Statistical Multiplexing of Upstream Transmissions in DOCSIS Cable Networks

César Heyaime-Duvergé and Vasant K. Prabhu, *Life Fellow, IEEE*

***Abstract*—The appearance of increased-bandwidth applications with quality-of-service (QoS) requirements has brought about the emergence of multi-service access networks. Data Over Cable Service Interface Specification (DOCSIS) networks are intended to support high quality audio, video, and interactive services over Internet protocol (IP) flows within a Hybrid Fiber/Coax (HFC) network context. In order to support large amount of users and demanded QoS for such applications, a careful management of resources is needed. In this paper we propose a centralized bandwidth allocation scheme for the return path of DOCSIS networks. The proposed scheme takes advantage of the application's traffic characteristics to minimize the bandwidth dedicated to control signals and achieve high bandwidth utilization. At the same time, the proposed approach complies with the QoS requirements of the serviced applications. The performance of our bandwidth allocation approach was evaluated using computer simulation. The performance metrics used are the average packet delay, data throughput, and number of users supported. Results are obtained for networks with a single service as well as for a mixed service scenario. As shown by the results, our approach achieves high bandwidth utilization, and still complies with the delay constraints imposed by the user applications. It increases the system capacity as compared to reference approaches.**

***Index Terms*—Cable modem (CM), collision resolution, DOCSIS, MAC, QoS, scheduling, statistical multiplexing.**

I. Introduction

A CLASS of users that has drawn much attention over the last few years is that of residential users. Lots of effort has been devoted to the development of options to support broadband services to residential areas. Cable networks, originally deployed to provide one-way broadcast services, have emerged as one of the main technologies employed to provide two-way communication services to the home.

The DOCSIS standard [1], [2] delineates physical and data link layer protocols as well as the functionality necessary to provide two-way communication services to residential users. DOCSIS also defines the mechanisms utilized to support resource allocation, and different quality assurances for the services offered.

Cable modems (CM), the end user equipment, access these upstream channels using a time division multiplexing approach that divides the channel into mini-slots. Resource management agents at the Cable Modem Terminating System (CMTS) control the use of the upstream mini-slots. Following DOCSIS, mini-slots can be used in contention using a contention resolution algorithm (CRA), or they can be dedicated to specific stations. Generally, the utilized approach is that in which single mini-slots are used to transmit bandwidth requests from the CMs to the CMTS, thus, effectively creating a *reservation* channel. As in the previous case, these single mini-slots can be used in contention, or dedicated to specific CMs.

Due to the asymmetric distribution of bandwidth in DOCSIS cable networks, the return path or upstream channel becomes the bottleneck for supporting the QoS requirements of delay sensitive applications for a large number of users [3], [4]. For this reason, the performance of protocols and applications in cable networks has lately been subject of study [5]–[8]. Thus, it is of great importance to define bandwidth management approaches that manage the upstream bandwidth efficiently in order to support applications with very stringent delay bounds, achieve increased capacity, and increase operation revenues. These bandwidth management algorithms should incorporate an effective priority mechanism that gives a higher priority to these applications to minimize access delay during the period of high contention.

Efficient bandwidth management involves reduction of the bandwidth spent in signaling messages for bandwidth reservation. At the same time, the employed approach needs to be flexible enough so that it provides the mechanisms to comply with the different QoS requirements of the applications being serviced by the network.

A. Previous Work

The design of resource allocation algorithm and medium access techniques for access networks has been drawing the attention of the research community for years. As a result, several MAC protocols, capacity studies, and bandwidth allocation algorithms have been proposed for DOCSIS cable networks.

1) MAC Protocols: Several MAC protocols were proposed as part of the effort to standardize the medium access control protocol for cable networks. Some of these propositions are: XDQRAP, proposed by Wu and Campbell; UniLINK, by Grobicki and Ulm; and CPR, proposed by Sala and Limb. They are described in the following paragraphs

XDQRAP [9] is a reservation based distributed scheme in which the upstream channel is divided into mini-slots and data slots. Mini-slots are used by the user stations to send requests for data slots in a contention mode. The CMTS reflects the information received in the mini-slots to a downstream channel. Stations listen to the reflected transmission in the downstream

Manuscript received November 11, 2009; revised May 05, 2010; accepted May 20, 2010. Date of publication June 28, 2010; date of current version August 20, 2010.

C. Heyaime-Duvergé is with Nokia Siemens Networks, 01210 México, DF, Mexico (e-mail: cesar.heyaime@nsn.com).

V. K. Prabhu is with the University of Texas at Arlington, Arlington, TX 76019 USA (e-mail: prabhu@uta.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TBC.2010.2051496

channel, and can detect if there has been a successful reservation or collision. This knowledge allows stations to control when to transmit by keeping track of all the reservations made. Thus, the system resembles a distributed queue environment.

The UniLINK protocol [10] uses time division multiplexing to partition the bandwidth into time slots, which are grouped into Block Sync Intervals or frames. A central controller or pacer decides the size as well as the use of the time slots within the frames. The Block Sync Interval is divided into three regions: *Contention region*, where stations transmit following the rules of a modified CSMA/CD. This region is used to transmit delay sensitive traffic and is also used to reserve bandwidth in the other regions; *Periodic dedicated region*, where time slots are reserved periodically for the duration of a station's session to provide jitter free bandwidth assignment; *Reservation dedicated region* provides dedicated slot assignments based on reservations on a per frame basis. The pacer indicates the station that can use the periodic and reservation slots and indicates contention mode for the rest of slots (contention region). The number of slots in each region varies dynamically based on the offered load. The CSMA/CD protocol used in the contention region is modified to efficiently operate in a system with long propagation delay.

Centralized Priority Reservation (CPR) [11] is the MAC scheme adopted by the DOCSIS standard to provide medium access control in cable networks. It uses a time division multiplexing approach to divide the upstream bandwidth into fixed size mini-slots.

In CPR a station with a packet available for transmission sends a bandwidth request in a contention mini-slot. After receiving the request, the CMTS echoes back an acknowledgement in the downstream channel. It also informs the station, at the appropriate time, of the mini-slots it can use to transmit the data packets. If an acknowledgement is not received by the station, it assumes that a collision has occurred and retransmits the request following certain CRA. As part of their work, the authors evaluate the performance of CRAs that employ p-persistent, and Pseudo-Bayesian estimation techniques among others.

2) QoS Provisioning: In the literature several proposals have addressed the support of QoS provisioning in cable networks, however, very few studies address the problem within the DOCSIS context, which is *de facto* standard for cable networks.

One of the first studies addressing QoS provisioning in a DOCSIS context is that by Droubi et al. In [3] the authors propose a scheduler based on the Self Clocked Weighted Fair Queuing algorithm to provide bandwidth guarantees for constant bit rate traffic. The proposed approach uses the MAC protocol defined by DOCSIS. They also address the issue of minimizing the scheduling portion of the total delay suffered by a packet to provide delay guarantees for delay sensitive traffic.

In [12] the authors use a concept very similar to Fair Queuing to develop scheduling protocols to efficiently multiplex constant bit rate traffic, such as voice over IP, and provide guaranteed delay bounds. In addition, the proposed approach offers minimum bit rate guarantees to best-effort traffic. As part of their work, the authors propose an approach to use a combination of direct polling and contention in the reservation channel in order to achieve a higher bandwidth utilization.

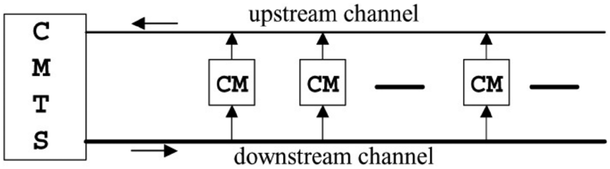


Fig. 1. Topology abstraction of a DOCSIS cable network.

In [13], the authors propose a scheduling architecture to support bandwidth and delay QoS guarantees within the DOCSIS context. More specifically, the authors layout the architecture to support several of the services proposed by DOCSIS.

Other studies exist that propose variations of the algorithms and architectures presented above. However, most of the work in the literature has focused on providing QoS guarantees for different services without addressing the issue of minimizing the used bandwidth. The exception is [14], where the authors propose a QoS MAC scheduling mechanism that provides QoS guarantees for real-time variable bit rate (VBR) traffic over DOCSIS cable networks.

In our work we do not directly extend any of the previous studies. We propose a novel general approach that focuses on providing a close QoS compliance while minimizing the used bandwidth in order to increase system capacity. In version 3.0 of DOCSIS multiple transmit channels in downstream and upstream are possible. Similarly, synchronous code division multiple access (S-CDMA) and time division multiple access (TDMA) are supported for the upstream channel. The proposed approach assumes a system with single transmit channel in both directions (upstream and downstream). It also assumes the use of TDMA mode in the upstream direction. While the proposed scheme can be extended to the case of multiple transmission channels, and S-CDMA scheme, this scenario is out of the scope of this work.

The rest of this paper is organized as follows. Section II provides an overview of DOCSIS procedures. Section III describes the concept of traffic-based bandwidth allocation proposed in this work. Section IV discusses the simulation procedures and system parameters implemented in the simulation. Simulation results are presented in Section V. Concluding remarks are presented in Section VI.

II. Overview of DOCSIS and CATV Networks

An abstraction of the tree-and-branch topology of cable networks is presented in Fig. 1. The upstream channel, controlled by the CMTS, is a multipoint-to-point channel. It is shared by all the CMs using time-division-multiplexing.

The upstream channel can be viewed as a stream of time slots or *mini-slots*. The mini-slot duration is the time needed for transmission of a fixed number of bytes, and is determined by the CMTS. Mini-slots represent the smallest transmission unit in the upstream channel.

Resource management agents at the CMTS decide the usage of upstream mini-slots. They inform the CMs of each mini-slot usage by means of the *Upstream Bandwidth Allocation MAP message* or *MAP message*. A *MAP message* carries transmission grants addressed to particular stations, or specification of mini-slots available for use in contention mode by several stations. *Transmission grants* define slots as being of the data, or request type (*Data or Request grants*). *Data slots* consist of a set of adjacent mini-slots used to carry the data packets transmitted by the CMs. *Request slots* are individual mini-slots assigned to a CM and used to send bandwidth request messages (or *Request messages*) to the CMTS. *Contention mini-slots* (CMS) are individual mini-slots used in contention to transmit bandwidth request messages to the CMTS. A CMS may be simultaneously used by several CMs, in which case collision occurs.

A. Cable Modem Protocol

The procedure followed by the CM to request bandwidth using request messages is summarized in the following paragraphs. The description is very generic and hides the specific details of some concepts involved in the communication between CM and CMTS. For a more detailed description see [1].

