Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2010, Article ID 645049, 13 pages
doi:10.1155/2010/645049

Research Article

An Evolutionary Video Assignment Optimization Technique for VOD System in Heterogeneous Environment

King-Man Ho, Wing-Fai Poon, and Kwok-Tung Lo

Department of Electronic and Information Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong

Correspondence should be addressed to King-Man Ho, enkmho@hotmail.com

Received 8 January 2010; Accepted 21 June 2010

Academic Editor: Markus Kampmann

We investigate the video assignment problem of a hierarchical Video-on-Demand (VOD) system in heterogeneous environments where different quality levels of videos can be encoded using either replication or layering. In such systems, videos are delivered to clients either through a proxy server or video broadcast/unicast channels. The objective of our work is to determine the appropriate coding strategy as well as the suitable delivery mechanism for a specific quality level of a video such that the overall system blocking probability is minimized. In order to find a near-optimal solution for such a complex video assignment problem, an evolutionary approach based on genetic algorithm (GA) is proposed. From the results, it is shown that the system performance can be significantly enhanced by efficiently coupling the various techniques.

1. Introduction

With the explosive growth of the Internet, the demand for various multimedia applications is rapidly increasing in recent years. Among different multimedia applications, Video-on-Demand (VOD) is playing a very important role. With VOD, customers can choose their desired video at arbitrary time they wish via public communication networks. Nevertheless, the VOD system is required to store several hundreds of videos as well as serve thousands of customers simultaneously. In order to build a cost-effective and scalable system, various designs have been proposed in terms of system architecture [1], bandwidth allocation [2], and transmission schemes [3]. Among different techniques, data broadcasting and proxy caching are two commonly used approaches.

To improve the scalability of a VOD system using data broadcasting, the broadcast capability of a network is exploited such that video contents are distributed along a number of video channels shared among clients. Staggered broadcasting [4] is the simplest way to support broadcast services in the early day. After that, a number of efficient broadcasting protocols [5–8] were proposed. Apart from data broadcasting, hierarchical architectures [3] have also been explored to reduce the resources requirement. To leverage the workload of the central server and reduce the service latencies, an intermediate device called proxy is sit between the central server and the clients. In such architecture, a portion of video is cached in the proxy. The request generated by a client is served by the proxy if it caches the requested portion of the video. Meanwhile, the central server also delivers the remaining portion of the video to the client directly. Existing caching mechanisms can be mainly classified into four categories [9]: sliding-interval caching [10], prefix caching [11], segment caching [12], and rate-split caching [13]. Content distribution network (CDN) is an extension of the proxy caching in which a number of CDN servers are deployed at the edge of the network core. Unlike proxy which only stores a portion of the video, a full copy of the video is replicated in each CDN server. Then, the clients request the video from their closest CDN servers directly. This architecture significantly reduces the workload of the central server and provides a better quality of service (QoS) to the clients. Nevertheless, most of the previous works mainly focused on providing VoD services in a homogeneous environment. In a practical situation, the clients can connect to the network, say Internet, with different communication technologies such as modem, ASDL, and wireless link. Their downstream rates vary from 56 kbps to 100 Mbps or even higher. To meet different clients' bandwidth requirement, the videos are encoded into different quality levels by the replication or layering approach. Replication [14] provides multiple versions of the video but at different data rates and one of them will be retrieved according to the requested video quality from the client. On the other hand, layering [15, 16] encodes the video into a number of layers and the client needs to retrieve several video layers concurrently to meet his/her requirements. To adapt such coding scheme, Kangasharju et al. [16] considered delivering layered video through proxy cache and developed a model for the layered video caching problem to determine which videos and which layers should be cached in order to maximize the revenue from the streaming services. The effectiveness of replication and layering for video transmission in a heterogeneous environment has been investigated in [17–19]. Kim and Ammar [17] compared the replication and layering approaches and the results showed that replication is better. However, they only focused on time-dependent streaming of a single video from the central server to the clients. Later, Hartanto et al. [18] studied the system performance with a proxy cache and compared replication with layering in a hierarchical framework. It was found that layering is more appropriate when a proxy server is used. In [19], the authors extended this work by exploring the proxy cache coupled with video broadcast technology. It was observed that layering can have further improvement in such framework. In addition, it was found that the proxy size, the efficiency of the broadcasting scheme, the bandwidth reserved for broadcasting as well as the layering overhead have significant impacts on the system performance. In general, the performance of layering is superior to that of replication. However, from the result in [19], replication performs better in some situations. For instance, replication should be used when the proxy size is zero. Thus, in this paper, we not only use both coding schemes to support different quality of video streams but also explore a hierarchical VoD system using proxy caching coupled with video broadcasting to further improve the system performance in a heterogeneous environment. Different from [19], in the proposed framework, the video streams with different quality levels can be encoded by replication or layering. Each of the video streams are then either cached in the proxy server or delivered over the broadcast/unicast channels. The objective of this work is to determine the appropriate coding strategy as well as the efficient transmission mechanism for a specific quality level of a video such that the overall system blocking probability is minimized. In order to find a near-optimal solution for such a complex video assignment problem, an evolutionary approach based on a genetic algorithm (GA) is proposed. GA has been successfully demonstrated as a powerful optimization tool for solving various real-world complex problems [20] and has been deployed in some VoD applications, such as those mentioned in [21, 22]. The main contribution of this paper is that we explore the benefits of complementary coding schemes for a hierarchical VoD system. To determine the appropriate encoding schemes and the efficient transmission strategies, a mathematic model is formally stated to represent this complex video assignment problem. Then, we present an evolutionary approach based on GA to solve the proposed system model.

