Performance Analysis of a Managed Bandwidth Service for ATM Networks

Christos Bouras **Chryssi Chantzi** **Vaggelis Kapoulas**
Alexandros Panagopoulos **Ioanna Sampraku** **Afrodite Sevasti**
Research Academic Computer Technology Institute, Patras, Greece, and
Computer Engineering Dept., Univ. of Patras, Greece
E-mail: bouras@cti.gr

Keywords: Managed Bandwidth Service

Abstract
In this paper we are considering methods to improve the performance of a bandwidth control scheme (i.e. Managed Bandwidth Service) for an ATM network infrastructure (with application to the Greek Research and Technology Network - GRNET). These methods try to increase the efficiency of the system and the utilisation of the available bandwidth. More specifically, we consider: a) a bandwidth-resizing algorithm for virtual paths in order to (constantly) keep the allocated bandwidth very close to the bandwidth actually used, b) a simple method to estimate the effective bandwidth for VBR paths that can be used in call admission, and c) a semi-offline call admission scheme where requests are gathered and considered for acceptance in regular intervals (with a further improvement, of allowing connections to be allocated a little before or after the time initially requested). All these methods were tested in a simulation setting (with the ATM-TN simulator), and results indicate that they lead to an increased number of accepted requests and better network utilisation.

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. 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. In this case, efficient management of bandwidth can contribute to the improvement of network utilisation from an economical point of view.

One of the main network resources is the available bandwidth of the communication channels. In order to use the network, the user requests a virtual connection to be established between the source and destination node. Although the rate of information flowing through such a connection may vary in time, the network has to guarantee that the connection will support at least the bit rate that was agreed upon the admission of the requested connection.

In this paper, we study methods to improve the performance of a Managed Bandwidth Service (i.e. increase the utilisation 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 simulations were run for the network topology of the Greek Research and Technology Network (GRNET) and the requests and traffic patterns are based on traces of actual request sequences and actual transmissions for different kinds of applications (e.g. video-conference, steaming video etc.).

We implement and study algorithms that allow us to address both the bandwidth management and admission control issues (i.e. which requests to satisfy and which ones to reject). 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.

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. 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 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 scheme is able to recognise 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. 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 a simulation environment so that it can 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, part of this bandwidth is wasted since it is allocated but hardly used.

In order to maximise the utilisation 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 [1]. 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 some bandwidth can be saved 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 utilisation and/or the number of the accepted requests depending on what we are trying to maximise. In an attempt to further improve the MBS, we allow the system to shift the time that a connection begins in order to avoid overlapping and, thus, serve more connections.

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 TEN-155. The TEN-155 Managed Bandwidth Service [2], is an end-to-end service which enables connectivity among research projects in countries connected to TEN-155. Various national net works have also developed services for bandwidth control. E.g. the SWITCHlan develops the SWITCH Managed Bandwidth Service (http://www.switch.ch/lan/mbs/).

The VCCS (Virginia Community College System) Intranet has adopted Policy-Based Management that allows for central control of bandwidth management and the enforcement of QoS policies [3].

A similar service implemented on an IP network, is the 'Reserved Bandwidth' service of the vBNS backbone network [4]. This service allocates bandwidth to application flows using RSVP as the signalling protocol.

Concerning the best way to implement a bandwidth allocation service and flow control, a variety of related algorithms have been proposed. These can be classified in five levels: cell level (priorities, cells flow control), flow level (burst control, etc.), connection level (Connection Admission Control – CAC), Virtual Path level (bandwidth allocation), and network level (routing of Virtual Paths).

As far as the connection level is concerned, [5] 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 [6], 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 [1] deal with Virtual Path management in ATM networks and propose a Virtual Path Bandwidth Resizing Algorithm that conducts utilisation driven virtual path bandwidth adjustment. Alternatively, the algorithm proposed in [7] adjusts the bandwidth of a VP when the number of VCCs reaches a certain limit.

Finally, the authors of [8] and [9] 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 the overall network status.

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 maximise link utilisation. The first one takes place in the call admission phase, and the second during the actual service of the VP connection.