Upon receiving a packet from the serviced application, the CM puts together a Request message that includes its identification, and the number of mini-slots required to transmit the packet. If there are no previous outstanding Request messages (a request not acknowledged as received by the CMTS) for that CM, the CM transmits the Request message using a CMS, or a Request slot previously assigned by the CMTS. Note that CMs can also send Request messages piggybacked in Data slots. However, piggybacking of Request messages only occurs when there is one or more packets in queue at the time a packet is being transmitted.

If the transmission was successful, the next MAP message will contain an acknowledgement for the transmitted request, or a Data grant. Data grants are allocated based on the priority of the CM and bandwidth availability. The allocated Data slot is for the exclusive use of the requesting CM.

If the request message is being transmitted using CMSs, the re-transmission attempts are decided by a CRA. The CRA defined by DOCSIS is based on a truncated binary exponential back-off. Following the CRA, a CM with a request to transmit must back off for 2^n CMSs before attempting to transmit the request. The CRA parameters are specified by the CMTS in the MAP message.

Once the request message is transmitted successfully, and a data grant is received, the CM waits for the start of the assigned data slot in order to transmit the corresponding packet. In our implementation, concatenation or fragmentation of packets is not employed. Thus, the CMTS must grant exactly the number of mini-slots requested by the CM.

B. DOCSIS Scheduling Services

In DOCSIS, the main mechanism to provide enhanced QoS is to classify packets traversing the network into a Service Flow. A *Service Flow* is a unidirectional flow of packets that provides a particular QoS. The CMTS and CM provide this QoS by scheduling, shaping, and prioritizing traffic according to the QoS parameter set that defines the QoS to be provided to that Service Flow. DOCSIS provides a set of services to support different Service Flows for CMs. These services are:

- *Unsolicited Grant Service* (UGS) offers Data grants of fixed size at regular intervals of time in an effort to meet the CM (application) real time transmission needs. This service is suitable for applications that generate fixed size data packets on a periodic basis, such as Voice over IP. In UGS, the CM can not use any CMSs, or piggyback Request messages in packets transmitted in Data slots.
- *Real Time Polling Service* (rtPS) provides Request grants to CMs on a periodic basis in order to meet the Service Flow real time needs. It is designed for upstream traffic generated at periodic intervals and with variable packet sizes. In rtPS, the CM can not use any CMSs, or piggyback Request messages in packets transmitted in Data slots.
- *Non-Real-Time Polling Service* (nrtPS) is designed to support non-real time service flows that require variable size data grants on a regular basis such as high bandwidth FTP. The service offers request slots on a periodic basis without the time restrictions of the rtPS service. In this service the CM is allowed to use CMSs and piggybacking of request messages.
- *Best Effort Service* (BE) does not offer periodic data grants or request slots to the CM. It allows CMs to use CMSs and piggybacking of request messages. In BE, the CMTS will issue request slots or unsolicited data slots only in the case they are needed in order to satisfy the minimum reserved bandwidth for the service.
- *Unsolicited Grant Service with Activity Detection* (UGS-AD) is a combination of the UGS and rtPS services. It is designed to support flows that may become inactive for substantial portions of time (i.e., tens of milliseconds or more). The service provides unsolicited data grants when the flow is active, and request grants when the service is inactive.

III. Traffic-Based Bandwidth Allocation Concept

The CRA algorithm employed by DOCSIS is not well suited for applications with strict delay constraints. During periods of congestion, the CRA may result in very high access delays for all the users. In addition, the increased number of collisions may result in an inefficient use of bandwidth. To address this problem, we introduce a bandwidth allocation approach that assigns bandwidth based on the traffic characteristics of the user application, and gives priority to services with strict delay constraints. It is described in the following paragraphs.

The traffic-based bandwidth allocation (TraBA) scheme for bandwidth allocation extends the UGS and rtPS concepts defined in DOCSIS. We call these new services: statistical Unsolicited Bandwidth Assignment (UBA) and Statistical Polling (POLL). It can be summarized as follows: the centralized bandwidth manager develops a statistical model of the traffic generated by each individual user stations being serviced. If using UBA, this model is used to automatically issue Data grants to the transmitting stations. In the case of a POLL scheme, the statistical traffic model can be used to decide when is the proper time to poll the transmitting stations. In both cases, the assignment of bandwidth to stations must ensure that the application's delay constraints are met.

A generic representation of the TraBa concept is shown in Fig. 2. The *Automatic Grant Generator* uses each station's traffic model to decide the time to either poll or issue Data

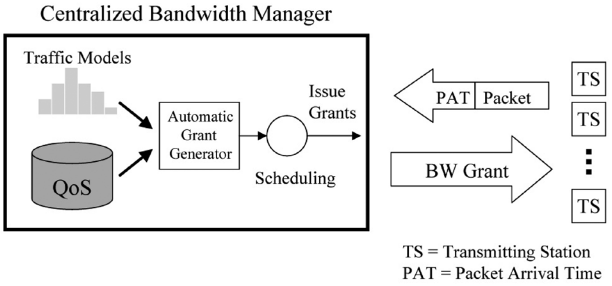

Centralized Bandwidth Manager
Traffic Models
QoS
Automatic Grant Generator
Issue Grants
Scheduling
PAT
Packet
BW Grant
TS
TS
⋮
TS
TS = Transmitting Station
PAT = Packet Arrival Time

Fig. 2. General concept of the TraBA approach.

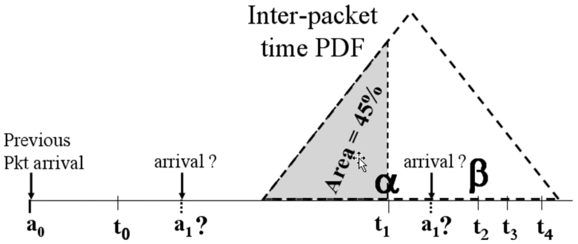


Fig. 3. Inter packet time PDF and new packet arrival.

grants to the stations. When stations with different priorities are serviced, the *Scheduler* makes sure that the Data grants are issued following the appropriate priority.

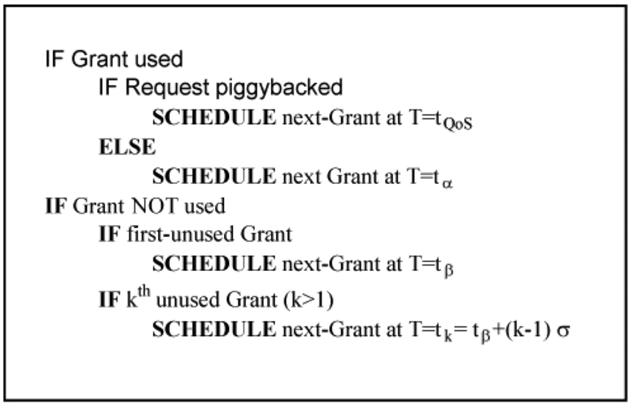
In the TraBA concept upstream packets transmitted by a CM contain a packet arrival time (PAT) field, as shown in Fig. 2. This field is used by the Automatic Grant Generator to decide the time of the next grant (Data or Request depending on the case) to be issued to that CM. It can also be used to develop statistical inter-packet time models of the upstream traffic. The information carried by the PAT field is determined at the time the packet is transmitted by CM. If at the moment a packet is transmitted there is only one packet in queue (the one being transmitted), then the PAT carries the packet's arrival time to the CM's transmission queue. If there are more packets in the transmission queue, then the PAT carries the time of arrival of the next packet in the queue (this is equivalent to piggybacking a Request message).

In our analysis, we will assume that the statistical model is already available to the bandwidth manager. This assumption is based on the existence of traffic models for different types of applications in the literature [15]–[19]. Furthermore, the TraBA concept implies the idea of systems that dynamically develop source traffic models based on the traffic generated by a user or application. These systems can then schedule data transmissions based on these self-developed traffic models. The development of traffic models is out of the scope of this work.

A. Grant Times as Percentiles of Probability Function

Assume that a packet arrives at the CM's transmission queue at time a_0, and is transmitted at time t_0, as depicted in Fig. 3. The arrival time a_1 ($a_1 > t_0$ is assumed) for the next packet is unknown. Automatic grants (Data or Request depending on the case) for the next packet will be issued for times $t_1, t_2, \ldots, t_k$, and so on, until a grant is used.

If we have knowledge of the inter-packet time statistical cumulative distribution function (CDF) $F(t)$, and probability distribution function (PDF), we can select the times t_k of the grants so as to achieve the desired packet delay and minimize the number of unused grants. One simple approach is to make the time of the first two grants, t_1 and t_2, the $100 \times \alpha^{th}$ and $100 \times \beta^{th}$ ($\beta > \alpha$) percentiles of $F(t)$, respectively; and $t_k = t_{k-1} + \sigma$, for $k > 2$. σ is the standard deviation of the inter-packet time distribution. Thus, the potential times for the grants are $t_1 = t_\alpha = F^{-1}(\alpha)$, $t_2 = t_\beta = F^{-1}(\beta)$, and $t_k = t_\beta + (k-2)\sigma$, for $k > 2$. The expectation is that, if β is large enough, with high probability packets will be transmitted using the first or second grants, and that only sporadically more than 2 grants will be needed.



```
IF Grant used
    IF Request piggybacked
        SCHEDULE next-Grant at T=$t_{QoS}$
    ELSE
        SCHEDULE next Grant at T=$t_\alpha$
IF Grant NOT used
    IF first-unused Grant
        SCHEDULE next-Grant at T=$t_\beta$
    IF $k^{th}$ unused Grant (k>1)
        SCHEDULE next-Grant at T=$t_k$= $t_\beta$+(k-1) σ
```

Fig. 4. Next grant time determination.

The procedure employed by the Automatic Grant Generator based on the previously described approach is shown in Fig. 4. The figure shows the decisions made by the Automatic Grant Generator as Data slots previously assigned to a particular CM arrive at the CMTS. D_{QoS} is the desired target (or constraint) packet delay for the serviced application in the serviced CM. In the procedure, the grant time t_{QoS} is the transmission time at which the packet delay D_{QoS} is achieved. If the PAT carries a piggybacked Request message, it is assumed that at the moment the PAT is received by the CMTS, the next upstream packet has been in the CM queue for less than the delay constraint D_{QoS} so that the delay constraint can be satisfied for that packet. Note that the described procedure shows a system issuing Data grants. A similar procedure is followed by a system issuing Request grants.

B. TraBA Packet Delay and Grant Times t_k

The packet delay analysis presented in this section assumes the use of UBA service. In [20] a very similar analysis is followed for the POLL case.

