Adaptive QoS Platform in Multimedia Networks

Mahmoud Sherif¹, Ibrahim Habib¹, Mahmoud Naghshineh², and Parviz Kermani²

¹ CUNY Graduate School and
Department of Electrical Engineering
The City College of New York, USA
{sherif, eeiwh}@ee-mail.engr.ccny.cuny.edu

² IBM Research Division,
T.J. Watson Research Center
Yorktown, USA

Abstract. In an adaptive multimedia environment, each of the multimedia substreams (i.e. video, audio and data) has its own distinct quality of service (QoS) requirements (e.g. cell loss rate, delay, jitter, etc.). These requirements constitute a certain QoS level. In contrast to the static approach, each substream declares a preset range of acceptable QoS levels (e.g., high, medium, low) instead of just a single one. This range of QoS levels is defined in a user-defined profile (UDP). In this paper, we suggest a channel borrowing algorithm based on an adaptive QoS platform. In a channel borrowing algorithm, an acceptor cell that has used all its nominal channels can borrow free channels from its neighboring cells (candidate donors) to accommodate new calls. In our suggested algorithm, an acceptor cell can borrow from any neighboring (donor) cell as long as this donor cell has some channels available after satisfying a minimum QoS (minQ) level defined in the UDP. A donor cell assigning QoS levels (to calls under its coverage) higher than the minQ levels defined in the UDP will declare those channels as available for borrowing by other acceptor cells. When a channel is borrowed, several other cells are prohibited from using it due to channel locking. The proposed channel borrowing algorithm differs in the way a free channel is selected from a donor cell to be borrowed by an acceptor cell. The criteria for choosing the free channel include not only the number of free channels but also the QoS levels in the donor cell. The criteria is also extended to include the effect of channel locking on the number of free channels and the QoS levels on the locked cells.

1 Introduction

In wireless multimedia networks, admission control is required to reserve resources in advance for calls requiring guaranteed services. In the case of a multimedia call, each of its substreams (i.e. video, audio and data) has its own distinct quality of service (QoS) requirements (e.g. cell loss rate, delay, jitter, etc.). The network attempts to deliver the required QoS by allocating an appropriate amount of resources (e.g., bandwidth, buffers). The negotiated QoS requirements constitute a certain QoS level

that remains fixed during the call (static allocation approach). Accordingly, the corresponding allocated resources also remain unchanged. In a previous paper [1], we presented and analyzed an adaptive allocation of resources algorithm based on genetic algorithms. In such an adaptive networking environment, calls can be admitted to the system even if the available bandwidth is not sufficient to satisfy their highest QoS guarantees. To accomplish this, the proposed algorithm will try to degrade the QoS levels of the existing calls in order to free some bandwidth. Each call is considered to have a pre-defined minimum QoS level (minQ) and a maximum QoS level (maxQ) that are defined in a user-defined profile (UDP). The (minQ) level corresponds to the minimum set of QoS requirements that the application is willing to tolerate. On the other hand, the (maxQ) level corresponds to the best QoS requirements a call can obtain whenever the bandwidth is available. In the case of a multimedia call, each of its substreams (i.e. video, audio and data) will have its own QoS level ranging from high to medium to low. For example, one user might be willing to tolerate a low video quality, but requires high audio quality and high speed data service. Such user will be granted an amount of bandwidth that corresponds to its declared minQ level. Whenever there is available bandwidth, the user might obtain an extra amount of bandwidth up to its declared maxQ level. It should be noted that the UDP approach also defines different “grades of service” at different costs that the users can subscribe to. For example, a network provider may choose to offer two grades of services with different cost structures. A subscriber to a premium service will pay high dollar for the call but will be allocated a UDP with the uppermost range of quality levels, whereas a subscriber to an economy service will have a UDP range of qualities that is significantly lower than the premium one.