This paper is organized as follows. The proposed system architecture and the system model will be first described in Section 2. In Section 3, the formulation of the problem will be derived and the conditions to minimize the system blocking probability will be discussed. The optimal video assignment strategy using GA, where the fitness function and chromosome representation for the problem will then be outlined and explained in Section 4. In Section 5, the experiment results will be presented. Finally, some concluding remarks will be given in Section 6.

2. System Model

In this section, we describe the system architecture for video streaming services. Before we go into the details, the notations used in this paper are defined and listed in Table 1.

Figure 1 shows a two-tier VoD system which consists of one central server and several proxy servers. The central server, which has a large storage space to store M videos for clients, is connected to the proxy servers that are physically located closer to the clients. The clients can connect to the network with different communication technologies such as modem, ASDL, or wireless link and their downstream rates vary from 56 kbps to 100 Mbps. To cater for the heterogeneous requirement, video m will be encoded into l different quality levels of video streams which will be delivered to the clients according to their capacity constraints. If the clients have a low bandwidth connection such as 56 Kbps, they will receive the videos encoded at a low bit rate. On the other hand, the high-quality video will be streamed to the customers having the broadband access capability. In the proposed architecture, jth quality of video m, v_{mj}, can be encoded by the replication or layering approach. Note that a layered-encoded video incurs around 20%–30% overhead compared with a replicated video for the same quality level [17, 18, 23] and thus it requires more transmission bandwidth. Let β be the overhead of the layered-encoded video where $\beta \geq 0$. Then, the relationship of the streaming rate of v_{mj} between these two approaches is given by $\sum_{k=1}^{j} \eta_{mk}^{L} = c_{mj}^{L} = (1+\beta)c_{mj}^{R}$.

It is assumed that the proxy servers are independent and a large group of heterogeneous clients is served by a single proxy server. The proxy server has a limited storage space of K bits to cache some of the popular videos for users' repeating requests in order to minimize the transmission cost. Let $\mathbf{b} = [b_{mj}]_{M\times l}$ denote a *proxy cache map* matrix, where b_{mj} is set to 1 if a copy of v_{mj} is stored in the proxy server. It is set to 0, otherwise. As mentioned, the videos can be layer-encoded or replicated with different quality levels and stored in the proxy server. For layering, the base layer can be decoded independently while the enhancement layers should be decoded cumulatively. That means, layer k should be decoded along with layer 1 to layer $k-1$. To find a feasible cache assignment solution, we define a *coding approach instance* as the vector $\mathbf{e}= (e_1, e_2, \ldots, e_M)$, where

