Dynamic Programming Optimization of Multi-rate Multicast Video Streaming Services

Nestor Michael C. Tiglao[1,3], Jânio Miguel E.F. Monteiro[1,2], António Manuel R. C. Grilo[1], Mário Serafim S. Nunes[1], João Manuel F. Xavier[4]

[1]Instituto de Engenharia de Sistemas e Computadores Investigação e Desenvolvimento, INOV, Instituto Superior Técnico, Lisboa, Portugal

[2]Universidade do Algarve, Faro, Portugal

[3] Electrical and Electronics Engineering Institute, University of the Philippines, Diliman, Quezon City, Philippines

[4]Instituto de Sistemas e Robótica, Instituto Superior Técnico, Lisboa, Portugal

*Corresponding author: Email: nestor@eee.upd.edu.ph

ABSTRACT

In large scale IP Television (IPTV) and Mobile TV distributions, the video signal is typically encoded and transmitted using several quality streams, over IP Multicast channels, to several groups of receivers, which are classified in terms of their reception rate. As the number of video streams is usually constrained by both the number of TV channels and the maximum capacity of the content distribution network, it is necessary to find the selection of video stream transmission rates that maximizes the overall user satisfaction. In order to efficiently solve this problem, this paper proposes the Dynamic Programming Multi-rate Optimization (DPMO) algorithm. The latter was comparatively evaluated considering several user distributions, featuring different access rate patterns. The experimental results reveal that DPMO is significantly more efficient than exhaustive search, while presenting slightly higher execution times than the non-optimal Multi-rate Step Search (MSS) algorithm.

Keywords: Quality of Experience (QoE), Dynamic Programming , Quality of Service (QoS), Internet Protocol Television (IPTV), Multi-Rate

INTRODUCTION

The possibility of ubiquitous Internet access brought about by the wide dissemination of mobile technologies led to a greater heterogeneity of access networks and terminal equipments with different capabilities, which, coupled with the natural unpredictability of Internet QoS, constitutes a technological challenge for Internet-based video distribution services. While current networking technologies and architecture have enabled a significant increase of the transmitted video quality, this can mainly benefit high capability terminals operating from high capacity access networks. However, in order to cover the entire market spectrum, the video distribution services must also satisfy users using lower capability terminals operating from lower capacity access networks (e.g. mobile handsets using GPRS).

In large scale IP Television (IPTV) and Mobile TV distributions, the video signal is typically encoded and transmitted using several quality streams, over IP Multicast channels, to several groups of receivers, which are classified in terms of their reception rate. As the number of video streams is usually constrained by both the number of TV channels and the maximum capacity of the content distribution network, the selection of the best transmission rate of each video stream should be dynamically adjusted trying to meet the maximum user level of satisfaction. For huge numbers of dynamically changing heterogeneous receivers and video streams, the exhaustive search for the optimal mix of stream quality levels is unbearable given the real-time adaptation requirements.

Regarding video quality metrics, while networking Quality of Service (QoS) metrics usually provide good hints regarding the true quality experienced by the user, their relationship is nonlinear. That is the reason why user centric metrics have been used in television services for more than twenty years to evaluate video quality, based on subjective assessment metrics that are obtained using a panel of human evaluators in standard defined methods. These metrics measure the impairment caused by a diversity of factors on the Human Visual System (HVS) and constitute what is called Quality of Experience (QoE) metrics. When available, the latter constitute more reliable criteria for the optimization of the video streaming service.

This paper proposes a QoE oriented multi-rate optimization algorithm designated the Dynamic Programming Multi-rate Optimization (DPMO) algorithm. As the name suggests, DPMO is based on Dynamic Programming concepts. It is able to find the set of video stream transmission rates that optimizes the average QoE of a set of users, featuring a computational complexity that is significantly lower than the exhaustive search. The proposed mechanism was comparatively evaluated considering several user distributions, featuring different access rate patterns.

RELATED WORKS ON MULTI-RATE SWITCHING

In the early video distribution systems over Internet, video streaming was provided at a single rate. This scheme was suitable for users with connection speeds close to the average media rate, which implies that users with large access speeds were unable to exploit their transmission capabilities and users with slower connections could not view any content at all.

A natural evolution of this single rate system is to offer content at individually encoded bitrates chosen in order to cover the available connection types (ADSL, Cable, WIMAX, WiFi, etc.). However, this proposal does not optimize the quality of the streams sent to the different users due to the fact that the bottleneck capacity is not defined only by the access link but also by the core network and the streaming server, which can also limit the available bandwidth due to congestion or high load.

