INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. 2003; **16**:151–169 (DOI: 10.1002/dac.574)

Performance issues of bandwidth management in ATM networks

Christos Bouras[1,2,*,†], Chryssi Chantzi[3], Vaggelis Kapoulas[1,2], Alexandros Panagopoulos[1,2], Ioanna Sampraku[2,4] and Afrodite Sevasti[1,2]

[1] *Research Academic Computer Technology Institute, Riga Feraiou 61, 26221 Patras, Greece*
[2] *Computer Engineering Department, University of Patras, 26500 Rion-Patras, Greece*
[3] *Agricultural Bank of Greece, Panepistimiou 23, Athens, Greece*
[4] *Atmel Hellas S.A., Ag. Varvarus Av. 34, 17563 Paleo Faliro, Athens, Greece*

SUMMARY

In our days, efficient management of the available network resources becomes a critical issue, both from a functional point of view (so that users can be provided with the bandwidth they need), and an economical point of view (so that carriers can satisfactorily and efficiently serve as many customers as possible and at the same time increase their revenue). In this paper we consider a bandwidth control scheme (i.e. managed bandwidth service) for an ATM network infrastructure which is applied to the Greek research and technology network (GRNET). We present some methods that we have tested (in a simulation setting) in order to increase the efficiency of the system and the utilization of the available bandwidth. More specifically, we consider a bandwidth-resizing algorithm for virtual paths, in order to keep the allocated bandwidth very close to the bandwidth actually used. This leads to an increased number of accepted requests and better network utilization. We, also, use the simulation results in order to get an estimation of the effective bandwidth for VBR paths that can be used in call admission. Finally, we consider a semi-offline scheme where requests are gathered and considered for acceptance in regular intervals. Simulation results show an increase in the utilization of resources. As a further improvement, we allow connections to be allocated a little before or after the time initially requested. This leads to further improvement in network utilization. All the improvement schemes were tested with the ATM-TN simulator and the results look promising. Copyright © 2003 John Wiley & Sons, Ltd.

KEY WORDS: managed bandwidth service; ATM networks; simulation

1. INTRODUCTION

In the last few years, great attention has been given to the design and implementation of mechanisms for network bandwidth management. The number of applications with remarkable demands in terms of network resources is constantly rising, resulting in underlying networks exhibiting denial of service and long delays, and facing several bandwidth availability problems.

*Correspondence to: Christor Bouras, Research Academic Computer Technology Institute, Riga Feraiou 61, 26221 Patras, Greece.
† E-mail: bouras@cti.gr

Received August 2001
Revised October 2002
Accepted October 2002

In cases of inadequate resources, the service provided does not meet the desired, promised or expected quality levels. Therefore, it is becoming increasingly vital to ensure high-quality network services for bandwidth critical applications.

At the same time, lack or ineffective allocation of resources results in limitations to the number of users that can be simultaneously served, regardless of whether they are running bandwidth critical applications over the underlying network or not. These limitations can be proved extremely hazardous for the networking facilities provider, especially when users are charged for their access to the network. In this case, efficient management of bandwidth can significantly contribute to the improvement of network use from an economical point of view.

In this paper, we study methods to improve the performance of a managed bandwidth service (i.e. increase the utilization of the network and/or increase the number of accepted requests etc.). The study is based on simulation experiments on the ATM-TN simulator. The simulation experiments were run for the network topology of the Greek research and technology network (GRNET) [1], and the request and traffic patterns are based on traces of actual request sequences and actual transmissions for various applications (e.g. videoconference, steaming video, etc.).

We implement and study two algorithms that allow us to address both the bandwidth management and admission control (i.e. which requests to satisfy and which ones to reject) issues. In both cases there are some strict performance guarantees: transmission starts at a specific time, ends at a specific time, uses a predefined amount of bandwidth, and is guaranteed to be successfully accomplished, once admitted into the network.

Currently the managed bandwidth service (MBS) of the GRNET uses a greedy on-line approach in serving requests for virtual connections with guaranteed bandwidth. Requests are placed in advance and concern connections that will be initiated in the future (reservation in advance). Requests are processed in a first-come-first-serve (FCFS) basis, are accepted if resources are going to be available during the requested time period, and are notified immediately. This way, the availability of resources can actually be guaranteed for the period of time during which the resources are needed. In this sense, reservation means that limited resources are reserved a certain amount of time in advance to get an assurance that the resources are available at the requested time. Comparing the negotiation of immediate and advance reservation, one of the differences is the specification of additional parameters for setting up a reservation in advance. As the resources for advance reservations will be reserved for a given time period in the future, one of the necessary parameters is the time at which the resources will be needed and the duration of the connection which describes for how long will the reservation be alive. Based on this information, the admission control is able to recognize whether the reservation is overlapping with others or not.

More specifically, we assume that requests for the establishment of connections arrive on-line; each request specifies the source and destination nodes, the requested bandwidth, and the duration. The algorithm, based on the availability of resources, either rejects the request, or accepts it. In the latter case, it makes an *a priori* reservation for the establishment of the connection, by reserving the required bandwidth along some path between the source and the destination nodes for the specified duration. This path is determined using the Dijkstra's routing algorithm. In fact, the system calculates and virtually reserves the required resources for the specified time interval in the future, however without immediately blocking them. We implemented this algorithm in the simulation environment to be used as the basis for comparisons.

In order to improve the call admission module we use the results of the simulation to get a simplified empirical formula for the effective bandwidth of VBR connections. The admission control module can use this formula to better calculate the allocated bandwidth (and thus the still available bandwidth). The current MBS of GRNET uses an over-estimation for this effective bandwidth in order to guarantee the quality of service (QoS). However, with this over-estimation bandwidth is wasted since it is allocated but hardly used.

In order to maximize the utilization of the network resources (especially bandwidth) and be able to accept more calls, we then enhance our algorithm by implementing the bandwidth resizing mechanism described in Reference [2]. We focus on the bandwidth allocation problem aimed at the virtual path (VP) and network levels, and investigate the usefulness of bandwidth resizing. The purpose of bandwidth resizing is to make better use of trunk capacity by dynamic adjustment of allocated bandwidth for each virtual path. We adjust the allocated bandwidth of VPs every time a VP is established or terminated, according to some bandwidth adjustment rules. We tested the enhanced algorithm on the ATM-TN simulator to see the effectiveness of the improvements. Results indicate that bandwidth can be freed for use by additional requests.

