Multimedia Systems 8: 92–104 (2000)

Multimedia Systems

Video placement and configuration of distributed video servers on cable TV networks

Constantinos Vassilakis[1,⋆], **Michael Paterakis**[1,⋆], **Peter Triantafillou**[2,⋆]

[1] Department of Electronics & Computer Engineering, Technical University of Crete, GR-731-00 Chania, Greece; e-mail: {cv,pateraki}@telecom.tuc.gr
[2] Multimedia Systems Institute of Crete (MUSIC/TUC), GR-731-00 Chania, Greece; e-mail: peter@ced.tuc.gr

Abstract. A large-scale, distributed video-on-demand (VOD) system allows geographically dispersed residential and business users to access video services, such as movies and other multimedia programs or documents on demand from video servers on a high-speed network. In this paper, we first demonstrate through analysis and simulation the need for a hierarchical architecture for the VOD distribution network.We then assume a hierarchical architecture, which fits the existing tree topology used in today's cable TV (CATV) hybrid fiber/coaxial (HFC) distribution networks. We develop a model for the video program placement, configuration, and performance evaluation of such systems. Our approach takes into account the user behavior, the fact that the user requests are transmitted over a shared channel before reaching the video server containing the requested program, the fact that the input/output (I/O) capacity of the video servers is the costlier resource, and finally the communication cost. In addition, our model employs batching of user requests at the video servers. We study the effect of batching on the performance of the video servers and on the quality of service (QoS) delivered to the user, and we contribute dynamic batching policies which improve server utilization, user QoS, and lower the servers' cost. The evaluation is based on an extensive analytical and simulation study.

Key words: Distributed VOD systems – Program placement – HFC distribution networks

1 Introduction

It is believed that interactive multimedia services on user demand (services like VOD, digital broadcast TV, remote working, distance education, teleshopping, teleworking, telebanking, high-speed Web accessing, and video conference), will be of the most important kind of services to residential and business customers in the near future provided over emerging high-speed networks [1–4].

1.1 Technological infrastructure

Large-scale delivery of multimedia services, requires the retrieval from storage server devices and the delivery to the users of huge amounts of time-sensitive data. The high bandwidth requirements imposed on the storage facilities, together with the large number of multimedia programs that must be available to the users on demand, make it necessary to provide a distribution network in which a number of video servers and switching nodes are interconnected via communications links of various capacities. The high communication bandwidth requirement makes fiber an ideal physical medium for the transport of the multimedia data. The most appropriate technology to support a wide-area broadband multimedia distribution network is the asynchronous transfer mode (ATM), since it is designed to accommodate the simultaneous transmission of data, voice and video streams of variable bit rates and different quality-of-service (QoS) requirements over high-bandwidth fiber-optic-based networks [5,6]. However, due to the high cost of the optical fiber, it is considered too expensive to extend the fiber-based backbone network to the residential premises. Instead, a hybrid distribution network consisting of fiber, coaxial cable or copper, or even wireless media, will typically be employed. For example, the cable TV industry is currently upgrading its coaxial plant to a hybrid fiber/coaxial (HFC) by adopting the fiber to the node strategy, according to which fiber will feed optical network units (ONU), each of which serves a neighborhood by using another medium all the way from it to the customer premises (see Fig. 1). The use of cable modems at the customer premises provides the bandwidth to support multimedia services over the coaxial cable [7,8]. In addition, the telephone companies have developed technologies like the asymmetric digital subscriber line (ADSL) to enable broadband delivery of multimedia data over copper phone lines [9,10]. Finally, the wireless solution can be realized through the use of microwaves or via communication satellites. For an up-to-date report of the technological

⋆ Research supported by the European Community under the ESPRIT Long-Term Research Project HERMES no. 9141
Correspondence to: M. Paterakis

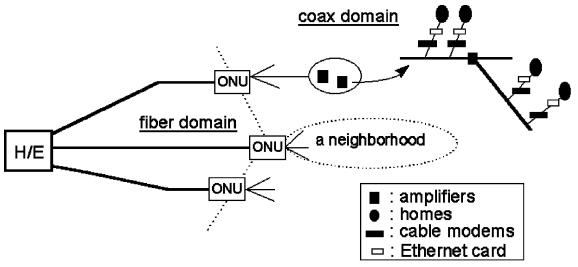


Fig. 1. A view of the network from the head end node to the customer premises

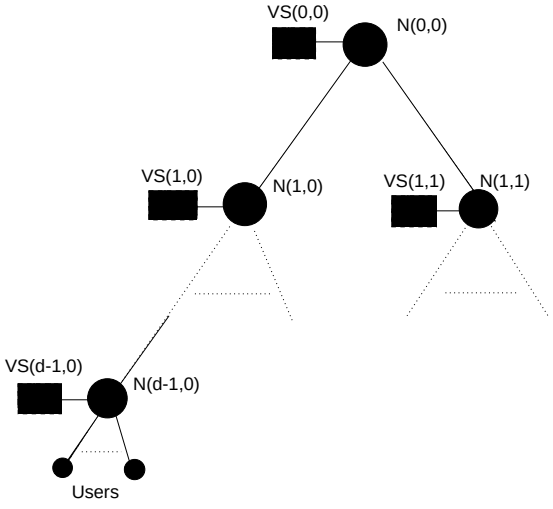


Fig. 2. An r-ary tree network topology with d levels consisting of video servers denoted by VS(i,j) attached on switching nodes denoted by N(i,j)

infrastructure and techniques for the distributed large-scale delivery of interactive multimedia services, the interested reader is referred to [4].

The large excess bandwidth available on today's cable TV HFC plants is an ideal candidate to provide the communications infrastructure for the distribution of multimedia services on user demand. For this reason, we will study a hierarchical architecture for the VOD distribution network which fits the existing tree topology used in today's cable TV distribution networks. Similar to [3, 19, 20], the distribution network topology is considered to be an r-ary tree associating a number of video servers and switching nodes connected via communication links. Referring to Fig. 2, there is a video server VS_{ij} associated with the switching node N_{ij} which can provide multimedia programs on demand to its children switching nodes. All the users are connected to the leaf nodes of the tree, also called headends. The links of the tree are considered bi-directional, although not symmetric. A communication link must be capable of transferring programs displayed by the local attached video server and by video servers located above the local one. Consequently, the communication link bandwidth increases as we move downwards in the tree hierarchy.

1.2 Open research issues

There are many interesting design issues in such distributed VOD systems, such as (i) the input/output (I/O) and storage configuration of the video servers located at the different levels of the hierarchical distribution network, (ii) the placement of the video programs at the different video servers in a way that ensures high performance, (iii) the bandwidth dimensioning of the communication links, and (iv) the effect of the number of levels in the tree topology of the distribution network. The above-mentioned design issues are studied in the paper.

1.3 Organization of the paper

The paper is organized as follows. In Sect. 2, we describe the system and user models. In Sect. 3, we present our results, starting from our study of the multiple-access channel used for the users' requests and following with the analytical and simulation results for the centralized and the distributed server cases. In Sect. 4, we present dynamic system operation techniques. Concluding remarks are offered in Sect. 5.

2 System model

We develop a model for the distributed VOD system considered, which takes into account the user behavior, the fact that the user requests are transmitted over a shared time-slotted, multiple-access channel before reaching the headend node, the fact that the I/O capacity of the video servers is the costlier resource, and a constraint in the QoS expressed in terms of a maximum user start-up latency. The I/O capacity of a server is defined as the number of program streams it must be capable to display simultaneously. Our model serves the user requests addressed to a video server in batches, in order to reduce the I/O server bandwidth requirements [17]. The playback of an initial request for a program is intentionally delayed by an amount of time, called the batching interval equal to T_B units of time, called slots, so that subsequent requests for the same program arriving within the batching interval may be serviced by using a single I/O stream. This, of course, introduces some latency, upper bounded by T_B. We define the user start-up latency, as the time interval from the instant a user generates a program request until the time the corresponding program is displayed. Propagation delays across the high-speed links are assumed negligible. We adopt batching over alternative techniques for reducing the I/O bandwidth requirements of a video server, like bridging [11–13] and adaptive piggybacking [18], because of its simplicity, which makes it easier to introduce it to a video server.

