead required to retrieve frames from the AP thanks to the usage of data frames as triggers. This is specially relevant for symmetric applications like VoIP, because then almost no QoS Null frames are required and therefore the significant PS-Poll overhead required by the standard 802.11 power save mode is significantly reduced.
- The third advantage is again an important reduction of the overhead required to retrieve frames from the AP depending on the maximum service period length (Max_SP_Length) per trigger. While for 802.11 power save mode a PS-Poll is required for obtaining each single frame buffered at the AP, this is not necessary for U-APSD which can deliver up to Max_SP_Length frames per trigger.

III. ADAPTIVE U-APSD (AU-APSD)

The U-APSD specification provided in the IEEE 802.11e standard introduces different mechanisms to control the QoS provided to a station for each different AC. The specific usage of these mechanisms to actually deliver the desired QoS is though, as usual, left open to allow differentiation between vendors. In the following we describe our proposed implementation of U-APSD which aims to be a *generic* solution to provide a QoS adapted to the different applications needs. Our proposal is an *Adaptive* U-APSD power saving algorithm that requires only information available at the MAC layer. This algorithm will be referred in the rest of the paper as AU-APSD.

A. Design Considerations

The U-APSD power saving mechanism, as the legacy 802.11 power save mode, is based on the usage of an AP as a buffering entity for stations being in doze mode in order to be able to enter in a power save mode without losing any data frame. The frames buffered at the AP are then retrieved by a station by indicating to the AP that they have changed to the awake state again. Although this mechanism is very efficient to reduce the power consumption of stations and to avoid frame losses, it introduces two performance challenges that have to be carefully addressed in order to guarantee that the power consumption improvement does not result in a significant degradation of the QoS provided or the wireless LAN capacity:

- Variable delay in the delivery of frames in the downlink direction (AP → station) depending on the trigger frame generation rate which has to be bounded to a value acceptable for applications requiring QoS guarantees.
- Signaling load introduced in the channel which can significantly degrade the wireless LAN capacity.

In order to limit the MAC downlink delay introduced by U-APSD the first problem that needs to be solved is to guarantee that triggers will be generated to meet the delay requirements even if the applications running at the station do not generate enough data frames (triggers). A possible solution to this problem is to schedule at fixed time intervals the generation of QoS Null frames to bound the maximum delay introduced by U-APSD. Additionally, to avoid the introduction of unnecessary overhead in the channel due to the generation of QoS Null frames in an instant where it would not have been necessary, we propose to re-schedule the QoS Null frame generation each time that the station sends a data frame in the uplink belonging to a trigger-enabled AC. In this way, we guarantee that a QoS Null frame will be generated to trigger a SP only after reaching the maximum delay we would like to allow.

Since our objective is to design an U-APSD algorithm generic enough to meet the different QoS requirements of *all* ACs we consider the case of setting *all* ACs for the stations as trigger- and delivery-enabled as the most favourable one. By doing this, we minimize the signaling load introduced in the channel by the stations since every data frame sent in the uplink acts as a trigger, and a single trigger from any AC makes the AP deliver the traffic buffered available from any of the AC. Moreover, the proposed configuration has the advantage of allowing stations to check in the Beacon whether frames have been buffered at the AP which enables for further power saving in some situations as we will show in the next sections.

Once a trigger from a certain station is received at the AP we start delivering first the frames of the AC with the highest priority and continue until all buffered frames for a certain station have been delivered or the maximum service period length has been reached. Since due to the maximum service period length it could happen that a SP concludes before all the frames of a certain delivery-enabled AC had been delivered to a station, we use the More Data (MD) information provided in the data frames to decide whether a new SP has to be started after receiving the end of service period (EOSP) indication.

The maximum service period length is another parameter that needs to be configured for U-APSD. In our proposed implementation we configure the maximum service period length (Max_SP_Length) such that it allows to send the maximum number of frames allowed by the 802.11e specification, i.e., all. We have chosen this option for two main reasons. First, because by retrieving the frames from the AP as soon as possible we maximize the chances that the traffic meets its delay requirements. Second, because the more frames sent per trigger the less signaling overhead introduced in the channel. Note that even if the Max_SP_Length has no limit, the maximum time that a station can keep control of the channel will continue to be bounded by the TXOP limit.