Multi-rate switching or stream replication (Li & Liu, 2003) is a dynamic extension of the individually encoded bitrates. It allows mid-stream switching between different rates according to the detected network conditions. Several commercial products (Real Networks, 2002) (Birney, 2000) use this scheme, as for instance SureStream technology from Real Networks (Conklin, et al., 2001). The innovation of this approach lies in the use of multiple representations of the original content (each one encoded at a different bitrate) optimized for different access network and load conditions. The result is a single file wherein all encoded streams are bundled. During the streaming session, the player monitors the bandwidth and the loss characteristics of the connection and instructs the server to switch to the stream that will provide the best perceptual quality.

More recently, Microsoft introduced Smooth Streaming (Zambelli, 2009), an adaptive streaming technique where the video/audio source is encoded at multiple bitrates, generating multiple chunks of various sizes each with 2 to 4 seconds of video. Because web servers usually deliver data as fast as network bandwidth allows, the client can easily estimate user bandwidth and decide to download larger or smaller chunks ahead of time. Smooth streaming could be used for stored content or for live content, which allows to dynamically adapt the rates of the different streams in real-time according to the network and user conditions.

In the OLYMPIC project (Patrikakis, et al., 2003), a platform that was able to deal with multi-rate switching was developed. It consisted of servers, reflectors (proxy streaming servers) and transcoders. Servers are responsible for streaming stored or live content and are considered as the point where any stream originates. Users may request a particular stream directly from a server or the request may be submitted to a proxy streaming server. In the latter case, content is transmitted to the proxy node before being forwarded to the client. A proxy streaming server node may serve a large number of clients by replicating and subsequently forwarding packets received from the server, thus reducing the server's workload. For example, assume that a large number of users request the same stream from a specific proxy streaming server. In this case, only one copy of the stream must

reach the proxy streaming server. There the stream is replicated and transmitted to the various clients, improving scalability as the system's capacity (in terms of users) can be increased by the deployment of additional proxy streaming servers.

Nunes, Monteiro, & Grilo (2009) proposed an algorithm called Multi-rate Step Search (MSS) to efficiently search the optimal set of streams bitrates that maximizes the QoE of a set of users (see section III). MSS fixes the lowest data rate to the value of the lower user access rate and then sequentially adds additional streams whose bitrates must always match available user access rates. For each added stream, an exhaustive search is performed so that its data rate will correspond to the *argmax* of the QoE when fixing the bitrates of the streams that are already on the system. After the new bitrate is found, all bitrates are sequentially adjusted until a local maximum is found. Then the algorithm proceeds to add another stream. The process continues until the target number of streams is reached. Although MSS does not provide any guarantees of finding the optimal set of streams, it is very efficient from the point of view of execution time and its results are usually not very far from the optimal ones.

Yang, Kim, & Lam (2000) have proposed the use dynamic programming to determine the optimal partitioning of multi-rate multicast receivers. They have formulated the partitioning problem as an optimization problem to maximize the sum of receiver utilities subject to some loss tolerance constraints for a general class of utility functions. In particular, they considered two different receiver utility functions: one is based on Inter-Receiver Fairness (IRF), which was first defined in (Jiang, Ammar, & Zegura, 2000), and the other based on isolated receiver rates. In this study, however, we used a different utility function, one that is based on the quality of experience (QoE) metric as explained in the next section.

QOE VIDEO MULTI-RATE STREAMING OPTIMIZATION: PROBLEM STATEMENT

The problem of rate adaptation as it is related to the OLYMPIC project platform (Patrikakis, et al., 2003) may be formulated as follows: Consider a multimedia stream S encoded at K different bitrates b_i, $i \in \{1, ..., K\}$, resulting in K streams $\{S_1, S_2. .., S_K\}$. A streaming server proxy node relays the streams to various clients, where the clients are categorized to K groups $\{G_1, G_2. .., G_K\}$. according to the received stream (for example, a terminal receiving stream S_i (encoded at bitrate b_i) belongs to group G_i). A group G_i contains n_i terminals $t_{i,j}$, where $j \in \{1, ..., n_i\}$,. This configuration is illustrated in Figure 1.

The objective is to develop algorithms that are able to respond to varying network and terminal conditions by dynamically adapting a subset of the parameters K and b_i, in order to achieve maximization of user perceived quality (QoE). The parameters that affect stream quality include the available bandwidth, network congestion, terminal CPU load and the number of served terminals.

Considering a fixed number of K streams, each stream encoded at a fixed bitrate b_i , a specific case of the generally stated problem emerges. The goal is to find the optimal distribution of terminals to available streams whereby minimization of terminal-side packet loss and maximization of network utilization and perceived quality is achieved. It is worth noting that the optimization goal is twofold. Firstly, we wish to obtain a specific (optimal) distribution of terminals as a function of their performance (specifically the sustained packet loss). Secondly, we wish to design an algorithm for enforcing this distribution dynamically over time, as a response to fluctuating terminal performance.

