Traffic Scheduling in Wireless ATM Networks

Nikos Passas Lazaros Merakos Dimitris Skyrianoglou
Communication Networks Laboratory
Department of Informatics
University of Athens
15784 Athens, Greece
E-mail: {passas,merakos,dimiski}@di.uoa.gr
Tel: +301 7248154 (x334)
Fax: +301 7219561

Abstract
Wireless ATM is enjoying enormous research interest in the last few years, because of its ability to combine multimedia applications support, together with the freedom of mobility. One of the key design issues is the medium access control (MAC) protocol for the radio interface. This paper presents the traffic scheduling algorithm used in the MAC protocol of the Wireless ATM Network Demonstrator (WAND) system being developed within project Magic WAND. Magic WAND is investigating wireless ATM technology for customer premises networks in the framework of the Advanced Communications Technologies and Services (ACTS) programme, funded by the European Union. The proposed algorithm is delay oriented to meet the requirements of the various traffic classes defined by the ATM architecture. Simulation results are presented to assess the performance of the algorithm.

1. Introduction

Broadband and mobile communications are presently the two major drives in the telecommunication industry. Asynchronous Transfer Mode (ATM) is considered the most suitable transport technique for the future Broadband Integrated Services Digital Network (B-ISDN) [1]. On the other hand, wireless communications have been developed in a level were offered services can be extended beyond voice and data [2]. The combination of wireless communications, together with ATM can provide freedom of mobility with service advantages and QoS guarantees.

ACTS Project Magic WAND (Wireless ATM Network Demonstrator) is aiming to introduce ATM over the air all the way to the mobile terminal, and to cover a wide range of functionalities, from basic data transmission, to shared multimedia applications. The main components of the WAND system, as shown in Figure 1, are:

- *Mobile Terminals* (MTs), the end user equipment, which are basically ATM terminals with a radio adapter card for the air interface,
- *Access Points* (APs), the base stations of the cellular environment, which the MTs access to connect to the rest of the network,
- an *ATM Switch* (SW), to support interconnection with the rest of the ATM network, and
- a *Control Station* (CS), attached to the ATM switch, containing mobility specific software, to support mobility related operations, such as location update and handover, which are not supported by the ATM switch.

An important system design issue for WAND, and wireless ATM systems in general, is the design of an efficient medium access control (MAC) protocol for the radio interface. This protocol must be able to support all or a useful subset of ATM services with often conflicting requirements, and guarantee a QoS for every connection. Accordingly, advanced traffic scheduling is required to fulfil these requirements. In this paper, we present traffic scheduling in the radio interface, as is currently worked out in the WAND project. The objective of the algorithm is to guarantee quality of service (QoS) for different kinds of connection, attaining at the same time efficient use of the available radio bandwidth.

The paper is organized as follows. In Section 2, the MAC protocol designed for WAND is presented. Section 3 discusses traffic scheduling requirements for the radio interface of WAND and proposes a novel scheduling algorithm. In Section 4, simulation results on performance of the scheduling algorithm are presented. Finally, Section 5 contains our conclusions.

2. MAC in the Radio Interface of WAND

The MAC protocol is a critical component of WAND as its role is to provide wired-like services to ATM connections, in addition to controlling the access to the radio medium. This transparency is a challenge, because standard ATM has clearly not been designed to work in a wireless environment. Assumptions that have been taken into account for the design of ATM, like quasi-error free transmission medium, full duplex and point-to-point connections, are clearly not valid any longer. The radio medium is characterized by a high bit error rate (BER), the transmission mode is intrinsically broadcast or at least multicast, and the scarcity of available radio bandwidth calls for a time division duplex (i.e., half duplex) mode.

The MAC protocol for the radio interface of WAND is based both on reservation and contention techniques and it is called **M**obile **A**ccess **S**cheme based on **C**ontention and **R**eservation for **A**TM, or MASCARA. The multiple access technique used in MASCARA for uplink and downlink transmissions (respectively, from the MTs to the AP of their cell and from the AP to its MTs) is based on time division multiple access (TDMA), where time is divided in variable length time frames, which are further subdivided in time slots. Time slot duration is equal to the time needed to transmit the ATM cell payload (i.e., 48 bytes) plus the radio and MAC specific header. The multiplexing of uplink and downlink traffic is based on time division duplex (TDD). Slot allocation is performed in a dynamic and adaptive manner so that bit rates can readily match current user needs, by allocating more or fewer time slots per time frame. This is critical in servicing ATM connections, especially those with variable bit rate, because bandwidth can be allocated dynamically, and the resulting statistical multiplexing gain yields high resource utilization.

The MAC protocol belongs to the MAC layer of each MT and AP, which rests between the ATM layer and the radio physical layer. Cells coming from the ATM layer are formed into MAC Protocol Data Units (MPDUs) and are delivered to the radio physical layer for transmission, while MPDUs coming from the physical layer are processed and ATM cells are extracted. Data link control is required, as the quality of the radio channel is significantly worse than the conventional wired media (bit error rate can reach values as bad as 10^{-3}). For this purpose, the MASCARA layer includes a Wireless Data Link Control (WDLC) sublayer, which is responsible for error control over the radio link. The selection of the WDLC technique depends on the exact constraints imposed on each ATM connection, such as delay or loss constraints. In any case, a WDLC overhead is required in each individual ATM cell transmitted.