Finally, to avoid that high priority traffic gets dropped in congestion conditions at the AP due to the low retrieval rate of low priority one, we propose to have separate power save mode queues at the AP (physical or logical) for each AC to guarantee that no high priority traffic is dropped because of having the power save mode queue full of low priority traffic.

B. AU-APSD algorithm

Based on the design considerations described in the previous section we have designed an $adaptive$ U-APSD algorithm for the stations that estimates at the MAC layer the lowest downlink frame interarrival time of all ACs. Note that we only need to schedule the transmission of QoS Null trigger frames for the AC with the minimum downlink frame interarrival time because the requirements of the other ones will be fulfilled by the most stringent one. We have chosen this approach because in general applications can cope with an end-to-end delay well above their frame generation rate. For instance a delay sensitive application such as VoIP, which codecs usually generate frames every 10-30ms, can deal with an overall delay of 150-300 ms. Considering that in most of the networks including wireless LAN access the main contributor to the overall delay is usually the MAC layer, upper bounding the MAC downlink delay to the downlink frame interarrival time should satisfy the most stringent application requirements.[3] The objectives of our proposed algorithm are as follows:

1) Provide a $soft$ upper bound of the MAC downlink delay according to the minimum downlink frame interarrival time observed of all ACs
2) Keep the bound guarantee even in the case of more than one application per station of the same AC
3) Guarantee a power saving efficiency similar or better than the one of standard 802.11 power save mode
4) Minimize the required signaling load

In order to do so, the algorithm consists of two basic steps. The first one is to decide which is the AC sending frames more often, this will be done at the end of each SP by simply keeping track of the frames received from each AC in successive SPs. The second one is the estimation of the downlink frame generation rate of the AC selected in the first step. The basic idea to estimate the downlink interarrival time of the traffic of a given AC is to count the number of frames retrieved from the AP corresponding to the selected AC, $n\_frames\_rcvd[AC]$, during a certain period of time, Δ_t, and then compute the AC downlink interarrival time estimation as:

$$\widehat{interarrival\_time} = \frac{\Delta_t}{n\_frames\_rcvd[AC]} \qquad (1)$$

[3]Note that the algorithm could be easily extended to provide an upper bound for the downlink delay proportional to the minimum estimated downlink interarrival time

Considering the way the estimation is computed in the previous equation, the longer Δ_t is considered the more accurate the estimation will be in a period where the downlink frame generation rate has been stable. Based on that, we define a $fine$ estimation that keeps updating the estimation at the end of each SP considering all the information collected during the period where the downlink frame interarrival time has been stable. An algorithm based only on this estimation though would not be able to react fast to sudden changes in the downlink frame generation rate because the weight of the information of the latest SPs as compared to the previous ones could be too low. Therefore, we use a second parallel estimation, $rough$, that considers the information only between $events$. These events indicate that our current estimation is above or below the actual value.

To select which of the two different estimations, $rough$ and $fine$, should be used at a certain point of time we defined two thresholds, $rough\_thr$ and $fine\_thr$, that are checked, each time there is an estimation update, to determine which one should be used as QoS Null trigger interval, $trigger\_interval$. When the algorithm is started, the *rough* estimation is used until the *fine* one is considered reliable enough, i.e., the variation within the last *fine* estimations is below the $fine\_thr$ threshold. Then, each time a change in the *rough* estimation of the minimum downlink frame interarrival time above $rough\_thr$ is detected, the *fine* estimation is initialized and the *rough* one takes over until the *fine* is considered reliable again.