The *packet delay* is defined as the amount of time it takes to transmit a packet once it has been generated by the application. The packet delay is given by the time the packet spends at the head of the queue (access delay), plus the time it spends waiting in queue (queuing delay). The average packet delay can be written as

$$D = D_{NQ}P_{NQ} + D_Q P_Q \tag{1}$$

where D_{NQ} is the delay experienced by the packet when there is no queuing (the access delay). D_Q is the delay experienced by a packet that is queued. P_Q is the packet queuing probabilities in the CM, and $P_{NQ} = 1 - P_Q$ is the probability of no queuing.

It is assumed that real-time applications with strict delay requirements have an inter-packet time larger than the delay constraint D_{QoS}. Otherwise, for inter-packet times smaller than the provided packet delay, transmission queues would build up at the CM. In the following paragraphs, we assume that P_Q is very small, and that queued packets use piggybacked requests and can be transmitted within the time constraints imposed by the application. Therefore, we assume we can approximate the average packet delay by using the packet delay in the non queuing case $(D \approx D_{NQ})$. The analysis will focus on finding the grant times t_k that comply with D_{QoS}.

Using conditional probability, and assuming a negligible transmission delay, the delay D_{NQ} is given by

$$D_{NQ} = \sum_{k=1}^{\infty} \overline{\tau_k} P(I_k) \tag{2}$$

where $\overline{\tau_k}$ is the conditional average delay experienced by the packet given that it arrives to the CM during the time interval k. $P(I_k)$ is the probability that the packet arrival time to the CM (a_1) is in the kth interval. Time intervals are defined as the time between two consecutive grants. The grant times t_k are: $t_1 = t_\alpha = a_0 + F^{-1}(\alpha)$, $t_2 = t_\beta = a_0 + F^{-1}(\beta)$, and $t_k = t_\beta + (k-2)\sigma$. For these grant times, the probabilities $P(I_k)$ are given by

$$P(I_1) = \Delta_\alpha = \alpha - F(t_0), \tag{3}$$

$$P(I_2) = \Delta_\beta = \beta - \alpha, \tag{4}$$

$$P(I_k) = \Delta_k = F(t_k) - F(t_{k-1}) \; for \; k > 2. \tag{5}$$

Similarly, the kth interval average delay $\overline{\tau_k}$ is given by

$$\overline{\tau_k} = \int_{t_{k-1}}^{t_k} (t_k - t) \frac{f(t)}{\Delta_k} dt, \tag{6}$$

where t_k and t_{k-1} are the limits of the interval, and Δ_k is the probability that the reference packet arrives in that interval. In these equations F(), and f() are the inter-packet time CDF and PDF, respectively. After rearranging, (2) can be written as

$$D_{NQ} = h(t_\alpha, t_\beta) = t_\alpha \Delta_\alpha + t_\beta \Delta_\beta + \sum_{k=3}^{\infty} t_k \Delta_k - \int_{t_0}^{\infty} t f(t) dt \tag{7}$$

C. Packet Delay, α, and β

From (7) we see that α, and β, the $100 \times \alpha^{th}$, and $100 \times \beta^{th}$ $(\beta > \alpha)$ percentiles of $F(t)$, and the corresponding grant (Data or Request) times (t_α, and t_β) determine the average packet delay to be achieved. Therefore, it is important to find the values of α, and β that make the average packet delay equal to the desired target delay $(D_{NQ} = D_{QoS})$.

The parameters that achieve a target delay D_{QoS} can be found using the following generic procedure (where we have used the fact that the delay is an increasing function of β):

```
Step 1) Select an initial value of $\beta$.
Step 2) Repeat
Step 3) Evaluate $D_{NQ} = g(t_\beta)$ (defined below).
Step 4) If $D_{NQ} > D_{QoS}$ decrease $\beta$ by $\epsilon_\beta$.
Step 5) If $D_{NQ} < D_{QoS}$ increase $\beta$ by $\epsilon_\beta$.
Step 6) Until $(D_{NQ} - D_{QoS}) \leq \epsilon$
```

In the procedure, the value ϵ is a very small value that ensures the algorithm does not run indefinitely. In practice, we have used $\beta = 0.5$ as initial value. The function $g(t_\beta) = h(t_{\alpha_{min}}, t_\beta)$ is a function of β (or t_β) only. $t_{\alpha_{min}}$ is the value of t_α that minimizes the delay given by (7) for a given value of t_β.

The value $t_{\alpha_{min}}$ is obtained by finding the root of $dh(t_\alpha, t_\beta)/dt_\alpha = 0$, where t_β is a fixed value. Thus, the evaluation of $g(t_\beta)$ in *Step 3* of the procedure above involves the sub-procedure

```
Step 1) Get $t_{\alpha_{min}}$ by finding the root of $dh(t_\alpha, t_\beta)/dt_\alpha = 0$.
Step 2) Evaluate $g(t_\beta) = h(t_{\alpha_{min}}, t_\beta)$.
```

After finding $t_{\alpha_{min}}$, α_{min} is given by the transformation $\alpha_{min} = F(t_{\alpha_{min}})$. Similarly, the relationship between β and t_β is given by $\beta = F(t_\beta)$.

Using the procedures above, searching for the α and β combination that achieves the target delay reduces to finding a value of β only, as the value of α (α_{min}) that minimizes the delay given by (7) is fixed for a given β.

The procedure finds the minimum average packet delay achievable for a given β. The average delay is an increasing function of β, thus, the procedure finds the largest value of β that complies with the target average packet delay D_{QoS}. Since larger values of β achieve higher bandwidth utilization (defined in later sections), the procedure is optimal in the sense that it finds the values of α and β that achieve the target delay while maximizing the bandwidth utilization in the system.

D. Effect of the Packet Size

For sufficiently high transmission rates, the effect of the packet size could be negligible. consider an application generating packets that are 500 bytes long in average. For a transmission rate of 2.5 Mbps, the transmission time is approximately 1.6 milliseconds. Thus, if we consider delay requirements on the order of 25 to 30 milliseconds, the packet transmission time is approximately 6% of the packet delay. In case of large packet sizes (or low transmission rates), the transmission delay may not be negligible. Following the example above, for 1500 bytes long packets the transmission delay is approximately 4.8 milliseconds, which is approximately 20% percent of the packet delay requirement.

Analysis with different packet size distributions shows that for large enough transmission rates the packet size distribution does not affect the achieved packet delays [20].

IV. System Simulation and Parameters

In order to evaluate the performance of the proposed approach we developed simulations programs in C language using the event-driven simulation technique. For comparison purposes, we defined a contention-based bandwidth allocation approach and used it as the reference performance. In the sections that follow the assumptions and parameters used in the evaluation of the proposed and reference approaches are discussed.

A. QoA and Performance Metrics in DOCSIS Networks

In the following sections we will evaluate the proposed bandwidth assignment approach for the cases of single and mixed services scenarios. For the case of a single service to real-time applications, the performance metrics will be the packet delay and the network capacity. The *packet delay* was defined in the previous section. The *network capacity* is defined as the number of users that can be supported by the system while complying with the QoS attributes of the user class, in this case, the packet delay.

In a mixed service environment, where delay-sensitive and delay tolerant applications are serviced simultaneously, the performance metrics will be the packet delay (for real-time applications), the real-time network capacity, or number of real-time applications supported, and the delay-tolerant total throughput. The *total throughput* is the total transmission rate achieved in the system by the delay-tolerant applications. Equivalently we will use the *average CM throughput*, or the average transmission rate per CM.

B. Source Traffic Models and QoS Constraints

1) Delay Sensitive Traffic Model: In [15], traffic models for action video game applications were described using the Gamma density function. Typical values for the inter-packet time mean ranged from 35 to 90 milliseconds, with standard deviations in the range 3 to 10 milliseconds. For comparison purposes we evaluate a system servicing real-time applications with different values for the mean and variance. In the first case we will use a mean of $\mu = 65 \text{ milliseconds}$, and standard deviation of $\sigma = 15 \text{ milliseconds}$. We will also consider an application with $\mu = 50 \text{ milliseconds}$, and $\sigma = 10 \text{ milliseconds}$.

The video game packet size was characterized as having a mean in the range 35 to 45 bytes, and standard deviation in the range 2 to 4 bytes. If we add the size of the underlying protocol headers, the packet size is close to the minimum protocol data unit (PDU) size in DOCSIS, which is 64 bytes. Taking into account the small standard deviation, and the fact that a mini-slot can carry 8 bytes, a fixed packet size of 64 bytes is a good approximation to the video game packet size. This is the value we will use in calculating the performance metrics for the case of a 65 milliseconds average inter-packet time. In this case, each CM offers a load of 7.88 kbps to the system. For illustrative purposes, we will use a variable packet length in the traffic model with a 50 milliseconds average inter-packet time. In this case, we assume that packets of size 64, 128, and 192 bytes are generated by the application with equal probability. Thus, the average packet length is 128 bytes, and the average offered load is 20.5 kbps per CM, approximately.

An analysis of the effect of delay on the performance of video game applications is out of the scope of this work. In [21] one-way delays of about 50 milliseconds are assumed for real time gaming applications. Out of this, 20 to 30 milliseconds are assigned for the delay in the access network. The remaining delay is budgeted for other sources of delay (e.g., backbone network). In our study, we will choose a 20 milliseconds target delay for the $\mu = 65 \text{ milliseconds}$ case. For illustration purposes, a more strict target delay of 10 milliseconds will be used for the case $\mu = 50 \text{ milliseconds}$.

Due to the fixed packet size nature of the real time video game application, and delay constraints imposed by its packet delay budget, a scheme that avoids the extra time required to send a Request and await for a Data grant is preferred. For this reason, the UBA service is utilized for the video game traffic in the simulations that follow. In the case when a variable packet size model is used for the same video game application, POLL service is used.

2) Delay Tolerant Traffic Model: Applications that generate asynchronous data traffic (e.g., file transfer, email) have been studied extensively. Network measurements on real traffic show high levels of self-similarity, that is, long-range dependence. Heavy tailed distribution functions, such as Pareto, are essential to model such a behavior [16], [22]. Thus, for the generation of data traffic we assume the inter-packet time to be random variables following a Pareto distribution with shape parameter $\theta(1.5 \leq \theta \leq 2)$, and cut-off parameter κ.

In our simulations, we will use an average inter-packet time of 200 milliseconds. Parameter θ values of 1.6, 1.8, and 2.0 will be used in order to observe any effects it may have on the calculated metrics.

In typical data traffic models, the packet length follows a truncated geometric distribution with average packet length of approximately 1,000 bytes. Truncation occurs at the maximum and minimum MAC PDU sizes allowable by the system being simulated. In our case we use a geometrically distributed packet length with a mean of 1024 bytes. The distribution is truncated below and above at 64 and 1,518 bytes, respectively, as given by DOCSIS's minimum and maximum PDU sizes. With these values of the mean inter-packet time and packet size, the offered load from CMs supporting data traffic applications is approximately 41 kbps per CM.