As shown in Figure 2, the MASCARA time frame is divided into a downlink period for downlink data traffic, an uplink period for uplink data traffic and an uplink contention period used for MASCARA control uplink information. Each of the three periods has a variable length, depending on the traffic to be carried on the wireless channel. The AP schedules the transmission of its uplink and downlink traffic and allocates bandwidth dynamically, based on traffic characteristics and QoS requirements, as well as the current bandwidth needs of all connections. The current needs of an uplink connection from a specific MT are sent to the AP through MT "reservation requests", which are either piggybacked in the data MPDUs the MT sends in the uplink period, or contained in special "control MPDUs" sent for that purpose in the contention period. At the end of a frame, the AP constructs the next frame, according to the MASCARA scheduling algorithm presented below, taking into account the reservation requests sent by the MTs, the arriving cells for each downlink connection, and the traffic characteristics and QoS requirements of all connections. By frame construction we mean the length of the frame and of each of its periods, and the position of the slots allocated to each downlink and uplink connection. This information is broadcast to the MTs in the frame header (FH period) at the beginning of each frame (Figure 2).

The physical layer overhead of the wireless medium is considerably larger than that of wired media. Hence, efficient data transmission can only be achieved if the length of transmitted data packets is not too small. On the other hand, the high bit error rate, characterizing the wireless media asks for not-too-large data packets to keep the packet error rate at tolerable values. To minimize the physical layer overhead, the MASCARA protocol uses the concept of a *"cell train"*, which is a sequence of ATM cells sent as the payload of a MPDU. More precisely, each MPDU consists of a MPDU header, followed by a MPDU payload containing ATM cells generated by the same MT or AP. The time required by the physical layer to initiate a MPDU transmission (referred to as physical overhead) plus the time needed to send the MPDU header is equal to one time slot. This way it is possible to follow the slot-based timing structure, whatever the number of transmitted cells contained in a MPDU. Figure 2 sums up the TDMA frame structure.

A more detailed description of the operation of MASCARA protocol can be found in [3].

3. The Scheduling Algorithm of MASCARA

In the environment under study, an arbitrary order of slot allocation from the AP, in accordance with some properties of the MASCARA protocol, such as uplink/downlink period separation and cell train construction, can alter the traffic pattern of a connection. This may result in violation of the contractual values of QoS and traffic characteristics, such as peak cell rate (PCR), cell delay tolerance (CDT), and cell delay variation tolerance (CDVT), and cause discarding of ATM cells deeper in the network, or late arrival at the receiver. The maintenance of contractual values for PCR and CDVT for uplink connections can be controlled with the use of a shaper at the fixed network port in each AP, while for downlink connections maintaining PCR and CDVT in the radio part is less important since this is the last hop of the connection. CDT values for both uplink and downlink can only be controlled by a traffic scheduler, located at the AP, that takes into account the delay constraints of individual connections in the allocation of bandwidth. The scheduling algorithm presented here is named **P**rioritized **R**egulated **A**llocation **D**elay **O**riented **S**cheduling (**PRADOS**), and is split into two main components: i) traffic regulation based on traffic characteristics, and ii) maintaining of the delay constraints of the connections transmitting in the radio interface.

By the beginning of each frame, PRADOS determines the MT that each slot of the next frame will be allocated to. The proposed algorithm combines priorities with a leaky bucket traffic regulator [4]. A priority is introduced for each connection, based on its service class:

Priority number	Service class
5	CBR (Constant Bit Rate)
4	rt-VBR (real time-Variable Bit Rate)
3	nrt-VBR (non real time-Variable Bit Rate)
2	ABR (Available Bit Rate)
1	UBR (Unspecified Bit Rate)

The service classes are defined in [5]. The greater the priority number, the higher the priority of a connection. Additionally, a token pool, located at the AP, is introduced for each connection. Tokens are generated at a fixed rate equal to the mean cell rate, and the size of the pool is equal to the "burst size" of the connection [5]. The burst size depends on the characteristics of each connection, and is the maximum number of cells that can be transmitted with rate greater than the declared mean. For every slot allocated to a connection, a token is removed from the corresponding pool. In this way, at any instance of time, the state of each token pool gives an indication of the declared bandwidth that the corresponding connection has consumed.

The priority leaky-bucket algorithm works as follows. Starting from priority 5 (CBR), and down to priority 2 (ABR), the scheduler satisfies requests of (uplink and downlink) connections belonging to each service class, as long as tokens are available. UBR connections have no guaranteed bandwidth, thus no token pools are maintained for them. At every priority class, it is very probable to have more than one connections requesting slots. In that case, the scheduler gradually allocates one slot at a time to the connection (or connections) which possesses the most tokens (i.e., highest token variable), decreasing the token variable by one. The rationale is that the connection with the most tokens has consumed less bandwidth than declared, and thus has higher priority for getting slots allocated. When the satisfaction of "conforming" requests is completed, and if there are still available slots, the scheduler tries to satisfy "exceeding" requests. To avoid possible later congestion, these excess uplink cells will have the cell loss priority (CLP) bit set to zero.

At this state, the token variables of all connections requesting slots are less than or equal to zero. The scheduler follows the same procedure as before, starting from priority 5 (CBR), down to priority 1 (UBR). If more that one connections belonging to the same priority class request slots, one slot at a time is allocated to the connection with the highest token variable. Since this is excess traffic, decreasing will result in negative values for the token variables. The procedure stops when all requests are satisfied or all available slots are allocated.