We assume a time-slotted system, with the slot size equal to the transmission time of an ATM cell (53 bytes) over the shared upstream multiple-access channel connecting the users with a single headend node. The bandwidth of the upstream channel is taken to be equal to 2.5 Mbps. Therefore, the slot duration is equal to $424/2.5 \times 10^6$ s = 0.00017 s. Results for different request packet size and/or upstream channel speed can be easily obtained by calculating the new slot duration. The user demand profile is assumed to be the same for all users, and the request load of each headend is assumed to be the same. The aggregate request arrival stream by all the users connected to a headend node is assumed

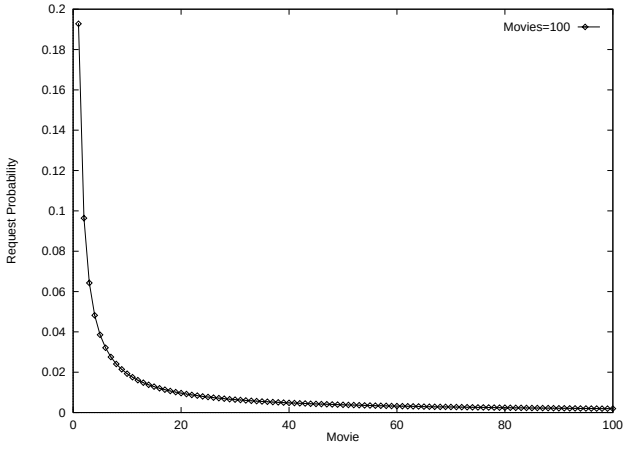


Fig. 3. Zipf's law distribution for a total of 100 movies

Poisson with rate λ requests/slot. We consider that an ATM cell is large enough to accommodate the various types of the requests that a user of the distributed VOD system can generate. Furthermore, most of today's cable TV modems are designed to transmit ATM cells over the shared channel connecting them to the headend node.

The stabilized controlled slotted ALOHA protocol [14], is used by the users to transmit their requests to the headend node over the shared channel because of its simplicity. In addition, slotted ALOHA is currently used in HFC CATV networks. For example, Scientific Atlanta uses the slotted ALOHA protocol for signaling between a subscriber's set-top terminal and the headend node [8].

Initially, we consider only the sequential-access (SEQ) service type, where a user initiates the playback of a program, but it does not have the ability to pause or do a scan on it (VCR-like access). The program popularity distribution is assumed to follow Zipf's law [16] (see Fig. 3). This distribution has been found to statistically fit video program popularities estimated through observations from video store statistics [17]. Given a total of M encoded programs (e.g., MPEG-2 format, in which case at least 4–8 Mbps are needed for an acceptable-quality broadcast of a program), the probability of requesting program m, $m = 1, 2, \ldots, M$, is given by

$$P_m = \frac{1}{mK_M} , \quad (1)$$

where

$$K_M = \sum_{m=1}^{M} \frac{1}{m} .$$

The duration of a video program for sequential access is considered to be uniformly distributed in the interval [90, 120] min. Each video server is considered to be capable of storing all, or a subset of the M programs.

Our overall approach

There are three major costs associated with the distributed VOD system considered. First, the I/O bandwidth requirements of the video servers. Second, the communication cost, which is proportional to the number of hops that it takes for a program from its storage location to reach a user. Third, the storage cost defined as the total number of program copies stored in the distribution network. The number of copies of programs stored in the same level of the tree topology grows exponentially as they are placed closer to the users (i.e., if a program is placed at level k, $k = 0, 1, \ldots, d-1$, where d is the tree depth and r is the tree degree, r^k copies of the program will be stored in the network).

Unlike related work [3,19,20], we focus on the I/O bandwidth requirements of the servers, and consider that as the most critical resource, instead of the storage and communication costs. We believe that this choice is justified given the present technology trends [21]. We conjecture that the overall performance of the VOD system will be better when all video servers in the system are highly utilized. For this reason, we are trying to balance the usage of all the video servers by requiring that the average number of I/O streams for each video server is the same, irrespective of the tree level it belongs. This criterion is important because it yields a balanced usage of all the video servers, and allows the deployment of identical- (or similar-) technology servers at the different levels of the hierarchical distribution network. Therefore, our placement criterion requires that the different programs are placed at the video servers located at different levels of the distribution network so that the average number of the I/O streams for each server is the same.

3 Results

3.1 Study of the multiple access channel

Since the stabilized controlled ALOHA protocol is used for the transmission of the user request packets across the shared upstream channel connecting them to the video server, we present in Table 1 simulation results for the average delay in slots experienced by a request packet from the time it is generated until it is successfully transmitted, for various values of the request packet arrival rate λ. In the same table, we present the 95% confidence intervals for the estimated average delay values as well as the exact analytical results from [15]. Notice the excellent agreement between the simulation and the analytical results.

We would like to point out that even the highest λ values considered in our study, when translated to the slot duration, correspond to very small values (e.g., 353 requests/min corresponds to 10^{-3} requests/slot). This implies that the ALOHA upstream channel is very lightly loaded, and as a result collisions of request packets occur very rarely. Therefore, the process of successfully transmitted request packets over the upstream channel (output process) is not expected to deviate considerably from the Poisson arrival request packet process (input process). This result might not hold when the users are allowed to engage in VCR-like access or to generate requests for other types of services (e.g., Web accessing)

Table 1. Simulation and analytical results

		95% Confidence interval		Analytical bounds	
λ (req/slot)	Average delay (slots)	Lower bound	Upper bound	Lower bound	Upper bound
0.05	1.651413	1.650977	1.651848	1.651631	1.651631
0.1	1.880150	1.878830	1.881469	1.87938	1.87938
0.15	2.243197	2.240917	2.245476	2.24265	2.24265
0.2	2.874727	2.870095	2.879358	2.87576	2.87576
0.25	4.159395	4.146791	4.171998	4.15097	4.15097
0.3	7.583717	7.556032	7.611402	7.57485	7.57485
0.35	31.290740	30.544981	32.036480	30.13219	33.28924

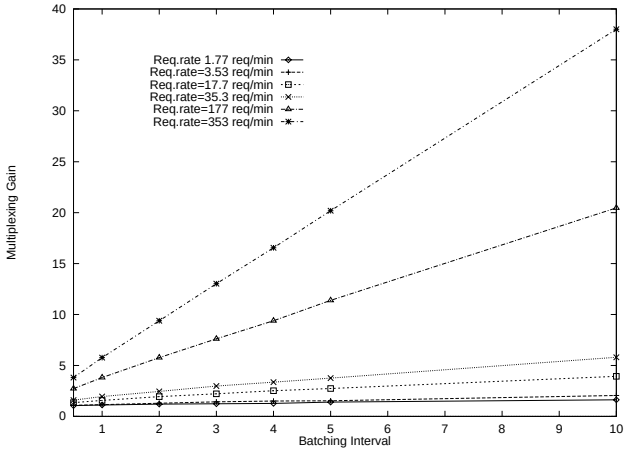


Fig. 4. Gain due to batching in a centralized video server containing 100 movies for various batching intervals and request rates

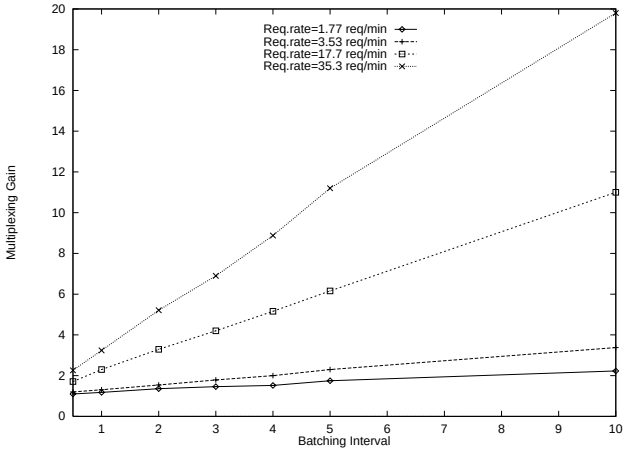


Fig. 5. Gain due to batching in a server containing the 10% most popular movies (out of a total of 100 movies) for various batching intervals and request rates

3.2 The centralized single-server case and motivation for distributed VOD systems

First, we motivate the need for a hierarchical network of the video servers and for the batch type of service for the user requests. In Fig. 4, we present simulation results of the video server multiplexing gain versus the duration of the batching interval for various values of the aggregate user request arrival rate λ (ranging from 1.77 requests/min to 353 requests/min). The multiplexing gain for a video server is defined as the ratio of the program requests addressed to the server divided by the number of I/O streams required to display the programs. In other words, the multiplexing gain is equal to the average batch size served by the server. The video server contains all of the 100 available programs. We notice the strong gains achieved for high and moderate request rates and large batching interval durations.

In practice, we expect that a customer will be willing to wait an average of 5 min from the time it submits a request until the display of the requested program begins. For this reason, the maximum batching interval considered in Fig. 4 is equal to 10 min. We would like to point out that the request arrival rates in the upper two curves of the figure are extremely high and it is not expected that in practice a single video server will have to sustain such request loads.

In Fig. 5, we consider the same system as in Fig. 4, with the only difference that the video server now stores only 10% of the most popular programs (out of 100). We notice the significant increase of the gain due to batching. This gain increase agrees with intuition, since a small number of high-popularity programs (program popularities are determined by the Zipf distribution) are stored in the server. Here, we would like to remark that the remaining 90% of the programs have to be placed at higher level servers of a hierarchical network.