C. System Parameters

Simulations were run for approximately 900 simulated seconds, with performance metrics gathered after the initial 30 seconds. In the simulation, each CM has only one service flow that can be either of the delay sensitive (also called real-time) or delay tolerant type.

The CMs are all assumed to be at a distance of 50 kilometers from the CMTS. Assuming perfect ranging and synchronization, the MAP message should be transmitted so that it is received by all CMs before the start of the MAP message's first described mini-slot. For the assumed propagation delay of 5 $\mu\text{secs/Km}$, a MAP must be transmitted at least 0.25 milliseconds, or 10 mini-slots, before the start of the first described mini-slot (assuming negligible processing delay at the CM).

We assume 0 bytes for the header of the MAC frames transmitted by the CM. This takes into account that the bandwidth allocation strategy used is independent of the MAC header. The bandwidth utilization computation is only concerned with the number of mini-slots (request or CMS) used for signaling out of the total number of mini-slots used per upstream transmitted packet. Similarly, we assume that the maximum protocol data unit (PDU) received by the CM for transmission is 1518 bytes long, and the minimum has a size of 64 bytes. The PDU is the payload of the MAC frame transmitted by MAC protocol.

TABLE I
SIMULATED SYSTEM PARAMETERS

Parameter	Value	Units
MAC Header	0	bytes
Mini-slot duration	0.025	milliseconds
Upstream data rate	2,560	Kbps
Propagation delay	5	$\mu sec/Km$
CMTS to CM distance	50	Kms
MAP max described mini-slots	1,800	mini-slots
MAP max no. of IEs	100	IEs/MAP
MAP max grant size	255	mini-slots
Max PDU size	1,518	bytes
Min PDU size	64	bytes
Mini-slot size	8	bytes
Request message size	1	mini-slot

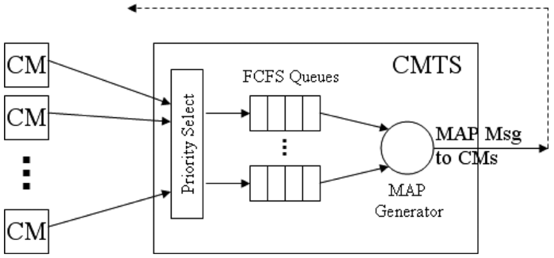


Fig. 5. Contention-based upstream channel scheduler.

At the CMTS, the MAP messages transmitted downstream can describe up to a maximum number of upstream mini-slots (*MAXSLOTS*). This limit is equal to 1,800. Similarly, they can have a given maximum of information elements (IE). This limit (also called *MAXIE*) is 100. In the MAP message, each grant (Data or Request) is limited to a size of 255 mini-slots or less. The values for these and other simulation parameters are shown in Table I. These parameter values were kept constant throughout a simulation run, and were the same for all simulations.

D. Reference Bandwidth Allocation Scheme

We defined a reference bandwidth allocation-medium access scheme that works with the time division multiplexing scheme of the upstream channel. The reference scheme uses CMSs and piggybacking in order to allow CMs to transmit request messages to the CMTS.

The structure of the CMTS scheduler for the reference protocol is shown in Fig. 5. It has two different components: first-come-first-serve (FCFS) queues, and a MAP Generator.

The FCFS queues store the requests messages sent by the CMs. These request messages may arrive at the CMTS in a CMS or piggybacked in a Data slot. There is one FCFS queue for each priority level supported by the system. In our study, two priorities levels are supported. Requests from applications that are delay sensitive are stored in priority 0 queue (high priority). Those arriving from delay tolerant applications use priority 1 queue (low priority). In this scheme, if only one application type is present, then only one queue is used.

The MAP Generator assembles the MAP messages that describe the use of the upstream mini-slots. It uses a simple prioritization process. Grants corresponding to high priority requests are served first, followed by grants for the request messages in the low priority FCFS queue. The MAP Generator serves the received requests without using concatenation or fragmentation. It always grants the number of mini-slots requested by the CM in the request message. Once all the FCFS queues are empty, the MAP generator fills the rest of the MAP with CMS slots (one IE per CMS) until it reaches MAXSLOTS or MAXIE. A MAP message is completed and transmitted when either MAXSLOTS or MAXIE is reached.

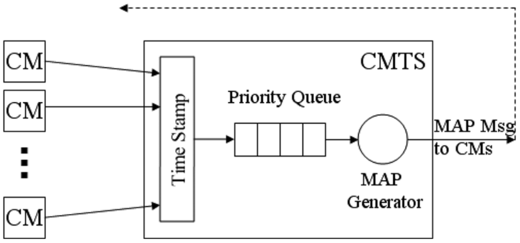


Fig. 6. TraBA based upstream channel scheduler using UBA.

V. Performance Evaluation-Real Time Application Using UBA

In this section we describe the CMTS structure used to implement the TraBA approach for the real time traffic applications as described in previous sections. We will employ the UBA service. An example using POLL service will be discussed in a later section.

In UBA, automatic Data grants of fixed length are provided in such a way that the average packet delay requirement is met. It uses the TraBA approach to decide the Data grant times and allows for piggybacking of requests in Data slots. The following sections show the CMTS structure employed as well as simulation results.

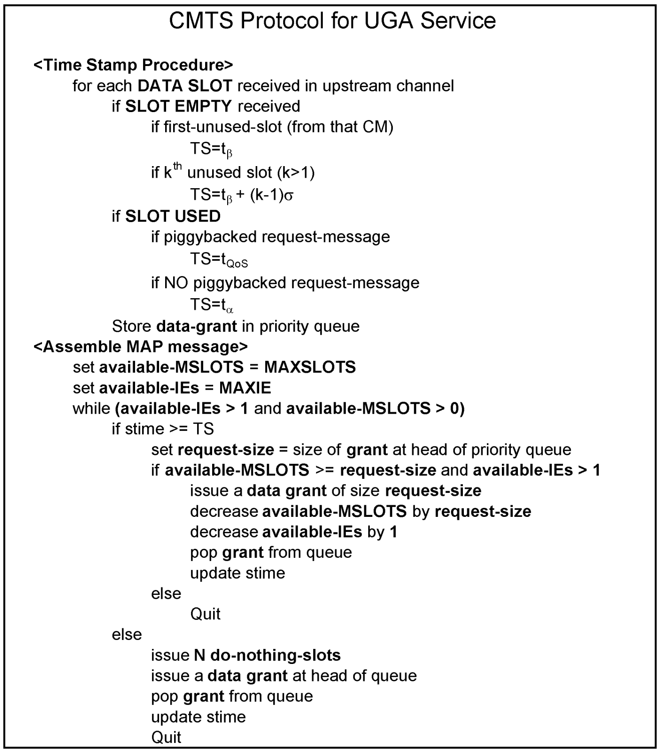
A. CMTS Scheduler Structure and Protocols

Fig. 6 shows the structure of the CMTS scheduler used to provide real time service with the UBA service. It has three components: a time-stamping mechanism, a priority queue mechanism, and a MAP Generator.

As upstream data Data slots (packet-carrying or empty) are received at the CMTS, the time-stamping mechanism computes the time-stamp of the next Data grant to be issued to the CM to which the received Data slot was assigned. The grant is stored in the *priority queue* in increasing order of time-stamps (the head of the queue has the nearest time-stamp). The time-stamps correspond to the system time at which the granted CM transmission must start.

The *MAP Generator* generates the MAP messages carrying the bandwidth grants to the CMs. It serves the received requests without using concatenation or fragmentation, so it always grants the number of mini-slots necessary to transmit the fixed size packets.

At the beginning of the MAP generation process, the MAP Generator serves the grant at the head of the queue first. If the first mini-slot to be described by the MAP starts at a time earlier than the time-stamp of the first request in the queue, the MAP Generator fills the gap with *empty mini-slots*, or mini-slots with no particular use assigned to them. As we will see later, these mini-slots could be used to provide other services, such



```
CMTS Protocol for UGA Service

<Time Stamp Procedure>
    for each DATA SLOT received in upstream channel
        if SLOT EMPTY received
            if first-unused-slot (from that CM)
                TS=$t_\beta$
            if k$^{th}$ unused slot (k>1)
                TS=$t_\beta$ + (k-1)$\sigma$
        if SLOT USED
            if piggybacked request-message
                TS=$t_{QoS}$
            if NO piggybacked request-message
                TS=$t_\alpha$
        Store data-grant in priority queue
<Assemble MAP message>
    set available-MSLOTS = MAXSLOTS
    set available-IEs = MAXIE
    while (available-IEs > 1 and available-MSLOTS > 0)
        if stime >= TS
            set request-size = size of grant at head of priority queue
            if available-MSLOTS >= request-size and available-IEs > 1
                issue a data grant of size request-size
                decrease available-MSLOTS by request-size
                decrease available-IEs by 1
                pop grant from queue
                update stime
            else
                Quit
        else
            issue N do-nothing-slots
            issue a data grant at head of queue
            pop grant from queue
            update stime
            Quit
```

Fig. 7. Pseudo-code of the CMTS time stamping and MAP assembly routines for UBA service.

as best-effort, in the case of a mixed services system. The MAP generator keeps serving data grants in the priority queue until the MAP message is complete (either reached MAXSLOTS or MAXIE). A MAP message may also be considered complete when there is a gap between the finish time of a grant (the last used mini-slot), and the time of the next grant in the queue.

Fig. 7 shows the protocols at the CMTS for the time stamps mechanism and the map generation process. The time stamps generation at the scheduler follows the procedure described in the following paragraphs. In the figure, the variable *TS* used in the time-stamp procedure is the time stamp assigned to grants inserted into the priority queue. In the MAP generation process, the same variable name TS is used for the time stamp of the grant at the head of the priority queue. The variable *stime* represents the time of the next mini-slot to be described by the MAP message being assembled. σ is the standard deviation of the source traffic's known inter-packet time distribution. Assuming that T_{tx} is a packet's transmission time, and t_0 is the time carried in the PAT field of the last received packet (either its own arrival time to CM, or arrival time of next packet in CM's queue), the times for the next Data grant are calculated as:

$$t_{QoS} = t_0 + D_{QoS} - T_{tx}, \tag{8}$$

$$t_\alpha = t_0 + F^{-1}(\alpha), \tag{9}$$

and

$$t_\beta = t_0 + F^{-1}(\beta) \tag{10}$$

where $F^{-1}()$ is the inverse of the inter-packet time CDF. α and β are the algorithm parameters. t_0 is the received packet's arrival time to the CM in (9) and (10), and the arrival time of the next packet in queue in (8) (Request piggybacking case).