What is not specified with the above algorithm is the exact order of allocation of slots per MT. To do that, the scheduler should consider the traffic and QoS characteristics of the connections. As already mentioned, an important QoS parameter to be maintained in the air interface is CDT. Scheduling in the radio interface should enforce the wireless hop CDT of each connection, and allocate slots so that the fraction of ATM cells whose delay exceeds this CDT is minimized. The wireless hop CDT can be evaluated by decomposing end-to-end CDT into CDT for each hop of the ATM connection path.

It is clear that a delay-oriented scheduler requires knowledge of the arrival time of ATM cells in the output queues of the AP and MTs. Due to the location of the scheduler, the arrival time of downlink ATM cells can be directly logged and used in the scheduling algorithm, while the arrival time of uplink ATM cells should be estimated. For piggybacked requests, let us denote by M_i^n the i-th transmitted MPDU of connection C_n. If it is assumed that requests in M_i^n correspond to cells created in the interval $[M_{i-1}^n, M_i^n]$, then a worst case estimation for the creation time of these cells is M_{i-1}^n. The estimated deadline is W_n time units after M_{i-1}^n, where W_n is the wireless hop CDT for connection C_n. The deadline of an ATM cell is not an absolute threshold but rather an indication of how quickly an ATM cell should be scheduled for transmission. The scheduler should try to schedule ATM cells before their deadlines, but if that is not possible, some specified maximum extra delay may be allowed.

PRADOS is based on the intuitive idea that, in order to maximize the fraction of ATM cells that are transmitted before their deadlines, each ATM cell is initially scheduled for transmission as close to its deadline as possible. This idea was first proposed for fixed ATM networks in [6]. To attain high utilization of the radio channel, the algorithm is "work-conserving", meaning that *"the channel never stays idle as long as there are ATM cells requesting transmission"* [7]. Consequently, the final transmission time of an ATM cell will be the earliest possible given the ATM cell's initial ordering. This way the deadline of an ATM cell in effect determines the transmission order of that ATM cell with respect to the others.

The construction of cell trains and the separation of uplink and downlink periods inside each frame play an important role in the operation of PRADOS. In that sense, cell trains are constructed gradually, according to the leaky bucket algorithm, and ordered according to their deadlines. The deadline of a cell train is considered equal to the deadline of its first ATM cell. An ATM cell is attached at the end of the corresponding cell train if this does not cause deadline violation of the existing cell trains.

Below we describe how PRADOS takes into account the deadlines in the radio interface. For simplicity, all times are measured in slot time units. We denote by $c_n(i)$ the i-th ATM cell of connection C_n. When an uplink or downlink cell $c_n(i)$ arrives at the MASCARA layer of a MT or AP it can be represented by the parameters given below. For the purposes of the algorithm, we assume that, at the beginning of each frame, the slots following or preceding the FH period are numbered, starting from 1 and -1 respectively, and the parameters are adjusted accordingly.

- $a(c_n(i))$ = the arrival time of $c_n(i)$. For downlink ATM cells, this is the actual arrival time, while for uplink ATM cells it is estimated.
- $m(c_n(i))$ = the maximum waiting time that $c_n(i)$ can wait before being transmitted in the radio interface. Here we assume that $m(c_n(i))=W_n$, the same for all ATM cells of connection C_n, although the algorithm can also be used for different $m(c_n(i))$'s.
- $d(c_n(i)) = a(c_n(i))+m(c_n(i))$, the latest time by which $c_n(i)$ can be transmitted without missing its deadline.

Assume that, during a frame, a number of requests have been issued to the scheduler. At the end of the operation of the algorithm, the exact position of the slots allocated to each MT in the next frame should be specified. We define the following notation:

- $D_n = \min\{d(c_n(i)) : c_n(i) \text{ is requesting slot in the current frame}\}$, i.e., D_n is the earliest deadline of all ATM cells of C_n requesting slots in the current frame,
- $F_n = \begin{cases} 0, & \text{if no slot is allocated to connection Cn in this frame} \\ \text{the first slot allocated to connection Cn}, & \text{otherwise} \end{cases}$

- $L_n = \begin{cases} 0, \text{ if no slot is allocated to connection Cn in this frame} \\ \text{the last slot allocated to connection Cn, otherwise} \end{cases}$
- $O(x)=\begin{cases} 0, \text{ if slot x is empty} \\ n: \ Fn \le x \le Ln, \text{ otherwise} \end{cases}$ (i.e., O(x) is the identifier of the connection that slot x belongs to)
- $D^-_n = \begin{cases} D_n, \text{ if } O(D_n) = 0 \\ FO(Dn) -1, \text{ if } O(D_n) \neq 0 \end{cases}$ (i.e., D^-_n is the slot preceding the first slot allocated to the connection owning D_n)
- $D^+_n = \begin{cases} D_n, \text{ if } O(D_n) = 0 \\ LO(Dn) + 1, \text{ if } O(Dn) \neq 0 \end{cases}$ (i.e., D^+_n is the slot following the last slot allocated to the connection owning D_n)
- $En = \sum_{i=1}^{Dn} (O(i) = 0)$ (i.e., E_n is the number of free slots in the interval $[1, D_n]$)
- Δ_n is the maximal allowed *extra* delay, for allocating slots to connection C_n, after the deadline D_n.
- N(x) is the first empty slot after slot x.
- ϕ is the length of the "MPDU overhead": for each MPDU transmitted over the air, a number of ϕ slots must be reserved for transmission of the physical and MPDU headers. In the WAND system, this overhead consumes only one slot (ϕ=1), but for keeping the description general, we will not give any constant value to this overhead. Nevertheless the value 1 is assumed in the following illustrative figures to keep them easy to understand.
- P is the length of the "period overhead": at the boundary between the DOWN and UP periods, the RF modem must switch between transmit and receive mode, an operation which is assumed to last for P slots. In the WAND system, this overhead consumes only one slot (P=1), but for keeping the description general, we will not give any constant value to this overhead. Nevertheless the value 1 is assumed in the following illustrative figures to keep them easy to understand.