Finally, we test variations of the admission control algorithm mainly by lifting the requirement for on-line decisions. Requests are examined at regular (short) time intervals leading to a semi-offline algorithm where notification of acceptance or rejection is delayed. This leads to an increase of the utilization and/or the number of the accepted requests depending on what we are trying to maximize (i.e. profits, user satisfaction, etc.). In an attempt to further improve the MBS, we allow the system to shift the time that a connection begins in order to serve more connections. This leads to further performance improvements.

The rest of the paper is organized as follows: In Section 2 a brief overview of the initiatives on the issue of network bandwidth management is provided. This overview contains references to several efforts that have been done to set-up MBS services, international projects and research work that can be used to improve the implementation of these services. In Section 3 we briefly present the infrastructure of GRNET and the current situation of the service. In Section 4 we give an analytical description of the algorithms used. Finally, in Section 5 we present the results of our simulations and conclude with the evaluation of our improvements and the future work that can be done in order to achieve better bandwidth management.

2. PREVIOUS WORK

Bandwidth allocation has been a popular research subject during the last years. In order to solve the bandwidth availability problems of today's networks, various managed bandwidth services, similar to the MBS of GRNET, have been developed in several national research networks.

One of the most known services of this kind has been developed by the TEN-155, a high-speed network for the pan-European academic and research community. The TEN-155 managed bandwidth service [3], is an end-to-end service which enables connectivity among research projects in countries connected to TEN-155. ATM connections are established between project participants' sites, creating in this way a virtual private network (VPN) based on ATM.

UKERNA has also developed a service for bandwidth control, called the SuperJANET managed bandwidth service [4], in order to meet the increasing demands for guaranteed QoS over the SuperJANET III backbone and the requirements of the research community. It is based

on the provision of dedicated ATM VCs or VPs across the SuperJANET III and into the TEN-155.

Another national network where an MBS is being developed is the SWITCHlan network, which is also based on an ATM backbone. The SWITCH Managed Bandwidth Service [5] allows the creation of VPNs with committed bandwidth between interested parties on top of the existing network infrastructure.

The Virginia community college system (VCCS) Intranet, in order to serve a variety of bandwidth-hungry applications that require bandwidth management to guarantee a certain level of customer expectations, has adopted policy-based management that allows for central control of bandwidth management and the enforcement of QoS policies [6]. It provides management of IP traffic from router-to-router, of IP traffic using separate virtual networks and of IP traffic from end-to-end, all via an ATM backbone network.

A similar service implemented on an IP network, is the 'reserved bandwidth' service of the vBNS backbone network [7]. This service allocates bandwidth to application flows using RSVP as the signalling protocol. The service is designed to meet the needs of the vBNS traffic characteristics and in particular super-computer to super-computer communication on the vBNS backbone.

The above services have some common characteristics. In particular, they are not automated and are characterized by long notification and usage times. In most of the cases, the CAC algorithms in the ATM switches are unknown or not available. In addition, CBR is the only service type that is provided at the moment by the managed bandwidth service of GRNET. Thus, the algorithms implemented by these services need to be enhanced with some additional characteristics.

In order to accomplish this and to overcome the above difficulties, a variety of related algorithms that aim to the best way of implementing resource allocation and flow control have been proposed. These algorithms can be classified in the following five levels: cell level (priorities, cells flow control), flow level (burst control, etc.), connection level (connection admission control-CAC that ensures the existing bandwidth is enough for guaranteed quality), virtual path level (bandwidth allocation), network level (routing of virtual paths).

As far as the connection level is concerned, [8] presents a multi-class CAC policy for high-speed ATM switches. According to this policy, traffic is described with respect to the usage parameter control (UPC) parameters. The authors present formulas for determining the bandwidth needed for maintaining the QoS guarantees that can be incorporated in their CAC for constant bit rate (CBR) and variable bit rate (VBR) types of service.

In [9], the authors propose three algorithms for implementing efficient bandwidth allocation and call admission control for VBR service using UPC parameters. The first algorithm, called TAP, takes into account is cell loss ratio (CLR). The second algorithm, called MEB, is based on the idea of effective bandwidth. The third algorithm, called MSM, is based on Lucent's CAC.

Concerning the virtual path and network levels, the authors of Reference [2] deal with virtual path management in ATM networks and propose a virtual path bandwidth resizing algorithm that conducts utilization driven virtual path bandwidth adjustment. Alternatively, the algorithm proposed in Reference [10] adjusts the bandwidth of a VP when the number of VCCs reaches a certain limit.

Finally, the authors of References [11] and [12] present the subject of bandwidth allocation for virtual paths (BAVP) that aims at the optimal bandwidth allocation (capacity, service rate) in virtual paths based on observations of the overall network status. At first, an evolutionary

programming optimization technique is applied in the problem of bandwidth allocation. Then, the authors propose a classical constrained optimization algorithm (CCO) and a genetic algorithm (GA), for solving the BAVP problem.

3. THE GRNET TESTBED

The first algorithm presented here is being implemented in the GRNET, the Greek research and technology network [12]. GRNET provides network services to the Greek academic and research community, and interconnects Universities and research centres in Greece, as well as other R&D departments of industrial organizations, through an advanced, high speed network. The GRNET backbone includes, at this stage, network nodes in 7 major Greek cities and participates actively in the Pan-European TEN-155 high-speed computer network. It provides IP connectivity, several basic and advanced services over IP, as well as non-IP services. It is based on an ATM backbone at the edges of which, IP routers are connected.

In this network, bandwidth reservation is implemented via the establishment of permanent virtual connections (PVCs) between two network nodes. Users may request for various types of service, e.g. CBR, VBR, ABR, defining their objective QoS requirements. It is the network's responsibility to guarantee the QoS requirements of a connection request, once the request is accepted. Due to high demand in QoS, there are two types of service that need to be supported by the network, the CBR and VBR types of service. In this paper we try to find ways to increase network utilization and the number of accepted calls, without violating the QoS guarantee already made to a user.