As a simple example of the aforementioned algorithm, consider a streaming server proxy node that relays 3 streams S_1 (128 kbps), S_2 (256 kbps) and S_3 (512 kbps). The streams are received by terminals $t_1 t_2$, and t_3 respectively. At some point the system is informed (by means of a reporting mechanism presumably established between the terminals and the streaming server proxy node, e.g. RTCP reports) that terminal t_2 is sustaining a significant packet loss that is deemed as unacceptable on the basis of the defined policy. As a response, the streaming server proxy node 'switches' the 512 kbps transmitted to t_2 with the 128 kbps stream (represented by the big vertical arrow in Figure 1). This results in a decrease of the packet loss experienced by the terminal that leads to a smoother terminal-side

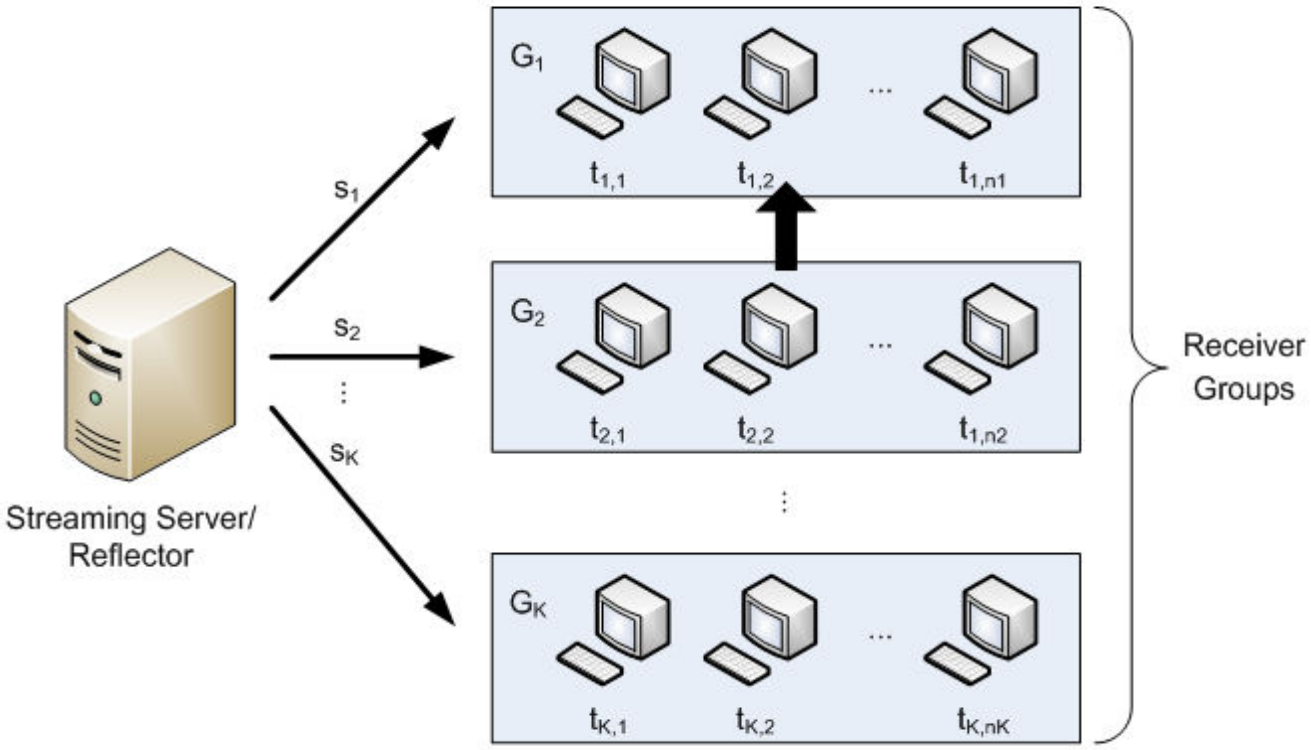

Figure 1. Terminal grouping according to the received stream bitrate.

playback of the received stream as well as better network utilization.

Furthermore, consider an algorithm that examines the packet loss of each terminal in a group and determines the worst and best performing terminals in the group. Then, the algorithm initiates a 'switch' of the stream received by the worst performing terminal with a stream encoded at a lower bitrate and another 'switch' of the stream received by the best performing terminal with a stream encoded at a higher bitrate.

