

# Adaptive MAC Protocol for a Cable Modem

Dolors Sala\*, John O. Limb, Sunil Khaunte\*  
College of Computing  
Georgia Institute of Technology  
Atlanta, GA 30332-0280  
{dolors,limb,sunil}@cc.gatech.edu

**GIT-CC-97/14**

[http://www.cc.gatech.edu/tech\\_reports/](http://www.cc.gatech.edu/tech_reports/)

*May 1997*

## **Abstract**

Cable plants were initially designed for one-way broadcast communication (from the head-end to the neighborhood). They are now being upgraded to provide an upstream path (from the home to the head-end). New challenges arise in using the upstream channel since the available bandwidth is low and the noise levels are high. In this paper we present a MAC protocol especially designed to efficiently share the scarce upstream capacity. The protocol dynamically adjusts operating parameters to the current workload on the system. The control mechanism does not require any framing structure and is built around a “sea of mini-slots”. The performance under both static and highly dynamic loads is close to optimum. The station implementation is particularly simple and the downstream control structure is also simple. Results are given here for fixed length data units (ATM cells) but the algorithm extends very simply to variable length MAC frames.

---

\*School of Electrical and Computer Engineering

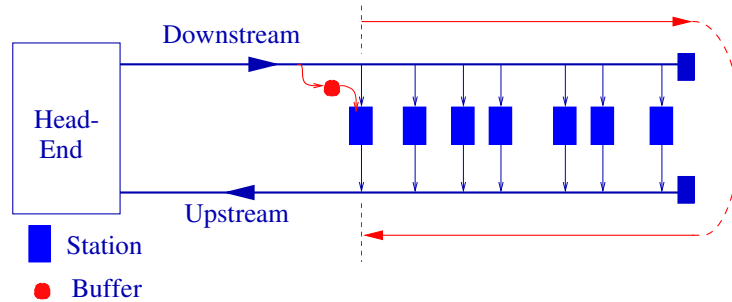


Figure 1: Logical Topology of an HFC System.

## 1 Introduction

An efficient way to provide a high-speed data channel to the home is to use a two-way cable television plant as a shared access medium. Communication takes place from one home to another by sending the data first to the head-end of the cable plant on one frequency and then translating the signal to another part of the spectrum for broadcast downstream to all stations (homes).

The topological abstraction of a cable TV plant is a folded unidirectional-bus (Fig. 1). The upstream bus (or channel) carries the information from the stations to the head-end while the downstream channel supports the communication from the head-end to the stations. Stations can be located at any distance from the head-end.

The upstream signal path, for historical reasons, is limited in bandwidth. It is also very noisy as a result of unwanted signals creeping into the cable and summing together as they flow towards the head-end of the cable plant (ingress). Furthermore, there is a significant overhead associated with each transmission from a given station to the head-end due to the need for guard bands and physical and MAC layer headers, mitigating against very short cells or packets. This is not the case in the downstream channel where there is only one transmitter and all the stations can remain synchronized to the head-end transmitter. The effective upstream capacity is rather limited compared to the downstream capacity. Nevertheless, it is possible to support aggregate data rates in excess of 10 Mbits/s in the upstream channel. Since this capacity is shared between many stations it is important that it be used efficiently. It is the Medium Access Control (MAC) protocol that ensures the fair and efficient use of the channel.

A number of MAC protocols have been proposed for cable data communication [1] and they are mostly a variation of the reservation ALOHA protocol [2, 3]. Note that if short cells are being transmitted (e.g. 53 byte ATM cells) then the size of a reservation slot can be a significant fraction of a data cell, due to the overhead associated with transmitting a message. It appears that one may be able to transmit between four and six reservations in the time it would take to transmit one ATM data cell.

The basic operation of the reservation ALOHA channel as applied to a cable modem is as follows: A station on receiving a message to transmit divides it into basic units (cells or 802 LLC frames) which occupy a single MAC-level slot in the upstream channel. The station then sends a request to the head-end by using a mini-slot. The mini-slot is written in by any station having a request to make at that time, and hence a collision may occur between stations trying to write in the same contention mini-slot. Assuming no collision occurs the head-end receives the request and (usually) immediately acknowledges the request. It then schedules a time for the station to transmit in a data slot. No collisions will occur in a data slot as a station can only write in a data slot when it receives a grant to do so, and the head-end will not schedule more than one station to write in a given slot. If a collision occurs in a contention slot then the station receives no acknowledgment from the head-end and a contention resolution algorithm (CRA) is used to resolve the collision. The CRA could be p-persistence (as in Aloha), or a tree-based algorithm [4]. Most CRAs require a knowledge of the number of stations competing for a slot to work most effectively. An algorithm to estimate this number is required.

One technique used to reduce the load on the reservation channel (those mini-slots allocated for making reservations) is to allocate a small field at the end of a data slot for attaching a request. This is referred to as “piggybacking” and under certain conditions this will increase the maximum load that can be carried by the channel.

A typical upstream frame structure is shown in Fig. 2. A group of data slots and contention mini-slots (CMSs) constitute a frame. The length of a frame may vary with traffic load [5] and/or the number of CMSs within a frame may be varied. As far as we are aware all adaptive protocols so far explored use some type of framing and clustering of slots. We refer to them collectively as frame-based algorithms. Contention mini-slots (CMSs) may be periodically inserted after each data slot or they may be lumped together at the end of a group of data slots.

The curve of delay versus applied load characterizes an important aspect of the static performance

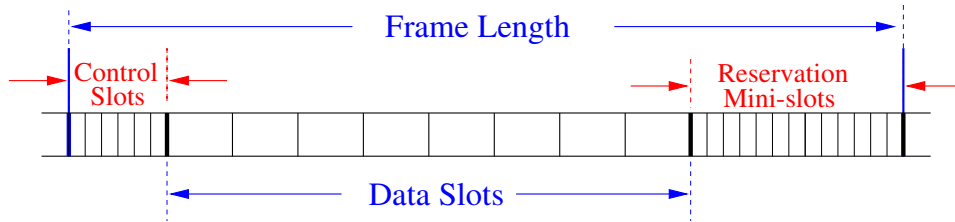


Figure 2: Typical Upstream Frame Structure

of a protocol. Of equal importance is the dynamic performance, since, in practice, a data load will vary widely from instant to instant. Since we have algorithms that are adaptive, if they do not adapt quickly, static tests may indicate good performance while the performance with realistic loads could be poor. For this reason we pay particular attention to the dynamic performance in this paper. There has been very little work on characterizing the dynamic performance of MAC protocols (See [6] for one dynamic test) and so we propose an impulse response test. A last measure that we focus on, more conceptually than quantitatively, is the idea of simplicity of implementation. Since we plan to implement the described protocols we will give preference to simple algorithms if there is little performance penalty involved.

In this paper we develop MAC protocols that are close to the best achievable bounds for the upstream channel of a cable modem. We have chosen a *frameless* structure in which CMSs are inserted continuously as they are required using what we refer to as the contention-slot allocator (CSA). Contention resolution is by means of a p-persistence algorithm.

## 2 Previous Work

The first MAC protocol proposed for HFC systems was (X)DQRAP [7, 8]. XDQRAP is a *distributed* scheme with the head-end reflecting the information arriving from the upstream channel to the downstream channel. Stations independently control access to the medium by monitoring the downstream channel. The CRA used is a type of tree algorithm. Since a number of CMSs may be in flight at one time a number of “instances” of the algorithm may be operating at the same time. Each “instance” is referred to as a contention resolution engine and interleaving is said to occur. The maximum number of engines that may be required, or equivalently the degree of interleaving, is equal to the number of CMSs that occur during the duration of a round-trip

(RTD) from the station to the head-end and back.

CPR [3] was designed as a centralized scheme with a very *simple* station protocol. Low cost is one of the main goals in this design. Therefore, as much functionality as possible is moved to the head-end. The CPR station is largely transparent to the control algorithms deployed in the head-end. It does not require knowledge of any framing that may be used by the head-end (e.g. frames used in support of CBR traffic [9]).