For the $rough$ estimation the challenge is thus to detect when our current estimation does not match the actual real value and react accordingly. The time interval between two of such indications will determine the value of Δ_t which therefore varies at each event. Ideally, if the computed trigger interval would perfectly match the minimum AC downlink frame interarrival time, when a trigger is sent to the AP a *single* data frame of that AC would be retrieved from the AP. When the trigger interval is not yet *adapted* though what happens is that, if the trigger interval is above the downlink frame interarrival time, the station will receive more frames than expected during a SP. While if the trigger interval is below the downlink data frame generation rate, then the station will either receive a QoS Null frame with the EOSP flag set or will simply retrieve frames from ACs different than the selected one. Therefore, our algorithm updates the estimation each time an event occurs of the following type, which will be referred as $Event\_rough_{AC}$:

1) A SP is started by a QoS Null trigger but no frame from the currently selected AC is received within the SP.
2) More than one frame from the currently selected AC is received within a SP.

It can be proved that the distance between these events, Δ_t, increases when the estimation gets closer to the actual downlink interarrival time. Therefore the $rough$ estimation tends to converge to the desired value.

Once the AU-APSD algorithm is started, the station sends periodically triggers to the AP starting from the configurable predefined value $trigger\_interval\_init$ and converging to the minimum downlink data frame interarrival time of all ACs following the procedure described in Algorithm 1. To avoid unnecessary signaling load, we disable the QoS Null trigger generation when we detect that an inactivity traffic period has started (the method to detect that is explained below). In order to determine when to start the QoS Null trigger generation, we take again advantage of our design requirement of configuring *all* ACs as delivery-enabled and monitor the TIM in the Beacon. Once traffic is detected, the QoS Null trigger generation is started again. Note that when the AU-APSD algorithm is started the station does not need to wake up to listen to Beacons anymore.

In the algorithm, $trigger\_interval$ is the actual interval used to schedule QoS Null trigger transmissions, $n\_fr\_rcvd\_SP[AC]$ counts the number of frames retrieved from the AP of each AC during a SP, $n\_fr\_rcvd\_fine[AC]$ and $n\_fr\_rcvd\_rough[AC]$ count the number of frames received for each AC since the last initialization of their respective estimation and $Event\_rough_{AC}$ is a condition that is true when one of the two events previously defined occur. The $asymmetry\_factor$ has been introduced to account for the error we consider acceptable for our estimation. Since to minimize the signaling it is preferable to have a estimation slightly above the actual value than below it, we add this factor to the $trigger\_interval$ value used.

Two exceptional cases have been considered in the algorithm to cope with sudden changes in the downlink frame generation rate. 1) When during a SP the number of received data frames is significantly higher than expected, a relevant modification of the AC downlink frame interarrival time has been detected. Hence, if this happens $Long\_Data\_Burst$ times in a row, the estimation is obtained by directly dividing the current $trigger\_interval$ by the number of frames retrieved in this SP instead of considering the time between the last event and the current one. In this way we avoid that an isolated burst instead of a real change in the AC frame interarrival time results in a sharp estimation modification. 2) On the other hand, when the number of consecutive SPs finished without receiving frames of the currently considered AC is above the $Long\_No\_Frames\_burst$ threshold, a period of inactivity of the downlink traffic has been detected and the generation of QoS Null frames is cancelled until a Beacon is received indicating that traffic has been buffered at the AP. If the threshold is not reached though, the $trigger\_interval$ is doubled since the algorithms interprets that a significant increase of the downlink interarrival time has occurred.

The complexity of the mechanism has been kept low by minimizing the parameters to be configured (6) and by distributing the computation load between the stations instead of centralizing it at the AP.

Algorithm 1 AU-APSD Adaptive algorithm

Data Frame Arrival of Access Category AC **& AU-APSD active**
$n\_fr\_rcvd\_SP[AC] \leftarrow n\_fr\_rcvd\_SP[AC] + 1$
$n\_fr\_rcvd\_fine[AC] \leftarrow n\_fr\_rcvd\_fine[AC] + 1$
$n\_fr\_rcvd\_rough[AC] \leftarrow n\_fr\_rcvd\_rough[AC] + 1$