In fig.1 it is shown that when a call is being admitted to the network, it presents its traffic parameters and its declared UDP to the admission controller/scheduler. The admission controller/scheduler will then calculate the required bandwidth according to the traffic parameters presented to it. The calculated bandwidth and the declared UDP are then saved in a database in order to be used by the proposed optimization algorithm. Fig.2 shows an example of the UDP where a call (i) is presenting both the maxQ and the minQ levels to the admission controller/scheduler. All QoS levels between the declared maxQ and minQ are allowed to be assigned to call i. In this example, the maxQ level corresponds to the highest video quality, the highest audio quality and the highest speed for data service. The minQ level corresponds to medium video quality, low audio quality and low speed for data service. Any QoS level between these declared levels are allowed to be allocated to this call i.

In this paper, we suggest a channel borrowing algorithm based on the adaptive allocation of resources algorithm described in [1]. In a channel borrowing algorithm, an acceptor cell that has used all its nominal channels can borrow free channels from its neighboring cells (candidate donors) to accommodate new calls. In our suggested algorithm, an acceptor cell can borrow from any neighboring (donor) cell as long as this donor cell has some channels available after satisfying the minimum QoS (minQ) level defined in the (UDP). A donor cell assigning QoS levels (to calls under its coverage) higher than the minQ levels defined in the UDP will declare those channels as available for borrowing by other acceptor cells. A channel can be borrowed by a cell if

the borrowed channel does not interfere with existing calls. Furthermore, when a channel is borrowed, several other cells are prohibited from using it. This is called channel locking. The number of such cells depends on the cell layout and the type of initial allocation of channels to cells. In contrast to static borrowing, channel borrowing strategies deal with short-term allocation of borrowed channels to cells. Once a call is completed, the borrowed channels are returned to its nominal cell. The proposed channel borrowing algorithm differs in the way a free channel is selected from a donor cell to be borrowed by an acceptor cell. In our suggested algorithm, the criteria for choosing the free channel include not only the number of free channels but also the QoS levels in the donor cell. The criteria is also extended to include the effect of channel locking on the number of free channels and the QoS levels on the locked cells.

Throughout the description of the suggested channel borrowing algorithm, we are going to consider the hexagonal planar layout of the cells. A cell cluster is a group of identical cells in which all of the available channels (frequencies) are evenly distributed. The most widely used plan is the N=7 cell cluster [2]-[4] where the number of available channels are distributed evenly among 7 cells, which then repeats itself over and over according to Fig. 3. We are going to consider N=7 throughout the discussion in this paper.

Fig. 1. Database used by the algorithm

	Video	Audio	Data
maxQ	High	High	High
	High	High	Medium
	High	High	Low
	...	...	...
	...	...	...
	Medium	Medium	Low
minQ	Medium	Low	Low

Fig. 2. Example of a User-Defined Profile (UDP)

2 Suggested Channel Borrowing Algorithm

As opposed to the *Borrow from the richest* [5] algorithm, the suggested algorithm takes into account the effect of channel locking when choosing a candidate channel for

borrowing. Furthermore, the suggested algorithm takes into account not only the number of available channels but also the average QoS level of each candidate cell. This allows the algorithm to try to maximize the average QoS level of the calls existing in the system in addition to minimizing the call blocking probability.

Fig. 3. A seven-cell reuse plan

Fig. 4 shows the block diagram of the adaptive allocation of resources algorithm when the channel borrowing algorithm is added. As shown, the algorithm consists of three main modules: genetic algorithm modules I and II in addition to the channel borrowing module III. The reason to use modules I and II is to divide the problem into smaller easier to handle sub-tasks since the search space is too large for a single GA. Module I assigns fair bandwidth allocations to the existing calls; whereas module II maximizes the capacity utilization by assigning any available bandwidth (left over from module I allocations) to the existing calls. Eventually, the system will not force a call to drop unless there is not enough bandwidth to satisfy its minQ level. Module III is added as an important enhancement layer to allow for channel borrowing to take place in such an adaptive QoS environment.