All subsequent proposals adopt the centralized approach but impose different frame structures. UniLINK [10] divides the frame into three regions: contention, reserved and periodic reserved (to support stream traffic). The boundaries that delimit the regions are dynamically adjusted depending on the load. Transmission in the contention region is based on CSMA with downstream collision detection. ADAPt [11] divides the frame into two regions, an STM region to support Synchronous data and an ATM region (further divided into subregions) for asynchronous data. MLAP [12] also defines a fixed frame size (called a *block*). The transmission slot unit is of variable length (in multiples of ATM cell units) adjusted to the size of the data frame to be transmitted (applying concatenation). The slot structure of the next frame is broadcast in the previous frame, specifying for every slot the type, and if it is reserved, the length, owner and offset location within the frame. It uses a tree-based CRA with free access for new comers (*msSTART* [13]). The Zenith protocol [14, 15] defines three regions in a fixed length frame. Two of these regions are used for contention: one that can be used for any contending station; and another region reserved for those stations that have already collided at least once. The third region transmits data in a collision free mode. The contention bandwidth is dynamically assigned depending on the head-end queue and on the probability of collision.

## 3 Basic Protocol

### 3.1 System Synchronization

Stations synchronize with the head-end during the start-up process. There is a ranging mechanism that determines the propagation time (or distance) between station and head-end. This time is used to determine the slot boundaries as related to the upstream channel in a common timing reference. The slot structure is dynamically controlled by the head-end (see Section 6)

and broadcast to stations through the downstream channel. A single bit per mini-slot is used to distinguish the two types of mini-slots (CMSs and data mini-slots). Note that there is a propagation delay between station and head-end. Therefore the head-end has to specify the type of the mini-slots in advance. The advance notice required is equal to the maximum RTD. This guarantees that the farthest station always receives the bit prior to the time the cell is to be used by any station (according to the global reference timing). Due to the differences in distance to the head-end, closer stations receive the broadcast earlier. Thus for them the bit corresponding to the next arriving mini-slot has already traveled to the EOL and visited all the downstream stations in the upstream channel. In order to derive correct timing, each station stores the bit values corresponding to the distance (in mini-slots) from the station to the end of the line (EOL) and back as shown for the closest station to the head-end in Fig. 1. In contrast to the padding approach typically used<sup>1</sup> [7], this extra storage does not introduce any additional access delay in the system (the station just points to the correct memory location in the buffer to recognized the type of the next mini-slot required).

### 3.2 Sea of Mini-Slots

The performance of a protocol depends more on the framing structure employed than the details of the CRA algorithm [5]. We have assumed that there is no frame structure used in the scheduling of data traffic<sup>2</sup>. At the lowest level there is just a sequence of mini-slots. A request to the head-end fits in exactly one mini-slot and a data-link frame or packet<sup>3</sup> fills an integral number of mini-slots<sup>4</sup>. After the head-end controller has scheduled a number of mini-slots to carry a packet any number of CMSs may be scheduled.

---

<sup>1</sup>Padding is another approach to overcome the different station locations. It virtually moves the physical location of the station to the EOL by adding some physical delay. This delay increases, by a proportional amount, the minimum access delay of the system.

<sup>2</sup>There could well be a frame structure used for the transmission of periodic traffic as described in [9]. But as shown there, stations need not be aware of this structure and the CSA need only be nominally aware.

<sup>3</sup>In order to clearly distinguish the frame structure concept with a MAC frame unit we will refer to a MAC frame as a packet.

<sup>4</sup>Stuff bits will usually be required to round the packet to an integer number of mini-slots.

### 3.3 How many CMSs?

When the load on the network is low, very few contention mini-slots (CMSs) are required. On the other hand, since the load *is* low, there will be unused mini-slots that could be used as CMSs. As the load increases, depending on the length of the packets, more mini-slots will need to be allocated as CMSs. As the load increases further, reservations may be made via the piggyback channel and hence the number of CMSs required will decrease. The solution to determining how many CMSs to allocate is rather simple: allocate all slots that are not being used for data as CMSs [16]. At low load many more CMSs will be allocated than are required. The surplus of CMSs reduces the probability of collision to a very low level which in turn reduces the access delay for the data<sup>5</sup>. This is a self regulating mechanism since if the number of CMSs is too low, the requests will not get to the head-end and more CMSs will be automatically allocated. If the number of CMSs is too high more successful requests will reach the head-end and the number of empty slots that can be allocated as CMSs will decrease.

If we consider a network of zero length and assume that traffic is queued at one or more stations each successful request will be followed by the transmission of a packet in one or more mini-slots, followed by a number of CMSs until a successful request is received. If we use the CMSs optimally then we cannot hope to decrease the delay of traffic through the network. As we shall see in our discussion of the Contention Slot Allocator (CSA), it is more difficult to use optimally the contention slots when the network has non-zero delay.

### 3.4 Optimal use of the Contention Mini-Slots

Several CRAs have been proposed in the past for multiple access systems [4, 17]. We have chosen to work with the p-persistence based algorithms mainly due to their simplicity and ease of implementation.

The simple p-persistence algorithm, using a fixed retransmission probability  $p$ , is shown to be unstable [17]. However a stable algorithm can be obtained if the retransmission probability at every station, is dynamically adjusted such that  $p = 1/N_a$  where  $N_a$  is the actual number of stations that are competing for an upstream CMS.

---

<sup>5</sup>Every collision adds one round-trip period to the access delay

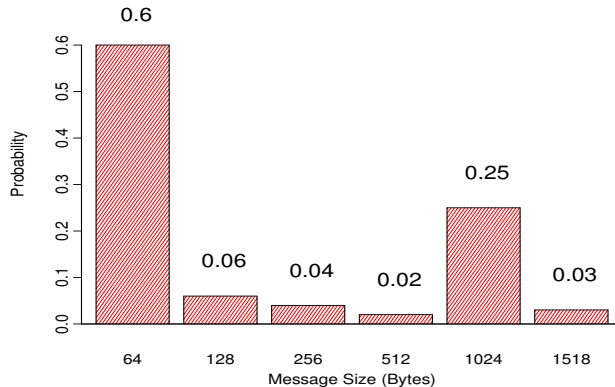


Figure 3: Message size distribution of Type 1 traffic. The average message size is 368.1 bytes (7.67 cell units)

Note that the operation of the CRA is rather independent of the CSA employed. If the CSA increases the number of CMSs in response to an increase in the number of active stations, each of the active stations execute p-persistence on each CMS exactly as it would have if there were fewer slots. The only difference in this case is that due to the increase in the number of opportunities (CMSs), the backlog of requesting stations is cleared more quickly.

## 4 Simulation Procedure and Bounds on Performance

### 4.1 Simulation Procedure

Our simulation process adheres closely to the procedure outlined in [18]. We use two of the sources defined in the document, i) a bursty source with a distribution of message sizes as shown in Fig. 3, (the average size of a message is 368.1 bytes or 7.67 48 byte cells) and ii) a Poisson source of single cell (48 byte payload) messages. In addition we use an extreme form of a transient load. We assume that each station instantaneously generates a one time load of two messages to be transmitted in addition to the existing load. This is purposely designed to stress the CRA both in terms of the stability of the algorithm and the transient response. The parameters of the network and the simulation are also taken largely from [18] and are given in Table 1.