Frame Arrival with EOSP & !MD & AU-APSD active
$AC \leftarrow decide\_lowest\_interarrival\_AC(n\_fr\_rcvd\_SP)$
if $fine\_estimation\_active$ **then**
  $\Delta t_{fine} \leftarrow current\_time - time\_first\_event\_fine$
  $fine(n) \leftarrow \frac{\Delta t_{fine}}{n\_fr\_rcvd\_fine[AC]}$
  $W(n) \leftarrow [fine(n) \ldots fine(n-k)]$
  if $min\,\{W(n)\} \geq (1 - fine\_thr) \times max\,\{W(n)\}\ \cap\ !using\_fine$ **then**
    $trigger\_interval \leftarrow max\,\{W(n)\} \times (1 + asymmetry\_factor)$
    $using\_fine \leftarrow TRUE$
  end if
else
  $fine\_estimation\_active \leftarrow TRUE$
  $time\_first\_event\_fine \leftarrow current\_time$
  $using\_fine \leftarrow FALSE$
  $n\_fr\_rcvd\_fine[AC] \leftarrow 0$
end if

if $Event\_rough_{AC}$ **then**
  $\Delta t_{rough} \leftarrow current\_time - time\_last\_event\_rough$

  if $n\_fr\_rcvd\_SP[AC] > 2$ **then**
    $n\_Long\_Data\_burst \leftarrow n\_Long\_Data\_burst + 1$
  end if
  if $n\_fr\_rcvd\_SP[AC] == 0$ **then**
    $n\_Long\_No\_Frames\_burst \leftarrow n\_Long\_No\_Frames\_burst + 1$
    if $n\_Long\_No\_Frames\_burst == Long\_No\_Frames\_burst$ **then**
      Disable AU-APSD
    end if
  end if

  if $n\_Long\_Data\_burst < Long\_Data\_burst$ **then**
    if $n\_fr\_rcvd\_rough[AC] > 0$ **then**
      $rough(n) \leftarrow \frac{\Delta t_{rough}}{n\_fr\_rcvd\_rough[AC]} \times (1 + asymmetry\_factor)$
    else
      $rough(n) \leftarrow 2 \times rough(n-1)$
    end if
  else
    $trigger\_interval \leftarrow \frac{trigger\_interval}{n\_fr\_rcvd\_SP[AC]}$
    $n\_Long\_Data\_burst \leftarrow 0$
  end if

  if $rough(n-1) \times (1 + rough\_thr) \geq rough(n) \geq rough(n-1) \times (1 - rough\_thr) \& !using\_fine$ **then**
    $trigger\_interval \leftarrow rough(n)$
  else
    $fine\_estimation\_active \leftarrow FALSE$
    $using\_fine \leftarrow FALSE$
    $trigger\_interval \leftarrow rough(n)$
  end if
  $n\_fr\_rcvd\_rough[AC] \leftarrow 0$
  $time\_last\_event\_rough \leftarrow current\_time$
end if

for $each\ AC$ **do**
  $n\_fr\_rcvd\_SP[AC] \leftarrow 0$
end for

IV. Performance Evaluation & Discussion

In this section we evaluate the performance of our proposed AU-APSD algorithm as described in Section III-B with respect to legacy 802.11 standard power save mode (StdPSM). The analysis has been performed using the OPNET [12] simulator.

EDCA-TXOP durations are configured for all ACs to allow the transmission of one data frame after gaining access to the medium. Note that this is a worst case scenario for U-APSD as compared to 802.11 standard power save mode since U-APSD could make use of TXOP mechanism together with the Max_SP_Length to further reduce the dowlink delay. The RTS/CTS mechanism has not been enabled to avoid its influence over the mechanisms being studied. The Beacon interval used is 100ms and the listen interval configured for the standard power save stations is 1. The maximum service period length configured allows to send all frames buffered for a certain station.

The 802.11b physical layer is used in our simulations and the AU-APSD algorithm is configured such that $trigger\_interval\_init$ is 20ms, $Long\_No\_Frames\_Burst$ is 3, $Long\_Data\_Burst$ is 2, $fine\_thr$ is 0.01, $rough\_thr$ is 0.1 and $asymmetry\_factor$ is 0.05.