The model

The ATM network, $G = (V, L)$, $|L| = m$, considered here consists of a set of nodes, V, and a set of links, L, connecting all these nodes. A user located at one end node makes a request for a PVC originating from this node S (source), and ending at another end node D (destination). 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, $i = 1,..,m$.
- C_i - the total capacity of link i, $i = 1,..,m$.
- Bw_i^{CBR} - the bandwidth of a request for CBR connection.
- Bw_i^{VBR} - the peak cell rate (PCR) of a request for a VBR connection.
- $Vp\_bw_i^{reserved}$ - the initially reserved bandwidth for VP i, $i = 1,..,n$, where n is the number of VPs, for which reservations have already been made.
- $Vp\_bw\_temp_i^{reserved}$ - 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 time 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.
- $C_i^{free}(T_f)$ - the free capacity of link i in period T_f, during which a number of VPs have been reserved belonging to this link, $i = 1,..,m$.
- P_i - the path of the reserved VP connection.

The CAC and Routing Algorithm

First we implement a method for deciding whether to admit or reject a connection request. 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 Bw_i^{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, Bw_i^{VBR}, in order to accept the requested connection. Finally, in case of an acceptance the initially allocated bandwidth will be $Vp\_bw_i^{reserved}$, and this will be equal to either Bw_i^{CBR} or Bw_i^{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. $Bw_i^{CBR} > C_i^{free}(T_f)$ for CBR connections, or $Bw_i^{VBR} > C_i^{free}(T_f)$ 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:

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

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:

$$link\_value_i = \min\_Vp\_bw_j^{reserved} / C_i^{free}(T_f).$$

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

The bandwidth resizing algorithm

The algorithm used for bandwidth resizing is based on the algorithm presented in [1]. It controls (and modifies) the allocated bandwidth of each VP, based on measurements of the utilisation of the VPs. This adjustment mechanism attempts to stabilise the network utilisation 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 utilisation.

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 utilisation of each VP in the previous time interval, and we decide whether a VP needs a bandwidth increase, or a bandwidth decrease. The utilisation is every time considered w.r.t. the previously reserved bandwidth of each VP*i*, $Vp\_bw\_temp_i^{reserved}$.

First we distinguish those VPs that need a decrease. We define a lower threshold, $thres^{low}$, for VP bandwidth utilisation. If the mean utilisation 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 lower the utilisation is, the bigger 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

$$thres^{low} > thres_1^{low} > thres_2^{low} > thres_3^{low} > 0$$

We, then, define the four different possible values of dec_step_i, that correspond to VP utilisation 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:

$$dec\_bw_i = Vp\_bw\_temp_i^{reserved} - dec\_step_i * Vp\_bw\_temp_i^{reserved},$$

and the bandwidth which will be reserved becomes

$$Vp\_bw\_temp_i^{reserved} = dec\_bw_i.$$

After this, we proceed with those VPs that need a bandwidth increase in the next interval. In computing the C_i^{free} for the next interval, we use the new decreased values of those VPs.

We define an upper threshold, $thres^{upper}$, for VP bandwidth utilisation. If the mean utilisation 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 utilisation of a VP is, the higher priority

will this VP get. 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 utilisation) to the least urgent (lower utilisation) 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^{free}=0), or the previously allocated bandwidth to this VP is equal to the initially reserved bandwidth of this VP, i.e.

$$Vp\_bw\_temp_i^{reserved} = Vp\_bw_j^{reserved},$$

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 $thres^{upper}$ that will be used in order to choose an appropriate inc_step_i for each one of those VPs. This threshold is denoted as $thres_1^{upper}$, and is such that

$$thres^{upper} < thres_1^{upper} < 1.$$

We, then, define the two different possible values of inc_step_i, for VP utilisation 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.

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 utilisation, the number of accepted calls etc.), we can postpone for a while the procedure of decision making, 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 characterisation).

The approach used to decide which requests to accept and which 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 all the queued requests are sorted in descending order based on their revenue. The requests are then examined in this order, using the previously mentioned CAC algorithm.

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, we use a simple algorithm to shift the starting time of the requested connections a little (in order to 'pack' them better). Again, a simple approach is followed.

Requests are queued and sorted as before. However, when a request gets accepted it is scheduled to start as early as possible. On the other hand, if a request cannot be accepted for the specified starting time, we examine whether it can be accepted either earlier or later. In case it can, it is accepted.

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 implementing more realistic traffic sources that follow patterns of real connections (e.g. videoconference connections and Video-on-Demand connections)

In our simulations we consider the exact topology of the GRNET.

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 traces of real requests.

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 giving to these requests 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:

Effective bandwidth = SCR + *c* (PCR-SCR)

Our objective is to calculate the value of the constant *c* (using simulation results) so that the above formula closely estimates the effective bandwidth (in fact we want the formula over-estimate 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 1 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.