The structure of the upstream channel is rather simple. The payload of an ATM cell fits in three mini-slots and the ATM header of 5 bytes plus the PHY overhead and guardband requires a further mini-slot. Thus, it requires four mini-slots to accommodate an ATM cell. A variable

Table 1: Network Configuration Parameters

Simulation Parameter	Value
Number of Active Stations	50
Distance from nearest/farthest Station to Head-end	25/40 Km
Spacing between closest and farthest Stations	Randomly Distributed
Upstream Data Transmission Rate	9 Mb/s
Propagation Delay (coax and fiber)	5 $\mu$ s/km
Length of Simulation Run	7.5sec
Length of Run Prior to Gathering Statistics	5% of simulated time
Guardband , pre-amble and PHY/MAC headers.	16 bytes
Size of mini-slot	16 bytes
Head-End processing delay	<u>0 ms</u> , 1.5 ms

length MAC frame, such as an 802 LLC frame would consist of one mini-slot of header plus an integral number of mini-slots to handle the MAC payload. We only simulate and consider ATM traffic in this paper. With the *sea of mini-cells* format, it is easy to extend the mechanism to variable length messages that can occupy a number of consecutive mini-slots, with the consequent saving in headers (and the additional potential blocking of traffic by a longer message).

A typical sequence of mini-slots transmitted on the upstream channel might appear as:

HDDDHDDD . 1HDDD . HDDD12HDDDHDDD . . 1HDD

where each symbol denotes a mini-slot, H denotes the 16 byte guardband and overheads, D denotes payload, “.” denotes an empty mini-slot and 1, 2, 3 etc. denotes the number of stations that compete for a CMS. Thus 1 denotes a successful request and “n” denotes that n stations tried to write in the CMS<sup>6</sup>. Hence the useful data are just the “D” mini-slots and the rest are overhead mini-slots. The goal is to achieve a slot structure with the maximum number of “D” mini-slots which, as we shall see, does not necessarily imply a small number of overhead mini-slots.

The simulation has been written in C and run both in native C (discrete time simulation) and in OPNET (event simulation). We tested the basic results presented here under the CSF (Common

---

<sup>6</sup>Neither the stations or the head-end have any way of knowing, or taking advantage of the number of stations colliding in a CMS. As we analyze the results of simulations it is frequently useful to use the above representation in order to understand the transient behavior of the upstream channel.

Simulation Framework) environment. The CSF has been used by the IEEE802.14 standard subcommittee in the MAC evaluation process.

## 4.2 Bounds on Performance

A simple bound on the maximum upstream throughput would be when only data slots were being transmitted, so that 48 bytes out of 64 were transmitting payload for an efficiency of 75%. This would be approached in practice if very large files were being transmitted so that the capacity of the reservation channel were negligible. (Of course, if we had not made the assumption that all data would be transmitted in ATM cells then the corresponding bound for very long packets would be 100%.)

We define access delay as the time between when a cell (or packet) is passed to the MAC layer and when the last byte of the cell (or packet) is transmitted.<sup>7</sup> Fig. 4 shows the access delay for a single cell when there is no other competing traffic. It is given by :

$$T_{min} = 2 * T_{sync} + T_{rtd} + T_g + T_{data} \quad (1)$$

where  $T_{sync}$  (1/2 mini-slot delay) is the delay in synchronizing to the next mini-slot boundary, after the message arrival in the beginning and before transmitting data.  $T_{rtd}$  is the round-trip propagation delay of the network,  $T_g$  (1 mini-slot) is the transmission time for the ack/grant and  $T_{data}$  is the transmission time for the data packet (4 mini-slots). Hence the minimal access delay at low loads reduces to  $T_{rtd} + 6$  mini-slots.

In practice we need to add to this time, the processing time in the head-end, which can be included in  $T_{rtd}$ . It will consist of physical layer processing (including processing for error correction/detection coding and security), MAC-level header processing, and scheduling. It appears that the error correction/detection coding/decoding will be by far the greatest component, approx. 2 to 6 ms [19]. At low values of throughput we should expect to approach an access delay of  $T_{min}$ .

---

<sup>7</sup>This definition differs from that used in [18], but is more consistent with common usage.

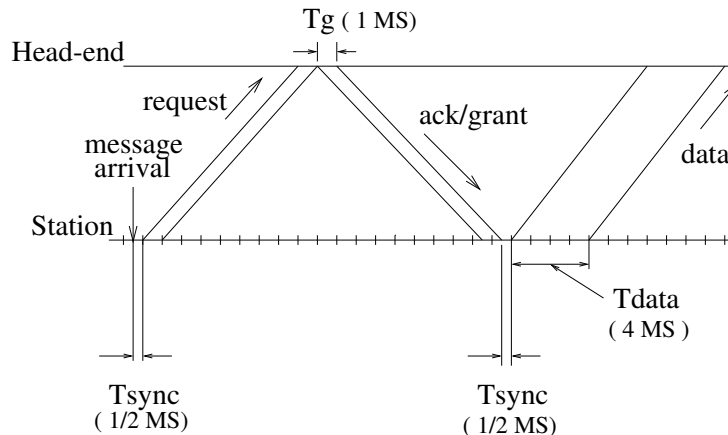


Figure 4: Components of the minimum access delay for a frameless MAC protocol.

## 5 Contention Resolution Algorithm

### 5.1 Pseudo-Bayesian Algorithm

Assuming p-persistence, the optimum usage of the CMSs will occur if the retransmission probability at the stations is adjusted to be  $1/N_a$  where  $N_a$  is the number of active stations that are contending for the CMSs [17]. Since neither the head-end nor the individual stations are capable of determining  $N_a$  exactly estimation algorithms are used. We use here the standard pseudo-Bayesian estimation algorithm proposed by Rivest [17]. In this scheme, unlike conventional slotted Aloha, a station having new arrivals is treated the same as a backlogged station immediately on arrival.

The Rivest algorithm operates by estimating the number of users (stations) waiting to transmit their requests at the beginning of each CMS. Each station then attempts to transmit with probability  $p = \max(1, 1/N_a)$ . The estimate of  $N_a$ ,  $\hat{N}_a$ , in the  $i+1$ -th slot is given in terms of the  $i$ -th slot by:

$$\hat{N}_a^{i+1} = \begin{cases} \max(\lambda, \hat{N}_a^i + \lambda - 1) & \text{Empty/Success in CMS}_i \\ \hat{N}_a^i + \lambda + \frac{1}{(e-2)} & \text{Collision in CMS}_i \end{cases} \quad (2)$$

where  $\lambda$  is the arrival rate of new data. While this rate is not known it could be estimated. We only need an accurate estimate of  $N_a$  when the system is near overload, because at lower data rates there will be a surplus of empty slots that can be used for requests. Thus we may assume

that  $\lambda$  has a maximum value of  $1/e$  requests per CMS. The resulting equations are then very simple to evaluate either at the head-end or in the station.

Implementing the algorithm at the head-end (centralized) or at the station (distributed) yields identical results. In the distributed scheme, every station updates its local estimate of  $N_a$  (and calculates  $p$ ) based on the CMS feedback received on the downstream channel. In the centralized scheme, the station directly receives the  $p$  calculated by the head-end on the downstream. This reduces processing at the station and avoids replication of the same algorithm at each node. In both cases the updated  $p$  used at the station is based on previous transmissions done over a period of time.

In a system where stations can be physically located in the same mini-slot a  $p$  value of 1 may introduce blocking problems. When two stations at the same location collide they get the feedback at the same time and both retransmit with probability 1 on the next mini-slot thus colliding again. This collision will only resolve if  $p$  is decremented. With a perfect estimator where the number of contending stations is always exactly known it is possible to always set  $p$  to  $1/N_a$ . With a realistic estimator that has a margin of error we need to set a maximum value of  $p$  smaller than 1 to avoid long blocking situations. In all experiments presented here we use a maximum  $p$  value of 0.3. This corresponds to an increment of 3.3 mini-slots in the minimum access delay in Equation 1. In most cases it is more advantageous to pay this entrance delay than to allow more collisions that each cost a further complete RTD. The most appropriate value to use will depend on various parameters of the system. There has been no attempt to optimize this value.