Since our focus is on the differences in the performance between the power saving mechanisms when being used together with EDCA and not on the EDCA parameters configuration, we assume a fixed configuration of the EDCA QoS parameters based on the 802.11e document recommendation [1], taking $aCWmin$ equal 127 and $aCWmax$ equal 1023.

The length of the simulations performed is 300 seconds. The number of seeds has been selected such that the variance between the different simulation runs is small enough to not affect the provided results.

To clearly observe the differences in the performance between standard 802.11 power save mode and our proposed AU-APSD algorithm we start with an scenario of four wireless LAN stations where each station is configured to send and receive traffic of a different AC from their corresponding pair in the wired domain, i.e., one station sends and receives AC_VO traffic, a second one sends and receives AC_VI traffic and so on. We study the impact of increasing the number of stations over: the uplink/downlink MAC throughput and delay differentiation between the different ACs, the power saving efficiency and the resulting power saving costs. Note that in general the downlink results will be always worse than the uplink ones since the power saving mechanisms achieve the power usage reduction by limiting the amount of time that a station is awake expecting frames buffered at the AP.

To evaluate the performance of our AU-APSD algorithm against the standard power save mode we consider an scenario with 'realistic' applications:

- AC_VO: G.711 Voice codec with silence suppression. Data rate: 64kbps. Frame Length: 20ms. Talk spurt exponential with mean 0.35s and silence spurt exponential with mean 0.65s.
- AC_VI: MPEG-4 traces of the movie 'Star Trek: First Contact' obtained from [13]. Target rate: 68kbps. Frame Length: 40ms.
- AC_BE: HTTP/Web browsing. Page interarrival time exponential with mean 10s. Page size 10KB plus 1 to 5 images of a size uniformly distributed between 10KB and 100KB.
- AC_BK: E-mail. Send interarrival time exponential with mean 120s. Receive interarrival time exponential with mean 60s. Size exponential with mean 100KB

A. QoS Differentiation

Figures 1 and 2 illustrate the impact of increasing the number of stations on the MAC uplink delay and throughput respectively. Regarding the delay, it can be observed that for all the access categories it is lower when using our AU-APSD algorithm than when the stations are configured to use legacy power save mode. The reason is that the medium is less congested because our AU-APSD algorithm, unlike standard power save mode, schedules the transmission of QoS Null triggers by the stations in a non synchronized way. Regarding the throughput, the only difference is that AC_BK starts to saturate in the standard power save mode case for 60 stations.

The corresponding results with respect to the downlink case are shown in Figures 3 and 4. As expected, the AU-APSD downlink delay outperforms again in general the legacy power save mode one. However, the reasons for some of the results are not obvious and require further explanation. The first surprising result is that when the network is not saturated yet the AC_VO and AC_VI applications are experiencing the same downlink delay, around 40ms. Considering the applications used for each access category this seems to be correct for the AC_VI case but not for the AC_VO one. The reason for the AC_VO result comes from the fact that we use a voice codec with silence suppression. When using this codec, the AU-APSD mechanism is enabled and disabled in order to reduce signaling load for each downlink active period. To realize that there is a new downlink active period starting when the algorithm is disabled, the wireless station must wait for a Beacon frame that in our scenario is sent each 100ms. Therefore, the first frames of each burst will see a higher delay. Note that the importance of this effect would get reduced if the active period length would become longer. Another thing to point out about the downlink delay is the fact that the Web browsing and E-mail applications, sent with AC_BE and AC_BK respectively, also have a very low delay when the network is not saturated. The reason is that these applications come in a bursty fashion, and the bursts are so dense that the wireless station can download the whole burst without going back to sleep, because of our Max_SP_Length parameter. Regarding the downlink throughput, AU-APSD also outperforms legacy power save mode because, by sending triggers more often, the AP queues are emptier than in the standard power save mode case resulting in a lower frame dropping rate in the AP.