4. OUR APPROACH

In this paper we implement and test two simple, yet effective, algorithms in order to make the appropriate decisions upon the acceptance of requests, improve the number of accepted calls, and maximize link utilization. The first one takes place in the call admission phase, and the second during the actual service of the VP connection.

The network model used in our case and the procedure that takes place upon the arrival of requests is presented in the sequel.

4.1. The model

The ATM network, $G = (V, L)$, $|L| = m$, considered here consists of a set of end nodes (traffic sources—destinations), E, intermediate nodes (switches), S, and links, L, connecting all these nodes, V. A user located at one end node makes a request for a PVC originating from this node (source) S, and ending at another end node (destination) D. Each PVC request i must define the QoS characteristics for this PVC, the time of creation of this PVC, and the duration.

We denote:

- S_i—the source of VP i.
- D_i—the destination of VP i.
- L_i—the network link i, $i = 1, \ldots, m$.
- C_i—the total capacity of link i, $i = 1, \ldots, m$.

- $\mathrm{Bw}_i^{\mathrm{CBR}}$—the bandwidth of a request for CBR connection, which will be used at the acceptance/rejection decision making module of the system.
- $\mathrm{Bw}_i^{\mathrm{VBR}}$—the peak cell rate (PCR) of a request for a VBR connection, which will be used at the acceptance/rejection decision making module of the system.
- $\mathrm{Vp\_bw}_i^{\mathrm{reserved}}$—the initially reserved bandwidth for VP i, $i = 1, \ldots, n$, where n is the number of VPs, for which reservations have already been made.
- $\mathrm{Vp\_bw\_temp}_i^{\mathrm{reserved}}$—the new bandwidth that will be reserved for VP i, during the next period of time, after a bandwidth adjustment of this VP.
- t_i^{arr}—the arrival of PVC (VP) i request.
- t_i^{cr}—the creation time of reservation of VP i.
- T_{f}—the future period (duration) for which the reservation of VP i is made.
- $C_i^{\mathrm{free}}(T_{\mathrm{f}})$—the free capacity of link i in period T_{f}, during which a number of VPs have been reserved belonging to this link, $i = 1, \ldots, m$.
- P_i—the path of the reserved VP connection.

Upon the arrival of a request, the system uses some algorithms to find out if the request can be accommodated and decides whether to accept or reject it. If a request is accepted the necessary resources are committed for pertinent interval.

4.2. The CAC and routing algorithm

First we implement a method for deciding whether to admit or reject a connection request, as we mentioned above. Users make connection requests with the QoS characteristics they desire their connection to have. In case of a CBR connection request, we examine whether the network can guarantee an amount of $\mathrm{Bw}_i^{\mathrm{CBR}}$ bandwidth, which is the requested amount of bandwidth for the CBR connection. In case of a VBR connection request, at the present we only consider the peak cell rate (PCR) requirement as the amount of bandwidth the network must be able to guarantee to this connection, $\mathrm{Bw}_i^{\mathrm{VBR}}$, in order to accept the requested connection. Finally, in case of an acceptance, the initially allocated bandwidth will be $\mathrm{Vp\_bw}_i^{\mathrm{reserved}}$, and this will be equal to either $\mathrm{Bw}_i^{\mathrm{CBR}}$ or $\mathrm{Bw}_i^{\mathrm{VBR}}$. In the sequel we present a more detailed description the first algorithm, which has already been implemented in the GRNET.

Upon an arrival of a PVC request from S to D for a T_{f} future period, we retrieve the data of the VP reservations already made for part or the whole of that period. We then find those network links that cannot accommodate the particular connection (e.g. $\mathrm{Bw}_i^{\mathrm{CBR}} > C_i^{\mathrm{free}}(T_{\mathrm{f}})$ for CBR connections, or $\mathrm{Bw}_i^{\mathrm{VBR}} > C_i^{\mathrm{free}}(T_{\mathrm{f}})$ for VBR connections), and consider an abstract network where these links are not included. For the remaining network topology, we calculate values (weights) of the links that represent their workload at that period. Then we execute the Dijkstra algorithm for the current topology, starting from the source node of the requested connection. If the destination node appears in the emerging routing table, the request is accepted. If not, it is rejected. In case it is accepted, the system makes a PVC reservation for that period, which means that in time t_i^{cr} we will establish a PVPC (VP), for T_{f} period, which has the same characteristics as the accepted PVC request.

The values (weights) that will express the workload of each link in the remaining topology, are used as input to the Dijkstra routing algorithm, in order to find the 'optimal' feasible path between the source and destination of the requested PVC.

More specifically, whenever the Dijkstra algorithm is executed in context of the decision-making module's algorithm, the algorithm itself assigns a numerical value to every link of the network topology. This value represents the load of the link for the future period T_f. The computation of the value of each link is based on the available (free) capacity $C_i^{free}(T_f)$ of each participating link, and on the minimum bandwidth a user can reserve for that period, min_Vp_bw$_j^{reserved}$. The value of $C_i^{free}(T_f)$ for each link i, is computed by removing the Vp_bw$_j^{reserved}$ of every VP j in link i, from the total capacity of this link, C_i. To be more specific, the value for link i, is calculated as follows:

$$C_i^{free}(T_f) = C_i - \sum_j \text{Vp\_bw}_j^{reserved},$$

where j indicates all VPs reserved for T_f in link i. Thus the value of each link of the remaining topology is computed as follows:

$$\text{link\_value}_i = \text{min\_Vp\_bw}_j^{reserved} / C_i^{free}(T_f).$$

After the execution of this algorithm, a decision is reached, as well as a selection of the route for the VP.

The CAC used prior to the introduction of the previously described CAC was greedy and it was manually executed (i.e. a person considered the request, checked if a route was available with enough bandwidth, and decided). In effect this person executed an shortest path algorithm for the network, and then checked every link in the shortest path to determine availability of the requested bandwidth. If bandwidth was available the request was accepted. However, no attempt was made to optimise the selection of the route, i.e. no weight was associated to the links.