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

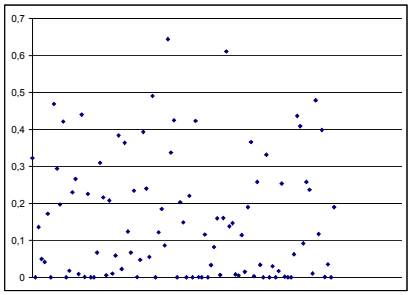

Figure 1. Constants *c* for various connections

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). Therefore the effective bandwidth can de calculated as:

Effective bandwidth = SCR + 0.3 (PCR-SCR)

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 utilisation.

Bandwidth Resizing

Our next step is to study the improvements gained from the implementation of bandwidth resizing. We set the upper threshold of utilisation equal to 0.8 and the lower threshold equal to 0.5. 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 utilisation, 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 (*BU* denotes bandwidth utilisation):

- If $0.8 < BU <= 0.9$, the increase step is set to 0.25
- If $BU > 0.9$, the increase step is set to 0.35
- If $0.4 < BU <= 0.5$, the decrease step is set to 0.2
- If $0.3 < BU <= 0.4$, the decrease step is set to 0.25
- If $0.2 < BU <= 0.3$, the decrease step is set to 0.3
- If $BU <= 0.2$, the decrease step is set to 0.35

The above values were found to be better after various simulations.

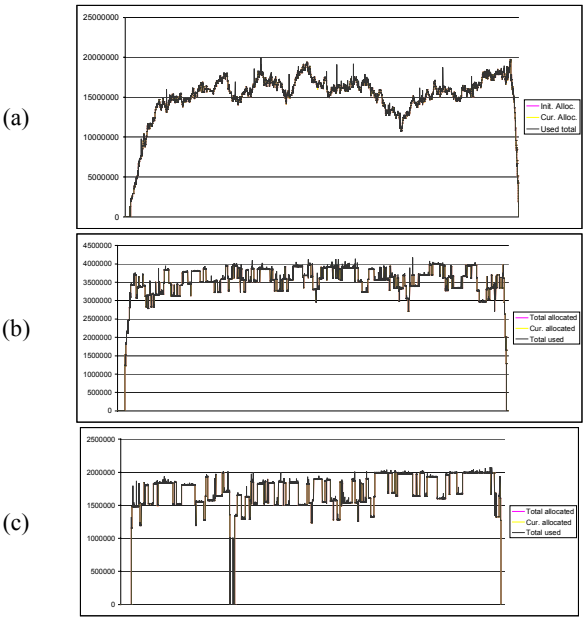

Figure 2. Allocated and used bandwidth for ideal traffic sources

First 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 2. Fig.2a corresponds to a 20 Mbps link, fig. 2b to a 4 Mbps link and fig 2c 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

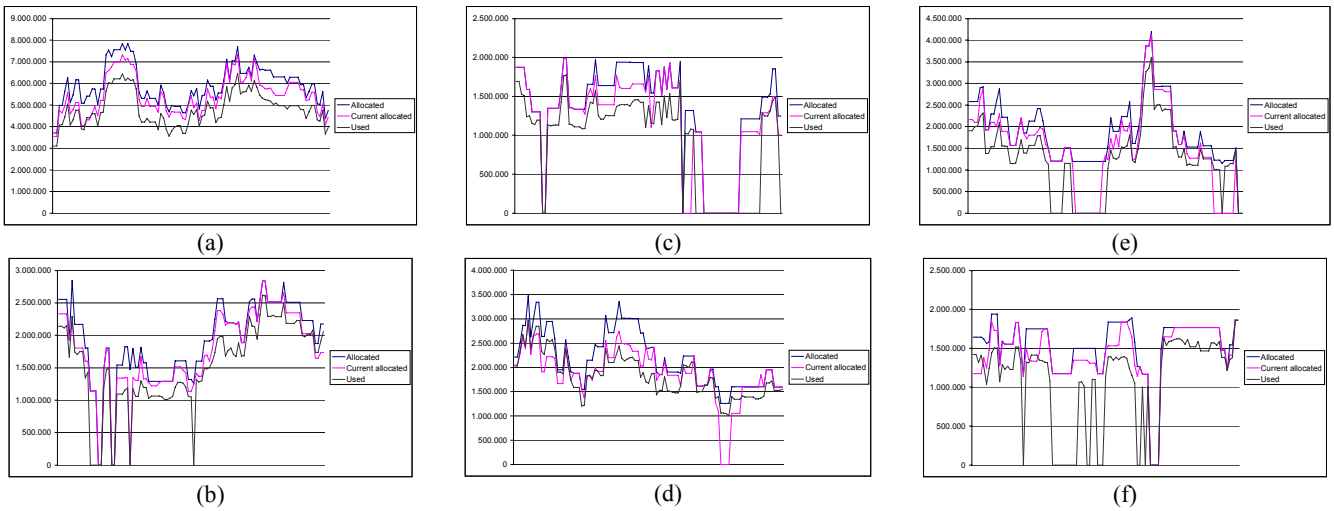

Figure 3. Allocated and used bandwidth for realistic traffic sources

resizing in these cases, we experiment with traffic sources that are more realistic and follow the patterns of observed actual transmissions. Some results of simulations for this case are shown in figure 3. 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 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).