The number of active I/O streams (or video server channels) after 30 min of operation as a function of the request arrival rate λ, for T_B = 5 min is presented in Fig. 6[1].

The three curves shown correspond to a video server storing all of the 100 available programs, 20% and 10% of the most popular programs, respectively. The first case corresponds to a single centralized server, while the other two cases correspond to servers belonging to a distribution network. From the results in the figure, we conclude that the centralized single server solution is very expensive, if not impossible to implement due to the extremely high required number of simultaneously active I/O streams. Due to the long program duration, the number of required active I/O streams further increases linearly as time progresses. Since an I/O channel can be released at minimum 90 min after it was occupied, we expect that the number of active I/O streams will be approximately three times that shown in Fig. 6. In contrast, the required number of active I/O streams is maintained at reasonable levels and grows very slowly

[1] A typical number of MPEG-2 streams (4 Mbps) , a commercial video server can support, varies from 12 to 600 [22]

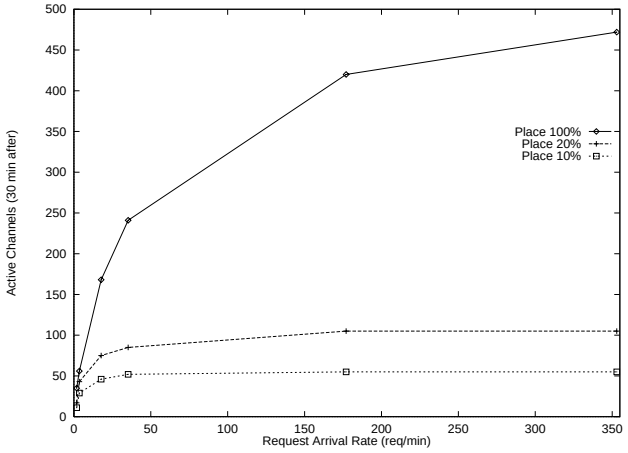


Fig. 6. Channels occupied in a server containing various fractions of the most popular programs, for various request rates. Server has been servicing requests for 30 min. (Total Movies=100, T_B=5 min)

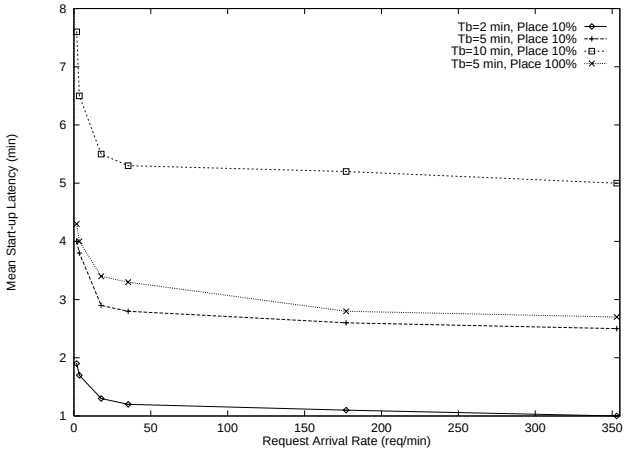


Fig. 7. Mean start-up latency for a user's request versus user request rate, for various widths of batching intervals T_B and various fractions of the most popular programs placed into the server

with time (due to the strong server gains, see Fig. 5) in the other two cases. The results in Fig. 6 motivate the need for the distribution of programs across a hierarchical network of video servers.

In Fig. 7 we present the user mean start-up latency versus the aggregate request rate λ, for various values of the duration of the batching interval and two different fractions of the available programs stored at the server. The results in the figure (together with other pertinent results, which are omitted for space reasons) demonstrate that the mean user start-up latency is rather insensitive to the fraction of programs stored at the server, and that it decreases approaching the value $T_B/2$ from above as the request rate increases.

3.3 Distributed VOD systems based on hierarchical networks of servers

Our analysis, presented in this section, produces the placement of the programs in the video servers located at the different levels of the distribution network so that the average number of I/O streams for each server is the same. We also provide expressions for the average user start-up latency when the user requests are served by a video server located at a given level of the hierarchy, and for the required upstream and downstream communication bandwidths of each link of the distribution network.

3.3.1 Video placement

The number of sequential programs placed in a server located at level i of the tree is denoted by $K^{[i]}$, $i = 0, 1, 2, \ldots, d-1$, $K^{[i]} = m' - m$, $(m' > m)$, where m is the lowest index program stored in server i and m' is the highest index program. Since the programs are placed sequentially in the servers, server i stores all programs m'' with $m \leq m'' \leq m'$. Our placement strategy keeps the most popular programs as close to the users as possible. Therefore, it starts by placing the highest popularity programs in each headend node. Less popular programs are placed at higher level servers, which serve the users attached to several headend nodes. The adopted placement strategy achieves a high server performance and balanced server utilization. In addition, placing the most popular programs at the headend nodes keeps the storage requirements very low, because the largest number of program replicas are made for a small number of popular programs. We define

$$P^{[i]} = \sum_{m \in VS_{ij}} P_m,\ j = 0, 1, .., \quad r^i - 1\ , i = 0, 1, \ldots, d-1 \tag{2}$$

as the cumulative probability that a user request corresponds to some program stored in a server located at level i of the hierarchical network. Notice that the above cumulative probability is the same for each server j located at level i of the network, since the network tree is assumed fully symmetric. The user request rate addressed to a headend server will be $\lambda P^{[d-1]}$, similarly the user request rate addressed to a server located at level i will be $\lambda r^{(d-(i+1))} P^{[i]}$. Based on the discussion in Sect. 3.1, we assume that the process of user requests addressed to program m placed in server i, is Poisson distributed with mean $\lambda_m = \lambda r^{(d-(i+1))} P_m$.

An initial user request for program m starts a batch and subsequent requests for the same program arriving within the batching interval belong to the same batch. The average number of requests in a batch for program m is given by

$$n_m = 1 + \lambda_m T_B\ , \tag{3}$$

where the constant 1 in the expression accounts for the initial request that starts the batch. To validate the above expression, we simulated a single server containing all the 100 available programs, with T_B=5 min and cumulative request rate λ=35.3 requests/min. The results we obtained together with the corresponding analytical results (from expression 3), are presented in Fig. 8. Notice the very good agreement between analytical and simulation results.

To obtain the average batch size served by server i, we need to average the quantities n_m in Eq. 3. Note, that the above average value is a weighted sum, since the fraction of batches for program m to the total number of batches for all programs stored in server i depends on the program index. To calculate this fraction, we first calculate the average

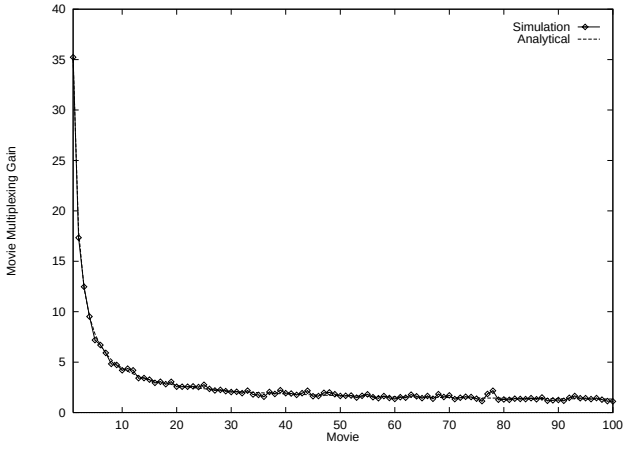


Fig. 8. Multiplexing gain due to a specific movie, from analysis and simulation

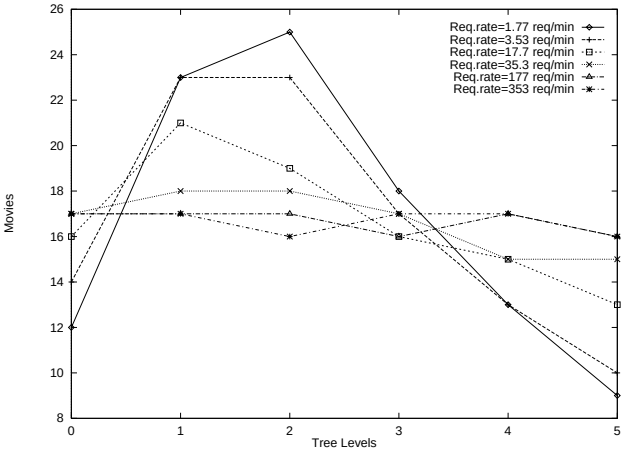


Fig. 9. Program placement in a binary tree with six levels for various request rates, a batching interval T_B equal to 5 min, and total number of movies equal to 100

number of batches for program m which form during a time interval of length T. This average is given by the ratio of the average number of requests for program m which arrive to the server within the interval T, $\lambda_m T$, to the average batch size for program m, $n_m = 1 + \lambda_m T_B$. Therefore, the fraction of batches for program m to the total number of batches for all programs stored in server i will be given by

$$\sigma_m = \frac{\frac{\lambda_m}{1+\lambda_m T_B}}{\sum_{m' \in VS_{ij}} \frac{\lambda m}{1+\lambda_{m'} T_B}}$$

$$= \frac{\frac{r^{(d-(i+1))} P_m}{1+\lambda_m T_B}}{\sum_{m' \in VS_{ij}} \frac{r^{(d-(i+1))} P_{m'}}{1+\lambda_{m'} T_B}} \ . \quad (4)$$