4.3. The bandwidth resizing algorithm

The algorithm used for bandwidth resizing is based on the algorithm presented in Reference [2]. It controls (and modifies) the allocated bandwidth of each VP, based on measurements of the utilization of the VPs. This adjustment mechanism attempts to stabilize the network utilization and free part of the allocated bandwidth so that it is available for other requests. This leads to an increased number of calls that can be accepted, and increased bandwidth utilization.

This algorithm is executed every time there is a VP creation (i.e. at the t_i^{cr} times of the VPs), or VP release event. We examine the utilization of each VP in the previous time interval, and we decide whether a VP needs a bandwidth increase, or a bandwidth decrease. The utilization is every time considered w.r.t. the previously reserved bandwidth of each VPi, Vp_bw_temp$_i^{reserved}$.

First we distinguish those VPs that need a decrease. We define a lower threshold, thres^{low}, for VP bandwidth utilization. If the mean utilization of a VP was below this lower threshold, this VP will get a decrease. Therefore, we compute a decrease step, $-$dec_step$_i$, for every VP i that needs a decrease. The lowest the utilization is, the biggest the decrease will be. We also define three other thresholds below the predefined thres^{low} that will be used in order to define the appropriate dec_step$_i$ for each one of those VPs. These thresholds are denoted as thres$_1^{low}$, thres$_2^{low}$, and thres$_3^{low}$, and are such that

$$\text{thres}^{low} > \text{thres}_1^{low} > \text{thres}_2^{low} > \text{thres}_3^{low} > 0$$

We, then, define the four different possible values of dec_step$_i$, that correspond to VP utilization in one of the four possible regions that are defined between the above four thresholds. The best

threshold values and the corresponding decrement steps are determined based on simulation results and are presented in the next section.

The new decreased bandwidth that will be reserved for each VP is computed based on the dec_step_i decided for each VP, and is the following:

$$\text{dec\_bw}_i = \text{Vp\_bw\_temp}_i^{\text{reserved}} - \text{dec\_step}_i * \text{Vp\_bw\_temp}_i^{\text{reserved}},$$

and the bandwidth which will be reserved for the next interval becomes

$$\text{Vp\_bw\_temp}_i^{\text{reserved}} = \text{dec\_bw}_i.$$

After calculating the new $\text{Vp\_bw\_temp}_i^{\text{reserved}}$ that will be reserved for each of those VPs during the next period, we update the reservation tables by inserting the new data. Then we proceed with those VPs that need a bandwidth increase in the next interval. Thus, in computing the C_i^{free} for the next interval, we use the new decreased values of those VPs.

We define an upper threshold, $\text{thres}^{\text{upper}}$, for VP bandwidth utilization. If the mean utilization of a VP was above this upper threshold, this VP is a candidate to get a bandwidth increase. Therefore we compute an increase step for each one of the candidate VPs. The higher the utilization of a VP is, the higher priority will this VP get, and the more additional bandwidth will be allocated to it. Therefore, we sort the candidate VPs from the one with the most urgent need for an increase, to the one with the less urgent need. We adjust the bandwidth of VPs beginning from the most urgent (higher utilization) to the least urgent (lower utilization) one.

For every VP, we first find the link that belongs to the path of this VP and has the minimum C_i^{free} in the next interval. If this minimum C_i^{free} cannot suffer an increase (minimum $C_i^{\text{free}} = 0$), or the bandwidth previously allocated to this VP is equal to the initially reserved bandwidth of this VP, i.e.

$$\text{Vp\_bw\_temp}_i^{\text{reserved}} = \text{Vp\_bw}_j^{\text{reserved}},$$

then we do not resize this VP.

If neither is the case, we proceed in finding the appropriate increase step, inc_step_i for this VP. We define one more threshold above the predefined $\text{thres}^{\text{upper}}$ that will be used in order to choose an appropriate inc_step_i for each one of those VPs. This threshold is denoted as $\text{thres}_1^{\text{upper}}$, and is such that

$$\text{thres}^{\text{upper}} < \text{thres}_1^{\text{upper}} < 1.$$

We, then, define the two different possible values of inc_step_i, for VP utilization in one of the two possible regions that are defined between the above three thresholds. The best values are also determined based on simulation results.

The new increased bandwidth, inc_bw_i, will then be computed similar to the decreased bandwidth. However if the increment calculated is greater than the minimum C_i^{free}, then the increment becomes equal to the minimum C_i^{free}. Now, if the increased bandwidth calculated is greater that the initially requested bandwidth, then the bandwidth, which will be reserved, does not increase beyond the initially requested bandwidth.

When all adjustments are made, the VPs continue with the new reserved bandwidths, until the next resizing event (a VP creation event), or until the end of their duration.

4.4. The semi-offline call admission algorithm

In the on-line case, all decisions are taken with no knowledge of future requests. This can lead to bad decisions. In order to improve the call admission procedure and maximise some revenue (e.g. the utilization, the number of accepted calls, etc.), we can postpone the procedure of decision making for a while, and queue the requests. Then, we can take the appropriate decisions for all the queued requests taking into account the parameters of all of them. This procedure takes place at regular intervals (hence, the semi-offline characterization).

The approach, used to decide which requests to accept and which ones to reject, is a greedy one. We opted for simplicity because the purpose is to examine whether the introduction of this semi-offline procedure can lead to substantial gaining. In the future, a more efficient algorithm can be introduced to replace the greedy one.

More specifically, a weight is assigned for each request, which corresponds to the revenue gained when serving this request. All the queued requests are sorted in descending order based on this weight (i.e. their revenue). Requests are then examined in this order. At each step, the algorithm selects the request with the largest weight, and examines if this request can be served. This is done using the previously mentioned CAC algorithm. If there is enough bandwidth to accommodate the requested VP, the creation of the VP is scheduled, otherwise the request is rejected. Then we proceed with the request which has the next largest weight, until all requests have been examined. This way, the algorithm gives a priority to the most profitable VP requests.

Figure 1 shows the pseudo-code that implements this algorithm.