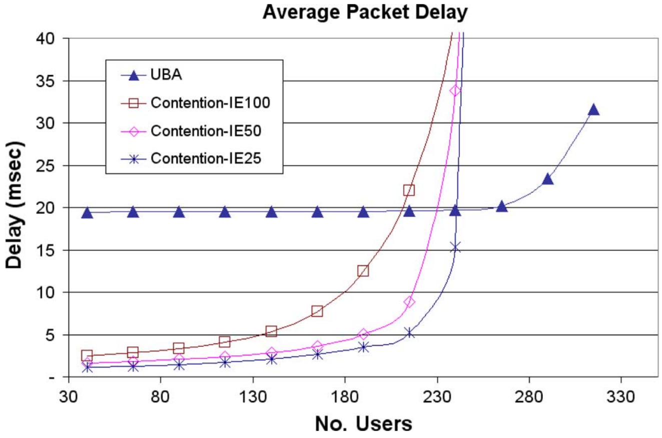


Fig. 8. Packet delay for contention-based and TraBA schemes.

B. α and β Calculation

As a first step for the simulation, we need to find the values of the TraBA algorithm parameters α and β that offer the desired packet delay D_{QoS}. The procedure to find the α and β was given in Section III-B. The following paragraphs show the computation of the target delay used to find the parameter values.

For UBA service, the total delay is given by $T_{total} = T_{TraBA} + T_{tx}$, where (T_{TraBA}) is the delay due to the TraBA approach (the time it takes to schedule the packet transmission). In order to achieve the target delay we must have $T_{total} = D_{QoS}$. Therefore the T_{TraBA} delay must satisfy

$$T_{TraBA} = D_{QoS} - T_{tx}. \tag{11}$$

Once we have determined the value of T_{TraBA}, we use the procedure presented in Section III.B to obtain the values of α and β that satisfy $D_{NQ} = T_{TraBA}$. In the current case, we are assuming packets that are 64 bytes long. For transmission rate of 2,560 kbps (see Table I) the transmission time for a packet is $T_{tx} = 0.2$ milliseconds, thus, assuming $D_{QoS} = 20$ milliseconds the TraBA scheme should achieve $D_{NQ} = T_{TraBA} = 19.8$ milliseconds, approximately. Analysis shows [20] that $T_{TraBA} = 19.8$ milliseconds is achieved by $\beta = 0.997$. The corresponding value of α is 0.75.

C. Simulation Results

Fig. 8 shows the average delay plotted against the number of users for the UBA service and the contention-based reference approach. The delay levels achieved by the UBA service agree with the target delay $D_{QoS} = 20$ milliseconds. As the load is increased the delay levels remain constant up to approximately 260 users. The delay starts increasing at the load that utilizes 100% of the system bandwidth (see for [20] load calculations). At this load, the data grants are issued at a time later than that intended by the MAP Generator, resulting in an increased delay.

Fig. 8 also shows the delay levels obtained when using the bandwidth allocation approach based on contention. In the figure the values $\text{MAXIE} = 25$, 50, and 100 are used for the reference scheme. For low system loads, the reference scheme

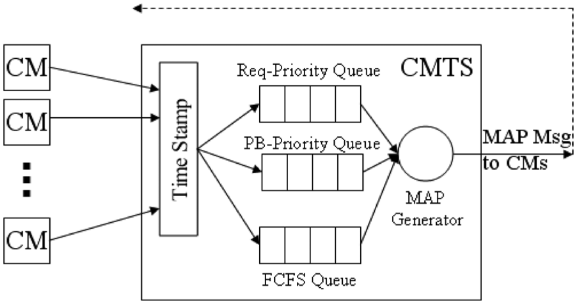


Fig. 9. TraBA based upstream channel scheduler using POLL.

achieves a considerably lower delay than that achieved by TraBA. However, as the load increases, the achieved average delay increases due to the reduction in the number of contention mini-slots issued, and the increase in the number of collisions in these mini-slots.

If we define capacity as the number of users supported while complying with the 20 milliseconds delay, we observe that the largest capacity obtained using the contention based approach is approximately 240 users. It is achieved by the contention scheme using $\mathrm{MAXIE} = 25$. For $\mathrm{MAXIE} = 50$ and 100, the achieved capacities are 220 and 200 users, respectively. Thus, comparing the TraBA and contention-based approaches, the TraBA approach achieves 9%, 18%, and 30% higher capacity, approximately.

VI. Performance Evaluation-Real Time Application Using Poll

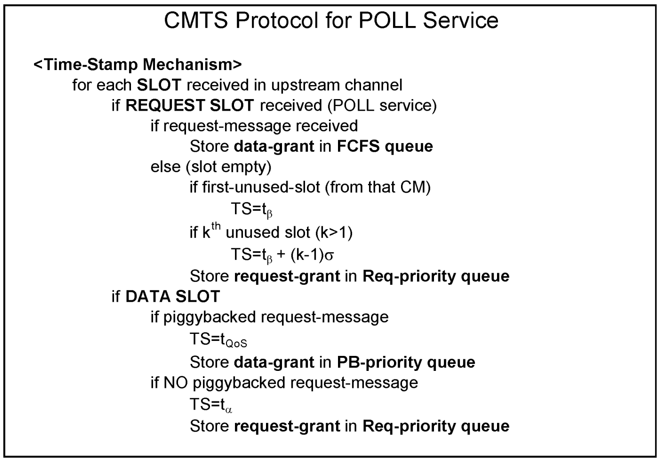
The POLL service is a extension of DOCSIS's rtPS service that uses TraBA approach. We will use POLL to service an application with Gamma inter-packet time having a mean $\mu = 50$ milliseconds, and $\sigma = 10$ milliseconds. We assume that packets are of variable size, with 64, 128, and 192 bytes long packets generated with equal probability.

In POLL Request grants are assigned to the CMs who use it to send Request messages when they have a packet to transmit. These grants are provided in a way that the real time need of the application being serviced is met. POLL allows for piggybacking of requests in Data slots, and uses the TraBA approach to decide the Request grant times.

A. CMTS Scheduler Structure and Protocols

Fig. 9 shows the structure of the CMTS scheduler used to provide real time service using the POLL. It replaces the single priority queue used for the UBA case, with three new components: two priority queues, one for automatically generated requests and one for data grants corresponding to piggybacked requests, and an FCFS queue.

The *Request priority queue* is used to store the time-stamped Request grants (poll messages) to be automatically issued by the scheduler. The *Piggybacked Request priority queue* (PB) stores the Data grants associated with piggybacked requests. Both queues store the grants in increasing order of the time-stamps. As upstream slots assigned to a CM are received at the CMTS (either empty or carrying information), the scheduler (time-stamp mechanism) computes the time-stamp of the next Data or Request grant to be issued for that CM and stores it in the corresponding priority queue. The grants are time-stamped so that the application's delay constraints are met. In case a Request message is received from a CM in a Request slot, the corresponding Data grant is stored by the scheduler in the *FCFS queue*.



```
CMTS Protocol for POLL Service

<Time-Stamp Mechanism>
    for each SLOT received in upstream channel
        if REQUEST SLOT received (POLL service)
            if request-message received
                Store data-grant in FCFS queue
            else (slot empty)
                if first-unused-slot (from that CM)
                    TS=$t_\beta$
                if $k^{th}$ unused slot (k>1)
                    TS=$t_\beta$ + (k-1)$\sigma$
                Store request-grant in Req-priority queue
        if DATA SLOT
            if piggybacked request-message
                TS=$t_{QoS}$
                Store data-grant in PB-priority queue
            if NO piggybacked request-message
                TS=$t_\alpha$
                Store request-grant in Req-priority queue
```

Fig. 10. Pseudo-code for the CMTS time stamping routine for POLL service.

The *MAP Generator* works by checking the different queues in a round robin fashion. For every data grant issued from the FCFS queue, it checks the time-stamp of the data grant at the top of the PB priority queue. If the next mini-slot to be described by the MAP starts at a time equal or later than the PB time-stamp, the MAP Generator adds the corresponding grant to the MAP. If the mini-slot starting time is earlier than that of the time-stamp, no grant is issued from the PB priority queue. After the PB priority queue, the same process is repeated with the Request priority queue which stores time-stamped automatic request grants. The FCFS queue is serviced next.

As in the reference case, the MAP generator keeps serving grants in the queues until the MAP message is complete (either reached MAXSLOTS or MAXIE). A MAP message may also be considered complete if the FCFS is empty, and there is a gap between the finish time of the last grant described in the MAP being assembled, and the time stamp of the next grant both priority queues.

The grant time-stamping and storing process as well as the procedure followed by the CMTS scheduler for generating the MAP messages are shown in Figs. 10 and 11.

In the time-stamp procedure shown in Fig. 10, TS is the time stamp assigned to grants inserted into the priority queues. Similarly, in the MAP generation procedure shown in Fig. 11 the variables TS_{PB} and TS_{Req} are the time stamp of the grants at the top of the PB and Request priority queues, respectively. The variable *stime* represents the time of the next mini-slot to be described by the MAP message being assembled. σ is the standard deviation of the source traffic's known inter-packet time distribution.

As shown in the figure, the MAP generation procedure gives similar priority to all queues. However, in practice, the effect of the Request grants on Data grant delays is very small. For not expired time-stamps at the head of the priority queues, Data grants will be served from the FCFS queue (if it is not empty). If the time stamps at the head of the priority queues are expired,

```
CMTS Protocol for POLL Service (cont.)

<Assemble MAP message>
    set available-MSLOTS = MAXSLOTS
    set available-IEs = MAXIE
    while (available-IEs > 1 and available-MSLOTS > 0)
        if FCFS not empty
            set request-size = size of grant at head of FCFS queue
            if available-MSLOTS >= request-size and available-IEs > 1
                issue a data grant of size request-size
                decrease available-MSLOTS by request-size
                decrease available-IEs by 1
                pop grant from queue
                update stime
            else
                <Wait another MAP message>
        if stime >= $TS_{PB}$
            set request-size = size of grant at head of PB-priority queue
            if available-MSLOTS >= request-size and available-IEs > 1
                issue a data or request grant of size request-size
                decrease available-MSLOTS by request-size
                decrease available-IEs by 1
                pop grant from queue
                update stime
            else
                <Wait another MAP message>
        if stime >= $TS_{Req}$
            set request-size = size of grant at head of Req-priority queue
            if available-MSLOTS >= request-size and available-IEs > 1
                issue a data or request grant of size request-size
                decrease available-MSLOTS by request-size
                decrease available-IEs by 1
                pop grant from queue
                update stime
            else
                <Wait another MAP message>
        if no grant to issue and no grants issued so far
            issue N do-nothing-slots
            Quit
```

Fig. 11. Pseudo-code for the CMTS MAP assembly routine for POLL service.

they will have equal priority as the FCFS queue, as the round robin mechanism will ensure that one grant is issued from each one of the queues. However, we note that Requests grants are 1 mini-slot long (0.0025 milliseconds), and will not introduce a significant delay to Data grants in the FCFS queue.