The QoE video streaming optimization aims to maximize the objective function Q_t that represents the sum of quality (Q_i) of each stream (s_i) that exist at one instant:

$$Q_t = \sum_{i=1}^{K} Q_i \cdot n_i \qquad (1)$$

ITU-T has standardized a parametric computational model, as Recommendation G.1070 (ITU-T, 2007), for evaluating QoE of video telephony. The model estimates the QoE of video-telephony services based on quality design/management parameters in terminals and networks. For H.264 the quality of a video stream without losses can be approximated by the expression (b_i in kbps):

$$Q_t = 1.2 \log_{10}(1 + b_i) \qquad (2)$$

Notice that Q_i is subject to the constraint , $b_i \leq AR_u$, $\forall_{u \in Gi}$, where AR_u is the access rate that user u experiences. A straightforward approach to solve this problem is to enumerate all possible bitrates combinations and choose the one that maximizes Q_t. However, this approach is combinatorial in nature and the time complexity of the solution can grow exponentially with the number of variables. The MSS algorithm (Nunes, Monteiro, & Grilo, 2009) tries to tackle this problem in an efficiently in a heuristic way, but it offers no guarantees of optimality.

DYNAMIC PROGRAMMING MULTI-RATE OPTIMIZATION

Dynamic Programming (DP) is an optimization technique for solving complex problems by breaking them into smaller sub-problems. It is especially useful in problems where one needs to make decisions one after another. In general, problems that can be solved through dynamic programming possess two characteristic ingredients: optimal substructure and overlapping sub-problems (Cormen, et al., 2001). A problem has an optimal substructure if the optimal solution of the problem can be derived from the optimal solution of the sub-problems. An optimization problem is said to have overlapping sub-problems if the algorithm solves the same problem over and over again.

The problem can be modeled as a system with an initial state and several possible final states. Starting from the initial state, a decision has to be made to move to a next state where a certain reward or benefit is associated with that decision, until a final state is achieved. The total reward is the sum of the rewards accumulated along the way from the initial state to the final state. An optimal solution is one that maximizes the total reward.

The application of DP to the optimal rate allocation problem is designated the DP Multi-rate Optimization (DPMO). Each user experiencing an access rate AR_j, $\in \{1, ..., M\}$ shall be assigned rate $R_j = b_i$, $i \in \{1, ..., K\}$. The DPMO algorithm analyses each user in turn, departing from its initial state at the lowest available access rate (AR_1) and then progressing to the highest access rate (AR_M). For users experiencing AR_1 , rate b_1 is introduced and this rate necessarily corresponds to AR_1, i.e. $R_1 = b_1 = AR_1$. For every other access rate AR_j, one of two possible options can be chosen:

(1) To assign the previous rate b_i again (i.e. $R_j = b_i$), keeping the number of used streams for the moment;
(2) To add rate $b_{i+1} = AR_j$ to the set of used stream rates, making $R_j = b_{i+1}$.

For example in Figure 2, given $M = 10$ and $K = 3$. At state 4, the reward in choosing the first option is $1.2 \log (1 + Ar_1) \cdot n_4$ whereas at state 5, the reward is $1.2 \log (1 + Ar_5) \cdot n_5$ taking the second option, assuming that where n_4 and n_5 are the number of users that experience access rates Ar_4 and Ar_5, respectively. Although it was not clearly evident that this problem can be solved through DP, realizing that decisions on when to jump to a higher rates subject to a finite number of jumps gives the clue that DP can be used. DPMO can be best understood with an example.

The DPMO implementation requires two state variables, $k \in \{1, ..., j - 1\}$ and $l \in \{0, 1, ..., K - 1\}$, where k is the index of the rate allocated to users

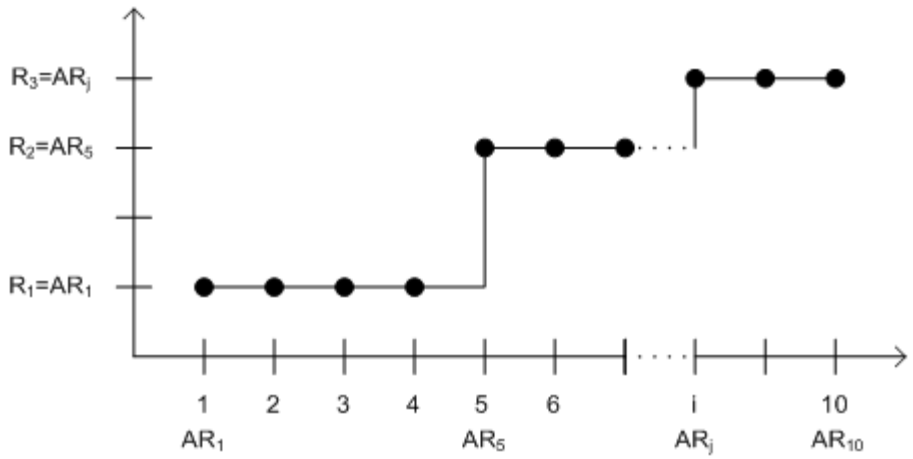

Figure 2. DP Optimization of multi-rate video streaming.

experiencing AR_j and l is the number of available steps (stream rates). These two variables serve as indices for the decision tables computed by the algorithm. For each <k, l> pair, two items are stored: one is the optimal option taken and the other is the corresponding reward for that option, where option 0 means the previous rate is maintained and option 1 means that there is a jump to a new rate. The rewards consist of QoE values calculated with expression (2).