4.5. The time shifting of connections

Simulation results showed that sometimes some requests were rejected because of a small overlap with other accepted connections and the consequent non-availability of bandwidth for this period of time. However, most of the times, the requests concern events (e.g. videoconferences) that can be re-scheduled to start a little earlier or a little later if this overlapping is to be avoided.

In order to examine whether this re-scheduling can be beneficial or not, we use a simple algorithm to shift the starting time of the requested connections a little (in order to 'pack' them better).

More specifically, when applying time shifting to the connections, a similar algorithm is used. Requests are still examined in a descending weight order, until all requests have been examined. However, now, when examining each request, the following is done: First, VPs that are already

```
Sort requests in descending weight order
For each request in this order
        Run Dijkstra algorithm
        If a valid path was found
                Schedule the creation of the requested VP
        Else
                Mark request as rejected
        End if
End for
```

Figure 1. Pseudo-code for the semi-offline call admission algorithm.


```
Sort requests in descending weight order
For each request in this order
        Shift left requests that start before the end time of the current request
        Shift right requests that end after the start time of the current request

        Find "critical" time moments within the given interval
        Add the requested creation time to this list of time moments
        For each of these times
                If the time interval starting from the current time and with the
                requested duration is valid
                        Run Dijkstra for this time interval
                        If a valid path was found
                                Schedule the creation of the VP
                                Break
                        End if
                End if
                If no valid path was found
                        Mark request as rejected
                End if
        End for
End for
```

Figure 2. Pseudo-code for the time shifting of connections.

scheduled but have not yet been served are 'shifted' away of the time interval that corresponds to the current request. That way there is more available bandwidth at that time interval, therefore chances that this request will be accepted are increased. When shifting requests in time, each already accepted request is moved as far as possible in time, without exceeding the capacity of the link and the time limits imposed by the request itself. Then, the time moments, when the available bandwidth of the links is changed, are found. We are interested only in times that are within the time interval available for serving the request (from the minimum start time till the maximum end time for this request). The request is examined for each interval that starts from one of these times and has the requested duration. The first valid time interval, when the request can be served, is accepted. If no such valid interval is found, the request is rejected.

Figure 2 shows the pseudo-code that implements this algorithm.

5. SIMULATION STUDIES

In this section, we provide our simulation results to illustrate the performance gains of the various improvements that we have adopted for the MBS of GRNET.

For our simulations we used the ATM-TN simulator. The simulator is not capable of handling VP construction and destruction during a run. This was essential for our study. So, we extended the simulator with a wrapper that handled all the changes in the allocation of VPs and used the ATM-TN simulator as a sub-routine. In addition we provided more traffic sources by

Figure 3. The network topology.

implementing more realistic ones that follow the patterns of real connections (e.g. videoconference connections and video-on-demand connections)

In our simulations we consider the exact topology of the GRNET for evaluating the improvements of the MBS. The ATM network topology used in the simulation is shown in Figure 3.

Requests are considered to come at random intervals following the geometric distribution, and the holding time is also geometrically distributed. The required bandwidth of PVCs follows the traces of real requests.

5.1. Calculation of the effective bandwidth

One of the problems of the current MBS implementation in GRNET is that it over-estimates the effective bandwidth for the VBR requests. This is done to avoid the problem of providing these requests with less QoS that requested. However, this over-estimation causes the service to allocate more bandwidth than needed and thus some bandwidth is wasted.

In order to improve the call admission control algorithm of the MBS we used the results of the simulation to come up with a simplified formula for making an estimation of the effective bandwidth.

This simplified formula is of the form:

$$\text{Effective bandwidth} = \text{SCR} + c\ (\text{PCR} - \text{SCR}).$$

Our objective is to calculate the value of the constant c (using data from the simulations) so that the above formula closely estimates the effective bandwidth (in fact we want the formula to over-estimate the effective bandwidth but as close as possible to the actual value).

The result of the calculation of the effective bandwidth for various connections shows that different connections correspond to a different constant c in the above formula. Figure 4 shows this constant c for various connections.

As we can see, most of the time the effective bandwidth does not exceed SCR by more than 50% of the difference between PCR and SCR. In fact, in almost 90% of the connections the effective bandwidth does not exceed SCR by more than 40% of the difference between PCR and SCR.

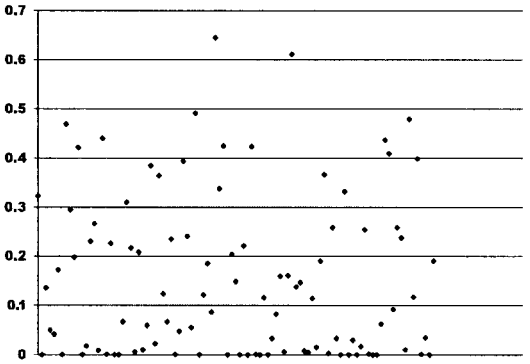

Figure 4. Constants c for various connections.

In average, the effective bandwidth is equal to SCR plus 15% of the difference between PCR and SCR.

Simulations show that a general value of $c = 0.3$ can be safely used to calculate the effective bandwidth (i.e. it does not cause situations where connections receive less QoS than guaranteed). In more detail, this value of the constant c, caused no cell drops in the simulation runs, effectively showing that the guaranteed QoS was not compromised. Higher values of c caused cell drops (more drops for higher values). This means that a higher value for c cannot be used without risking that some request will get less quality than guaranteed at some point in time. Therefore we propose that the effective bandwidth be calculated as:

$$\text{Effective bandwidth} = \text{SCR} + 0.3\ (\text{PCR} - \text{SCR}).$$

Alternatively, one can use higher values for c, and set-up a system for compensating requests when ever the guaranteed QoS is not provided.

The effective bandwidth calculated this way is less than the currently used over-estimation, and the adaptation of this formula leads to an increased number of accepted requests and increased bandwidth utilization.

5.2. *Bandwidth resizing*

Our next step is to study the improvements gained from the implementation of bandwidth resizing. These results show us how much we can expect to gain and whether the implementation of bandwidth resizing is worthwhile.