Finally, the average batch size served by server i is

$$n^{[i]} = \sum_{m \in VS_{ij}} \sigma_m n_m \ . \quad (5)$$

Since we are interested in placing the available programs to the different levels of the distribution network so that the average number of I/O streams is the same, we need to solve the following equations for the rate of I/O stream establishment at the different servers of the distribution network

$$\frac{\lambda P^{[d-1]}}{n^{[d-1]}} = \frac{r\lambda P^{[d-2]}}{n^{[d-2]}} = \ldots = \frac{r^{(d-(i+1))}\lambda P^{[i]}}{n^{[i]}}$$
$$= \ldots = \frac{r^{d-1}\lambda P^{[0]}}{n^{[0]}} \ , \quad (6)$$

where level 0 corresponds to the tree root and level $d-1$ to the tree leafs (headend nodes). The above set of equations simplifies to

$$\sum_{k \in VS_{d-1,j}} \frac{P_k}{n_k} = r \sum_{k \in VS_{d-2,j}} \frac{P_k}{n_k} = \ldots = r^{(d-(i+1))}$$
$$\sum_{k \in VS_{i,j}} \frac{P_k}{n_k} = \ldots = r^{d-1} \sum_{k \in V_{0,0}} \frac{P_k}{n_k} \ . \quad (7)$$

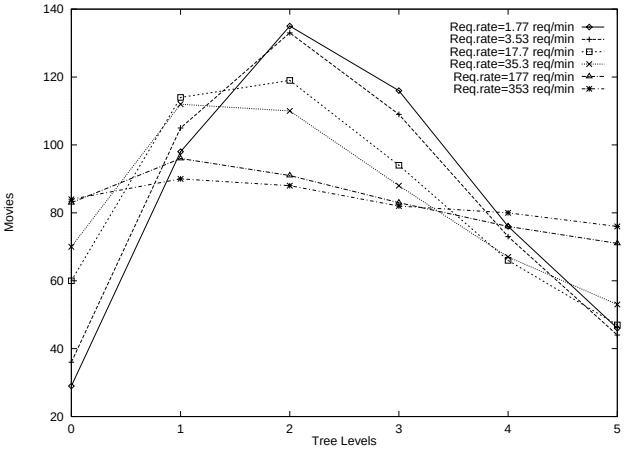


Fig. 10. Program placement in a binary tree with six levels for various request rates, a batching interval T_B equal to 5 min, and total number of movies equal to 500

3.3.2 Video placement dependencies on system and user parameters

Using Maple V, Release 3, we solve the above set of equations for the program placement at the different levels of the distribution network. In Fig. 9–11, we present the analytical results for the placement of 100, 500 and 1000 programs, respectively. We consider a six-level distribution network, and user request arrival rates for each headend node ranging from 1.73 req/min to 353 req/min. We observe that as the user request rate increases, the program placement becomes more and more even among the different levels. The same behavior has been observed in a tree distribution network of degree 3. For light-to-moderate request arrival rate values, the program placement is uneven, placing more programs in servers located at levels 1 to 3 than in servers located at level 0, 4 and 5.

The same behavior is observed when we examine the program placement as a function of the batching interval T_B, for a given request arrival rate (λ=35.3 req/min in Fig. 12). As T_B increases, the program placement becomes even. For small T_B values, the program placement can be very uneven.

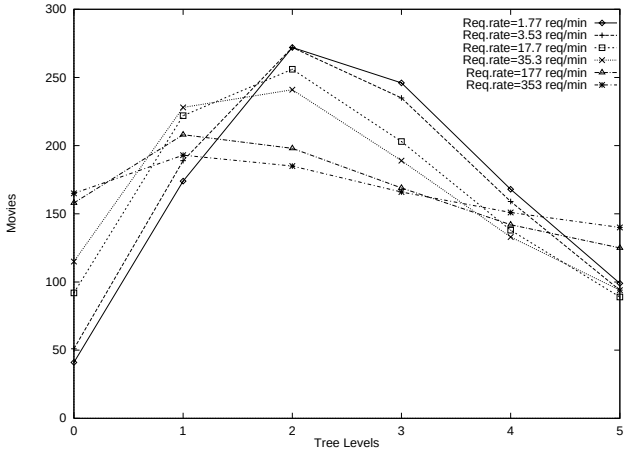


Fig. 11. Program placement in a binary tree with six levels for various request rates, a batching interval T_B equal to 5 min, and total number of movies equal to 1000

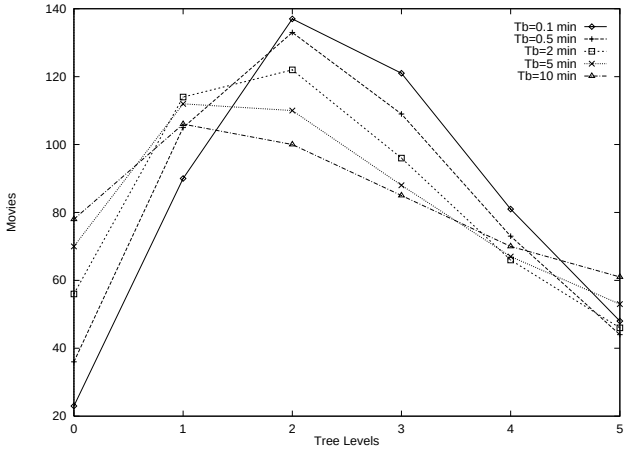


Fig. 12. Program placement in a binary tree with six levels for various batching intervals T_B, a user request rate equal to 35.3 req/min, and total number of movies equal to 500

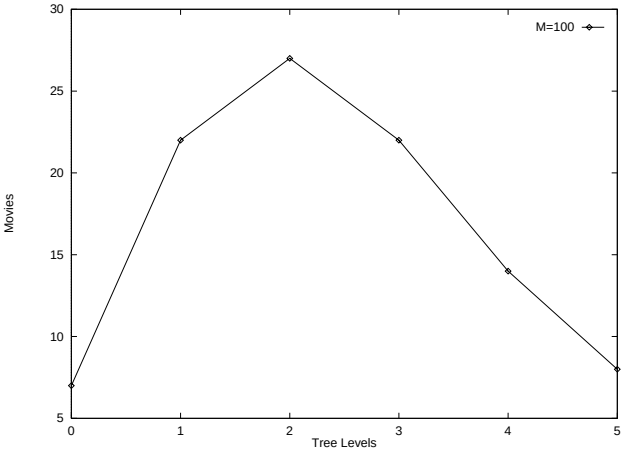


Fig. 13. Program placement in a binary tree with six levels, with the batching interval equal to zero, and total number of movies equal to 100

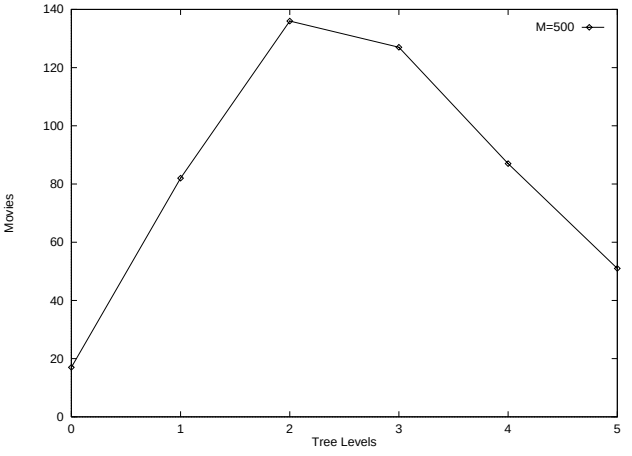


Fig. 14. Program placement in a binary tree with six levels, with the batching interval equal to zero, and total number of movies equal to 500

The explanation is that the parameters λ and T_B affect the program placement through only their product; therefore, the limiting placement solutions as either T_B increases or λ increases are the same. This observation is further supported by the fact that the average batch size served by a video server can be increased by either increasing the request arrival rate or increasing the batching interval.