Semi-online call admission

The way the current MBS works is that every request is examined immediately after it is submitted. This usually leads to some bad decisions. However, the service requires that reservations are 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. 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.

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 maximise the network utilisation.

Results from the simulations are shown in figure 4. The bandwidth usage is shown for two different sizes of the time interval. Also the bandwidth usage for the on-line algorithm is shown for comparisons.

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

Time-shifting of connections

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

Some results for this case are shown in figure 5. 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.

CONCLUSIONS – FUTURE WORK

We have presented few 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 improvements are worth being implemented.

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.

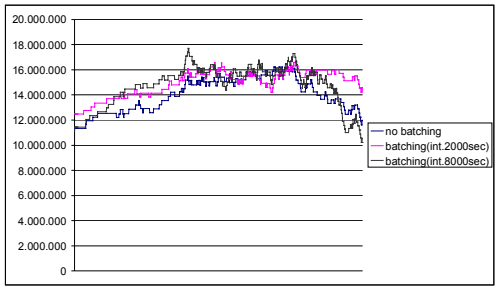

Figure 4. Bandwidth usage for semi-batch request serving

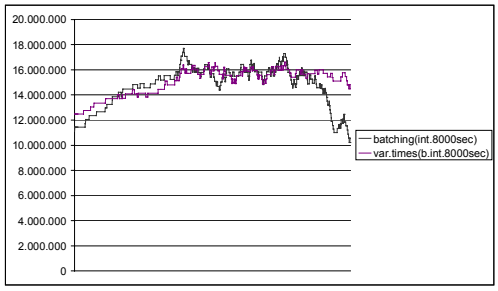

Figure 5: Bandwidth usage when time-shifting of connections is allowed

Acknowledgement

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

REFERENCES

[1] Y.-D. Lin, W.-J. Su, and C.-C. Lo, "Virtual Path Management in ATM Networks", in *Proc. of the Int. Conference on Communications*, Dallas, USA, June 1996

[2] DANTE , "External Operational Procedures for the TEN-155 Managed Bandwidth Service", 2 August 1999, QUA-99-063 OPS-99-071 Version 1, http://www.dante.net/mbs/project_procedures.phtml

[3] Roney E. Boyd Jr., "VCCS Network management White Paper", Virginia Community College System, August 1999, http://www.so.cc.va.us/its/projects/VCCS-NW-MANGT-WP1.htm

[4] Laura Cunningham, "QoS Development in the vBNS", in the Communications Magazine, Vol. 36, No. 5, May 1998,

[5] G. Ramamurthy and Q. Ren, "Multi-Class Connection Admission Control Policy for High Speed ATM Switches," in *Proc. of the Conference on Computer Communications* (IEEE Infocom), Kobe, Japan, April 1997, pp. 963–972

[6] D. Wu and H. J. Chao, "Efficient Bandwidth Allocation and Call Admission Control for VBR Service Using UPC Parameters," *International Journal of Communication Systems*, Vol. 13, no. 1, February 2000, pp. 25–50

[7] S. Ohta, and K.-I. Sato, "Dynamic Bandwidth Control of the Virtual Path in an Asynchronous Transfer Mode Network", *IEEE Transactions of Communications*, Vol.40, No.7, July 1992, pp. 1239–1247

[8] A. Pitsillides, G. Stylianou, C. Pattichis, A. Sekercioglu, A. Vasilakos, "Bandwidth Allocation for Virtual Paths (BAVP): Investigation of Performance of Classical Constrained and Genetic Algorithm Based Optimisation Techniques", INFOCOM 2000, Tel-Aviv, Israel, March 2000, pp.1501–1510

[9] A. Pitsillides, G. Stylianou, C. Pattichis, A. Sekercioglu, A. Vasilakos, "Investigation of the Performance of EP-BAVP (Evolutionary Programming for Virtual Path Bandwidth Allocation)", Int. Conf. On Telecommunications, Port Carras, Greece, 1998, pp. 310–314