DPMO solves the problem backwards (i.e., from the final state to the initial state), such that $j \in \{M, M-1, \ldots, 1\}$. Referring to the example in Figure 2, DPMO calculates the table for $j = 10$ first. For $l = 0$, the previous rate indicated by k is maintained while for $l > 0$, jumping to a higher rate will always provide a higher Q_t since $AR_1 < AR_2 < \cdots < AR_M$ (see Table 1). Each $\langle k, l \rangle$ value in the decision table is designated as $V_{10}^{k,l}$. Note that column 10 is empty and is blackened in Table 1.

Table 1. Decision table for $j = 10$.

	k											
l	1			2			3		…	9		10
0	0	$1.2\log(1+AR_1)\cdot n_{10}$	0	$1.2\log(1+AR_2)\cdot n_{10}$	0	$1.2\log(1+AR_3)\cdot n_{10}$	..	0	$1.2\log(1+AR_9)\cdot n_{10}$		
1	1	$1.2\log(1+AR_{10})\cdot n_{10}$	1	$1.2\log(1+AR_{10})\cdot n_{10}$	1	$1.2\log(1+AR_{10})\cdot n_{10}$	..	1	$1.2\log(1+AR_{10})\cdot n_{10}$		
2	1	$1.2\log(1+AR_{10})\cdot n_{10}$	1	$1.2\log(1+AR_{10})\cdot n_{10}$	1	$1.2\log(1+AR_{10})\cdot n_{10}$	..	1	$1.2\log(1+AR_{10})\cdot n_{10}$		

Moving to $j = 9$, the table is calculated in the same manner but for each <k, l> pair, the algorithm considers which option provides a higher accumulated QoE value ($V_9^{2,0}$) considering the reward in j = 10, and stores the corresponding option and accumulated reward in the table. For example, for k = 2, $l = 0$, there is no other option but option 0, because no jumps in the rate are possible. The corresponding accumulated value is ($V_9^{2,0}$) = 1.2 log $(1 + AR_2) \cdot n_9 + (V_{10}^{2,0})$, which goes into the respective table entry.

For $k = 2$, $l = 1$, two options are possible:

- Option 0[1]: ($V_9^{2,0}$) = 1.2 log $(1 + AR_2) \cdot n_9 + (V_{10}^{2,1})$
- Option 1[2]: ($V_9^{2,1}$) = 1.2 log $(1 + AR_2) \cdot n_9 + (V_{10}^{9,0})$

The algorithm chooses the option that gives the higher ($V_9^{2,1}$) and stores it in the respective entry in the table for $j = 9$. A similar procedure is carried out for the other $\langle k, l \rangle$ pairs.

The analysis will now jump to the last two steps. Table 2 shows the decision table for $j = 2$. Columns for $k = 2,\ldots, 10$ are empty and are blackened. The DPMO algorithm terminates and arrives at the optimal solution when it reaches $j = 1$, where the optimal value is 1.2 log $(1 + AR_1) \cdot n_1 + (V^{1,2})$. The optimal rates are then determined by checking each decision table from $j = 1$ to $j = 10$ where option 1 is taken.

Table 2. Decision table for $j = 2$.

	k					
l	1		2	…	9	10
0	0	$1.2\log(1+AR_1)\cdot n_2 + V_3^{1,0}$		…		
1	1	$1.2\log(1+AR_2)\cdot n_2 + V_3^{2,0}$		…		
2	1	$1.2\log(1+AR_2)\cdot n_2 + V_3^{2,1}$		…		

To visualize the operation of DPMO, consider three peaks access rate distribution (described in section V). Executing DPMO results in a decision graph with 62 states as shown in Figure 3. DPMO starts computing

[1] The value of l remains equal to 1 in $V_{10}^{2,l}$ because no jump was made.