Fig. 4. Block Diagram of the Proposed Adaptive QoS Platform

Module I is triggered whenever a call arrival or departure takes place. The inputs to module I include: (1) the capacity of the system (physical cell capacity), (2) the number of calls (N) after the arrival/departure and (3) the bandwidth information for these calls and their preset UDPs (which are fetched from the database). Furthermore, QoS levels from the previous allocation are also fed back to module I.

Module I searches for the QoS levels that correspond to the bandwidth allocation closest to the "fair" bandwidth. The evaluation of the optimization function is based upon the following rules:

$$B_{fair} = \frac{C_{tot}}{N} \quad (1)$$

Where

B_{fair} is the fair bandwidth Allocation, and

C_{tot} is the total physical cell capacity.

In Eq.2, the calculation of the fair bandwidth is applicable to calls of similar classes (i.e., having the same type of multimedia substreams)

For each of the existing call, the following equations are applied:

$$B_{\text{modl}}(i) \in B_i(Q_i) \quad (2)$$

$$B_{\text{modl}}(i) = \text{minimum}(\text{absolute}(B_{\text{fair}} - B_i(Q_i))) \quad (3)$$

$$B_{\text{modl}}(i) \leq B_{\text{fair}} \quad (4)$$

Where

i is the call number,

$B_{\text{modl}}(i)$ is the bandwidth requirement corresponding to the output QoS level from module I,

$B_i(Q_i)$ is the bandwidth requirement corresponding to the QoS level Q_i , and

B_{fair} is the fair bandwidth.

Once the fair allocations are determined by module I, if any of the assigned QoS levels in module I is less than the corresponding minimum QoS level minQ , then this minQ level is assigned to the corresponding call. Then this data is sent to the module chooser for further processing. Fig. 5 shows the module chooser. It starts by calculating the total bandwidth needed by all existing calls if granted the QoS levels assigned by module I. If the total bandwidth exceeds the cell capacity, then module III is triggered to try to borrow some bandwidth from the neighboring cell. Otherwise, module II is triggered to try to take advantage of any bandwidth left over from module I.

Module II, on the other hand, is designed to take advantage of any available bandwidth left over from module I. It tries to maximize the link capacity utilization and thus maximizing the QoS levels for the calls

In the following section, we are going to describe module III (channel borrowing algorithm) in more details.

2.1 Module III

Throughout the description of module III, we are going to consider the cell layout shown in Fig. 6. The cell layout shown is a seven-cell cluster surrounded by a second tier of 11 cells. The total number of cells in this two-tier layout is 19 cells. The cell reuse plan is $N=7$. Each color in Fig. 6 represents a distinctive set of channel frequencies.

The reason for choosing this specific layout is that module III is going to consider the first two tiers of the neighboring cells only. Cell A is the acceptor cell. It is surrounded by two tiers of cells. Cell A can borrow from any of its first tier neighbors (cell 1, cell 2, ..., cell 6). Furthermore, it can borrow channels from one and only one neighbor (no multiple donor cells are allowed). Borrowing any number of channels from any of these 6 cells will cause these channels to be locked in another two cells. To illustrate the concept of channel locking, let us assume that cell A is going to choose to borrow a number of free channels from cell 2. This will cause these borrowed channels to be locked in both cells 14 and 17. For this cell layout ($N=7$), the number of cells affected by channel locking is always 2 cells. We are going to denote

these 2 cells as *affected_cell1* and *affected_cell2*. In the suggested channel borrowing algorithm, each cell is required to keep track of the following parameters: (1) The number of available free channels if the existing calls are assigned their minQ levels, (2) the number of available free channels in *affected_cell1* if the existing calls are assigned their minQ levels and (3) the number of available free channels in *affected_cell2* if the existing calls are assigned their minQ levels. The information regarding the available number of channels in the *affected_cell* can be collected either periodically or on a need basis (a message sent from the cell to the *affected_cell* requesting immediate information regarding the number of available free channels).