B. Power Saving Cost

In this section we evaluate which are the costs of the power saving schemes under study in terms of signaling load and channel capacity. The upper part of Figure 5 shows the total signaling load introduced in the channel by the different mechanisms, i.e., PS-Poll frames for the standard power save mode case and QoS Null frames for AU-APSD. Clearly, a higher signaling load is introduced in the AU-APSD case as compared to standard power save mode. The reason for such difference is mainly the signaling load we have in the downlink for AU-APSD. In U-APSD, if a wireless station sends a trigger in order to start a SP but there is no data buffered addressed to that station in the power save buffer, the AP has to send a QoS Null to the station to finish that SP. This situation is very likely to happen in our scenario because of the AC_VO application that uses a codec with silence suppression in the uplink and

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE ICC 2006 proceedings.

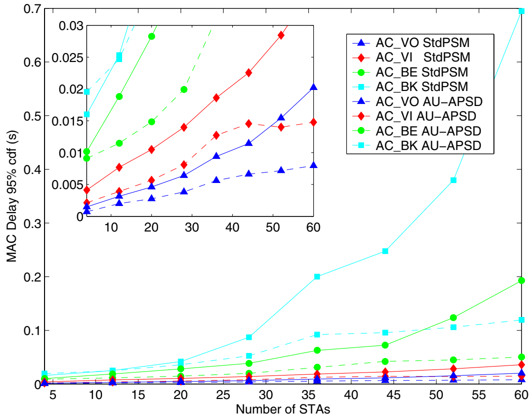

Fig. 1. Impact of the number of stations on the MAC uplink delay

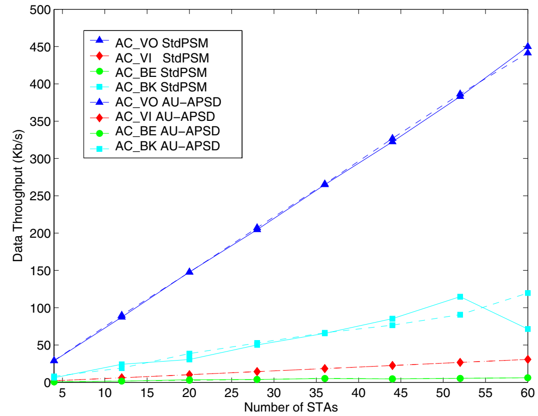

Fig. 2. Impact of the number of stations on the MAC uplink throughput

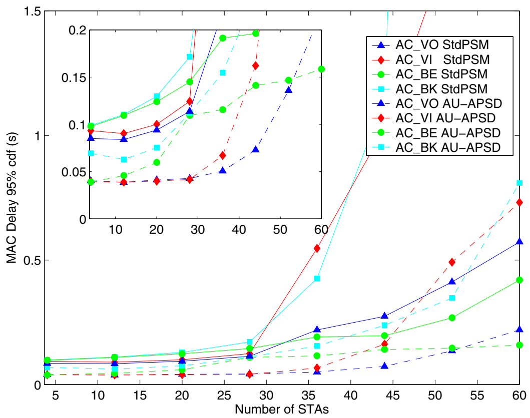

Fig. 3. Impact of the number of stations on the MAC downlink delay

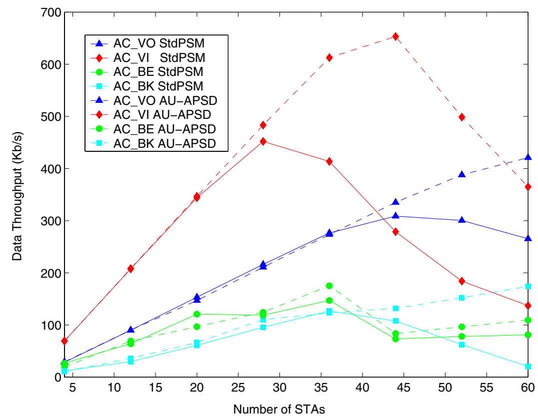

Fig. 4. Impact of the number of stations on the MAC downlink throughput

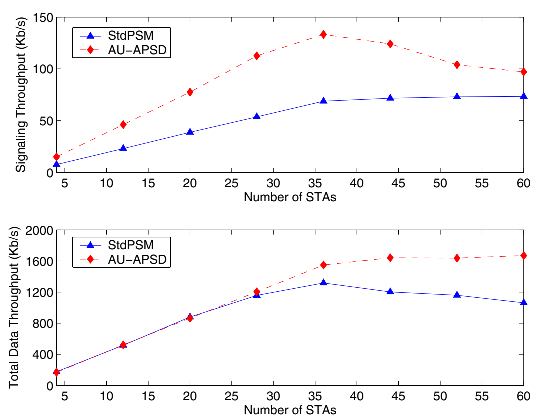

Fig. 5. Impact of the number of stations on the Signaling and Total Data throughput