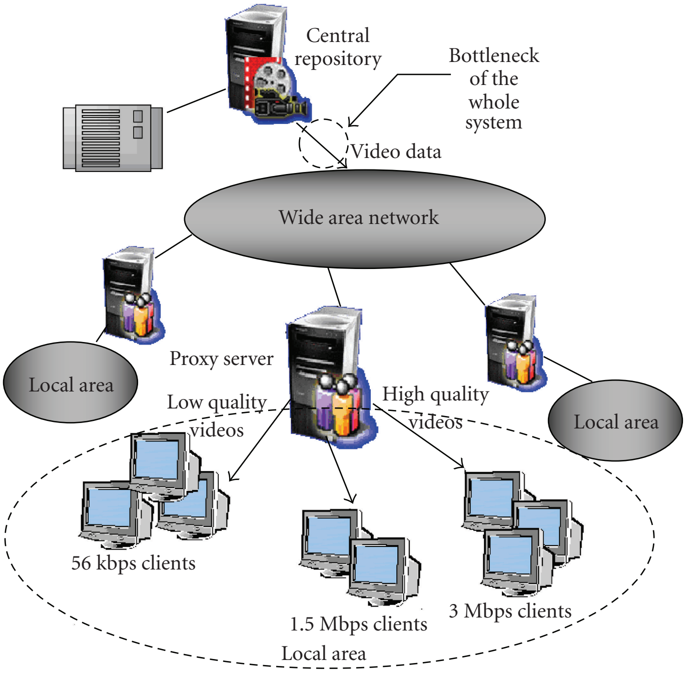


Figure 1: Hierarchical VoD architecture.

$e_m = [0, l]$ indicates the highest quality level of video m encoded by the layering approach reconstructed correctly. In addition, to satisfy the storage space constraint in the proxy server, we have $\gamma \leq K$ where $\gamma = \sum_{m=1}^{M}(\sum_{j=1}^{e_m} s_{mj}^{L} b_{mj} + \sum_{k=e_m+1}^{l} s_{mk}^{R} b_{mk})$. The first term and the second term calculate the storage requirement in the proxy server for the layered video and the replicated video for video m, respectively.

Upon receiving the user's request, the proxy server will acknowledge the request if the requested item has been already cached. Otherwise, it will bypass the request to the higher level. Because the storage capacity of the proxy server is limited, some videos cannot be cached and eventually should be delivered from the central server. It is clearly seen that the system is not scalable as the bandwidth requirement will linearly increase with the arrival rate. Because of recent deployment of IP multicast delivery [24], to further enhance the system performance, broadcasting capability in such a hierarchical architecture is also exploited. Apart from storing the popular videos in the proxy server, some videos will also be broadcast to the clients over the backbone network. Thus, it is assumed that a generic network infrastructure that supports broadcasting operations is used to implement the broadcasting protocols. Since our focus is on the performance of the whole architecture, the broadcasting techniques are not our major concern. In general, any efficient protocols, such as those mentioned in [4–8], can be applied to the system framework. Let H^X be the number of channels required for the protocol x to broadcast a video such that the start-up delay is insensitive to the clients. Given the bandwidth reserved for broadcasting (B_{rsv}), we define $\mathbf{w} = [w_{mj}]_{M\times l}$ as a *broadcast map matrix* to indicate which quality level of a video should be sent over the broadcast channels. w_{mj} is set to 1 if a copy of v_{mj} is broadcast over the broadcast channels. Otherwise, it is set to 0. Therefore, the bandwidth required for broadcasting is equal to $\chi = \sum_{m=1}^{M}(\sum_{j=1}^{e_m} H^x c_{mj}^{L} w_{mj} + \sum_{k=e_m+1}^{l} H^x c_{mk}^{R} w_{mk})$ and $\chi \leq B_{rsv}$. We can then construct a *cache-broadcast map* matrix $\mathbf{o} = [o_{mj}]_{M\times l}$, where $o_{mj} = b_{mj} \mid w_{mj}$ to indicate whether v_{mj} is cached in the proxy server or delivered over the broadcast channels. o_{mj} is equal to 0 if v_{mj} is simply transmitted over unicast channel.