For the calculation of the grant times t_{QoS}, t_α, and t_β we note that in the POLL case the total delay experienced by a packet arriving to the head of the CM transmission queue is given by

$$T_{total} = T_{TraBA} + T_{sched} + T_{tx}, \tag{12}$$

where T_{TraBA} is the time it takes the CM to transmit the Request message (due to the scheduling of Request grant issued in POLL service), T_{sched} includes the round trip, processing, and Data grant queuing delay at the CMTS, and T_{tx} is the packet transmission time. Thus, for an assumed value of T_{sched} and T_{tx}, α and β are given by the percentiles of F() that make $T_{TraBA} = D_{QoS} - T_{tx} - T_{sched}$.

As before, the values t_α, and t_β are calculated using (9), (10), and t_{QoS} is givey by

$$t_{QoS} = t_0 + D_{QoS} - T_{sched} - T_{tx}, \tag{13}$$

where t_0, the time carried in the PAT field of the last received packet, is the arrival time of the next packet in queue at the CM (in (9) and (10), it is the received packet's arrival time).

B. α and β Calculation

The total delay experienced by a packet in POLL is given by (12). Following the terminology used in the UBA case, T_{TraBA} is the time it takes the CM to receive a Request grant used to transmit a Request message. Thus, in the POLL case, the delay due to TraBA must satisfy

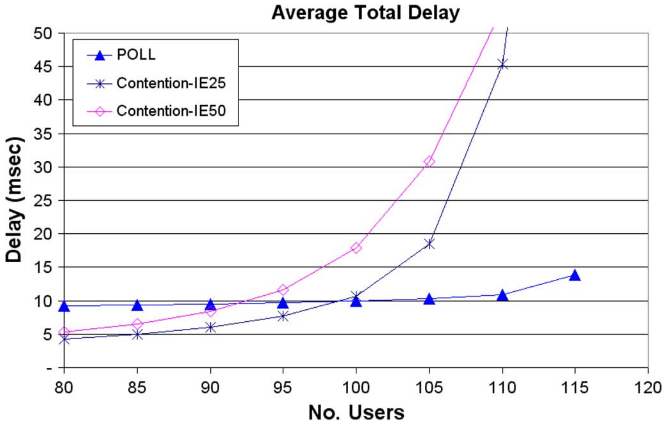


Fig. 12. Packet delay for contention-based and POLL schemes.

$$T_{TraBA} = D_{QoS} - T_{tx} - T_{sched} \tag{14}$$

where as before T_{sched} includes the round trip, processing, and bandwidth grant queuing delay at the CMTS, and T_{tx} is the packet transmission time.

The values for α and β can be calculated using the procedure presented in Section III-B to obtain the values of that satisfy $D_{NQ} = T_{TraBA}$. In this case, the target delay of $D_{QoS} = 10$ milliseconds is assumed. For the transmission rate of 2,560 kbps assumed, a 192 bytes packet is transmitted in 0.6 milliseconds, approximately. In addition, the round trip delay is approximately 0.5 milliseconds. If we include the processing delay and queuing that might take place at the CMTS, a value $T_{tx} + T_{sched} = 2$ milliseconds is a reasonable assumption, yielding $T_{TraBA} = 8$ milliseconds. Note that this values is just an approximation to the transmission and processing delay, and is not an accurate specification of the delay due to system load conditions. The system load may affect the packet delays in two different ways: late Request grants issuance, and larger Data grant issuance delays in the FCFS queue due to congestion. The simulation results will show the effects of the load on the achieved packet delays.

Analysis shows [20] that the target delay $T_{TraBA} = D_{QoS} - T_{tx} - T_{sched} = 8$ milliseconds is achieved by $\beta = 0.92$. The corresponding value of α is 0.55.

C. Simulation Results

Fig. 12 shows the average delay plotted against the number of users for the POLL service and the contention-based service. In both cases the user's traffic model is that described at the beginning of Section VI. The POLL service employs $\alpha = 0.55$ and $\beta = 0.92$ as previously suggested. The delay levels achieved by the POLL service agree with the target delay $D_{QoS} = 10$ milliseconds. As the load is increased the delay levels remain close to 10 milliseconds up to approximately 110 users. This is the load at which 100% of the system bandwidth is utilized for the target delay levels. The figure also shows the delay levels obtained by the reference contention based approach. In the figure two values of the parameter MAXIE are

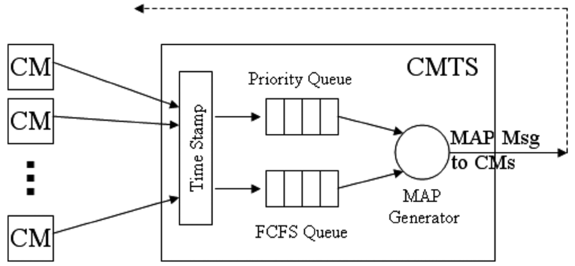


Fig. 13. TraBA upstream channel scheduler for the mixed services case.

used for the reference scheme. The value $\mathrm{MAXIE} = 25$ results in the best performance for the reference scheme. Thus, it is the most significant for the purpose of comparing to proposed scheme. The value of $\mathrm{MAXIE} = 50$ shows how the performance degrades as MAXIE is increased. Using $\mathrm{MAXIE} = 100$ (not shown in the figure) results in the worst performance.

For low system loads the contention-based scheme achieves a considerably lower delay than that achieved by POLL. However, as the load increases, the achieved average delay increases as well. The largest capacity obtained using the contention based approach is approximately 100 users. It is achieved by the contention scheme using MAP messages using $\mathrm{MAXIE} = 25$. For $\mathrm{MAXIE} = 50$, the achieved capacities is 92 users. Thus, comparing the TraBA-base POLL and contention-based approaches, the POLL approach achieves 10%, and 20% higher capacity, approximately.

VII. Mixed Real Time and Delay Tolerant Services Case

In this section we evaluate the performance of a system that employs the TraBA approach to simultaneously service real-time and delay tolerant applications. As in the real time only service case, we compare the performance of the proposed approach with that of the reference contention based system offering the same services under the same conditions. The performance metrics to be used are those explained in Section IV-A: packet delay for delay sensitive applications, real-time network capacity, total throughput, and average throughput per CM for delay tolerant applications.

For the real-time application, we will choose the real-time traffic model having a Gamma distributed inter-packet time with an average of 65 milliseconds and a standard deviation of 15 milliseconds. We will use a fixed packet size of 64 bytes as before. This application will use the UBA service. Following Section IV-B-2 the delay tolerant traffic model has a Pareto inter-packet time with a 200 milliseconds mean and $\theta = 1.8$, and a geometrically distributed packet length with a mean of 1024 bytes. In this case, we will use POLL service.

A. CMTS Upstream Scheduler Structure

The CMTS scheduler structure employed to simultaneously service real time and delay tolerant applications is shown in Fig. 13. It contains a mechanism for time-stamping, one priority and one FCFS queue, and a MAP Generator.

The *priority queue* is used to store the time-stamped Data grants issued by the scheduler for the real time application. These data grants may be the automatic grants issued by the scheduler, or those generated from Request messages piggybacked in Data slots. The priority queue also stores the Request grants automatically issued by the scheduler in the POLL service utilized to serve the delay tolerant application. The priority queues stores the grants in increasing order of the time-stamp. The *FCFS queue* stores the Data grants associated with the delay tolerant service. These grants are generated from the Request messages received in request slots as well as from those piggybacked in Data slots.

```
CMTS Protocol for Mixed Services

<Time-Stamp Procedure>
    for each SLOT received in upstream channel
        if REQUEST SLOT received (POLL service)
            if request-message received
                Store data-grant in FCFS queue
            else (slot empty)
                if first-unused-slot (from that CM)
                    TS=$t_\beta$
                if k$^{th}$ unused slot (k>1)
                    TS=$t_\beta$ + (k-1)$\sigma$
                Store request-grant in priority queue
        if DATA SLOT
            if piggybacked request-message
                if REAL-TIME service (using UGA)
                    TS=$t_{QoS}$
                    Store data-grant in priority queue
                if DELAY-TOLERANT service (using POLL)
                    Store data-grant in FCFS queue
            if NO piggybacked request-message
                if REAL-TIME service (using UGA)
                    TS=$t_\alpha$
                    Store data-grant in priority queue
                if DELAY-TOLERANT service (using POLL)
                    TS=$t_\alpha$
                    Store request-grant in priority queue
```

Fig. 14. Pseudo-code of CMTS time stamping routine for the mixed services case.

```
CMTS Protocol for Mixed Services (Cont.)

<Assemble MAP message>
    set available-MSLOTS = MAXSLOTS
    set available-IEs = MAXIE
    while (available-IEs > 1 and available-MSLOTS > 0)
        if stime >= TS
            set request-size = size of grant at head of priority queue
            if available-MSLOTS >= request-size and available-IEs > 1
                issue a data or request grant of size request-size
                decrease available-MSLOTS by request-size
                decrease available-IEs by 1
                pop grant from queue
                update stime
            else
                <Wait another MAP message>
        else if FCFS not empty
            set request-size = size of grant at head of FCFS queue
            if available-MSLOTS >= request-size and available-IEs > 1
                issue a data grant of size request-size
                decrease available-MSLOTS by request-size
                decrease available-IEs by 1
                pop grant from queue
                update stime
            else
                <Wait another MAP message>
        else
            issue N do-nothing-slots
```

Fig. 15. Pseudo-code of CMTS MAP assembly routine for the mixed services case.

As upstream slots are received at the CMTS from a CM (either empty or carrying information), the scheduler computes the time-stamp of the next Data or Request grant to be issued for that CM and stores it in the priority queue (for real time applications). Those corresponding to the delay tolerant case are not time-stamped, and are placed in the FCFS queue.

Figs. 14 and 15 show the procedures followed by the CMTS as it receives data or request slots in the upstream channel. Fig. 15 shows the procedure followed by the *MAP Generator* to put together the MAP messages. In the procedures, TS is the time stamp assigned to grants inserted into the priority queue. σ is the standard deviation of the source traffic inter-packet time distribution. The variable *stime* represents the time of the next mini-slot to be described by the MAP message being assembled. MAXSLOTS and MAXIE are as defined in Section IV-C. The values t_{QoS}, t_{α}, and t_{β} are calculated in the same way as for the UBA and POLL cases shown before.