## 6 Contention Slot Allocator

### 6.1 Simple CSA

As mentioned in section 3.3, an efficient way to allocate contention slots is to have all unscheduled mini-slots allocated as contention mini-slots (we refer to this scheme as S). This simple strategy works extremely well in many instances as shown in Fig. 6. Curve *a* shows the result without piggybacking for Type 2 traffic (i.e. a Poisson source of single cells) with a *very short* RTD (1 mini-slot). If the p-persistent CRA were being used optimally (maximizing throughput) we would need  $e$  ( $= 2.718$ ) mini-slots on average (assuming a large number of contending stations)

for each data cell that was being transmitted and the efficiency would be:

$$\eta_{max} = \frac{3 \text{ payload MS}}{3 \text{ payload MS} + 1 \text{ overhead MS} + e \text{ CMS}} = 44.7\%$$

This bound is approached closely, as seen in Fig. 6, where the scale on the abscissa goes from 0 to 44.7%. While not shown here, the lower bound of the set of curves obtained using fixed  $p$   $p$ -persistence with a range of values of  $p$  is a close approximation to this curve. The pattern of activity on the channel is indicated by the typical sequence below:

```
HDDD.3.211HDDDHDDD11HDDDHDDD.11HDDDHDDD332.2311
HDDDHDDD2.1HDDDHDDD...2...3...211HDDDHDDD11HDD
DHDDD2.1.HDDD...1.HDDD11HDDDHDDD2.22..1.HDDD211
HDDDHDDD1.HDDD3.11HDDDHDDD.11HDDDHDDD2..221.HDD
D11HDDDHDDD13HDD21.HDDD13HDD11HDDDHDDD.1.HDDD
3211HDDDHDDD.11HDDDHDDD42211HDDDHDDD.11HDDDHDDD
```

As can be seen, one or two data slots are followed by about three to six CMSs. For longer RTD delays this simple scheme does not work quite as well. Curve  $b$  shows results for the same condition as before but with longer RTD value (29 mini-slots or 40 Km). The reason for this is seen clearly in the printout of cell usage shown below:

```
11.....1.11HDDDHDDDHDDDHDDDHDDDHDDDHDDDHDDDHDD
DDHDDDHDD212231.2.1311.1.11311.2111..412HDDDHDD
DDHDDDHDDDHDDDHDDDHDDDHDDDHDDDHDDDHDDDHDDDHDDDH
DDD322131.2..11...1..21112..211...11HDDDHDDDHDD
DHDDDHDDDHDDDHDDDHDDDHDDDHDDDHDDDHDDDHDD17..1221.1
2.1.1..1.1.2.1111.21HDDDHDDDHDDDHDDDHDDDHDDDHDD
DHDDDHDDDHDDDHDDDHDDDHDD24131..1.22.31.1...2.1
...221.2.HDDDHDDDHDDDHDDDHDDDHDD32HDD312.111
11.11..11121.22.2..1...4HDDDHDDDHDDDHDDDHDDDHDD
DHDDDHDDDHDDDHDDDHDDDHDD1.22411.1.21211111
211.21.1...1HDDD.HDDDHDDDHDDDHDDDHDDDHDDDHDDDHDD
DDHDDDHDDDHDDDHDDDHDD7222.1..211...11.4.2
..11...11.2...HDDDHDDDHDDDHDDDHDDDHDDDHDDDHDD
HDD2631323112.3.1....1.11....32..1HDDDHDDDHDD
DDHDDDHDDDHDDDHDD5HDD.32212121221111..1111.12
1.2...1...HDDDHDDDHDDDHDDDHDDDHDDDHDDDHDDDHDDDH
DDDHDDDHDDDHDDDHDD34133.13..2.41..3..11.1.11.1
2.23HDDDHDD13.HDD12HDDDHDDDHDDDHDDDHDDDHDDDHDD
```

It is apparent that there is a cyclic distribution of mini-slots between data and contention channels. We see a long burst of CMSs followed by a long burst of data cells. A stream of CMSs allows a burst of requests to arrive at the head-end which are immediately served generating a burst of data. A burst of data will lead to requests being held at the stations until the burst is complete. So that now the burst of data is followed by a burst of requests which will, in turn, increase the following burst of data. The length of the cycle is related to the size of the round-trip delay because the data associated with a successful request is transmitted one round-trip delay after the request is made, breaking up the CMS burst. And, indeed, the length of the CMS portion of the cycle is close to the RTD period of 29 mini-slots. The maximum throughput in this case is achieved since the CMSs are still efficiently used (the probability of success, collision and empty are close to the theoretical values). When the RTD is further increased the number of *empty* CMSs increases since CMSs appear when stations are waiting for feedback and they cannot transmit. Empty CMSs translate to low efficiency in the contention channel reducing the number of data cells to be transmitted and, in turn, increasing even more the bandwidth assigned to the (inefficient) contention channel. This behavior implies that it is very important to assign the capacity to the correct channel (contention or data) at any instant of time and suggests that it is better to allocate the required bandwidth first to the contention channel (overhead channel) and then to the data channel.

## 6.2 Forced Mini-slots CSA

We have investigated a number of schemes to reduce the cycles. We will only describe the most effective one here, referred to as Forced Mini-slots (FMS). A number of CMSs are periodically forced between data slots. The number is based on the number required at maximum load. For messages that are but a single ATM cell long we require “ $e$ ” CMSs on average. The fact that at lower loads we require fewer is irrelevant. Since the load on the system is lower, and we convert all unused mini-slots to CMSs, there will be more CMSs available than required.

The number of FMSs ( $N_{FMS}$ ) required is given by the maximum throughput of the slotted Aloha system. Under a finite number of stations the throughput is greater than  $1/e$  and equal to the probability of success:

$$P_s = N_a * p * (1 - p)^{(N_a - 1)}$$

Hence at lower loads (lower than the edge of the system capacity) less FMSs than “ $e$ ” are

required. We have investigated different values of  $N_{FMS}$  close to the  $\epsilon$  bound and we have seen that the system is not very sensitive to its value. Setting  $N_{FMS}$  to 2 we obtain a small reduction in throughput but a lower delay in the knee region. (Insignificant differences are observed outside this region). In general it is better to under estimate the  $N_{FMS}$  in order to better tolerate small fluctuations in load and accuracy of the adaptive mechanisms, than to over estimate and limit the system capacity. Using 2 FMSs under the same conditions as before the flow on the channel is:

```
HDDD.12HDDD.3HDDD.1HDDD..1142HDDD1.HDDD....121.
1HDDD.22HDDD2.HDDD11HDDD41HDDD.1HDDD..HDDD1.1..
.HDDD..HDDD13HDDD13HDDD.4HDDD1.HDDD221.11HDDD11
HDDD11..111.HDDD13.1HDDD11HDDD1.HDDD34HDDD.1HDD
D.1HDDD.2HDDD1.HDDD13HDDD51HDDD21HDDD1.HDDD.2HD
DD21HDDD.3HDDD1.HDDD2.HDDD1.HDDD2.HDDD12HDDD12H
DDD21HDDD1.HDDD21HDDD31HDDD21HDDD..HDDD11HDDD..
HDDD11HDDD.1HDDD2.HDDD32HDDD.1HDDD11HDDD2.HDDD1
1HDDD2212.11HDDD1.HDDD..HDDD.2.HDDD.11HDDD2.HDD
D21HDDD31HDDD..HDDD.1HDDD1.HDDD11HDDD..HDDD.12.
31.HDDD31HDDD.2HDDD11HDDD1.HDDD211HDDD1.HDDD221
11HDDD12HDDD11HDDD.1HDDD112HDDD..HDDD22HDDD.3HD
```

The performance improvement obtained by breaking the cycles is shown in Fig. 6 curve *c*. The difference in performance of the S and FMS schemes varies with the round-trip delay, number of users and piggybacking.

The effect of the round-trip delay is shown in Fig. 7. Curves *a* and *b* belong to the system with distances (25, 40) Km. Curves *c* and *d* include the head-end (HE) processing delay of 1.5 msec. This delay increases the RTD by 105 mini-slots. For reasonably short distances the degradation is in delay while for longer RTD the throughput is also reduced<sup>8</sup>. Note that the cyclic behavior becomes important at earlier loads for longer RTDs, curves *a* and *b* split at 35% load while curves *c* and *d* split at 25% load.