Fig. 5. Module Chooser

Assuming that Cell A (the acceptor cell) needs N_b number of channels to borrow. The number of channels needed (N_b) should be calculated from the following equation:

$$N_b = (B_{\text{tot}} - C_{\text{tot}}) / Ch_{\text{size}} \quad (5)$$

Where
 B_{tot} is the total bandwidth needed by existing calls if assigned their minQ level,
 C_{tot} is the total physical cell capacity and Ch_{size} is the size of each channel.

Fig. 6. System Under Study

Cell A will then send a *borrow_request* message to each of its first tier neighbor cells. Each of the neighboring cells receiving the *borrow_request* message will calculate the number of available channels (N_{av}) if its calls are granted their minQ level and compare it to the requested number of channels (N_b). If $N_{av} > N_b$, then the cell will send back a negative acknowledge (NACK) message informing the acceptor of the denial of its request. This is shown in Fig. 7 (a). Otherwise, it will send a message to each of the affected cells (due to channel locking) requesting information regarding the number of available channels ($N_{av}(\text{affected_cell})$). The information regarding the available channels does not have to be gathered when the *borrow_request* message is received. As mentioned before, this information can be gathered periodically and kept ready for any *borrow_request* message. If the returned values $N_{av}(\text{affected_cell}) > N_b$

then this means that the requested number of channels is available for borrowing and that the locked channels (in the affected_cells) are also available. An acknowledgement (ACK) message is then sent back to the acceptor cell. Otherwise, a NACK message is sent back to the acceptor cell denoting that the locked channels will cause some calls to be dropped in the affected_cell. The following routine can be executed by the neighboring cells upon the receipt of a *borrow_request* message.

Fig. 7. Flow of messages between acceptor, candidate and affected cells (a) $N_{av} < N_b$, (b) $N_{av} > N_b$

Fig. 7 (b) shows the flow of the messages if there are enough available channels in the neighboring (candidate) cell ($N_{av} > N_b$). If the affected cells return an ACK message, this implies that the locked channels due to the borrowing process will not cause any calls to be dropped in these cells. This is due to the fact that when the candidate cell sends a *check_no_channel* message to any of the affected cells, it will calculate the number of available channels after assigning the minQ levels to the calls existing in this cell. Therefore, immediately afterwards, the candidate cell send an ACK message to the acceptor cell informing it that it can fulfill the *borrow_request* message.

The candidate cell will then wait for each of the affected cells to send a message carrying information regarding the average QoS (*average_qos*) levels of the existing calls and the number of channels that would be available if the borrowing process is executed (N_{av}). The *average_qos* is calculated assuming the borrowing process has been executed. This will give an indication of how the borrowing process affects the QoS levels of the existing calls. In the meantime, the candidate cell will calculate the average QoS levels of its existing calls based on the same criteria. Notice that the candidate cell can calculate the average QoS level while the other affected cells are doing the same thing in parallel. Once all the information is available for the candidate cell, it will send it in a message to the acceptor cell.

Once all messages from all the neighboring (candidate) cells are received, the acceptor cell will start processing the information searching for the best candidate cell to become the donor. First, all cells sending back NACK messages are removed from the

candidate list. The information of each of the remaining cells in the candidate list include: (1) a vector of average QoS levels of the candidate cell, affected cell 1 and affected cell 2 ($average_qos[]$), and (2) a vector of available number of channels (N_{av}).

The acceptor cell will use the following equation to calculate the cost of borrowing the requested number of channels from each cell. The cost value of each cell in the current candidate list is denoted by $borrow_cost$.

$$borrow_cost = \left(\frac{\min(N_{av})}{\max_channels} - \frac{\min(average_qos)}{64} \right) \quad (6)$$

Where $\max_channels$ is the maximum number of channels in each of the cells. Note that the value of $average_qos$ ranges from 1 to 64 (1 being the best QoS level, and 64 being the least). Therefore, the value of $borrow_cost$ ranges from -1 to 1. The higher the $borrow_cost$ value, the better is the candidate cell. Therefore, the acceptor cell will choose the one with the highest $borrow_cost$ value.