From the procedure we note that priority is given to the grants (Data or Request) stored in the priority queue. In this queue, the Data grants are those corresponding to real time services, thus, justly, they have priority over those for delay tolerant applications in the FCFS queue. The priority queue also stores Request grants issued to delay tolerant application. In normal conditions, these grants would have the same priority as real time data grants. These grants are only one mini-slot long, and will not greatly affect the performance of the real time service. In overloaded conditions, the lower priority delay tolerant Data grants in the FCFS queue will accumulate. Due to the CMTS protocol, the CMs starving service will not receive Request grants, and most of the bandwidth will be used for the real time service.

B. Simulation Set Up

In evaluating the performance of the system servicing both applications simultaneously we will consider two scenarios. In the first scenario, a fixed total load of approximately 2,060 kbps or 80% system bandwidth is offered, but the percentage of the load due to each application is varied, and the effects of this variation on the performance observed. This scenario will allow us to observe the effect of each application's load in the performance metrics. In the second scenario we will keep the delay tolerant data traffic total offered load fixed, and increase the system load by increasing the load due the real time application. It will also show the behavior of the system as its maximum bandwidth is reached.

As before, the first step for the simulation is to find the values of the TraBA algorithm parameters α and β that we need to use with each one of the services. For the real time application using UBA service, we will use the same 20 milliseconds constraint used in previous sections. For the delay tolerant (data) application, using POLL service, we will have no delay constraint. Instead we will choose a delay lower than the average inter packet time so that we prevent queues to build up at the CMs transmission queues.

The procedure to select the values of α and β was described in previous sections for both, UBA and POLL services. Following Section V-B, we will use $\alpha = 0.75$ and $\beta = 0.997$ to provide the same delay levels for the real time application. For the data traffic case using the POLL service, $\beta = 99\%$, $\alpha = 90\%$ give a delay of approximately 120 milliseconds, which is considerably lower than the mean inter-packet time. These are the values of α and β we will use throughout our simulation.

C. Simulation Results

For the first scenario, in which we keep the total system load constant, the number of real time CMs was varied from 43 to 204, while the number of data traffic CMs varied from 42 down to 30. The results are shown in Figs. 16 and 17, which plot the average delay and the throughput per CM against L, the percentage of the total offered load due to real time applications. Fig. 16 shows the average delay achieved by the TraBA-based and contention-based services for CMs with real time applications. For the TraBA-based service, the average delay remains at the target delay of 20 milliseconds regardless of the value of L. For the contention-based service, the delay levels are 25% to 100% higher. Moreover, the delay increases with increasing L. This increase is due to the increase in the number of lower traffic real time CMs. For the same 80% system load, more CMs are active, which increases the collisions in the contention slots and increases the delay. Note that the contention-based approach is using MAXIE = 100 IEs. Similar trends are observed for the case of MAXIE = 25.

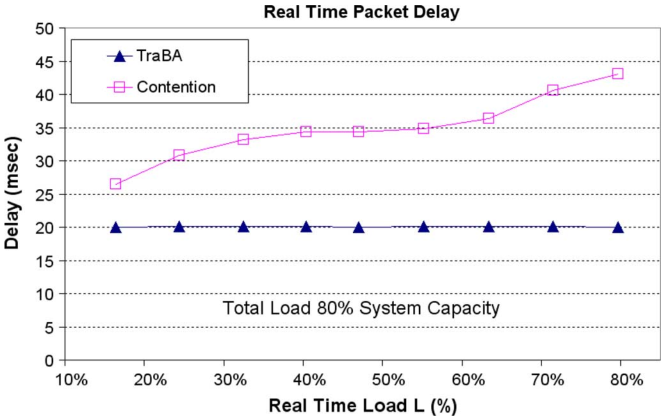


Fig. 16. Packet delay as a function of the parameter L.

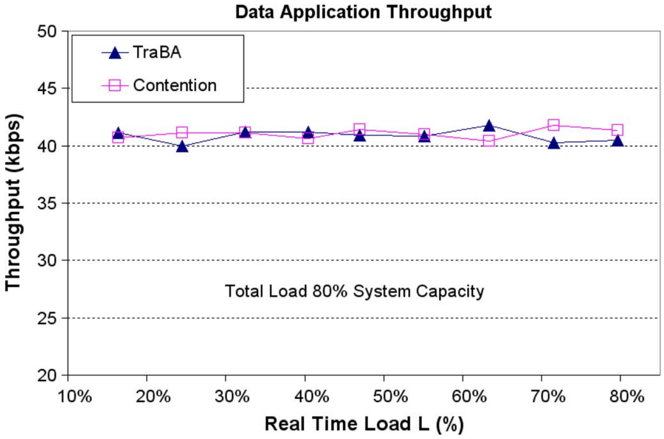


Fig. 17. Average CM throughput as a function of the parameter L.

For data traffic applications, Fig. 17 shows the average CM throughput achieved against L. As we observe in both cases the data throughput is maintained at the same levels (41 kbps). This is because neither of the bandwidth allocation schemes is fully utilizing the system bandwidth, thus it can fully comply with the demand of the serviced CMs.

In the second scenario, there are 30 data traffic CMs, each offering a 41 kbps load each, offering a total data traffic of 1,230 kbps (48% of the total system bandwidth). In this scenario we keep fixed the data traffic load, and vary the real time offered load. The number of real time CMs is varied from 40 to 180, each offering a load of 7.88 kbps. Thus, the effective total load

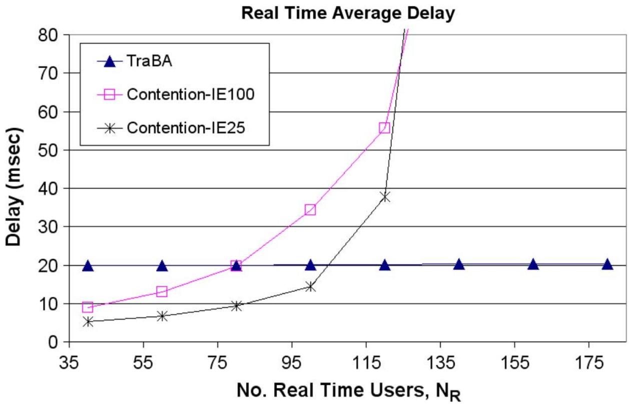


Fig. 18. Packet delay as a function of the number of real time users.

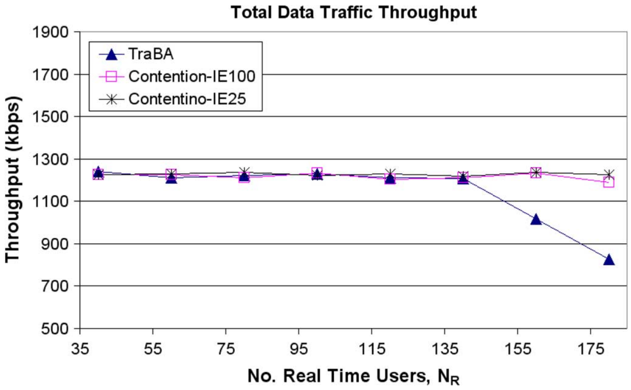


Fig. 19. Data throughput as a function of the number of real time users.

on the system is varied from 60% to 103.5%, approximately. The results are shown in Figs. 18 and 19 for both the TraBA-based and contention based approaches.

In Fig. 18 we observe that the TraBA approach achieves the target delay for all loads. In the contention-based approach the real time user delay increases beyond the target delay for more than 80 real time users, for the $\mathrm{MAXIE} = 100$ case, and 110 users for $\mathrm{MAXIE} = 25$. The increase in delay is due to the reduction number of contention mini-slots available to transmit request messages as the load is increased. The CMTS protocol gives priority to the data grants stored in the FCFS queues and, as a consequence, as the load is increased fewer mini-slots are dedicated to contention. These used values of MAXIE show the better $(\mathrm{MAXIE} = 25)$ and worst $(\mathrm{MAXIE} = 100)$ performance of the contention scheme.

As shown in Fig. 19, the delay compliance of the TraBA approach comes at the detriment of the data throughput, which decreases after the load reaches 140 users, approximately. On the other hand, the contention-based approach maintains the data throughput relatively constant for the offered loads. In comparing the performance of the TraBA approach against that of the reference approach we note that both schemes achieve the same data throughput levels up to 140 real time users. However, only TraBA achieves the target delay of the real time application. Thus, for similar throughput levels, we can say that TraBA achieved a 27%, and 75% increase in real time user capacity over the contention approach using $\mathrm{MAXIE} = 25$, and 100, respectively.

VIII. Conclusions

In this paper we presented the methodology to implement the new bandwidth allocation approach described in Section III within the context of a DOCSIS cable network. TraBA, the proposed bandwidth allocation approach, is based on the source traffic model, and QoS requirements of the serviced applications.

We considered a single service scenario where a single type of real time user application is serviced. We also showed the methodology to support a mixed service scenario where the real time user application and a delay tolerant application are serviced simultaneously.

The new approach was shown to offer increased capacity when compared to the contention based approach. For the single service case (real time application), a capacity increase in the range 10% to 30% was obtained, depending on the parameters used for the contention based approach.

For the case of mixed services, the TraBA approach was shown to comply with the target delays for very high system loads. This compliance comes at the detriment of the data traffic user class, which sees a reduction in the total throughput. In particular, for the fixed load (80% system bandwidth) case, the TraBA-based approach was shown to handle well variations in traffic conditions. As the percentage of the load due to one application was varied, the TraBA approach maintained the target delay levels as well as the total data throughput. On the other hand, the contention-based approach showed variations in the delay as the real time load percentage was varied. At this load, it did not comply with the target delay levels. For the variable load scenario studied, TraBA was shown to achieve a 27%, and 75% increase in real time user capacity over the contention approach using $\mathrm{MAXIE} = 25$, and 100, respectively.

In the presented results, one advantage offered by the TraBA approach (both service, UBA and POLL) is the gradual degradation of the delay levels, as opposed to the sudden delay increase shown by the reference contention based-approach.

Lastly, note that in general the TraBa approach is geared towards complying with QoS (delay) constraints, and at the same time making a better utilization of the upstream bandwidth. The approach also takes into account different service levels for different applications. It provides advantages especially at medium to high system loads. For applications that do not have such delay constraints, other approaches (such as contention) can be used satisfactorily.