The minimum delay bound is related to the round-trip delay required to send the request and receive the response. This is indeed the value we obtain for both systems at very low loads. It is interesting to see that this minimum delay is achieved over a significant range of loads. This is due to the excess of contention bandwidth, achieved by setting all unreserved mini-slots to

---

<sup>8</sup>This is true in a real system where the number of stations is finite. Under the assumption of infinite stations the delay increases but the throughput is maintained since there are always stations ready to send.

CMSs (referred to as UMS - Unforced mini-slots), thus no bandwidth is wasted.

The problem with cycles is the lack of opportunity to send a request during data transmission. Piggybacking allows requests to be sent, at virtually no bandwidth cost, along with the data cells. These requests enlarge the data part of the cycle and break up the CMSs part of the cycle. The gain in throughput is two fold: piggybacking reduces the load on the contention channel reducing the number of CMSs required; the length of the data part of the cycle increase compared to the length of the CMS part of the cycle. However, an unfairness problem arises since active stations, applying piggybacking, maintain low access delay while stations becoming active increase their delay (they have to wait until the others totally finish their data transmissions – empty the queue). The cycle periodicity determines the rate at which new arrivals can engage the access mechanism and therefore the minimum average delay. This long blocking period also affects the adaptability of the dynamic mechanisms and thus the reaction time to abrupt changes increases. Fig. 8 compares the S and FMS schemes when piggybacking is used.

The actual number of FMSs ( $N_{FMS}$ ) required not only depends on the load carried by the contention channel but also on the data load represented by each request. When each data request requires, for example, two data cells to be transmitted the contention load is halved compared with that required by single cell messages, and thus the number of FMSs required is also halved. On the other hand, if one third of the requests are carried by the piggybacking mini-slot then the maximum load on the contention channel is reduced by the same amount. Therefore, the expressions for the number of FMSs and maximum throughput generalize to the following:

$$N_{FMS}(\lambda_p, l) = (1 - \lambda_p) \frac{e}{l} \quad (3)$$

$$\eta_{max}(\lambda_p, l) = \frac{3}{4 + N_{FMS}(\lambda_p, l)} \quad (4)$$

where  $l$  is the average message length and  $\lambda_p$  is the piggybacking load.

A simple estimation of  $l$  can be derived from the head-end queue size as the ratio between the total number of requested mini-slots versus the actual number of requests (entries) in the queue. This measure is somewhat noisy when the head-end queue is small but the accuracy in this region is not critical since UMSs (unforced mini-slots) are available. Fig. 10 curve *a* shows the results obtained for Type 1 traffic (which has an average message size of 8 mini-slots) with no piggybacking.

The maximum theoretical throughput bound of 69.12% is approached for very high delays (not shown) but it is already close to the bound at reasonable delay levels. The difference between curves *a* and *b* Fig. 10 gives the improvement obtained by using piggybacking. The theoretical bound for piggybacking is 75% (when all the load is piggybacked and the contention channel disappears). It is possible to reach a utilization of 72%, with large delay of the order of 2700 mini-slots. For comparison, curve *c* shows the performance of Type 2 traffic with piggybacking.

In a fixed-slot structure piggybacking improves the maximum throughput as well as the access delay [3]. In a variable slot structure, the contention bandwidth is so high at low and medium loads that piggybacking is rarely used. It only helps at loads that are higher than the contention channel can carry. Comparing curves *a* and *b* we see the small difference it makes for this traffic type.

It should be possible to get better performance with fewer stations. The maximum throughput bound for 5 active stations is 46.55%. And indeed, the system approaches this bound very closely with a 46% throughput at a delay of 650 mini-slots (in a system with a  $RTD = 1$  minis-slot). On the other hand, when the round-trip delay is higher the throughput is smaller with fewer number of active users. This is due to our assumption (although it is not necessary) of having only one outstanding request per station at any instant of time. Fig. 9 shows the performance when varying the number of stations for the default round-trip delay configuration (29 mini-slots). 20 stations are not yet enough to fill the pipe with requests and therefore the delay is slightly higher (instead of smaller).

As described in Section 3.1 it is advantageous to keep stations at their respective locations rather than padding them out to be virtually at the end of the line. Fig. 11 compares the two approaches. Curve *a* corresponds to the default configuration. When stations are logically moved to the EOL the system becomes virtually equivalent to a system with all stations at the farthest distance (curve *b*). There is a significant improvement in delay between *a* and *b* (especially at high loads). The driving factor at high loads is the average access delay and not the maximum round-trip distance. For comparison, curve *c* shows the performance of a system with a round-trip delay equal to the average round-trip for curve *a* and all the stations at the end of the line.

The multiplicity of collision (number of stations that write in the same mini-slot) is of great importance in an HFC system. Laser clipping can occur when the amplitude of the composite

signal exceeds the dynamic range of the laser amplifier. Multiplicity of collision increases the probability of laser clipping. Fig. 12 shows the multiplicity histogram for a severely stressed contention channel (Type 2 traffic, load 43%, no piggybacking). We see that the first two bins (corresponding to empty and success mini-slots), contain the majority of the samples (80.25%). The maximum multiplicity occurring in this experiment is 8 and the probability of having more than 4 stations transmitting in the same mini-slot is 0.0056 (half percent).

### 6.3 Dynamics of the System

While analyzing the performance of adaptive algorithms like the CSA and the Rivest estimation algorithm, it becomes important to look at the dynamic response of the system to study the adaptation process.

Fig. 13 illustrates the tracking of the actual backlogged stations by the head-end estimator. Note the small number of backlogged stations in spite of having 50 active stations at 72% load. The actual number of backlogged stations is not available in a real system but has been computed here by checking the state of the stations at every mini-slot. It is evident that the estimator at the head-end tracks the actual number of backlogged stations very well with a slight lag due to the propagation delay in the feedback loop.

We can define an ideal system by using the actual number of backlogged stations to set  $p$  to  $1/N_a$  and  $N_{FMS}$  to  $P_s(N_a)$ . The performance of our system using the estimated values is extremely close to that of the ideal system with only one or two percent difference at very high loads.

Fig. 14 captures the dynamics of the adaptive system under the same conditions as before (Type 1 traffic, 72% load, no piggybacking) over a longer period. The upper plot of the figure illustrates the temporal variation of the actual backlogged number of stations. The lower plot shows the corresponding multiplicity of collision (The successful transmissions are not shown for better readability of the plot). We observe significant fluctuation in the actual number of backlogged stations. However the multiplicity of collision tends to remain fairly invariant. We would expect this if the estimated value of  $p$  is accurate since the CRA mechanism will then distribute the backlogged stations among the slots in a ratio close to optimum.

## 6.4 Transient Response

To test the stability of the adaptive mechanisms, we have used an extreme form of transient overload as shown in Fig. 15. After the network was allowed to run for a period at a low (5%) Type 2 traffic load, each station (200) was forced to generate two new packets at the same instant in addition to the existing low load. This corresponds to a load of 400 cells being applied instantaneously to the contention channel. The system tracks the applied load very well. The minimum time to digest this impulse is 2924 mini-slots, which consists of 1600 mini-slots of “impulse” data, 134 mini-slots of background (5%) load, and the remainder are the CMSs at the rate of  $\epsilon$  mini-slots per data slot. The system took about 3200 mini-slots to transmit the load, marginally greater than the minimum of 2924.

The estimation of the backlogged stations closely follows the actual value. The estimate increases at its maximum rate when the impulse load is applied and slows down when successes start to occur. The tracking after this point is due to the correct balance of the pseudo-Bayesian estimator.