PRADOS operates in frames, and can be divided into three steps.

Step A: The scheduler starts satisfying requests according to the leaky-bucket algorithm described above. When a request corresponding to cell $c_n(i)$ is selected for service, the scheduler tries to allocate as many slots as needed for the transmission of $c_n(i)$. We consider two cases:

Case 1: Request for cell $c_n(i)$ is the first serviced request for connection C_n in the current frame. In this case, if slots $[D^-_n-\phi, D^-_n]$ are all empty, the scheduler allocates them to the current request. Otherwise, we distinguish two subcases:
Case 1a: $E_n>\phi$, i.e., there are at least ϕ+1 empty slots in the interval $[1, D_n]$. From the definition of D^-_n and the operation of the scheduler, which results in no more than one cell train per connection in each frame, it is derived that all the E_n empty slots are in the interval $[1, D^-_n]$. In this case, the algorithm has to free ϕ+1 positions, as close to D_n as possible, without splitting any existing cell train, to place the new ATM cell. These positions are $D^-_n-\phi$ up to D^-_n (Figure 3). Since the allocations move to the left, none of them exceeds its deadline.
Case 1b: $E_n\le\phi$ (i.e., there are ϕ or less empty slots in the interval $[1,D_n]$). In this case, there is not enough available "free space" prior to the deadline to allocate slots for connection C_n. Therefore the allocation can only be done on the "right side" of the deadline, while ensuring that the introduced delay does not cause exceeding of the "extended deadline". The extended deadline of a connection C_i is defined as the sum of D_i plus Δ_i. The proposed approach is first to fill the E_n empty slots before the deadline D_n by shifting to the left the allocations before D^+_n and then start the allocation on the new position of $D^+_{n,}$ ensuring that shifting to the right existing allocations, if needed, does not violate their extended deadlines. Therefore the strategy consists of the following steps.

1. Verify that $D^+_n + \phi - E_n \le D_n + \Delta_{n,}$ to ensure that the extended deadline of C_n is not exceeded. If found false, no allocation is performed.
2. For all connections C_i, that have allocations on the right of D_n, and need to be shifted to the right to make space for the new allocation, verify that after the shifting $L_i \le D_i + \Delta_i$ for all connections. If this is not the case even for one connection, no allocation is performed. Otherwise:

3. Allocations up to D^{+}_{n}-1 are moved to the left to fill all the empty E_n slots, while allocations after D^{+}_{n}-1 are shifted to the right, to make ϕ+1 slots available for the new allocation.
4. The allocation is performed starting at D^{+}_{n} (note that the value of D^{+}_{n} after step 3 differs from the value before step 3, as it has been decreased by E_n).

An example is shown in Figure 4.

Case 2: Request for packet $c_n(i)$ is not the first serviced request for connection C_n in the current frame. We distinguish three sub-cases.

Case 2a: There is at least one empty slot in [1, L_n]. In this case, all allocations between the last empty slot before L_n and L_n shift one position to the left to leave L_n empty for the new allocation (Figure 5). Moving to the left, does not violate any deadlines.

Case 2b: There are no empty slots in [1, L_n] and the slot L_n+1 is empty. In this case, the allocation is performed in slot L_n+1, if $L_n+1 \leq D_n+\Delta_n$ (Figure 6), otherwise it is not performed.

Case 2c: There are no empty slots in [1, L_n] and the slot L_n+1 is not empty. In this case, the allocation can only be performed in slot L_n+1 if connection C_n and all the connections occupying the slots up to the first empty slot have not yet reached their extended deadline. The strategy consists of the following steps (see Figure 7):

1. If $L_n=D_n+\Delta_n$ then no allocation is performed.
2. Check that the following relation is true, for all connections C_i occupying slots in the interval [L_n+1, N(L_n+1)-1]:

$$L_i < D_i + \Delta_i$$

If found false, even for only one connection, then no allocation is performed. Otherwise:

3. Shift to the right by one position the allocations in the interval [L_n+1, N(L_n+1)-1] (this step frees the slot L_n+1).
4. The allocation is performed in slot L_n+1.