downlink direction. When a station sends a trigger and there is no downlink activity, the AP will then generate QoS Nulls as a response. An option to alleviate this problem would be, instead of sending a QoS Null, to use the more data bit of the ACK, currently not used, to indicate to the station that no frames are buffered at the AP. Regarding AC_VI, it uses a highly asymmetric application with most of the traffic sent in the downlink direction. Therefore, the signaling introduced by U-APSD is similar to the one introduced by standard power save mode. In addition, AU-APSD can not benefit from the Max_SP_Length configuration until the network gets congested. A saturation effect is also observed in the AU-APSD signaling load. The reason for this is a reduction in the number of QoS Nulls sent in the uplink by the stations running AU-APSD when the network gets congested. As the AP queues get full and the downlink delay grows, the number of frames retrieved in the same SP increases, because of our Max_SP_Length configuration, and thus the signaling load gets reduced.

The lower part of Figure 5 shows that, even if the signaling load is lower in the standard power save mode case, the wireless channel capacity is increased when using AU-APSD. The reasons for this are the same as the ones we have explained for the downlink throughput results.

C. Power Saving

Figure 6 shows the percentage of time the stations spend in active mode. It can be clearly observed that all access categories save more power when they are configured with AU-APSD than with standard 802.11 power save mode. For instance, in the AC_VO case the power saving improvement ranges between 26% to 65% depending on the number of

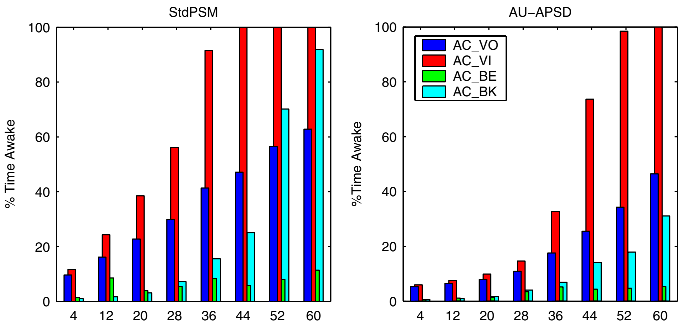

Fig. 6. Impact of the number of stations on the Power Saving Efficiency

stations. As already mentioned in [10], legacy 802.11 power save mode has a basic problem regarding the probability of collision when accessing the channel which comes from the fact that transmission of PS-Polls of stations is synchronized after the Beacon reception. This results in a higher collision probability and therefore less power saving. Any power save method where the decision of sending a trigger is not centralized but distributed, like our AU-APSD algorithm, reduces this problem and is likely to have a power save improvement with respect to standard 802.11 power save mode. The exact values observed in Figure 6 are determined by the amount of traffic each AC receives, e.g, the AC_VO stations are awake more time than the AC_BE because they receive more traffic.

V. Summary & Conclusions

The introduction of wireless LAN access capabilities in future mobile devices, e.g., cellular phones, poses QoS and power saving performance challenges that need to be addressed. The combination of the IEEE 802.11e EDCA and U-APSD mechanisms represent a potential solution to these challenges and therefore its feasibility has to be evaluated. Since the efficiency of the EDCA functionality to provide QoS differentiation has been thoroughly studied in the past, our focus has been the evaluation of the expected performance obtained when combining it with the U-APSD mechanism. The objective would be to assess whether the additional complexity would be justified by a significant QoS and power saving improvement as compared to simply using legacy 802.11 power save mode.