3. Problem Formulation

In this section, the optimization problem of the proposed system is formally defined. It is reported in [25] that the interarrival time of client requests in multimedia streaming applications are exponentially distributed. Thus, the client requests follows a Poisson process with a rate of λ. Let p_m and r_j be the popularity of video m and the probability of client requesting jth quality of video, respectively, where $\sum_{m=1}^{M} p_m = 1$ and $\sum_{j=1}^{l} r_j = 1$. As the request arrival processes for different videos with different quality levels are mutually independent, the request rate of v_{mj} is given by $\lambda p_m r_j$. It is assumed that the video popularity follows Zipf's distribution [26] with the skew parameter θ. Then $p_m = \Omega/m^{1-\theta}$, where $\Omega = (\sum_{i=1}^{M}(1/i^{1-\theta}))^{-1}$. Without loss of generality, it is further assumed that the service time of each unicast channel handled by the central server is exponentially distributed with mean $T = 1/\mu$ by considering the varying length of different videos.

As mentioned in Section 2, some of the requests can be satisfied by the proxy server and the broadcast channels but

Table 1: Summary of notations.

Symbol	Meaning
M	Number of videos in the system
B	Access bandwidth of the central video server (bits/s)
K	Proxy size (in bits)
χ	Bandwidth for broadcasting (bits/s)
λ	System arrival rate (reqs/s)
μ	System service rate
p_m	Popularity of video m
l_m	Number of quality levels of video m
r_j	Probability of customers requesting jth quality of videos
λ_S	Arrival rate for the dedicated streams (reqs/s)
d_S	Average rate of the dedicated streams (bits/s), replication and no broadcast
c^R_{mj}	Streaming rate of replicated video m having jth quality level (bits/s)
s^R_{mj}	Size of replicated video m encoded into jth quality (bits)
c^L_{mj}	Streaming rate of layered video m having jth quality level (bits/s)
η^L_{mj}	Streaming rate of layer j of video m (bits/s)
s^L_{mj}	Size of layer j of video m (bits)
v_{mj}	jth quality of video m
$\mathbf{b}$	Proxy cache map matrix, $\mathbf{b} = [b_{mj}]_{M\times l}$
$\mathbf{e}$	Coding approach instance, $\mathbf{e} = (e_1, e_2, \ldots, e_M)$
$\mathbf{w}$	Broadcasting map matrix, $\mathbf{w} = [w_{mj}]_{M\times l}$
$\mathbf{o}$	Cache-Broadcast map matrix, $\mathbf{o} = [o_{mj}]_{M\times l}$
σ	Crossover rate of GA
δ	Mutation rate of GA
Z	Population size of Gas
Qj	jth quality level

Table 2: Parameters of the experiment.

Parameter	Nominal value (range)
Number of videos (M)	50
System arrival rate (λ)	0.3, 0.8 (0.1–1 reqs/s)
Proxy size (K)	5% (5% of the total video storage requirement in the replication system is about 5 Gbits.) (0%, 5%, 10%)
Number of broadcast channels (H^x)	10
Access bandwidth of the central video server (B)	100 Mbps
Layering overhead (β)	0.25
Proportion of bandwidth reserved (p_{rsv})	0.1 (0, 0.1, 0.5)
Crossover rate of GA (σ)	0.6
Mutation rate of GA (δ)	0.01
Population size of GA (Z)	20
Skew parameter (θ)	0.271 (0, 0.217, 1)
Layer stream rate (η^L_{mj})	η^L_{m1}

the central server still opens the dedicated channels to serve the clients due to the small proxy storage capacity and the limited broadcasting bandwidth. Equation (1) calculates the requests that go up to the central server for the dedicated streams:

$$\lambda_S = \lambda\left(1 - \sum_{m=1}^{M}\sum_{j=1}^{l} p_m r_j o_{mj}\right). \tag{1}$$