For $\lambda T_B \gg 1 \Rightarrow n_k \simeq \lambda T_B r^{(d-(i+1))} P_k$, therefore all the equations in (7) simplify to

$$\sum_{k \in VS_{d-1,j}} \frac{1}{\lambda T_B} = \ldots . \sum_{k \in VS_{i,j}} \frac{1}{\lambda T_B} = \ldots . = \sum_{k \in VS_{0,0}} \frac{1}{\lambda T_B} , \quad (8)$$

which yields the even program placement distribution.

We also evaluated the program placement when T_B=0 (i.e., no batching is used). The results are shown in Figs. 13–15, for M = 100, 500, and 1000 programs respectively. We again observe an uneven program placement, which is also independent of the request arrival rate λ (since for T_B=0, n_k=1).

The effect of the tree degree r on the program placement is presented in Fig. 16, when T_B=5 min and λ=35.3 req/min. We observe that the peak of the curves moves to the left as r increases. Even though, the program placement for r = 5 or 6 is quite uneven, we recall that, as explained before, if λ and/or T_B is increased, the program placement will become even.

To validate our analytical placement results, we simulate a distribution tree network with d = 4 levels, r = 2, λ=35.3 requests/min and T_B=5 min. The experiment simulates 30 min of network operation, at which time it stops and we observe the number of active I/O streams at each network server. From the results in Fig. 17, we observe that the number of active I/O streams versus server level and index is pretty much flat. Furthermore, the simulation results closely match the analytically expected results. The analytical result (225 in this case) is obtained as follows. Once we obtain the placement of the programs by solving (7), we evaluate any of the ratios in the above equation and we multiply by network operation time.

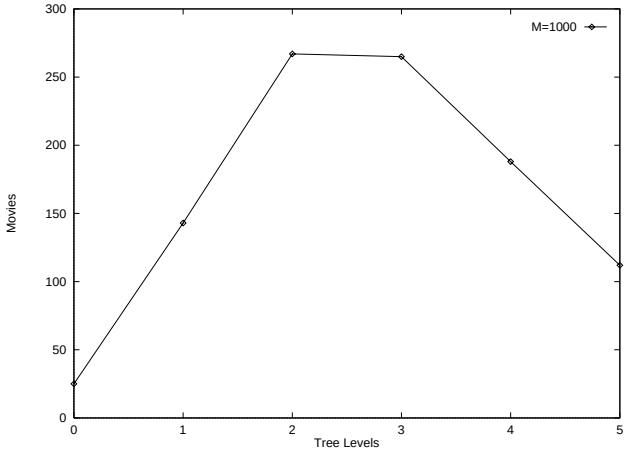


Fig. 15. Program placement in a binary tree with six levels, with the batching interval equal to zero, and total number of movies equal to 1000

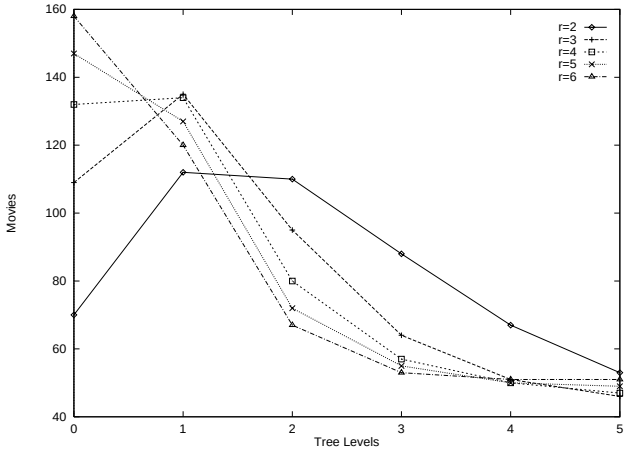


Fig. 16. Program placement in a tree with six levels, various degrees r, a batching interval T_B equal to 5 min, and a request rate equal to 35.3 req/min

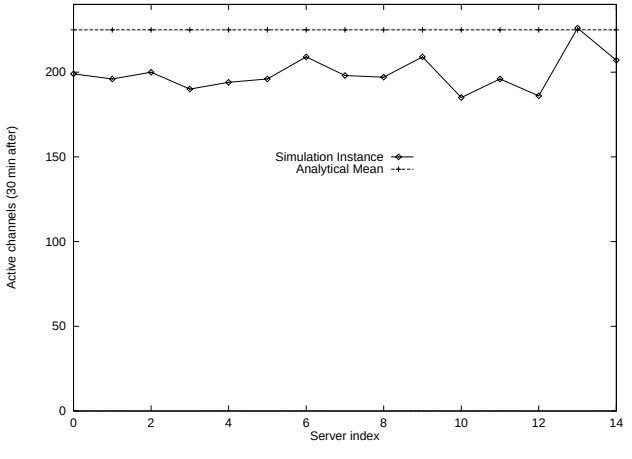


Fig. 17. I/O channels occupied in each server of the binary four-level tree, for T_B = 5 min, total movies equal to 500 and a request rate equal to 35.3 req/min. The placement is obtained from our analysis. Server index is 0..7 for Level 3 , 8..11 for Level 2, 12..13 for Level 1 and 14 for the root

Table 2. Analytical results

Level 3	Level 2	Level 1	Level 0
3.2 min	4.6 min	4.8 min	4.9 min

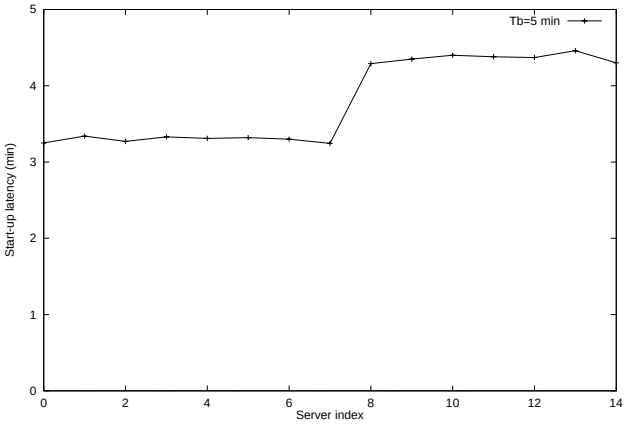


Fig. 18. Start-up latency for a user serviced by a server of the binary four-level tree, for a batching interval T_B equal to 5 min, total movies equal to 500 and a request rate equal to 35.3 req/min

3.3.3 User start-up latency

Regarding the average user start-up latency, let S_m denote the average start-up latency for users requesting program m. Then it can be easily shown that

$$S_m = \frac{T_B}{2} + \frac{T_B}{2(1 + \lambda_m T_B)} \,. \tag{9}$$

Therefore, the average user start-up latency for requests addressed to a server located at level i is given by

$$S^{[i]} = \sum_{m \in VS_{ij}} \frac{S_m P_m}{P^{[i]}} \,. \tag{10}$$

In Fig. 18, we present simulation results of the mean user start-up latency for requests addressed to each server in the network, for a distribution tree network with r = 2, d = 4, T_B=5 min, and λ = 3.53 request/min. In Table 2, we present the corresponding analytical results. We notice that mean user start-up latency when submitting program requests to the headend nodes is smaller than the same quantity when submitting program requests to nodes located above the headend nodes in the distribution network.

3.3.4 Communication bandwidth requirements

Regarding the required upstream and downstream link bandwidth requirements, we provide analytical expressions for their evaluation. For the upstream direction the required bandwidth for a link located *below a server* at level i is given by

$$UT^{[i]} = \begin{cases} \lambda r^{(d-(i+2))} \left(1 - \sum_{j=i+1}^{d-1} P^{[j]}\right) & , 0 \le i \le d-2 \\ \lambda & , i = d-1 \end{cases} \tag{11}$$

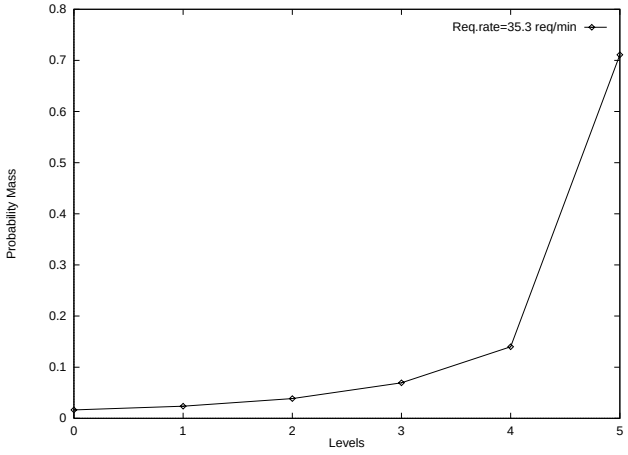


Fig. 19. Probability mass of the movies contained in a server at level i of a binary six-level tree for the placement obtained from our analysis. The batching interval is 5 min, the total movies are 500 and the request rate is equal to 35.3 req/min