Step B: The purpose of this second step is to build the DOWN interval of the MAC frame. This operation consists of three sub-steps:

1. When all requests have been processed, the scheduler packs as close to the beginning of the frame as possible, all allocations between the beginning of the frame and the first slot allocated to an uplink connection (clearly all these allocations correspond to downlink connections) (Figure 8 step b).
2. As an output of the former sub-step, some slots may be left empty before the first uplink slot. If this number of empty slots is equal to E, then a comparison is made between E and P:
   - if E≥P, then the last P slots before the first uplink slot are allocated for the so-called "period overhead" (Figure 8 step c).
   - if E<P, then the last P-E slots of the last downlink cell train preceding the first uplink slots are re-allocated to the period overhead. If the resulting slot train is less than ϕ+1 slots long, then this whole slot train must be deallocated (to avoid keeping an incomplete MPDU).
3. In the space left empty between the last downlink allocation and the period overhead, the algorithm tries to pack as many downlink allocations as possible by moving them to the left (Figure 8 step d).

Step C: The purpose of this step is to build the UP interval of the MAC frame. The operation is analogous to step B, except for the following differences:

1. Packing of uplink is performed between the end of the DOWN interval, as produced from step B, and the next slot allocated to a downlink connection.
2. P in sub-step 2 of step B is replaced by L_C+P+L_{FH}, where L_C is the length of the contention period and L_{FH} is the length of the FH of the next frame.

An example is shown in Figure 9. In this figure, the contention period is limited to the minimum for drawing purposes. Concerning allocations that cannot be packed during steps B and C, due to insufficient space in the DOWN and UP intervals, the corresponding requests are serviced before new requests in the next frame, according to the same algorithm (Figure 9 step d).

Step D: At the end of the procedure, the CONTENTION period is packed right next to the UP period (Figure 10). Packing preserves the "work-conserving" property that was mentioned before, since no slots are left empty.

4. Scheduling Algorithm Performance Results

In this section, we give simulation results on the performance of PRADOS. We also consider and compare it with a simpler algorithm, in which the delay constraints of the different connections are not taken into account.

The simulation models were built using the OPNET tool [8]. The models use the frame structure defined in section 2, except for the following simplification: the frame header and the contention period have constant lengths. The incoming traffic consists of identical VBR connections. To model the traffic for these connections, we used independent discrete-time batch Markov arrival processes (D-BMAP). D-BMAP, proposed in [9], is the discrete-time analogue of the batch Markov arrival process, introduced by Lucantoni in [10]. The number of connections was equally divided between uplink and downlink. For odd numbers, the extra connection was considered uplink. The system parameters used in the simulations are shown in Table 1.

Channel Characteristics	Connections Characteristics	Overheads
channel capacity = 19.2 Mbps slot duration = $22\cdot10^{-6}$ (time needed for one ATM cell)	mean rate = 512 kbps deviation (σ) = 256 kbps wireless hop CDT = 4.4 msec max. extra delay = 0.44 msec	cell train overhead = 1 slot period overhead = 1 slot frame header length = 5 slots contention period = 6 slots

Table 1: Simulation parameters

Finally, to focus on the performance of the scheduling algorithm, we have assumed that allocation requests sent over the contention period are successful in their first transmission attempt, i.e., no collisions.

PRADOS was compared with a simpler algorithm called **P**rioritized **R**egulated **A**llocation **S**cheduling (PRAS), which uses the virtual leaky bucket mechanism described in section 3, but allocates slots in a first-in-first-out manner, without taking into account cell transmission deadlines. The performance measures obtained from the simulation model are the following:

- P_{loss}: This is the ratio of the expired ATM cells (i.e., cells that experience delays higher than the maximum allowed, $W+\Delta$) over the total number of cells generated, and corresponds to the loss probability.
- Utilization: This is the fraction of the used channel capacity (i.e., the fraction of the total number of slots that are used to carry data traffic).
- Throughput: This is the fraction of the channel capacity used for the transmission of “useful” (i.e., not expired) ATM cells. Clearly, $\text{Throughput}=\text{Utilization}\cdot(1-P_{loss})$
- Mean delay: The mean delay experienced by the non-expired ATM cells. The delay of a cell is defined as the time from generation until transmission completion.
- Mean frame length.
- Mean cell train length: The mean number of slots is a cell train.

Figure 11 plots P_{loss} versus the number of connections. Observe that the loss probability remains almost equal to zero for low traffic, i.e., for as many as 18 connections, in both algorithms. However, for heavy traffic the loss probability of PRADOS remains comparatively low, while the loss probability for PRAS follows a more abrupt curve. This can be explained by the waste of network resources due to the transmission of expired ATM cells when PRAS is used.

An important objective of the scheduling algorithm is fairness in handling uplink and downlink connections. As shown in Figure 12, the loss probability of uplink and downlink connections is almost identical, indicating fair treatment of uplink and downlink connections by PRADOS.

Figure 13 gives the mean cell delay (irrespectively of direction) versus the number of connections, for both PRADOS and PRAS. As it can be seen from the figure, PRADOS attains better mean delay performance compared to PRAS as the traffic load increases. The reason for this behaviour is twofold: i) since PRADOS transmits only non-expired cells (i.e., the cell delay is upper bounded by $W+\Delta$), the induced cell delay variance is smaller than that under PRAS, and 2) the transmission of expired ATM cells in PRAS wastes resources.

Figure 14 gives the mean cell delay in the downlink and the uplink direction for both PRADOS and PRAS. Note that, in both algorithms the mean downlink delays are smaller than the uplink ones, which is attributed to the MASCARA frame structure, which places the downlink period before the uplink period. From Figure 14, we also note that the difference in delays between downlink and uplink ATM cells is smaller under PRADOS compared to PRAS, especially under heavy traffic load.