The CSA also adjusts the bandwidth as required. In the beginning when the load is very low, the contention channel bandwidth is high as the CSA converts unused mini-slots to CMSs. However when the impulse load is applied, the bandwidth requirement of the data channel rises suddenly and the CSA responds quickly by trading off contention channel bandwidth for data bandwidth as seen in the drop in the contention bandwidth. Note that the reaction of the CSA is shifted with respect to the CRA keeping the contention bandwidth high to allow the CRA to adjust more rapidly and generating the data bandwidth only when successes start to appear. Once the queues at the head-end clear out, the contention bandwidth becomes high again to serve the low background load.

## 7 Comparison

### 7.1 Performance

Comparing the performance of the protocol described here with other frame-based protocols described in the published literature is difficult for a number of reasons: there are very few

results actually reported; the parameters of the simulations differ; different CRAs are used. One comparison that can be made is to compare the lower bound on access delay.

The round-trip delay is an important component of the access delay. However in today's applications the RTD could be less than one slot (e.g. a data rate of 1Mbps with a round-trip length of 10 Km) in which case it is unimportant. However, we would like the protocol to still work well when the data rate is 30 Mbps and the round-trip length is 100Km (approximately 230 slots of RTD).

For the frameless protocol the bound has been described in section 4.2 and is equal to one RTD plus a few mini-slots. To provide an idea of the basic access delay mechanism for framed MAC protocols we consider an elementary implementation as shown in Fig. 5. The CMSs are clustered at one end of the frame, forming the contention region, as found in most of the current practical frame structures. For such clustered frame structures, the minimum access delay is given by :

$$T_{min} = 1/2 * F + T_{rtd} + T_{data}$$

where  $F$  is the length of the frame. The data region that precedes the contention region of the frame must have a duration larger than the worst case round-trip delay of the network. This ensures that all the stations that transmitted requests in the previous frame, can get their feedback in time for retransmission in the contention region of the current frame, if required. The boundary between the data region and the contention region is dynamically controlled by the head-end, based on the load. The maximum length of the contention region is a design parameter, but one would like to have the ability to convert up least 50% of the frame into a contention region if required<sup>9</sup>. This forces the length of the frame to be greater than  $2 T_{rtd}$ . For this frame length, the access delay at low loads is 2 round-trips as against the one round-trip delay for the frameless implementation. The access delays in practical frame-based systems may be 3 round-trips or even higher. Additional delay can be introduced by particular block-oriented CRA implementations[13], or by the frame control mechanisms which can sometimes require a station to skip the frame in which it receives the feedback before transmitting in the data portion of the subsequent frame. It may be possible to closely achieve the minimum access delay given in Equation 1 with a frame based-system if the contention region is distributed over the entire frame and p-persistence is used. In this case however the notion of a frame has almost disappeared.

The maximum throughput bound is hard to compare because the throughput depends on the

---

<sup>9</sup>For comparison our protocol uses up to 55% of all slots for contention

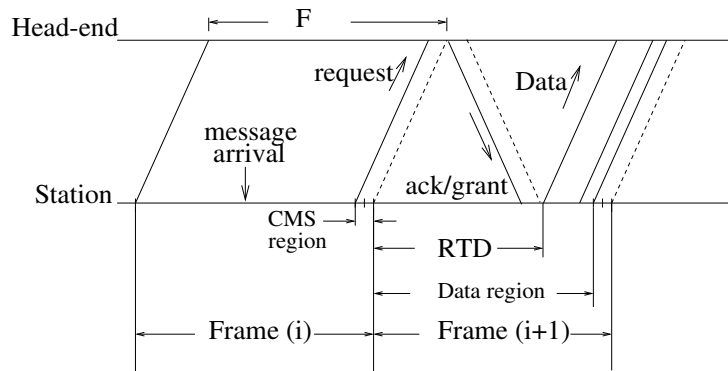


Figure 5: Components of the minimum access delay for a typical frame-based MAC protocol.

performance of the CSA, and the CSA in turn, interacts with the CRA. Based on the fact that the frameless protocol converges closely to the bound allowed by the CRA it would be unlikely that a frame-based protocol using the same CRA would perform any better than the protocol described here<sup>10</sup>.

## 7.2 Flexibility

Because the frameless protocol does not commit to any prior structure on the upstream channel CMSs and data slots can be assigned rather arbitrarily. For example, if it were desired to insert a periodic mini-cell into the data stream, the head-end scheduler would avoid using this mini-slot. The periodic slot could then occur anywhere, even in the middle of a data slot. No changes would be necessary to the CRA and a minor change may be necessary in the CSA.

## 7.3 Multiplicity of collision

In frame-based protocols, where blocking CRAs are used (i.e. newcomers are not admitted until the end of the resolution cycle), new stations can accumulate during the cycle and produce a large multiplicity immediately after the cycle ends. One way to address this problem is to admit stations throughout the cycle using a p-persistence like algorithm [16]. Using a continuous CRA as described here obviates the necessity of an additional mechanism.

<sup>10</sup>We plan to explore further whether potentially more efficient CRAs can be adapted to the frameless approach.

## 7.4 Implementation

We believe there are two aspects of the protocol that will reduce the cost and complexity of implementations, more particularly when the data rates are high<sup>11</sup>.

The first is the extreme simplicity of the station protocol and the relative simplicity of the head-end protocol. The focus on simple station design is an extension of previous work where this issue was examined more fully in [3]. There are indirect benefits of simplicity such as having to deal with fewer possible error conditions. An examination of the robustness of CPR was made in [3]. There are essentially two additional signals that are transmitted from the head-end to the station relative to CPR; the value of  $p$  and the indication of mini-slot/data slot. Without performing a detailed analysis, we believe that an error in the former will reduce performance for a short period (until the next time  $p$  is broadcasted), and an error in the latter may lead to a data slot being overwritten. The error effects do not propagate.

The second aspect is that the three primary functions performed at the head-end, contention resolution, contention slot allocation and scheduling are largely independent of one another and suggest an architecture where very high-speed performance may be obtained by implementing each function in a separate module operating with very loose coupling.

## 8 Conclusion

We have shown how the advantages of using an adaptive mechanism can be retained while at the same time reducing delay to close to the round-trip delay time while still retaining a very simple structure. We do this by avoiding any frame structure and operating slot-by-slot. A contention slot allocator at the head-end dynamically distributes the overall channel bandwidth between the contention channel and the reservation channel. An adaptive  $p$ -persistent contention scheme was selected for its simplicity and ease of application to a frameless protocol. Simulations reveal that the delay and throughput measures of performance closely approach the limiting bounds at low loads and high loads, respectively. We examined the transient performance of the system when subjected to a severe impulse load. The system showed no tendency of instability and transmitted the impulse in approximately 110% of the ideal performance. In a centralized implementation

---

<sup>11</sup>Ideally, many of the functions performed at the head-end and station should be executed within a mini-slot for best performance.

of the algorithm, two common pieces of control information need to be fed to each station, a periodic update of the transmit probability of the  $p$ -persistence algorithm and a bit to denote whether each upstream slot is for data or reservation. While we have not formally analyzed the behavior of the protocol in the presence of channel errors, we believe it is robust because of the minimal amount of state that is maintained in the stations.

## References

- [1] “Formal MAC Proposals” *IEEE802.14 Cable TV Protocol Working Group*, November 1995.
- [2] L. G. Roberts, “Dynamic Allocation of Satellite Capacity Through Packet Reservation”, *AFIPS Conference Proc., National Computer Conference*, 42, pp. 711-716, June 1973.
- [3] D. Sala and J.O. Limb, “A Protocol for Efficient Transfer of Data over Fiber/Cable Systems”, *Proc. INFOCOM’96*, pp. 904-911, San Francisco, March 24-28, 1996.
- [4] P. Mathys and P. Flajolet “Q-ary Collision Resolution Algorithms in Random-Access Systems with Free or Blocked Channel Access” *IEEE Trans. on Information Theory*, Vol. IT-31, No. 2, March 1985.
- [5] P. Jacquet, P. Muhlethaler, P. Robert “Asymptotic average access delay analysis: adaptive  $p$ -persistence versus tree algorithm” *contribution to the IEEE802.14 WG*, No. IEEE 802.14-96/248, December 1996.
- [6] B. Mukherjee, A.C. Lantz, N.S. Matloff, and S. Banerjee “Dynamic Control and Accuracy of the  $p_i$ -Persistent Protocol Using Channel Feedback” *IEEE Trans. on Communications*, Vol. 39, No. 6, June 1991.
- [7] W. Xu and G. Campbell, “A Distributed Queuing Random Access Protocol for a Broadcast Channel” *ACM SIGCOMM’93*, pp. 270-278.
- [8] C-T. Wu and G. Campbell, “Extended DQRAP (XDQRAP) A Cable TV Protocol Functioning as a Distributed Switch”, *Proc. 1994 1st International Workshop on Community Networking*, pp. 191-198, July 13-14, 1994.