[2] The value of l is decremented by 1 and becomes 0 in $V_{10}^{2,l}$ meaning that further jumps will be possible.

the decision table of the final state, $j = 62$. Then, it proceeds to compute the tables for the other states until it terminates at $j = 1$, obtaining the optimal Q_t. The optimal decision as well as the corresponding optimal bitrates b_i, $i \in \{1, 2, ..., K\}$ can be obtained by going through the decision tables tracing from state $j = 1$ to state $j = 62$ where option 1 is chosen. Take note that at $j = 1$, option 1 is chosen automatically. At

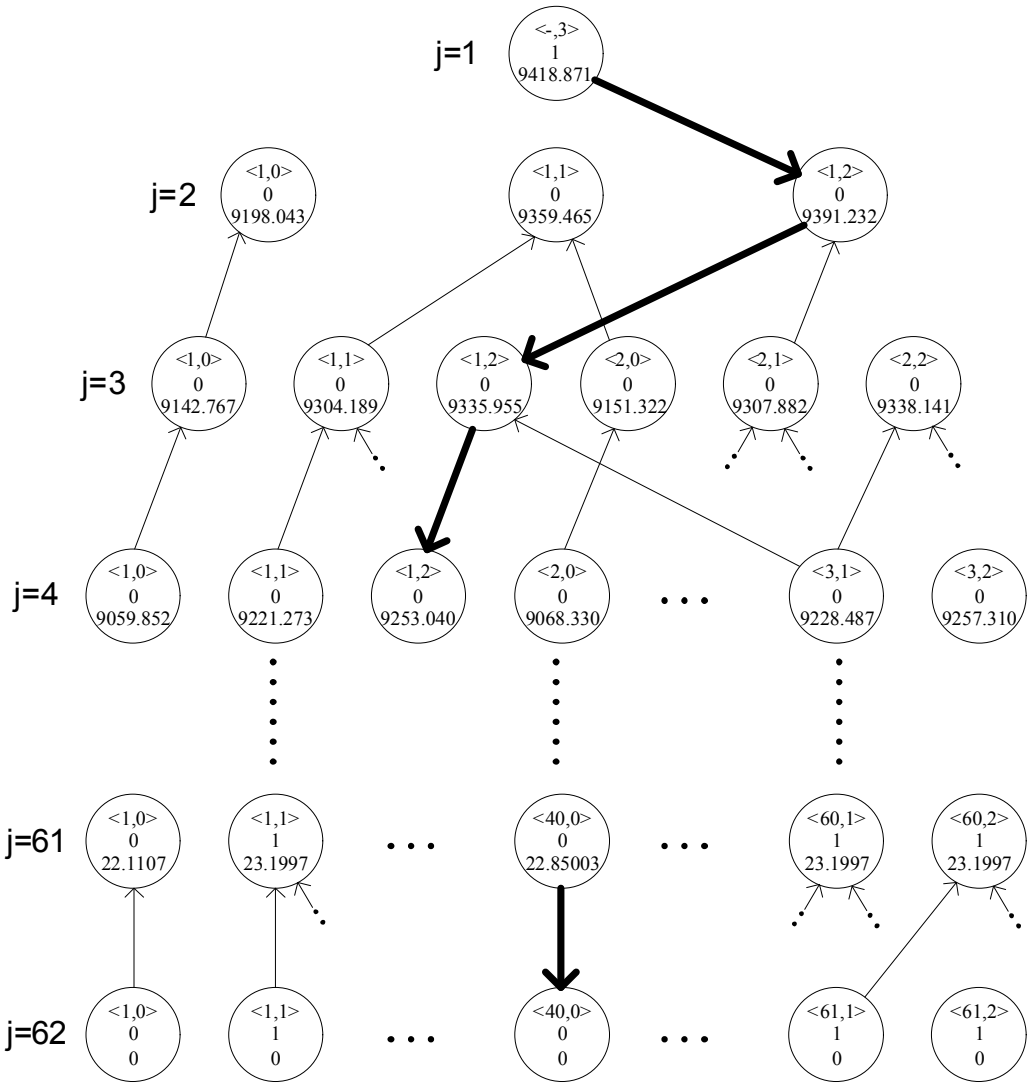

Figure 3. Decision graph for the three peaks access rate distribution. Within each circle, three items are indicated: <*l, k*> pair, best decision and the accumulated Q_t value.