Our contributions are as follows. First, we provided a description of the U-APSD functionality and described its advantages with respect to legacy power save mode. Second, we designed an *adaptive* distributed algorithm (AU-APSD) for the stations to make use of the U-APSD mechanisms that aims to be generic enough to provide a QoS according to the different traffic requirements, without any knowledge from the higher layers, while improving the power saving efficiency of legacy power save mode. Third, we provided quantitative results of the performance differences to be expected when AU-APSD is used as compared to 802.11 power save mode with respect to QoS perceived by the users, power saving efficiency, signaling load and available channel capacity. Finally, we provided the reasoning behind the performance differences pointing out the elements to be taken into account when deciding which power saving mechanism is more approapriate.

The main conclusions that can be drawn from our results are i) AU-APSD significantly outperforms 802.11 power save mode in the QoS differentiation provided, e.g., the downlink delay degradation of voice users starts at 44 stations (11 voice stations) instead of at 28 (7 voice stations), ii) the distribution by the AU-APSD mechanism of the transmission of the power saving signaling frames and their transmission at a rate according to the downlink traffic characteristics results in a less congested channel and thus higher power saving, e.g., in the AC_VO case the power saving improvement ranges between 26% to 65% as compared to legacy power save mode, iii) the solution does not improve only the overall performance of the higher priority traffic but also of the lower one yielding an overall better channel usage, i.e., more users/applications can be accepted in the system and iv) in cases where frames are sent by a station in the uplink acting as triggers and no frames are buffered at the AP, the required QoS Null response to end the service period results in an unefficient use of the channel that chould be easily improved.

VI. Acknowledgments

The authors would like to thank their colleagues from the development team of the 3G/WLAN NEC mobile terminals group for inspiring the direction of this research work.

References

[1] IEEE 802.11 WG, “Wireless medium access control (mac) and physical layer (phy) specifications amendment 8: Medium access control (mac) quality of service enhancements,” IEEE 802.11e, November 2005.
[2] G.R.Hiertz O.Klein B.Walke S.Mangold, S.Choi, “Analysis of IEEE 802.11e for QoS Support in Wireless LANs,” IEEE Wireless Communications Magazine, December 2003.
[3] IEEE, “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” IEEE Standard 802.11, June 1999.
[4] “Wi-Fi Alliance,” http://www.wi-fi.com.
[5] A. Banchs and X. Pérez-Costa, “An Assured Rate Service Extension for 802.11 Wireless LAN,” in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, Orlando, FL, March 2002.
[6] R.Krashinsky and H.Balakrishnan, “Minimizing energy for wireless web access with bounded slowdown,” in *Proceedings of the eighth Annual International Conference on Mobile Computing and Networking (MOBICOM)*, september 2002.
[7] D.Qiao and K.G.Shin, “Smart Power Saving Mode for IEEE 802.11 Wireless LANs,” in *Proceedings of IEEE INFOCOM*, March 2005.
[8] Y.Chen, N-Smavatkul, and S.Emeott, “Power Management for VoIP over IEEE 802.11 WLAN,” in *Proceedings of IEEE WCNC*, March 2004.
[9] X.Pérez-Costa, D.Camps-Mur, and T.Sashihara, “Analysis of the Integration of IEEE 802.11e Capabilities in Battery Limited Mobile Devices,” IEEE Wireless Communications Magazine, December 2005.
[10] X.Pérez-Costa and D.Camps-Mur, “APSM: Bounding the Downlink Delay for 802.11 Power Save Mode,” in *Proceedings of IEEE International Conference on Communications (ICC)*, May 2005.
[11] X.Pérez-Costa, A.Vidal, and D.Camps-Mur, “SU-APSD: Static IEEE 802.11e Unscheduled Automatic Power Save Delivery,” in *Proceedings of European Wireless Conference (EW)*, April 2006.
[12] “OPNET Simulator,” http://www.opnet.com.
[13] F.H.P. Fitzek and M.Reisslein, “MPEG-4 and H.263 Video Traces for Network Performance Evaluation,” IEEE Network,Vol. 15,No. 6,pages 40-54, November/December 2001.