- [9] J. O. Limb and D. Sala “An Access Protocol to Support Multimedia Traffic over Hybrid Fiber/Coax Systems”, *Proc. 1995 2nd International Workshop on Community Networking*, pp. 35-40, June 20-22, 1995.
- [10] C. Grobicki and J.M.Ulm “UniLINK as a Media Access Protocol for Community Cable TV” *Proc. 1995 2nd International Workshop on Community Networking*, pp. 41-48, June 20-22, 1995.
- [11] J.Dail, M.Dajer, C-C Li, P.Magill, C.Siller, K.Sriram and N.Whitaker “Adaptive Digital Access Protocol: A MAC Protocol for Multiservice Broadband Access Networks” *IEEE Communications Magazine*, pp. 104-12, March 1996.
- [12] C. Bisdikian, B. Neil, R. Norman and R Zeisz “MLAP: A MAC Level Access Protocol for the HFC 802.14 Network” *IEEE Communications Magazine*, pp. 114-21, March 1996.
- [13] C. Bisdikian “Performance Analysis of the Multi-slot n-ary Stack Random Access Algorithm (msSTART)” *contribution to the IEEE802.14 WG*, No. 802.14-96/117, May 1996.
- [14] R. Citta and D. Lin “Formal MAC Layer Protocol Proposal: Adaptive Random Access Protocol for CATV Networks” *contribution to the IEEE802.14 WG*, No. 802.14/95-144, October 1995.
- [15] C.C. Lee, R. Citta and D. Lin “Performance Analysis of Adaptive Random Access Protocol, Part I” *contribution to the IEEE802.14 WG*, No. 802.14-96/070, March 1996.
- [16] C.C. Lee, R. Citta and D. Lin “Performance Analysis of Adaptive Random Access Protocol, Part II” *contribution to the IEEE802.14 WG*, No. 802.14-96/071, March 1996.
- [17] D. Bertsekas and R. Gallager “Data Networks”, *Prentice Hall* 1992.
- [18] J.O. Limb et. al. “Performance Evaluation Process for MAC Protocols” *contribution to the IEEE802.14 WG*, No. 802.14-96/083R2, March 1996.
- [19] “Cable Modem Baseline Document” *DAVIC 1.1 Specification*, Baseline Document No 18, 1996.

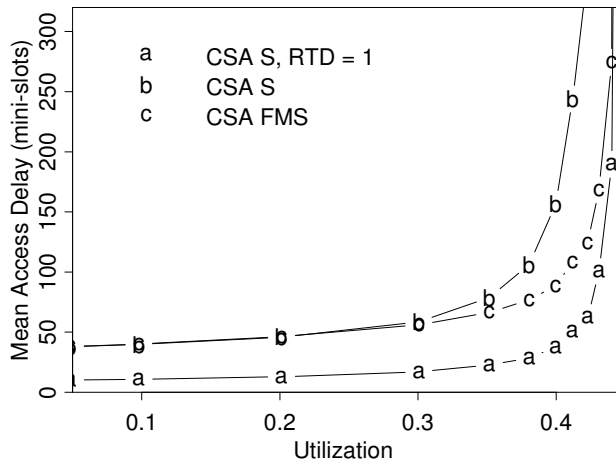


Figure 6: Mean access delay for the default configuration given in Table 4.1 with Type 2 traffic (Poisson Traffic) and no piggybacking. The existence of cycles in S CSA increases the access delay respect FMS CSA in this RTD delay (29 minis-slots).

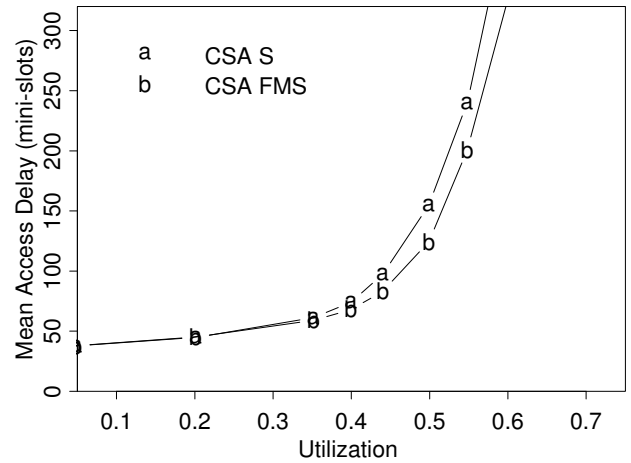


Figure 8: Delay comparison of S and FMS using piggybacking. Piggybacking reduces the length of the CMS part of the cycle.

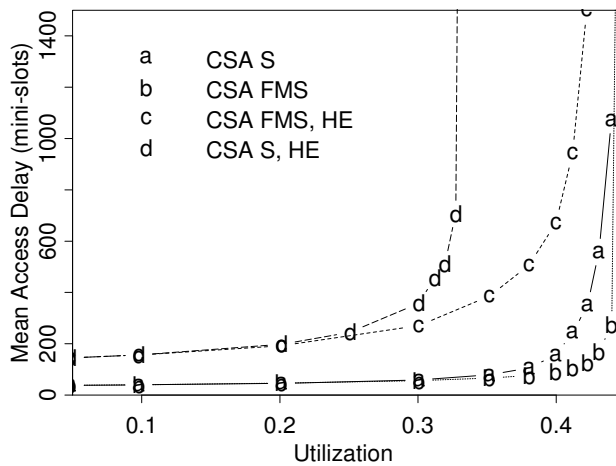


Figure 7: Effect of the head-end processing delay in the two CSA schemes. The RTD is more than 4 times higher when the head-end processing delay of 1.5 ms is considered. The cyclic behavior translates to a big reduction in throughput (with this number of users).

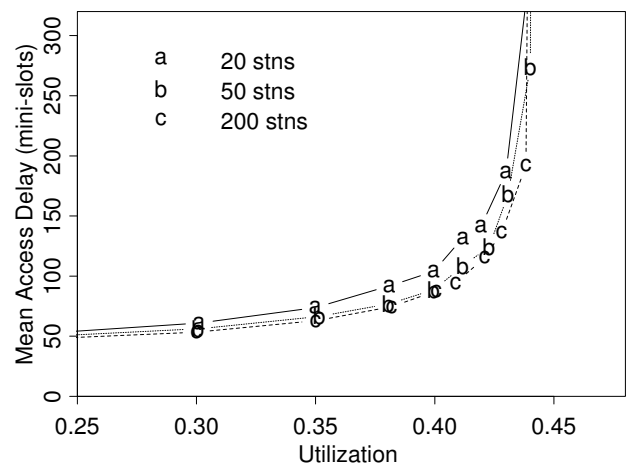


Figure 9: Performance of FMS CSA with different number of stations and RTD = 29, Dist(25,40). Few stations cannot seize the longer round-trip since they are restricted to at most one outstanding request.