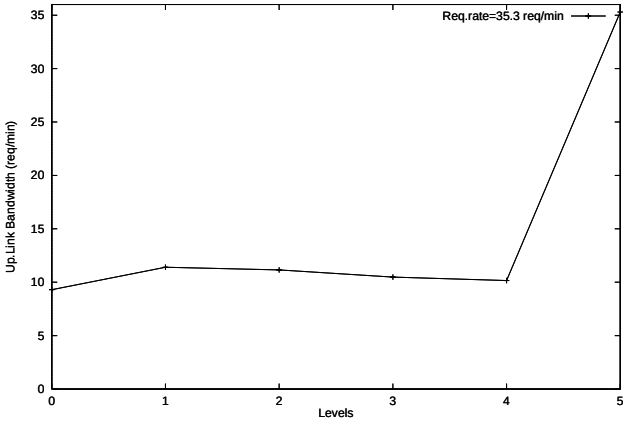


Fig. 20. Required communication bandwidth for an upstream link below level i of a binary six-level tree. The batching interval is 5 min, the total movies are 500 and the request rate is equal to 35.3 req/min

expressed in request packet transmissions per min. For the downstream direction, the required bandwidth for a link located *below a server* at level i is given by

$$DT^{[i]} = \begin{cases} \frac{R}{r} & , i = 0 \\ \frac{DT^{[i-1]}+R}{r} & , 0 < i \leq d-2 \\ DT^{[d-2]} + R & , i = d-1 \end{cases} \quad (12)$$

where R denotes the value of any of the ratios in (7) for the given program placement. This throughput is expressed in playback streams per minute. Notice that $DT^{[d-1]}$, corresponds to the downstream throughput of the link connecting a headend to its users.

In Fig. 20, we present the required upstream link bandwidth for a network with six levels, $r = 2$, $T_B = 5$ min, and $\lambda = 35.3$ req/min. We observe that the required link bandwidths are very close to each other for all the links located at different levels of the network above the headend nodes, and it is higher for the links connecting the headend nodes to the users. See also Fig. 19, where we plot the cumulative probability $P^{[i]}$ versus the network level i, for the program placement we are considering.

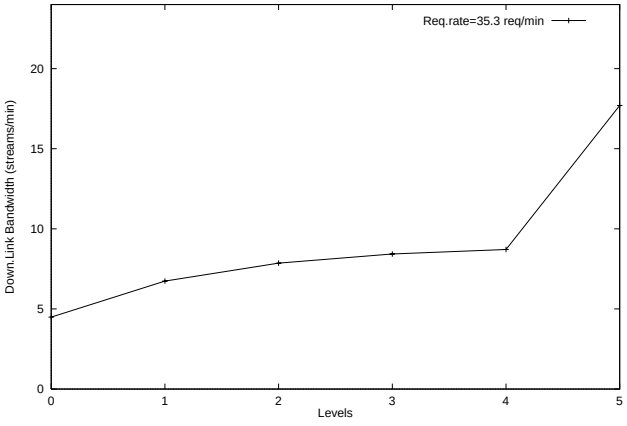


Fig. 21. Required communication bandwidth for a downstream link below level i of a binary six-level tree. The batching interval is 5 min, the total movies are 500 and the request rate is equal to 35.3 req/min

In Fig. 21, we present the required downstream link bandwidths for the same distribution network with the one in Fig. 20. Since each server in the downward path contributes some playback streams to the already accumulated traffic, the required link bandwidth increases as we move downstream.

The analytical results presented in this section are helpful in configuring the VOD distribution network, and in providing answers to interesting design questions related to the selection of the tree degree, the number of servers and their I/O capacity, the required link bandwidths, and the quality of service delivered to the users.

4 Adaptive system operation

Clearly, if we observe a VOD system during a day period, we will discover that the users' request rate is not constant. Various VOD trials have shown that there are fluctuations of the users' request rate, occurring mainly during a rush-hour period at the end of the day. During these hours, the user request rate as a function of time can be approximately modelled by a triangle shape. The duration of the rush-hour period is about 6 h, and its starting hour depends on the user community we examine. During the remaining hours of the day, the request rate can be approximated by a constant (actually an upper bound), although there may be small rate variations. We call these hours, idle hours.

It is critical to examine the behavior of the system through the day, because it is expected that an overload will take place during the rush hours, while the system will be underutilized during the idle hours. We adopt the cumulative user request shape shown in Fig. 22, which has been found to describe sufficiently well the users' behavior through a day period (see user behaviors from trials in [23, 24]). We refer to this user behavior profile as the *tailed-triangle profile*. The rush-hour period beginning and end may differ from community to community, however our approach is still applicable.

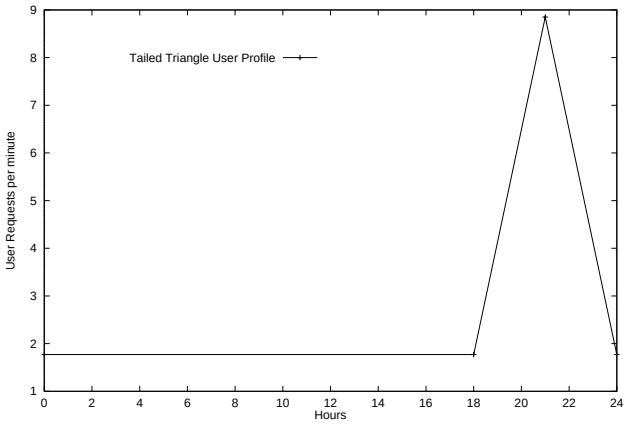


Fig. 22. Cumulative user request rate during a 24-h period (tailed triangle profile)

4.1 Customizing video placement to the user request rate

Before we proceed further, recall our goal which was to have an I/O balance through the servers of the r-ary tree network. In order to continue satisfying this goal, we must develop a new technique which will have the ability to adapt the program placement to the request rate variations. We refer to this technique as the *multirate placement* (MRP) technique. Using this technique, we will continue to have I/O balance through the day.

It has been observed that as λ decreases approaching very small values, the program placement producing the same server utilization at any tree level, tends to be the same with the one for $T_B = 0$. When λ increases, assuming high values, the program placement tends to be even (same number of movies placed at any level of the tree). Having in mind that λ varies, we want an adaptive program placement. Therefore,

1. we solve the balance equations for every possible discrete λ value that we want our system to handle in a way that will sustain I/O balance. From this process, we keep for each level of the tree the maximum and the minimum values of the different placement solutions.
2. We then store at each level all the programs with indexes between the minimum and the maximum value. Thus, some video programs are replicated across tree levels. Therefore, at each level of the tree, we keep a superset of programs required for a given λ.
3. Given a discrete λ value, the server accesses only the relevant set of programs.

The request rate is estimated at the end of constant-length time periods. Within a time period, the rate is assumed to be equal to the current estimate. We have validated through simulations the MRP technique for a binary tree network of servers with four levels and 500 available programs. We assumed a batching interval T_B = 5 min and we used a tailed-triangle user profile with minimum request rate λ = 1.77 req/min and peak rate λ = 8.85 req/min [2]. The analytically produced placements used in the simulations for given discrete λ values are shown in Table 3, while the program supersets stored at each level are shown in Table 4.

Table 3. Analytically produced placements

λ (req/min)	Level 3	Level 2	Level 1	Level 0
1.77	1-34	35-160	161-342	343-500
2.95	1-39	40-170	171-349	350-500
4.13	1-45	46-180	181-355	356-500
5.31	1-48	49-186	187-359	360-500
6.49	1-51	52-191	192-362	363-500
7.67	1-57	58-200	201-367	368-500

Table 4. Program supersets

Level 3	Level 2	Level 1	Level 0
1-57	35-200	161-367	343-500

We simulated the system for 24 h and observed the occupied I/O channels in each server of the four-level binary tree network at that instant. The results are presented in Fig. 23 and demonstrate that the MRP technique achieves balanced I/O despite the λ value variations. Had we wanted our VOD system to be able to adapt to any λ request rate value in the range [0,∞], the required program placement is shown in Table 5, and obviously corresponds to the worst case in terms of storage requirements.