As it can be seen in Figure 15, resource utilization seems better in PRAS for heavy traffic; however, a portion of the transmitted ATM cells are actually useless, since they are expired (recall the high loss probability under heavy traffic load, in Figure 11, when PRAS is used). A more interesting measure is the throughput of the two algorithms, which is given in Figure 16. Note that PRADOS is almost the same for both algorithms, reaching a value of 0.60 for 18 connections.

5. Conclusion

In this paper, we have given an overview of MASCARA, the MAC protocol designed for ACTS project WAND, and described in detail the traffic scheduling algorithm used in this protocol. MASCARA is a TDMA-based protocol using both reservation and contention for accessing the medium. Most parameters of the protocol, such as time frame length and periods length were left variable and adjustable to traffic conditions, to attain better performance. The scheduling algorithm (PRADOS) focuses on satisfying delay constraints of the various connections. It is based on the intuitive idea that, in order to minimize the fraction of expired ATM cells, each ATM cell should be initially scheduled as close to its deadline as possible.

To evaluate the performance of the scheduling algorithm, we have compared it with a simpler algorithm (PRAS) that does not take into account delay constraints. The obtained simulation results show that the increased complexity of the proposed algorithm is worth while, since the loss and delay performance is significantly improved. Work in progress includes improvement of specific operations of the scheduler, such as deadline estimation and cell train arrangement, as well as dynamic calculation of the contention period length.

6. References

[1] O. Kyas, “ATM Networks”, International Thomson Publishing, 1995.
[2] J.D. Gibson, Editor-in-Chief, “The Mobile Communications Handbook”, CRC Press, 1996, ISBN 0-8493-8573-3.
[3] N. Passas et al., “MAC Protocol and Traffic Scheduling for Wireless ATM Networks”, submitted for publication, ACM Mobile Networks and Applications Journal, special issue on Wireless LANs.
[4] D.C. Cox, “Wireless Personal Communications: What Is It?”, IEEE Personal Communications, Vol. 2, No. 2, pp. 20-35, April 1995.
[5] The ATM Forum, “User-Network Interface (UNI) Specification”, Version 3.1, September 1989.
[6] T. Ling and N. Shroff, “Scheduling Real-Time Traffic in ATM Networks”, in Proc. IEEE INFOCOM 96, pp. 2b.4.1-2b.4.8, 1996.
[7] H. Zhang and S. Keshav, “Comparison of Rate-Based Service Disciplines”, in Proc. ACM SIGCOMM’91, pp. 113-121, 1991.
[8] OPNET Modeler, MIL 3, Inc., 3400 International Drive NW, Washington, DC 20008, 1993.
[9] C. Blondia and O. Casals, “Performance Analysis of Statistical Multiplexing of VBR Sources”, in Proc. INFOCOM ‘92, pp. 828-838, 1992.
[10] D.M. Lucantoni, “New Results on the Single Queue with a Batch Markovian Arrival”, Process. Stochastic Models, vol. 7, 1991.

7. Acknowledgements

This work has been performed in the framework of the project ACTS AC085 The Magic WAND, which is partly funded by the European Community and the Swiss BBW (Bundesamt für Bildung und Wissenschaft). The Authors would like to acknowledge the contributions of their colleagues from Nokia Mobile Phones, Tampere University of Technology, VTT, Ascom Tech AG, University of Lancaster, Lucent Technologies, Robert BOSCH GmbH, University of Ulm, Compagnie IBM France, Eurecom, ETH Zurich, INTRACOM Hellenic Telecommunications and University of Athens.