Appendix A
Abbreviations and Acronyms

BE	best effort
CBR	constant bit rate
CDF	cumulative distribution function
CM	cable modem
CMS	contention mini-slot
CMTS	cable modem termination system
CPR	centralized priority reservation

CRA	contention resolution algorithm
CSMA/CD	carrier sense multiple access with collision detection
DOCSIS	Data-Over-Cable System Interface Specification
FCFS	first-come first-serve
FTP	file transfer protocol
HFC	hybrid fiber/coaxial
IE	information element
IP	Internet Protocol
MAC	medium access control
nrtPS	non-real time polling service
PB	piggybacked
PDF	probability distribution function
PDU	protocol data unit
POLL	statistical polling
QoS	quality of service
RTD	round trip delay
rtPS	real time polling service
TraBA	traffic based bandwidth allocation
UBA	unsolicited bandwidth assignment
UDP	User Datagram protocol
UGS	unsolicited grant service
VBR	variable bit rate

Appendix B
List of Variables

α, β	percentiles of $F(t)$ used for grant assignment
α_{min}	value of α that achieves lowest average packet delay and for a given β
σ	standard deviation of the inter-packet time distribution
a_1	arrival time of a packet to the CM transmission queue
D_{NQ}	delay experienced by packet arriving to CM and transmission queue when there is no queuing (D_{NQ})
D_{QoS}	target packet delay for the serviced application
I_k	kth interval into which time is divided by grant and times t_k
$f(t)$	probability distribution function (PDF) of the and inter-packet time
$F(t)$	cumulative distribution function (CDF) of the and inter-packet time
MAP	Upstream Bandwidth Allocation MAP message
$MAXSLOTS$	maximum number of slots in a MAP
$MAXIE$	maximum number of IEs in a MAP
$P(I_k)$, Δ_k	probability that a_1 is in the kth interval of and time defined by the by grant times t_k
PAT	field in uplink transmitted packets carrying the arrival and time of a packet to CM queue
$\overline{\tau_k}$	conditional average delay experienced by the packet and given that the it arrives in time interval k
$stime$	time of the next mini-slot to be described and in MAP Generator
$t_{\alpha_{min}}$, t_β	scheduling times of the transmission grant issued and at times α_{min} and β of $F(t)$
t_{QoS}	grant time to achieve D_{QoS} packet delay
T_{tx}	packet transmission time
T_{total}	total delay experienced by a transmitted packet
T_{TraBA}	delay experienced by a packet due to TraBA approach and out of the total delay
t_k	scheduling time of the kth transmission grant
TS	time stamp of grants inserted in the priority queues and in CMTS time stamping procedures
TS_{PB}	time stamp of the grant at the top of the and piggybacked-request priority queue
TS_{Req}	time stamp of the grant at the top of the Request and priority queue
T_{sched}	delay to schedule a grant (Data or Request)

References

[1] Cable Television Laboratories, Inc., "Data over cable service interface specifications MAC and upper layer protocols interface specification," CM-SP-MULPIv3.0-I12-100115, Jan. 2010.

[2] Cable Television Laboratories, Inc., "Data over cable service interface specifications physical layer specification," CM-SP-PHYv3.0-I08-090121, Jan. 2009.

[3] M. Droubi *et al.*, "Dynamic bandwidth allocation for the HFC DOCSIS MAC protocol," in *Proc. 9th Int. Conf. Comput. Commun. and Netw.*, Oct. 2000, pp. 54–60.

[4] W.-T. Lee *et al.*, "DOCSIS performance analysis under high traffic conditions in the HFC networks," *IEEE Trans. Broadcast.*, vol. 52, no. 1, pp. 21–30, Mar. 2006.

[5] K.-C. Chang and W. Liao, "The contention behavior of DOCSIS in CATV networks," *IEEE Trans. Broadcast.*, vol. 53, no. 3, pp. 660–669, Sep. 2007.

[6] W. Liao, "The behavior of tcp over DOCSIS-based CATV networks," *IEEE Trans. Commun.*, vol. 54, no. 9, pp. 1633–1642, Sep. 2006.

[7] D. B. Lee *et al.*, "An effective channel control algorithm for integrated iptv services over DOCSIS CATV networks," *IEEE Trans. Broadcast.*, vol. 53, no. 4, pp. 789–796, Dec. 2007.

[8] C. Xiao, "Measured QoS performance of the DOCSIS hybrid-fiber coax cable network," in *Proc. 13th IEEE Workshop Local and Metropolitan Area Netw.*, Apr. 2004, pp. 23–27.

[9] C. Wu and G. Campbell, "Extended DQRAP(XDQRAP) a cable TV protocol functioning as a distributed switch," in *Proc. 1st Int. Workshop Community Netw. Integr. Multimedia Serv. to the Home*, Nagoya, Japan, Jul. 1994, pp. 191–198.

[10] C. Grobicki and J. Ulm, "UniLINK as a media access protocol for community cable TV," in *Proc. 2nd Int. Workshop Community Netw. Integr. Multimedia Serv. to the Home*, Nagoya, Japan, Jun. 1995, pp. 41–48.

[11] J. O. Limb and D. Sala, "A protocol for efficient transfer of data over hybrid fiber/coax systems," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 872–881, Dec. 1997.

[12] R. Rabbat and K. Y. Siu, "QoS support for integrated services over CATV," *IEEE Commun. Mag.*, vol. 37, no. 1, pp. 64–68, Jan. 1999.

[13] M. Hawa and D. W. Petr, "Quality of service scheduling in cable and broadband wireless access systems," in *Proc. IEEE 10th Int. Workshop Quality of Service*, May 2002, pp. 247–255.

[14] W. Kuo, “Improved priority access, bandwidth allocation and traffic scheduling for DOCSIS cable networks,” *IEEE Trans. Broadcast.*, vol. 49, no. 4, pp. 371–382, Dec. 2003.

[15] C. Heyaime-Duverge and V. K. Prabhu, “Modeling action and strategy internet-games traffic,” in *IEEE VTC*, May 2002, vol. 3, pp. 1405–1409.

[16] Z. Hadzi-Velkov and L. Gavrilovska, “Performance of the IEEE 802.11 wireless LANs under the influence of hidden terminals and Pareto distributed packet traffic,” in *Proc. 4th Int. Conf. Telecommun. Modern Satellite Cable and Broadcast. Serv.*, Oct. 1999, vol. 1.

[17] P. T. G. *et al.*, “Source traffic modeling of wireless applications,” *AEU—Int. J. Electron. Commun.*, vol. 55, no. 1, pp. 27–36, 2001.

[18] A. R.-L. *et al.*, “A page-oriented www traffic model for wireless system simulations,” in *Proc. 16th Int. Teletraffic Congr. (ITC’16)*, Jun. 1999, pp. 1271–1280.

[19] M. S. Borella, “Source models of network game traffic,” *Comput. Commun.*, vol. 23, no. 4, pp. 403–410, Feb. 2000.

[20] C. Heyaime-Duverge, “Bandwidth allocation in multi-services access networks using statistical traffic models,” Doctoral Dissertation, Univ. of Texas at Arlington, Arlington, TX, Dec. 2003.

[21] *Universal Mobile Telecommunications System (UMTS); LTE; Policy and Charging Control Architecture*, ETSI TS 123 203 V9.4.0, ETSI, Mar. 2010.

[22] W. M. Yin and Y. D. Lin, “Statistically optimized minislot allocation for initial and collision resolution in hybrid fiber coaxial networks,” *IEEE J. Sel. Areas Commun.*, vol. 18, no. 9, pp. 1764–1773, Sep. 2000.

[23] D. Sala *et al.*, “Adaptive control mechanism for cable modem MAC protocols,” in *Proc. 17th Annu. Joint Conf. IEEE Comput. and Commun. Soc.*, Apr. 1998, vol. 3, pp. 1392–1399.

[24] V. Sdralia, “Performance characterization of the MCNS DOCSIS 1.0 CATV protocol with prioritised first come first served scheduling,” *IEEE Trans. Broadcast.*, vol. 45, no. 2, pp. 196–205, Jun. 1999.

[25] J.-L. C. Wu, “Traffic modeling and bandwidth allocation for MPEG video sources in ATM networks,” *IEEE Globecom*, vol. 3, pp. 2237–2241, Nov. 1995.

[26] W. Liao and H.-J. Ju, “Adaptive slot allocation in DOCSIS-based CATV networks,” *IEEE Trans. Multimedia*, vol. 6, no. 3, pp. 479–488, Jun. 2004.

[27] Y. Boger, “Fine-tuning voice over packet services,” [Online]. Available: http://www.protocols.com/papers/voip2.htm

César Heyaime-Duvergé was born in Santo Domingo, Dominican Republic, on February 21, 1970. He received the Ingeniero de Sistemas (Magna Cum Laude) degree in 1992 from the Universidad Nacional Pedro Henriquez Ureña, Dominican Republic. He received the M.S. and Ph.D. degrees from the University of Texas at Arlington (UTA), in 1997 and 2003, respectively.

At UTA, he was in the Department of Electrical Engineering, first as a Teaching and Research Assistant, and then as an Instructor. Earlier work involved designing and programming of information systems, and teaching duties as Assistant Professor at the Universidad Nacional Pedro Henrıacute;quez Ureña in Dominican Republic. From 1997 to 2000 he worked for Motorola in its Cellular Infrastructure Division. Since 2003, he has been with Nokia Siemens Networks (then Nokia Networks) where he currently holds the position of Principal Expert and works on performance evaluation and optimization of cellular networks.

Dr. Heyaime-Duvergé was awarded a Fulbright Scholarship in 1994, and the Texas Tec Fellowship in 2000 while at the University of Texas at Arlington. In 1998, he received a Recognition for his contributions to Motorola’s Cellular Infrastructure Group in the area of System Integration and Testing. He is a Member of Tau Beta Pi, and Eta Kappa Nu.

Vasant K. Prabhu (S’64–M’66–F’85–LF’03) was born in Kumta, India, on May 13, 1939. He received the B.Sc. (Hons.) degree from the Karnatak University, Dharwar, India, and the B.E. (Dist.) degree in electrical communication engineering from the Indian Institute of Science, Bangalore, India. He received the S.M. and Sc.D. degrees from the Massachusetts Institute of Technology (MIT), Cambridge, in 1963 and 1966, respectively.

At MIT, he was a Member of the Research Laboratory of Electronics, first as a Research Assistant and then as an Instructor in Electrical Engineering. From 1966 to 1991, he was a Member of Technical Staff at AT&T Laboratories where he made numerous contributions. Since retiring from AT&T Bell Laboratories in 1991, he has been a Professor of Electrical Engineering at the University of Texas at Arlington. Since 2009, he has been appointed as Professor Emeritus.

Dr. Prabhu received the IEEE Centennial Medal from IEEE in 1985. In 1998, he was awarded the Robert Q. Lee award for Excellence in Engineering Teaching from the University of Texas at Arlington. He is a Member of Tau Beta Pi, Eta Kappa Nu, and Sigma Xi.