3 Simulation Results

The system under study is shown in Fig. 6. The system consists of an acceptor cell (cell A) surrounded by two tiers of cells. The cell layout is a seven-cell cluster system ($N=7$). Each cell has a physical capacity of 60,000 ATM cells(packets)/sec. This is equivalent to 25 Mbps. For a 30 Kbps channels, each cell is assigned 848 channels ($\max_no_channels=848$). The dotted line represents a street with high call traffic load. Along the street are cells 15, 5, A, 2 and 9. In our simulation, the traffic load in these cells is characterized by a high traffic load. As shown in the cell layout, each of the following 6 sets represent a group of cells assigned the same set of channel frequencies: (1) cells {1,12,15}, (2) cells {2,14,17}, (3) cells {3,7,16}, (4) cells {4,9,18}, (5) cells {5,8,11} and (6) cells {6,10,13}. Therefore, the $no_of_affected_cells=2$ (i.e. when one cell is a donor cell, 2 other cells are affected due to channel locking).

The traffic load is multimedia traffic. In our simulation, the video component of the multimedia call was obtained using the MPEG-1 coded movie *Star Wars* as an example of high activity video traffic. Voice and Data substreams were generated according to an exponential distribution. The MPEG-1 coded movie generates a frame sequence of *I B B P B B P B B P B B*. There are 24 frames per second [6]. This set of data is described in [7]. Each call has an average duration of 5 minutes of movie time. Therefore the movie was actually divided into 24 different calls. To generate three QoS levels for the video substream, we adopted the algorithm described in chapter 2 where the B frames are dropped to obtain a lesser QoS level, then both the B and P frames are dropped to obtain the least video quality. Whereas, the high, medium and low QoS levels for audio were generated by varying the average bit rate of the audio source from 32 to 16 to 8 Kbps, respectively. Similarly, QoS levels for data were generated by varying the data rate from 64 to 32 to 16 Kbps.

Fig. 8. . Call Blocking versus offered load

Fig. 8 shows the percentage call blocking rate of the system under study versus the percentage offered load. Using the suggested channel borrowing algorithm reduced the call blocking rate of the system. This takes place under an offered load between 40% to 80%. Below 40%, there is no need for using the channel borrowing algorithm as there are enough channel in the system to satisfy at least the minQ levels of the exist- ing calls.

Fig. 9. QoS enhancement versus offered load

Fig. 9 shows the enhancement in the QoS offered versus the offered load. This curve shows another contribution of the suggested algorithm. Due to the adaptive QoS platform of the suggested system, the suggested channel borrowing algorithm is also able to enhance the QoS level of existing calls. The enhancement starts when the offered load is at 40%. It reaches a peak value of 7.8% at 45% offered load. It then starts to decrease reaching 0% at 78% offered load.

References

1. Sherif, M.R., Habib, I.W., Naghshineh, M., Kermani, P.: Adaptive Allocation of Resources and Call Admission Control in Wireless ATM Using Genetic Algorithms. *IEEE Journal Select. Areas Comm.* February, 2000.
2. Michalewicz, Z.: *Genetic Algorithms + Data Structures = Evolution Programs*. 3rd edn. Springer-Verlag, Berlin Heidelberg New York (1996).
3. Farouque, S.: *Cellular Mobile Systems Engineering*. Artech House, Boston, 1996.
4. Holtzman, J., Goodman, D.: *Wireless Communications, Future Directions*. Kluwer Academic Publishers, 1993.
4. Rappaport, F.: *Wireless Personal Communications*. Kluwer Academic Publishers, 1993.
5. Anderson, L.: A Simulation Study of Some Dynamic Channel Assignment Algorithms in High Capacity Mobile Telecommunications System. *IEEE Trans. On Vehicular Tech.* Vol VT-22, 1973, p.210.
6. Garret, M., Fernandez, A.: Variable Bit Rate Video Bandwidth Trace Using MPEG Code. Bellcore, <ftp://ftp.bellcore.com/pub/vbr.video.trace/>, 1992.
7. Garret, M.: *Contributions Towards Real-Time Services on Packet Networks*. Ph.D. Dissertation. Columbia University. May 1993