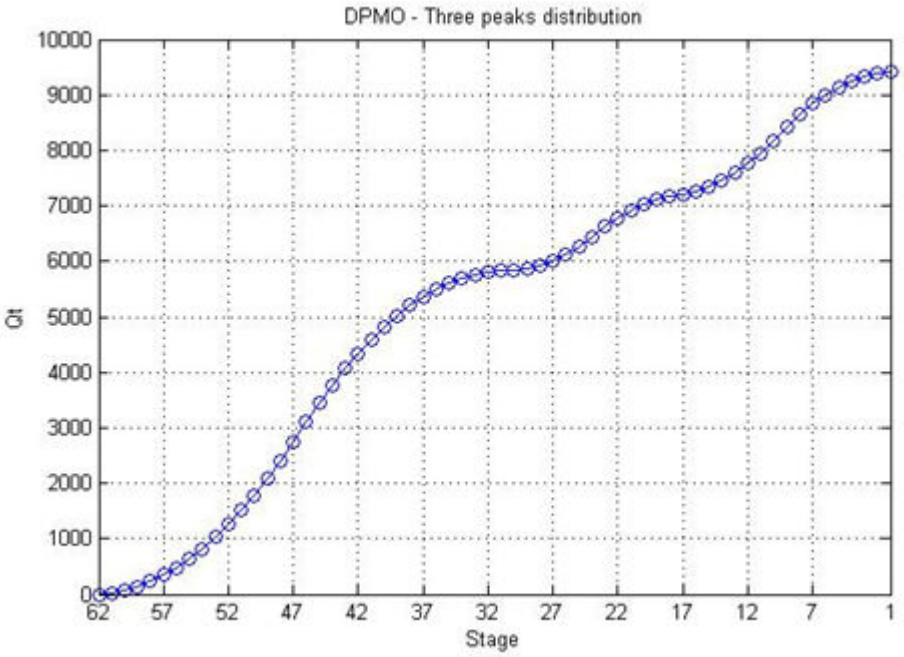

Figure 4. Evolution of Q_t as calculated by DPMO.

each succeeding state, the optimal decision and the corresponding reward value is already solved by DPMO. In this example, the resulting optimal streams rates are 200 kbps, 219 kbps and 239 kbps.

Figure 4 plots the value of Q_t as DPMO progresses from state $j = 62$ where $Q_t = 0$ to initial state $j = 1$ where $Q_t = 9418.871$, the optimal value.

RESULTS

The DPMO algorithm was compared with the MSS algorithm (Nunes, Monteiro, & Grilo, 2009) and exhaustive search in terms of execution time and the obtained . The algorithms were implemented and tested in MATLAB (2009) and performance results were obtained considering a fixed number of streams, $K = 3$. The algorithm can be easily adapted to other values of K. We considered the five different distribution profiles (also used in (Nunes, Monteiro & Grilo, 2009)) of user access rates AR_j to analyze the performance of the algorithms under various computational scenarios ranging from easy to difficult as follows:

- **Uniform:** the integer access rate values (Ar_j) are uniformly spaced within the interval $[AR_1, AR_M]$.
- **Three peaks:** three wide triangular-shaped peaks spanning uniformly spaced access rates, with maxima at 208 kbps (90 users), 222 kbps (70 users) and 245 kbps (≈128 users), minima at 200 kbps (10 users), 216 kbps (10 users), 229 kbps (0 users) and 260 kbps (8 users).
- **Three clusters:** three tiny triangular-shape pulses spanning uniformly spaced access rates, located in intervals [102 kbps, 140 kbps] (maximum at 108 kbps with 70 users), [251 kbps, 283 kbps] (maximum at 267 kbps with 85 users) and [501 kbps, 529 kbps] (maximum at 515 kbps with 75 users).
- **Six clusters:** three tiny triangular-shape pulses spanning uniformly spaced access rates, located in intervals [100 kbps, 165 kbps] (maximum at 130 kbps with 99 users), [175 kbps, 245 kbps] (two maxima at 205 kbps and 210 kbps with 99 users) and [250 kbps, 315 kbps] (maxima at 280 kbps and 285 kbps with 99 users), [325 kbps, 390 kbps] (maximum at 360 kbps with 99 users), [400 kbps, 465 kbps] (maximum at 435 kbps with 100 users) and [475 kbps, 530 kbps] (maximum at 410 kbps with 100 users).
- **Random:** Randomly chosen 300 access rates from the interval [10 kbps, 1000000 kbps], with random total number of users, where each access rate is assigned to a maximum of 1000 users. All the three algorithms (i.e. exhaustive search, DPMO and MSS) are executed using the same random scenarios.

Table 3. Access rate distribution profiles

Distribution	AR_1 (kbps)	AR_M (kbps)	Number of access rates (M)	Total number of users	Max users in each AR_j	Total number of users	Number of Runs
Uniform	250	440	20	20	1	20	1
Three peaks	200	261	62		128	3338	1
Three clusters	102	529	75		85	3060	1
Six Clusters	100	530	87		100	4494	1
Random	10	1000000	300	N/A	1000	Random	20

Table 3 provides a summary of the different access rate distribution profiles considered in this study.

The execution times are listed in Table 4 for all the considered user rate distributions. For each distribution except the Random, the optimal Q_t is listed together with the set of rates that achieve it are shown. In addition, we also included the average difference and the standard deviation of the difference between the optimal calculated by DPMO and the calculated by MSS.

While the exhaustive and DPMO algorithms always find the optimal Q_t, MSS sometimes fails to find the optimal solution in random distributions, since it can become stuck at local maxima. However, the error is very low and it consistently presents a lower execution time than DPMO. This suggests that MSS may still be useful in extremely demanding settings featuring huge numbers of heterogeneous users and target streams (K). Both MSS and DPMO are significantly more efficient than exhaustive search.