We set the upper threshold of utilization equal to 0.8 and the lower threshold equal to 0.5. Different pairs of thresholds result in different performance. Through several tests, we decided to choose the above values as the base parameters in the following analysis because they produce a higher call acceptance rate and a higher utilization, while they do not cause any problems, such as allocating less bandwidth than needed and causing cells to drop.

Then, we select the increase and decrease steps as follows:

- If $0.8 < \text{bandwidth utilization} <= 0.9$, the increase step is set equal to 0.25.
- If bandwidth utilization > 0.9, the increase step is set equal to 0.35.

- If $0.4 <$ bandwidth utilization $<= 0.5$, the decrease step is set equal to 0.2.
- If $0.3 <$ bandwidth utilization $<= 0.4$, the decrease step is set equal to 0.25.
- If $0.2 <$ bandwidth utilization $<= 0.3$, the decrease step is set equal to 0.3.
- If bandwidth utilization $<= 0.2$, the decrease step is set equal to 0.35.

These values, as above, were found to be better after various simulations.

Firstly, we experiment with traffic sources that follow closely the parameters they advertise (i.e. the traffic produced is shaped as expected). Some results of the simulations are shown in Figure 5. Figure 5(a) corresponds to a 20 Mbps link, Figure 5(b) to a 4 Mbps link and Figure 5(c) to a 2 Mbps link. In each subfigure the parameters depicted are: the bandwidth that would have been allocated without VPM, the currently allocated bandwidth using VPM, and the actual bandwidth used.

As expected, the type of the traffic sources selected makes the actually used bandwidth to closely follow the requested one, and thus it does not allow VPM to change the initially requested bandwidth too much during the run. Obviously in this case there is no gain.

However, most of the traffic sources in real applications do not follow closely their advertised behaviour. In order to test bandwidth resizing in these cases, we experiment with traffic sources that are realistic. These sources generated traffic identical to that traffic generated by actual transmissions of videoconference events and streaming media events that took place over the GRNET. The traces of these actual events were taken from the GRNET's routers. Several traces were taken and the simulated traffic sources choose a trace randomly each time a new stream had to be simulated. The simulator read the traces of the selected actual transmission and generated the equivalent simulated traffic. Some results of simulations for this case are shown in Figure 6. As above each subfigure corresponds to a different link.

We observe that, in the case of realistic traffic sources, bandwidth management can save bandwidth, i.e. allocate less than requested (but more than actually needed). This saved bandwidth can be allocated to other requests (i.e. more requests can be accepted).

Notice that most of the time the total allocated bandwidth (as decided by the VPM algorithm) is more than the one actually used by the realistic traffic sources. But sometimes the actual traffic gets higher than the bandwidth allocated at that time. However, this does not happen often and more importantly it causes no cell drops.

As a conclusion, bandwidth management can improve the performance of MBS by making the allocated bandwidth follow more closely the actually used bandwidth, and by freeing bandwidth to be used by other requests (i.e. it increases the number of accepted requests).

5.3. Semi-online call admission

The way the current MBS works is that every request is examined immediately after it is submitted, and a decision is reached (i.e. the decision algorithm is on-line). This usually leads to some bad decisions because the on-line algorithm does not have any knowledge of the future requests.

However, the service requires that reservations be made in advance. In such a setting, the on-line requirement may be relaxed a little bit, since the requested connection does not have to be immediately available (i.e. the request is for a future time). Therefore it makes sense to delay a little bit the decision (and the notification) in order to get more requests, and then decide based on this additional knowledge.

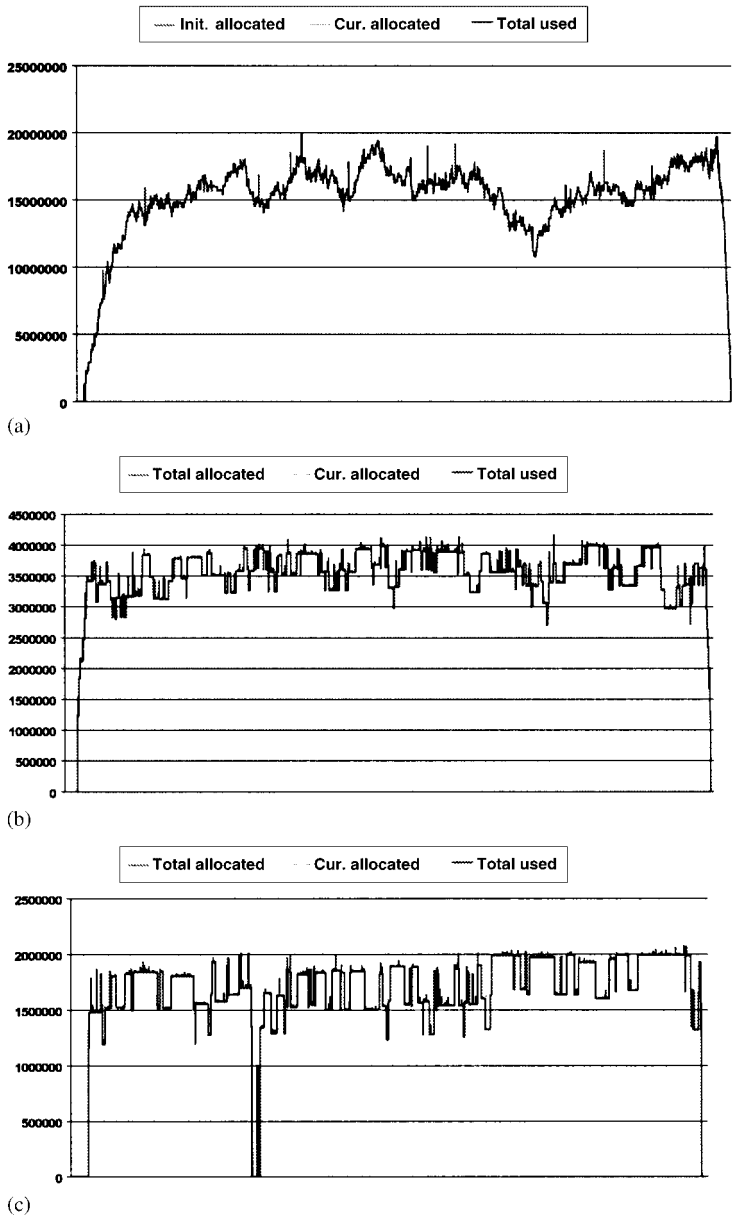

Figure 5. Allocated and used bandwidth for traffic sources that work as advertised.