Since multiple qualities of video streams are delivered at different data rates from the central server to the clients, the average streaming rate of the dedicated channels can thus be found by

$$d_S = \frac{\lambda}{\lambda_s}\left(\sum_{m=1}^{M}\left(\sum_{j=1}^{e_m} p_m r_j \sum_{k=1}^{j} \eta^L_{mk}\overline{o_{mj}} + \sum_{j=e_m+1}^{l} p_m r_j c^R_{mj}\overline{o_{mj}}\right)\right), \tag{2}$$

where $\overline{o_{mj}}$ is the complement of o_{mj}. The first term calculates the average bandwidth of the dedicated channels required for the layered-encoded videos while the second term computes that for the replicated videos.

To evaluate the performance of the central server, denote B as the available bandwidth between the central and proxy servers. Therefore, on average, the central server can support N virtual channels concurrently for the clients, where $N = \lceil (B-\chi)/d_S \rceil$. According to the Erlang's loss formula [27], the system can thus be modeled as an $M/G/N/N$ queueing system and the blocking probability is equal to

$$P_S = \frac{(\lambda_S/\mu)^N/N!}{\sum_{j=0}^{N}(\lambda_S/\mu)^j/j!}. \tag{3}$$

Table 3: Coding scheme and broadcast-cache map for different system configurations.

(a) SCENARIOA

Video ID	Q1	Q2	Q3	Q4	Q5	Q6	Q7
Layering System							
1	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)
2	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)
3	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(0, L)
4	(1, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
5	(1, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
6	(1, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
7	(1, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
8	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
9	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
10	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
11	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
12	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
13	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
14	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
15	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
16	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
17	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
18	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
19	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
20	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
21	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
22	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
23	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
24	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
25	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
Replication System							
1	—	(1, R)	—	—	—	—	(1, R)
2	—	(1, R)	—	—	—	—	(1, R)
3	—	(1, R)	—	—	—	—	(1, R)
4	—	(1, R)	—	—	—	—	(0, R)
5	—	(1, R)	—	—	—	—	(0, R)
6	—	(0, R)	—	—	—	—	(0, R)
7	—	(0, R)	—	—	—	—	(0, R)
8	—	(0, R)	—	—	—	—	(0, R)
9	—	(0, R)	—	—	—	—	(0, R)
10	—	(0, R)	—	—	—	—	(0, R)
11	—	(0, R)	—	—	—	—	(0, R)
12	—	(0, R)	—	—	—	—	(0, R)
13	—	(0, R)	—	—	—	—	(0, R)
14	—	(0, R)	—	—	—	—	(0, R)
15	—	(0, R)	—	—	—	—	(0, R)
16	—	(0, R)	—	—	—	—	(0, R)
17	—	(0, R)	—	—	—	—	(0, R)
18	—	(0, R)	—	—	—	—	(0, R)
19	—	(0, R)	—	—	—	—	(0, R)
20	—	(0, R)	—	—	—	—	(0, R)
21	—	(0, R)	—	—	—	—	(0, R)
22	—	(0, R)	—	—	—	—	(0, R)

(a) Continued.