CONCLUSIONS

This paper has proposed the DPMO, a Dynamic Programming based algorithm whose objective is to calculate the optimal set of multicast video data rates that maximizes the overall QoE for a set of users with heterogeneous access network rates.

DPMO guarantees that the optimum solution is reached by employing Dynamic Programming concepts. Nevertheless, experimental results have demonstrated that despite not providing any optimality

Table 4. Execution times and Q_t error

Distribution	Optimal Q_t	Optimal b_i (kbps), $K = 3$	Execution time (seconds)			Average difference between calculated and optimal Q_t (MSS only)	Standard deviation of difference between calculated and optimal Q_t (MSS only)
			Exhaustive	DPMO	MSS		
Uniform	59.9	250, 310, 380	0.65	0.07	0.04	0	0
Three peaks	9418.9	200, 219, 239	1.93	0.28	0.26	0	0
Three clusters	8994.2	102, 252, 501	3.49	0.38	0.22	0	0
Six Clusters	12692.7	100, 345, 195	5.40	0.47	0.34	0	0
Random	N/A	N/A	228.05	6.05	4.24	$0.63 \times \max(Q_t)$	$1.02 \times \max(Q_t)$

guarantees, the solutions presented by MSS are optimal in most configurations or at least approach the optimum solution with a very low average error. The results show that both MSS and DPMO are significantly more efficient than exhaustive search in terms of execution time, with MSS being slightly but consistently more efficient than DPMO, suggesting that MSS may still be useful in extremely demanding settings featuring huge numbers of heterogeneous users and target streams.

While the present work targets at dynamic video multicast environments where users and access rates dynamically change over time, both DPMO and MSS assume that the user configurations are static most of the time and that they must be executed anew each time the user configuration changes. Future work shall focus on adapting the DPMO algorithm in order to remain efficient in highly dynamic user configurations.

ACKNOWLEDGMENT

Nestor Tiglao would like to acknowledge the funding support of the Department of Science and Technology – Science Education Institute (DOST-SEI) and the University of the Philippines Engineering Research and Development for Technology (UP ERDT).

REFERENCES

Birney, B. 2003. Intelligent Streaming, Microsoft Corp. (http://www.microsoft.com/windows/windowsmedia/howto/articles/intstreaming.aspx)

Broadband Forum. 2006. TR-126 Triple-Play Services Quality of Experience (QoE) Requirements.

Conklin, G. J., G.S. Greenbaum, K.O. Lillevald, A.F. Lippman & Y.A. Reznik. 2001. Video coding for streaming media delivery on the Internet. *IEEE T. Circ. Syst. Vid.* 11(3):269-281.

Cormen, T., C. Leiserson, R. Rivest, & C. Stein. 2001. Introduction to Algorithms. 2nd edition. MIT Press.

Patrikakis, Ch. Z., Y. Despotopoulos, A.M. Rompotis, N. Minogiannis, A.L. Lambiris & A.D. Salis. 2003. An implementation of an overlay network architecture scheme for streaming media distribution. Proc. 29th Euromicro Conf. 207-214.

ITU-T Recommendation G.1010. 2007. Opinion model for video-telephony applications, International Telecommunication Union.

Jiang, T., M. Ammar, & E.W. Zegura. 2000. On the use of destination set grouping to improve inter-receiver fairness for multicast ABR sessions. Proc. 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2000), 1:42-51.

Li, B & J. Liu. 2003. Multirate video multicast over the Internet: an overview. *IEEE Network,* 17(1):24- 29.

MATLAB, Version 7.8.0.347 (R2009a), MathWorks, February 2009.

Nunes, M., J, Monteiro, & A. Grilo. 2009. A Quality of Experience Optimization for Multi-rate Internet TV Services. Proc. 8th Intl. Conf. on Decision Support for Telecommunications and Information Society – DSTIS.

RealNetworks.2002. Introduction to streaming media with RealOne player (http://service.real.com/help/library/guides/realone/IntroGuide/HTML/htmlfiles/title.htm).

Winkler, S. & P. Mohandas. 2008. The Evolution of Video Quality Measurement: From PSNR to Hybrid Metrics. *IEEE T. Broadcast.*, 54(3):1-9.

Yang, Y.R., M. S. Kim, & S. S. Lam. 2000. Optimal partitioning of multicast receivers. Proc. 8th Intl. Conf. Network Protocols (ICNP'00) pp.129-140, 2000.

Zambelli, A. 2009. IIS Smooth Streaming Technical Overview. Microsoft Corporation (http://www.microsoft.com/downloads/en/details.aspx?displaylang=en&FamilyID=03d22583-3ed6-44da-8464-b1b4b5ca7520).