4.2 Effect of dynamically adjusted batching intervals on user QoS and server utilization

Having demonstrated that the I/O balance is achieved through the use of our multirate placement technique, we observe through simulation the number of active I/O channels for each server through the day. We simulate the system's operation for a number of days in order to reach a steady state

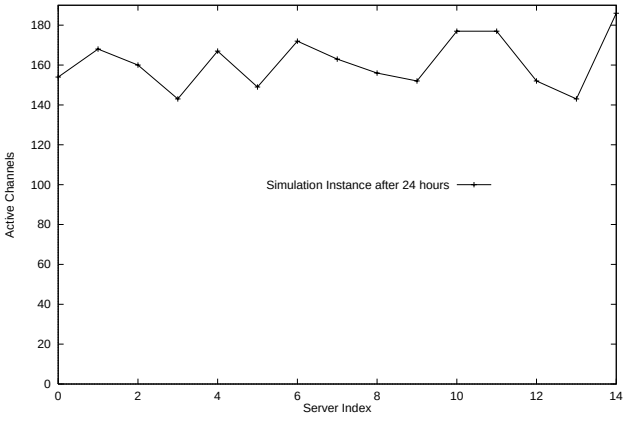


Fig. 23. I/O channels occupied in each server of the binary four-level tree, for T_B=5 min, total movies 500, request rates varying from 1.77 to 8.85 req/min according to tailed triangle user profile, assuming the multirate program placement technique

[2] The ratio 1:5 of the minimum to peak rate is taken from observed request profiles in actual VOD trials. Although the minimum rate has been chosen arbitrarily and may seem low, on the contrary, the number of requests submitted to the VOD system during a 24-h period is about 25,000. Therefore, our system can serve at least 25,000 users in one day, assuming that each user requests on average one movie per day. Usually, the average number of user requests per week is about 2 to 3.

Table 5. Program placement

Level 3	Level 2	Level 1	Level 0
1-125	22-253	129-386	319-500

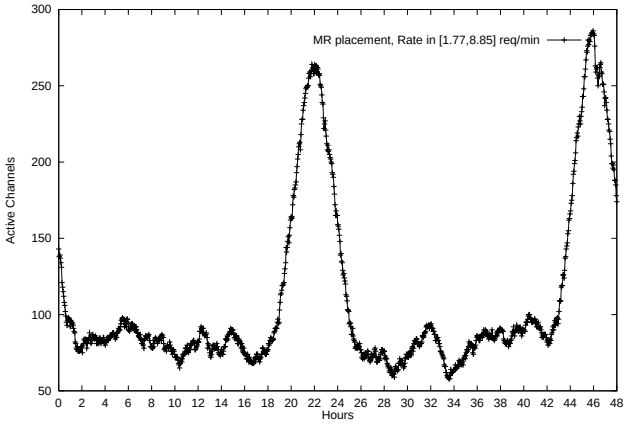


Fig. 24. Number of active channels in a server of a binary four-level tree network of servers during a 48 hour period, for T_B=5 min, 500 stored programs, a tailed triangle user behaviour profile (1.77 to 8.85 req/min minimum to peak request rate), assuming the multirate placement

and then take two consecutive 24 h of operation, the results of which are presented in Fig. 24.

We observe the extreme rise in the number of active I/O channels during the rush hours, while during the idle hours there is an undesired severe server underutilization. Specifically, the server is underutilized more than 75% of the time and there is a very sharp increase in the I/O requirements during the rush period. This means that we must deploy a very-high-capacity expensive server, which will be severely underutilized most of the time. Furthermore, we are using a constant batching interval for servicing requests even during the idle period, penalizing the users. To solve these problems, we adopt a different approach. First, we do not use batching during the idle periods (improving the user QoS). Second, we investigate whether by dynamically adjusting the batching interval during the busy period we can have the same server I/O requirements as during the idle period (with batching turned off). We use the discrete request rates from Table 3. For each of these values we have a set of balance equations with an extra unknown parameter, which is the duration T_B of the batching interval for that λ rate. We equate the first term of each set of these equations. The resulting system of equations consists of 24 equations in 24 unknowns. The results are obtained through the use of Maple V, Rel. 3 and are presented in Table 6.

Table 6. Results

λ (req/min)	T_B (min)	Level 3	Level 2	Level 1	Level 0
1.77	0	1-21	22-128	129-318	319-500
2.95	16.1	1-58	59-202	203-368	369-500
4.13	35.6	1-82	83-231	232-381	382-500
5.31	51.6	1-96	97-242	243-385	386-500
6.49	63.7	1-103	104-248	249-385	386-500
7.67	73	1-108	109-250	251-385	386-500

Table 7. Program placement

λ (req/min)	T_B (min)	Level 3	Level 2	Level 1	Level 0
1.77	0	1-21	22-128	129-318	319-500
2.95	10	1-50	51-189	190-360	361-500
4.13	10	1-56	57-198	199-366	367-500
5.31	10	1-61	62-205	206-369	370-500
6.49	10	1-65	66-210	211-372	373-500
7.67	10	1-68	69-215	216-374	375-500

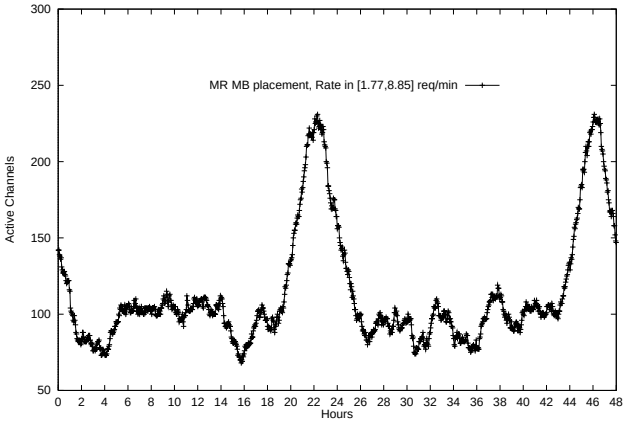


Fig. 25. Active channels in a server of a binary four-level tree during a 48-h period, for a total of 500 programs, a tailed triangle user behaviour (1.77 to 8.85 req/min), assuming a multirate placement and using multiple batch sizes (0 min during idle hours, 10 min during rush hours)

Our results are rather disappointing. Its is clear that the obtained batching interval durations are very long to be used in a real system. For this reason, we use an upper bound on the batching interval (worst case user start-up latency). Specifically, for those request rates for which the batching interval exceeds the upper bound, we impose the upper bound. We then solve again the set of balance equations to obtain the new program placement. The corresponding results are shown in Table 7 with the upper bound equal to 10 min. We simulated once again the resulting system, and observed the number of active I/O channels for each server (the corresponding results are shown in Fig. 25). Comparing these results to those presented in Fig. 24, we conclude that

1. the distribution of the server's I/O requirements versus time is considerably less skewed, resulting in a higher server utilization,
2. we now need a server with a considerably smaller I/O capacity (roughly 30% less).

5 Conclusions

We have developed a model for a distributed interactive VOD system based on a hierarchical server interconnection topology that fits the existing topology used in today's HFC CATV plants. The model is useful in the video program placement, configuration, and performance evaluation of such systems. A user activity model is integrated with a program popularity distribution, a simple and stable random-access protocol for the transmission of the user requests to

the video servers, and a batch-type service of the user requests at the video servers. Results from analytical and extensive simulation studies are presented.

Our major conclusions and contributions are

1. through experimentation we determined that a centralized single-server solution is very expensive if not impossible to implement. In contrast, building large-scale VOD systems based on a distributed hierarchical network of servers, in which servers at the lower levels (closer to the users) store more popular videos, is preferable. This is so, because the servers' I/O requirements are maintained at reasonable levels and grow very slowly with increased request rates.
2. We contributed an analytical solution to the problem of placing video programs across the servers located at the different levels of the hierarchy, so that the average server I/O requirements are the same. Our findings are that
   - the video placement depends on the request rate λ and the batching interval T_B only through their product. In particular, when $\lambda * T_B \gg 1$, the program placement should be even across levels. For small-to-moderate product values, the program placement should be uneven, placing more videos at the intermediate levels of the hierarchy than before.
   - When increasing the degree of the hierarchy, the program placement should be uneven, with more videos placed at higher levels and fewer at the lower levels.
3. From our analytical expressions and our experimentation, for the average user start-up latency we notice that it is smaller for user requests serviced by the headend nodes. This is attributed to the fact that the latter nodes store more popular items for which user requests do not typically wait a whole batching interval.
4. Regarding the communication bandwidth requirements, our results for the upstream links show that above the headend level the required link bandwidths are very similar, irrespective of levels, and about one third of the required bandwidth for the links connecting the headend nodes to the users. This is attributed to the fact that servers at higher levels store less popular videos on the one hand, and on the other hand, they serve larger user communities.

   Regarding the downstream link bandwidth requirements, they increase as we move toward the users, since each server in the downstream path contributes some playback streams to the already accumulated traffic.
5. Given that in real systems the user request rate will be fluctuating over time, we contributed the multirate placement technique (MRP technique), which introduces some video program replicas at different levels and achieves the same I/O server requirements, irrespective of the levels and the request rate. Furthermore, we showed how by dynamically adjusting the batching interval duration we can improve the user QoS during the idle periods, achieve higher server utilization, while requiring less expensive servers (with smaller I/O capacities).