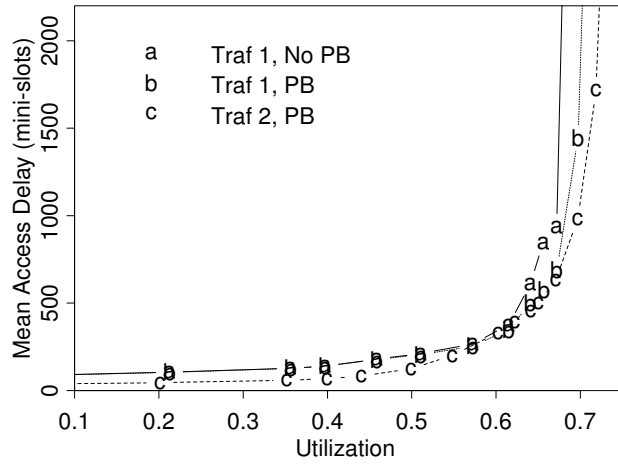


Figure 10: Maximum Throughput can be achieved with longer message length or with piggybacking (PB).

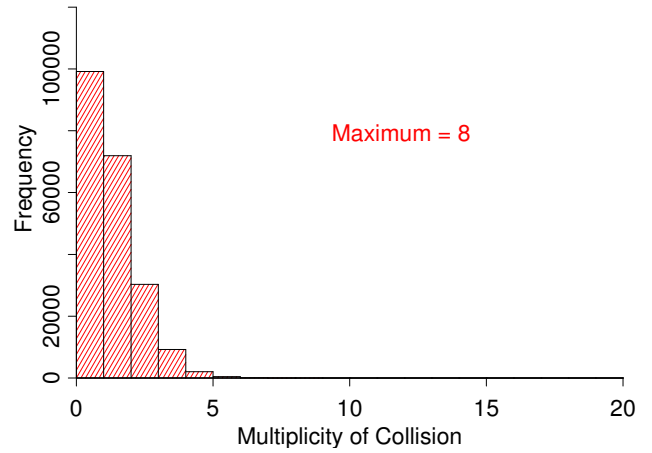


Figure 12: Multiplicity of collision histogram at 43% load with Type 2 traffic. Only half percent of the samples are greater than 4 and the maximum is 8.

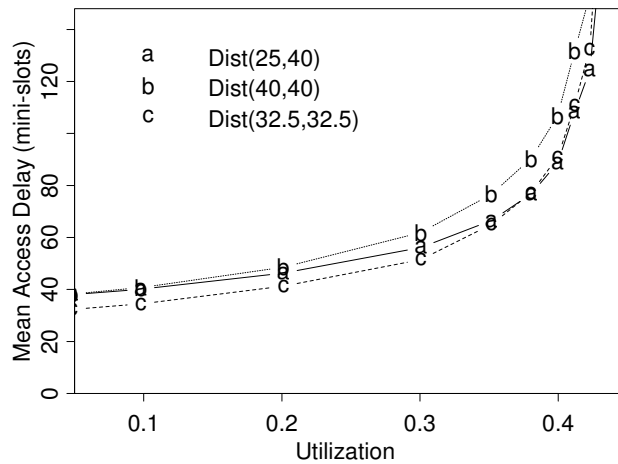


Figure 11: Padding effects: Stations in *a* are moved to be virtually at the EOL in *b* and at the average RTD in *c*. The driving factor at high loads is the average time it takes to receive the collision feedback.

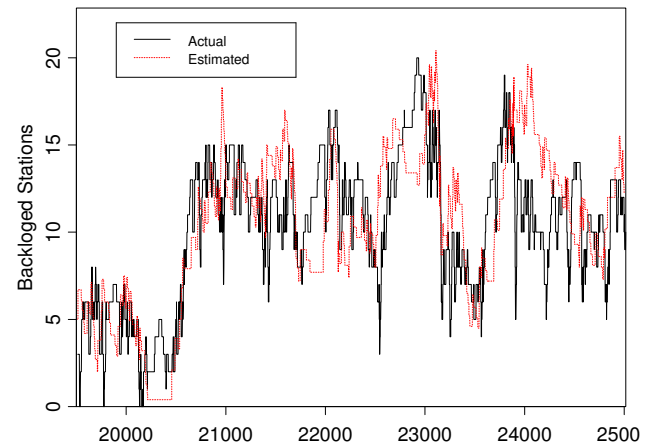


Figure 13: Actual and estimated backlogged stations at load 72% with Type 1 traffic. There is a shift in time between the actual and estimated due to the round-trip delay.

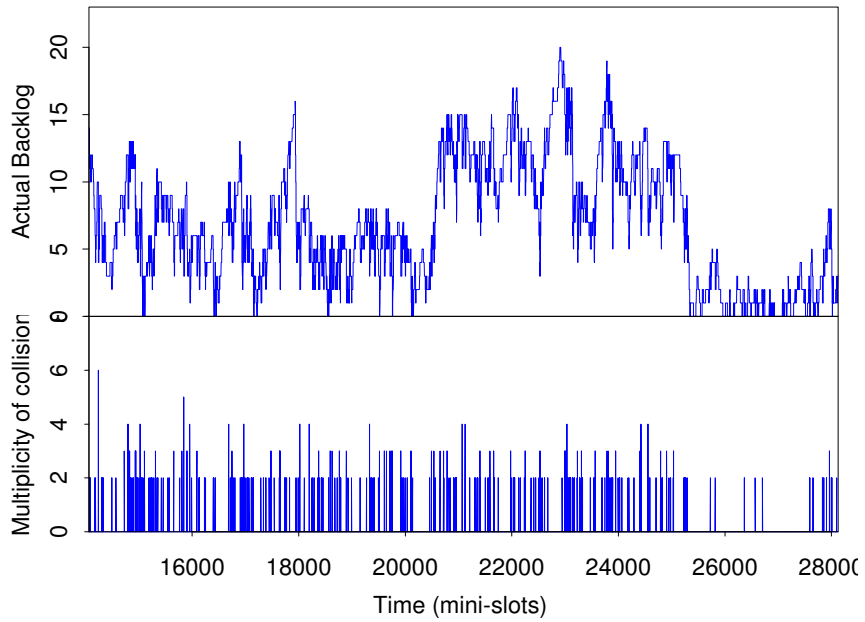


Figure 14: Backlogged stations and Multiplicity of collision for the same experiment as in Fig. 13.

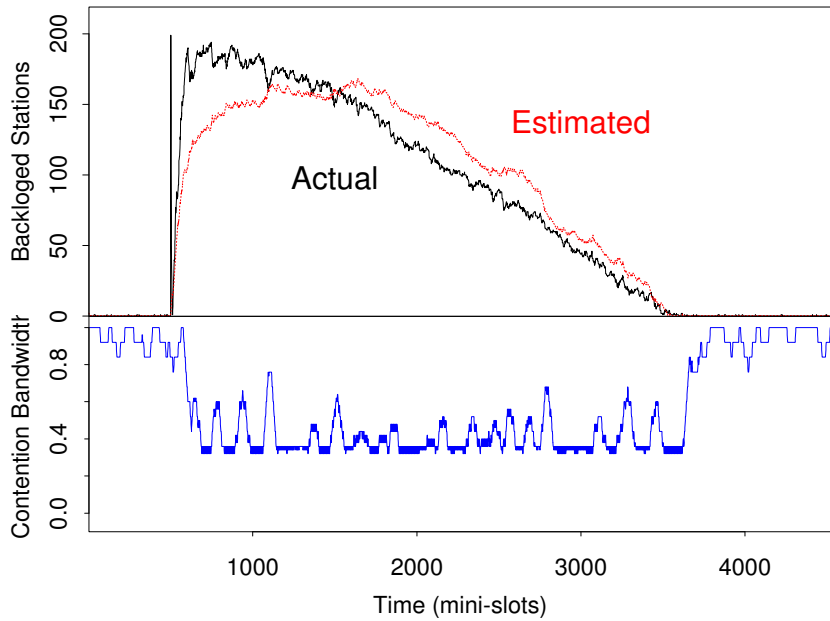


Figure 15: Stability test of the adaptive mechanisms. The system recovers with a time close to optimal from an abrupt increase in load (from 5% to full load - 2 arrivals at 200 stations).