Based on this observation our next step was to study the case where requests were queued and then examined in a semi-batch mode at regular time intervals. The algorithm used is a greedy one and at each time tries to accept these requests (among the queued ones) that maximize the network utilization.

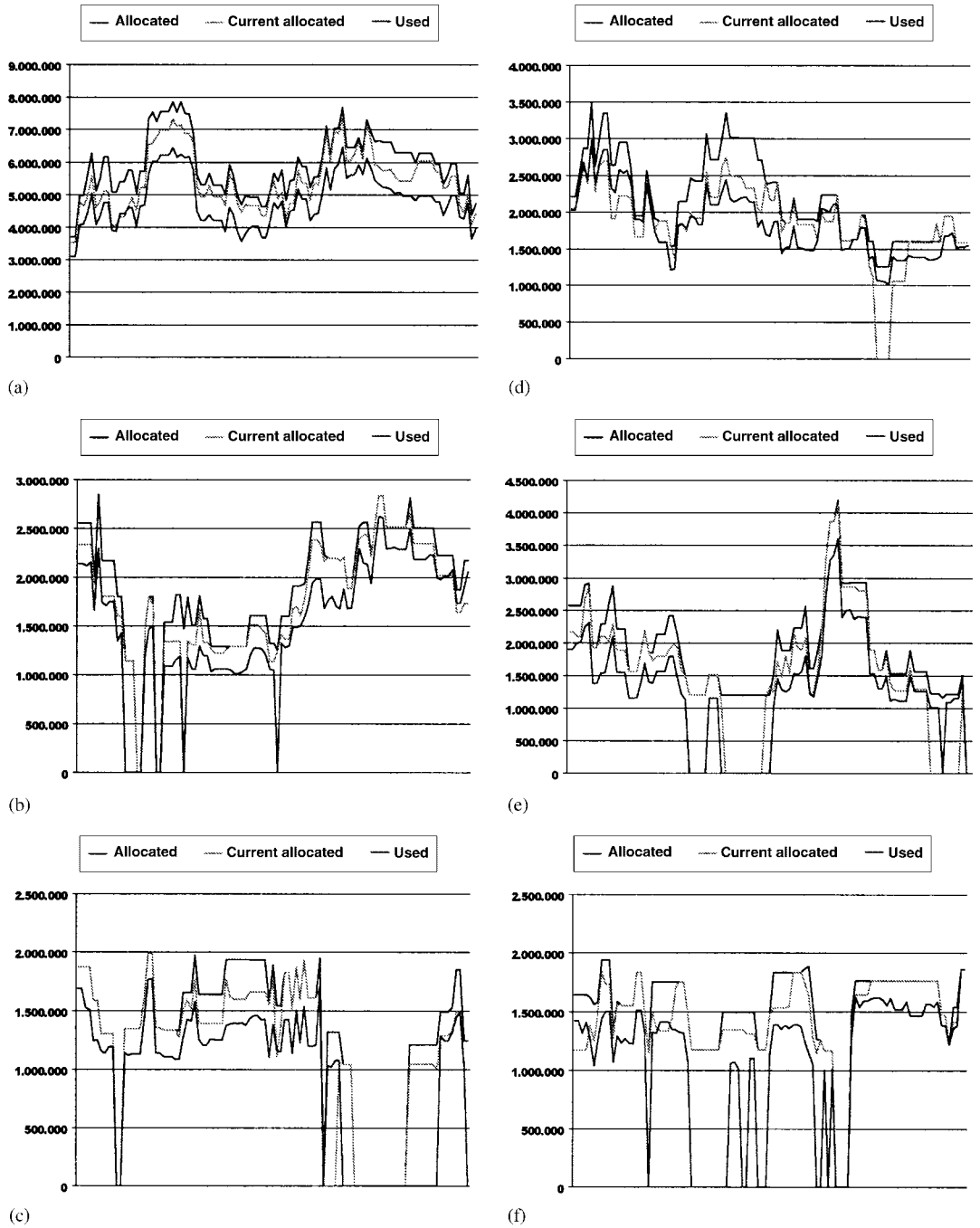

Figure 6. Allocated and used bandwidth for realistic traffic sources.

Results from the simulations are shown in Figure 7. The bandwidth usage is shown for two different sizes of the time interval. The bandwidth usage for the on-line algorithm is, also, shown for comparisons.

Results indicate that serving requests in batches increases the utilization of the network.

5.4. Time-shifting of connections

Our final step is to try to improve even more the utilization of the network by allowing the MBS to shift the time period a connection takes place at (actually up to 10% of its duration).

Some results for this case are shown in Figure 8.

As we can see time-shifting seems to offer improvement only when the overall usage of the network increases or decreases (e.g. during early in the morning or late in the afternoon of a working day). When the usage of the network remains high the improvements are not very big.

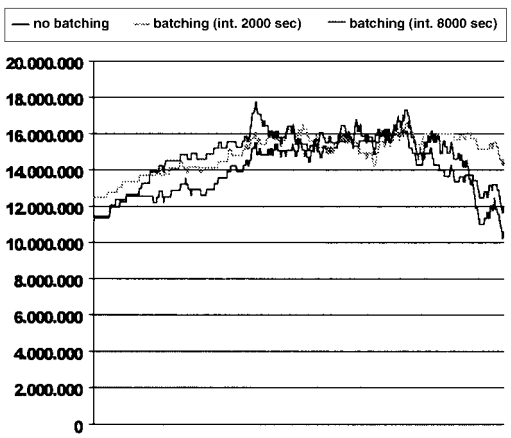

Figure 7. Bandwidth usage for semi-batch request serving.