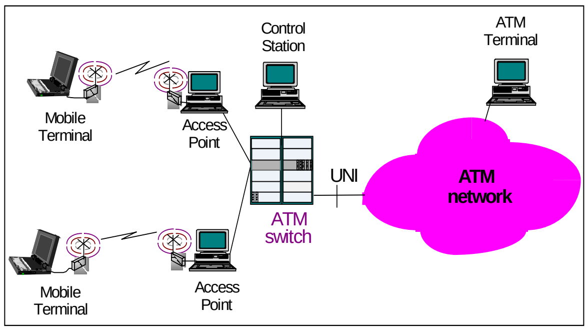

Figure 1: A WAND system

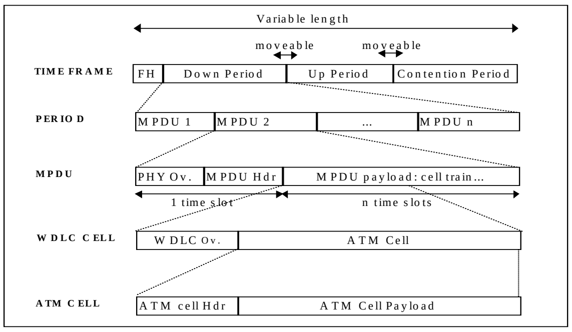

Figure 2: Time Frame Structure

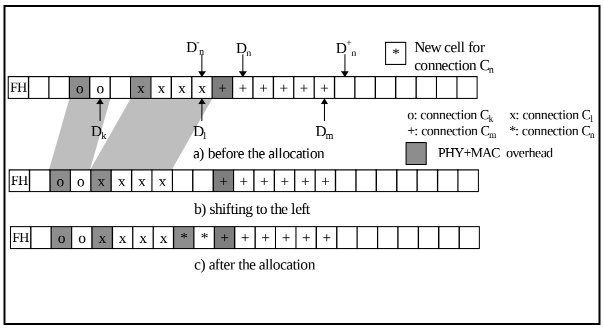

Figure 3: Allocation when there are at least ϕ+1 free slots

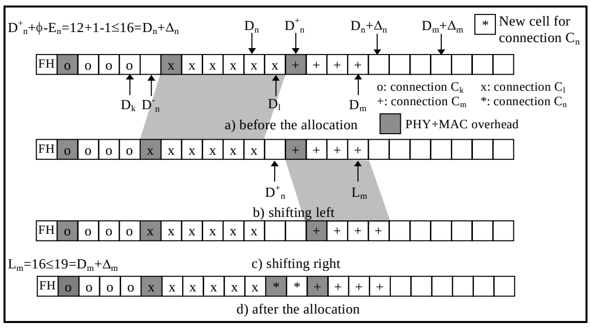

Figure 4: Allocation when there is no more than ϕ slots

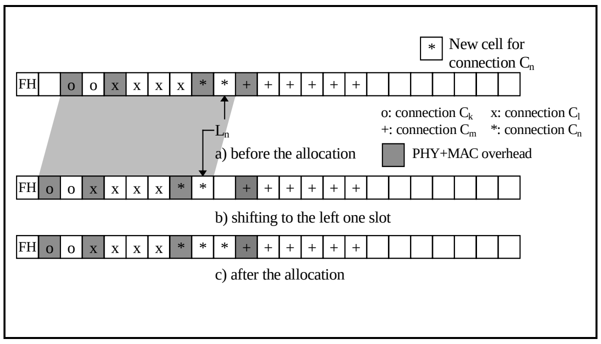

Figure 5: Allocation when there is at least one free slot before L_n

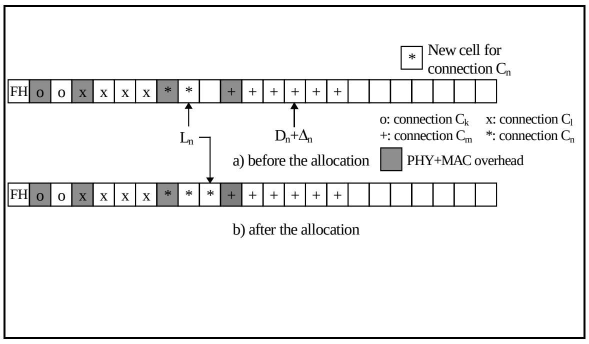

Figure 6: There is no free slot before L_n and L_n+1 is free

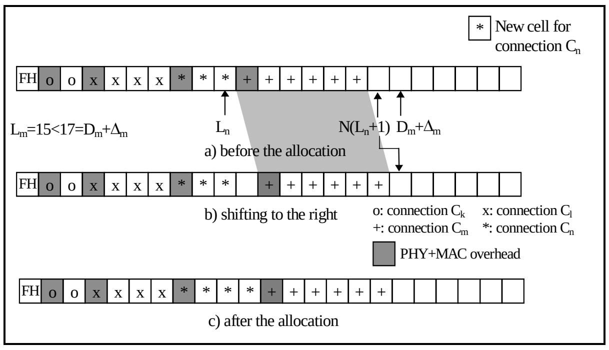

Figure 7: Allocation when there is no free slot before L_n and L_n+1 is not free

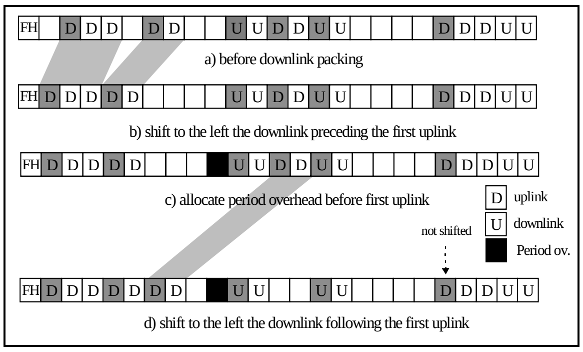
FH
a) before downlink packing
b) shift to the left the downlink preceding the first uplink
c) allocate period overhead before first uplink
D uplink
U downlink
Period ov.
not shifted
d) shift to the left the downlink following the first uplink

Figure 8: Packing of downlink allocations

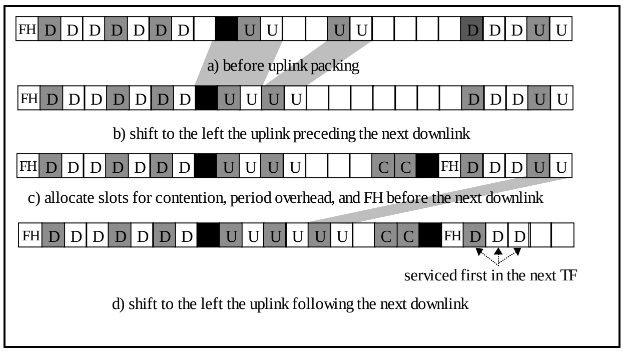
FH D D D D D D D U U U U D D U U
a) before uplink packing
FH D D D D D D D U U U U D D U U
b) shift to the left the uplink preceding the next downlink
FH D D D D D D D U U U U C C FH D D U U
c) allocate slots for contention, period overhead, and FH before the next downlink
FH D D D D D D D U U U U U U C C FH D D D
serviced first in the next TF
d) shift to the left the uplink following the next downlink