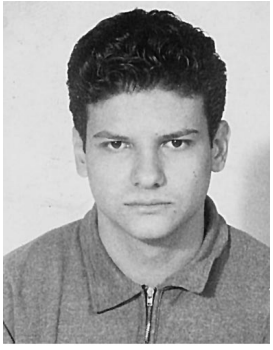
Although the methods presented in this paper are tested over a symmetric r-ary tree network topology of homogenous servers, they can be easily employed for studying other network topologies of heterogeneous servers as well. In addition to the above, the work in this paper can be extended in other ways. If the I/O capacities of the servers are given a priori, then a user request to a video server can be blocked due to the servers finite I/O capacity. In such case, there is a number of policies that can be employed such as discarding the request (i.e., the user has to try again at some later time), or queuing the request at the server for some time hoping that a video stream currently displayed will terminate soon, or hybrids of the above two policies.

So far, we have considered one class of service type, the sequential access (SEQ), where the whole program is displayed. An interesting extension of this work will be to study VCR-like and random accesses. In such cases, it is expected that our system will be considerably affected due to the fact that users will tend to leave their batches.

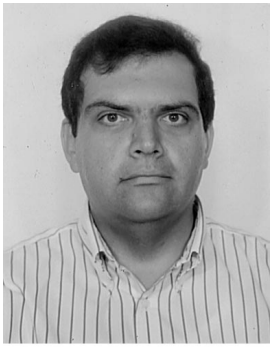
References

1. Bisdikian C, Maruyama K, Seidman D, Serpanos D (1996) Cable Access Beyond the Hype: On residential broadband data services over HFC networks. IEEE Commun Mag 34(11): 128-135
2. Li V, Liao W, Qiu X, Wong E (1996) Performance Model of Interactive Video-on-Demand Systems. IEEE J Sel Areas Commun 14(6): 1099–1109
3. Nussbaumer JP, Patel BV, Schaffa F, Sterbenz JPG (1995) Networking Requirements for Interactive Video on Demand. IEEE J Sel Areas Commun 13(5): 779–787
4. Vassilakis C, Paterakis M (1997) On the Distributed Delivery of Broadband Multimedia Services to Residential/Business Customers on Demand. Technical Report. ECE Department, Tech. Univ. of Crete, Crete, Greece
5. Onvural R (1995) Asynchronous Transfer Mode Networks, Performance Issues, 2nd Edition. Artech House Books, Boston, Mass.
6. ATM Forum (1999) http://www.atmforum.com. ATM Forum
7. AT&T Paradyne (1999) http://www.paradyne.att.corn. AT&T Paradyne
8. Minoli D (1995) Video Dialtone Technology: Digital Video Over ADSL, HFC, FTTC and ATM. McGraw-Hill, New York
9. Kyees PJ, McConell RC, Sistanizadeh K (1995) ADSL: A New Twisted-Pair Access to the Information Highway. IEEE Commun Mag 4: 52–59
10. ADSL Forum (1999) http://www.adsl.com. ADSL Forum
11. M. Kamath, K. Ramamritham, and D. Towsley, Kamath M, Ramamritham K, Towsley D (1994) Buffer Management for Continuous Media Sharing in Multimedia Database Systems. Technical Report 94-11. University of Massachusetts, Amherst, Mass.
12. Dan A, Sitaram D (1994) Buffer management policy for an on-demand video server. IBM Research Report RC 19347. T.J. Watson Research Center, Yorktown Heights, New York
13. Dan A, et al. (1995) Buffering and caching in large-scale video servers. In: Proceedings of the IEEE CompCon Conference, 1995, pp 217–224
14. Fayolle G, Gelenbe E, Labetoulle J (1977) Stability and Optimal Control of the Packet-Switching Broadcast Channel. J Assoc Comput Mach 24(3): 375–386
15. Georgiadis L, Merakos L, Papantoni Kazakos P (1987) A method for the delay analysis of random multiple access algorithms whose delay process is regenerative. IEEE J Sel Areas Commun 5(4): 1051–1062
16. Knuth D (1973) The Art of Computer Programming, Vol. 3. Addison-Wesley, Reading, Mass.
17. Dan A, Sitaram D, Shahabuddin P (1996) Dynamic batching policies for an on-demand video server. Multimedia Syst 4: 112-121
18. Aggarwal C, Wolf JL, Yu PS (1996) The Maximum-Factor Queue-Length Batching Scheme for Video-on-Demand Systems. IBM Research report RC 20621. T.J. Watson Research Center, Yorktown Heights, New York

19. Bianchi G, Melen R (1996) Performance and Dimensioning of a Hierarchical Video Storage Networks for Interactive Video Services. Eur Trans Teleccommun 7(4): 349–358
20. Bisdikian C, Patel BV (1995) Issues on Movie Allocation in Distributed Video-on-Demand Systems. In: Proceedings of the IEEE International Conf. On Communications, 1995, pp 250–255
21. Wu T-H, Korpeoglu I, Cheng B-C (1997) Distributed Interactive Video System Design and Analysis. IEEE Commun Mag 3: 100–108
22. EU ESPRIT HERMES Project (1999) Technical Report TR5, Digital Libraries: A Survey of Developments in Required Technology;. EU ESPRIT HERMES Project, Technical Univ. of Crete, Multimedia Systems Institute, Chania, Greece. http://www.ced.tuc.gr/Research/Projects.htm
23. Haar PG de, et al. (1997) DIAMOND Project: Video-On-Demand System, and Trials. Eur Trans Teleccommun 8(4): 337-344
24. Bell Atlantic (1996) Fact Sheet: Results of Bell Atlantic Video Services Video-On-Demand Market Trial, Trial Results (May-November 1995). http://www.ba.com/ilr/96/mar/vod-results.htmI



CONSTANTINOS C. VASSILAKIS received his diploma degree from the University of Athens, Department of Informatics, and his M.Sc. degree in Electronics and Computer Engineering from the Technical University of Crete, in 1995 and 1999, respectively. His M.Sc. thesis research was partially supported by the European Community ESPRIT Long-Term Research Project HERMES. Since 1997 he has been working as Telecommunications/Computer Networks Engineer for the University of the Aegean where he participates in several research and development projects funded by the European Community and the Greek government. Currently, he is a Ph.D. student in the Department of Informatics, University of Athens. His research interests include computer communication networks, modeling and performance evaluation of high-speed networks and interactive multimedia services delivery systems.



MICHAEL PATERAKIS received his Diploma degree from the National Technical University of Athens, Greece, his M.Sc. degree from the University of Connecticut, and his Ph.D. degree from the University of Virginia, in 1984, 1986, and 1988, respectively, all in Electrical Engineering. He was an Assistant Professor and later an Associate Professor in the Department of Computer and Information Sciences at the University of Delaware from September 1988. Since 1995, he is a faculty member in the Department of Electronic and Computer Engineering at the Technical University of Crete, Greece, where he is currently a Professor and Department Chairman. His research interests include computer communication networks with emphasis on protocol design, modeling and performance evaluation of braodband high-speed networks, of multiple-access wireless microcellular communication systems, and of packet radio networks; centralized and distributed multimedia information delivery systems; queueing and applied probability theory and their application to computer communication networks and information systems. He has published extensively in scientific journals, refereed conference proceedings and edited books, in the above-mentioned technical areas. He served on the Technical Program Committees of the 1991 Int. Conference on Distributed Computing Systems, the 1992 and 1993 IEEE INFOCOM Conferences, the 1997 IEEE Workshop on the Architecture and Implementation of High-Performance Communication Subsystems (HPCS'97), and the 6th IFIP Workshop on Performance Modeling and Evaluation of ATM Networks (IFIP ATM'98). Professor Paterakis is a Senior Member of the IEEE. He is also a member of the IEEE Technical Committee on Computer Communications, the Greek Chamber of Professional Engineers, and the Greek Association of Electrical and Electronic Engineers.

PETER TRIANTAFILLOU was born in Toronto, Canada in 1963. He received the Ph.D. degree from the Department of Computer Science at the University of Waterloo, Waterloo, Canada in 1991. From September 1991 till August 1996, he was an Assistant Professor at the School of Computing Science at Simon Fraser University, Vancouver, Canada. From September 1994 till January 1996, he was on a leave of absence at the Technical University of Crete, Greece. Since January 1996, he has been with the Department of Electronic and Computer Engineering at the Technical University of Crete, Greece. Currently, Professor Triantafillou's research focuses on the area of high-performance intelligent storage systems (including disk drives, disk arrays, and robotic tape/disk libraries) and on distributed servers, with a special emphasis on supporting multimedia applications. His research activities in the past have focused on multidatabases, distributed file systems, and highly available distributed databases. Prof. Triantafillou has been invited to serve in the Program Committees of EDBT'2000, ACM MobiDE'99, ACM SIGMOD'99, FODO'98, RIDE'98, and EDBT'98 international conferences and as a reviewer in many relevant international journals.