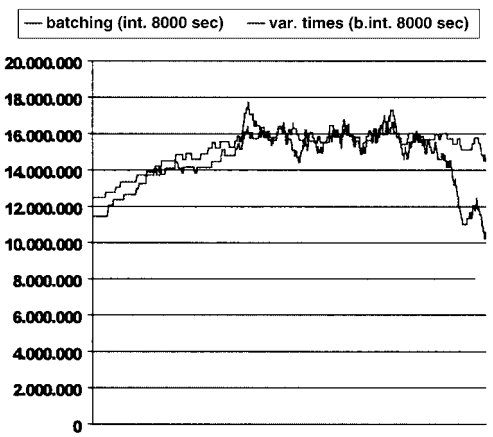

Figure 8. Bandwidth usage when time-shifting of connections is allowed.

6. CONCLUSIONS

We have presented a set of improvements for the MBS of GRNET. These improvements were studied using simulation and various results were presented. Most of the results are promising and indicate that the improvement worth being implemented.

7. FUTURE WORK

Our future plans are: (a) to further study various other improvements, (b) to substitute the greedy algorithms used with more efficient ones, and (c) to make similar studies for networks that offer IP-based Quality of Service.

ACKNOWLEDGEMENTS

The authors would like to thank Giorgos Mourlas for his valuable help in debugging the code written for the simulation experiments.

REFERENCES

1. GRNET: Greek Research and Technology Network. http://grnet.gr/index_en.html.
2. Lin Y-D, Su W-J, Lo C-C. Virtual path management in ATM networks. *Proceedings of International Conference on Communications*, Dallas, USA, June 1996.
3. DANTE. External Operational Procedures for the TEN-155 managed bandwidth service. QUA-99-063 OPS-99-071, Version 1, August 1999. http://www.dante.net/mbs/project_procedures.phtml.
4. SuperJANET Management Bandwidth Service Terms and Conditions. ND/STG/MBS/DOC/001, Version 2, March 2000. http://www.ja.net/development/man_band/man_band.html.
5. SWITCH Managed Bandwidth Service. http://www.switch.ch/lan/mbs/.
6. Roney EB Jr. VCCS Network management White Paper. Virginia Community College System. August 12, 1999. http://www.so.cc.va.us/its/projects/VCCS-NW-MANGT-WP1.htm
7. Cunningham L. QoS development in the vBNS. *Communications Magazine* 1988; **36**(5). http://www.vbns.net/presentations/laura-qos.
8. Ramamurthy G, Ren Q. Multi-class connection admission control policy for high speed ATM switches. *Proceedings of the Conference on Computer Communications* (*IEEE Infocom*), Kobe, Japan, April 1997; 963–972. http://www.ieee-infocom.org/1997/papers/ren.pdf.
9. Wu D, Jonathan Chao H. Efficient bandwidth allocation and call admission control for VBR service using UPC parameters. *International Journal of Communication Systems* 2000; **13**(1):25–50.
10. Ohta S, Sato K-I. Dynamic bandwidth control of the virtual path in an asynchronous transfer mode network. *IEEE Transactions of Communications* 1992; **40**(7):1239–1247.
11. Pitsillides A, Stylianou G, Pattichis C, Sekercioglu A, Vasilakos A. Bandwidth allocation for virtual paths (BAVP): investigation of performance of classical constrained and genetic algorithm based optimisation techniques. *INFOCOM 2000*, Tel-Aviv, Israel, March 26–30, 2000; 1501–1510.
12. Pitsillides A, Stylianou G, Pattichis C, Sekercioglu A, Vasilakos A. Investigation of the performance of EP-BAVP (evolutionary programming for virtual path bandwidth allocation). *International Conference on Telecommunications*, Port Carras, Greece, 1998; 310–314.

AUTHORS' BIOGRAPHIES

Christos Bouras is currently an Assistant Professor in the Computer Science and Engineering Department of Patras University (Greece). He is, also, the scientific advisor of Research Unit 6 and director of Networking Technologies Sector, in Research Academic Computer Technology Institute (CTI), Patras, Greece. His research interests include Networks and Protocols, Educational Technology, Telematics and New Services. He has published 130 papers in various well-known refereed conferences and journals. He is a co-author of five books in Greek. He is, also, member of the team of experts in the Greek Research and Technology Network (GRNET), ACM, IEEE, EDEN, AACE and New York Academy of Sciences.

Chryssi Chantzi is currently a Computer Engineer of the Data Management Group in the Agricultural Bank of Greece and her interests include Data Management, Data Warehousing, Database Design and Implementation, Telematics and New Services, Bandwidth Allocation algorithms and implementation of Managed Bandwidth Services.

Vaggelis Kapoulas is currently a PhD candidate at the Computer Science and Engineering Department, School of Engineering, University of Patras (Greece). He is, also, an R&D Engineer at Research Unit 6: Networks Telematics and New Services, Research Academic Computer Technology Institute (Greece). His research interests include Networks, Telematics and Algorithms for Distributed Systems. He has published 30 research papers in various well-known refereed conferences and 4 research papers in scientific journals.

Alexandros Panagopoulos is currently attending the fourth year of undergraduate studies at the Computer Engineering and Informatics Department of the University of Patras (Greece). He is also affiliated to Research Unit 6 of the Research Academic Computer Technology Institute in Patras (Greece). His interests include computer networks, telematics, networked virtual environments, 3d graphics and virtual learning environments.

Ioanna Samprakou is currently an R&D Computer Engineer of the System Concept & Design Group at Atmel Hellas, Greece. She is also in the final phase of obtaining her MSc degree in 'Computer Science & Technology' at the Computer Engineering and Informatics Department of the Patras University. Her research interests include Telematics and New Services, Bandwidth Allocation algorithms, QoS and Wireless & Mobile Protocols & Communications.

Afrodite Sevasti is currently an R&D Computer Engineer in Research Unit 6 at the Computer Technology Institute, Patras, Greece. She is also attending the second year of studies to obtain her PhD degree at the Computer Engineering and Informatics Department of Patras University (Greece). Her research interests include Telematics and New Services, Virtual Reality Services, Multimedia Editing environments, Database design, implementation of Managed Bandwidth Services, QoS and pricing of network services as well as Video on Demand algorithms. She has published 16 papers in well-known refereed conferences and journals.