Figure 9: Packing of uplink allocations

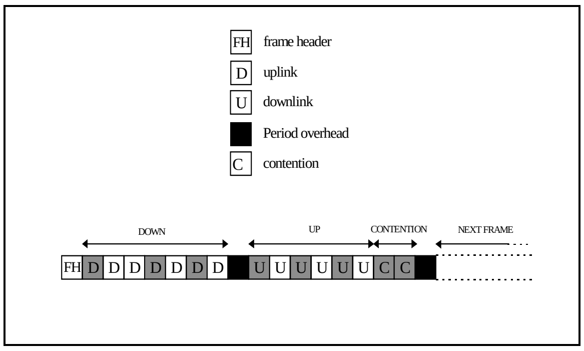

Figure 10: Final frame structure

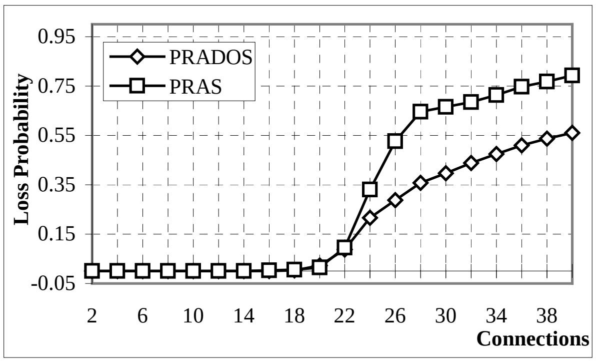

Figure 11: Loss probability

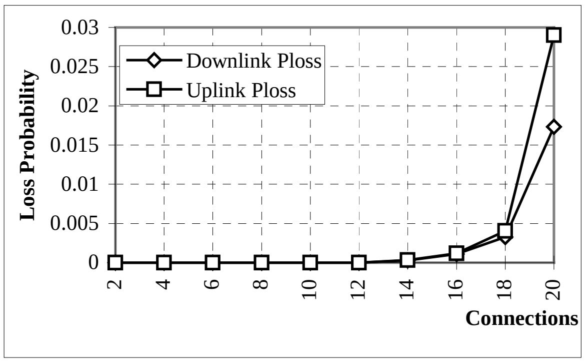

Figure 12: Uplink vs downlink loss probability for PRADOS

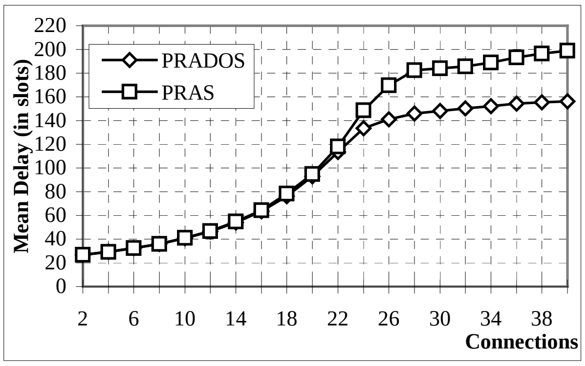

Figure 13: Mean delay

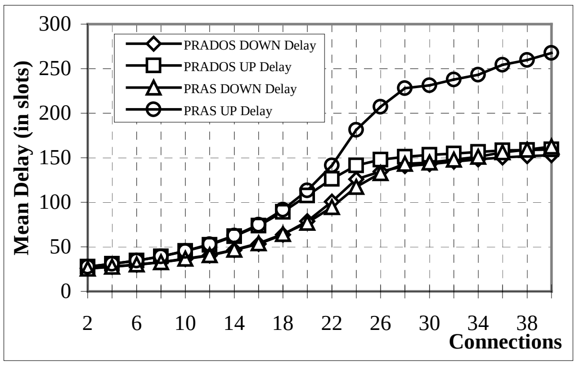

Figure 14: Downlink and uplink mean delay

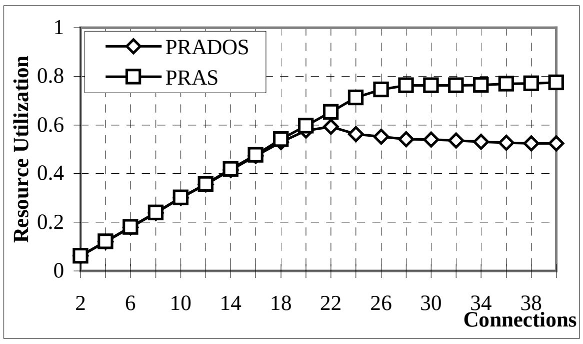

Figure 15: Resource utilization

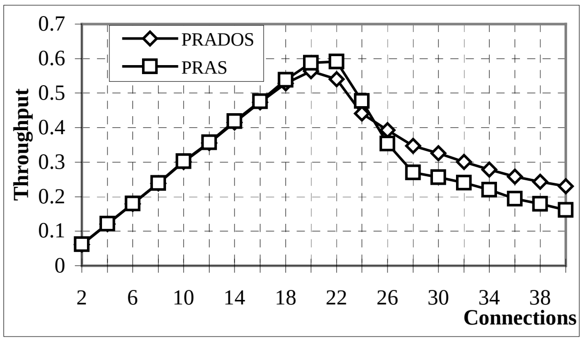

Figure 16: Throughput