Video ID	Q1	Q2	Q3	Q4	Q5	Q6	Q7
23	—	(0, R)	—	—	—	—	(0, R)
24	—	(0, R)	—	—	—	—	(0, R)
25	—	(0, R)	—	—	—	—	(0, R)
			Mixed System				
1	(0, L)	(1, L)	—	—	—	—	(0, R)
2	(0, L)	(1, L)	—	—	—	—	(0, R)
3	(0, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(1, L)
4	(0, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(1, L)
5	(0, L)	(1, L)	—	—	—	—	(0, R)
6	(0, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(1, L)
7	(0, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(1, L)
8	(0, L)	(1, L)	—	—	—	—	(0, R)
9	(0, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(1, L)
10	(0, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(1, L)
11	(0, L)	(1, L)	—	—	—	—	(0, R)
12	(0, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(1, L)
13	(0, L)	(1, L)	—	—	—	—	(0, R)
14	(0, L)	(1, L)	—	—	—	—	(0, R)
15	—	(0, R)	—	—	—	—	(0, R)
16	(0, L)	(1, L)	—	—	—	—	(0, R)
17	—	(0, R)	—	—	—	—	(0, R)
18	(0, L)	(1, L)	—	—	—	—	(0, R)
19	(0, L)	(1, L)	—	—	—	—	(0, R)
20	(0, L)	(1, L)	—	—	—	—	(0, R)
21	(0, L)	(1, L)	—	—	—	—	(0, R)
22	(0, L)	(1, L)	—	—	—	—	(0, R)
23	(0, L)	(1, L)	—	—	—	—	(0, R)
24	—	(0, R)	—	—	—	—	(0, R)
25	—	(0, R)	—	—	—	—	(0, R)

(b) SCENARIO(B)

Video ID	Q1	Q2	Q3	Q4	Q5	Q6	Q7
			Layering System				
1	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(0, L)
2	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(0, L)	(0, L)
3	(1, L)	(1, L)	(1, L)	(1, L)	(0, L)	(0, L)	(0, L)
4	(1, L)	(1, L)	(1, L)	(1, L)	(0, L)	(0, L)	(0, L)
5	(1, L)	(1, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)
6	(1, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
7	(1, L)	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
8	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
9	(1, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
10	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
11	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
12	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
13	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
14	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
15	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
16	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
17	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
18	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
19	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)

(b) Continued.

Video ID	Q1	Q2	Q3	Q4	Q5	Q6	Q7
20	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
21	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
22	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
23	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
24	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
25	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)	(0, L)
Replication System							
1	(1, R)	(1, R)	(1, R)	(1, R)	(1, R)	(1, R)	(1, R)
2	(1, R)	(1, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
3	(1, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
4	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
5	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
6	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
7	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
8	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
9	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
10	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
11	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
12	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
13	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
14	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
15	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
16	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
17	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
18	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
19	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
20	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
21	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
22	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
23	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
24	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
25	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
Mixed System							
1	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)
2	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)
3	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)
4	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)	(1, L)
5	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
6	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
7	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
8	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
9	(0, L)	(0, L)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
10	(0, L)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
11	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
12	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
13	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
14	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
15	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
16	(0, L)	(0, L)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
17	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
18	(0, L)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)

(b) Continued.

Video ID	Q1	Q2	Q3	Q4	Q5	Q6	Q7
19	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
20	(0, L)	(0, L)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
21	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
22	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
23	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
24	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)
25	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)	(0, R)

If the bandwidth from the proxy server to the clients is large enough and no requests will be blocked, the overall blocking probability of the system (P_O) is given by

$$P_O = \frac{\lambda_S P_S}{\lambda}. \tag{4}$$

Considering the coding approach (replication and layering) and transmission strategy (caching and broadcasting), the optimization problem (OPT) can thus be formally stated as follows:

$$\text{Minimize} \quad P_O$$

$$\text{Subject to} \quad \sum_{m=1}^{M}\left(\sum_{j=1}^{e_m} s_{mj}^{L} b_{mj} + \sum_{k=e_m+1}^{l} s_{mk}^{R} b_{mk}\right) \leq K, \tag{5}$$

$$\sum_{m=1}^{M}\left(\sum_{j=1}^{e_m} H^x c_{mj}^{L} w_{mj} + \sum_{k=e_m+1}^{l} H^x c_{mk}^{R} w_{mk}\right) \leq B_{rsv}. \tag{6}$$

Equation (5) indicates the constraint that the total size of the cached videos is less than or equal to the proxy size and (6) shows that the broadcasting bandwidth is not larger than the bandwidth reserved for broadcasting.

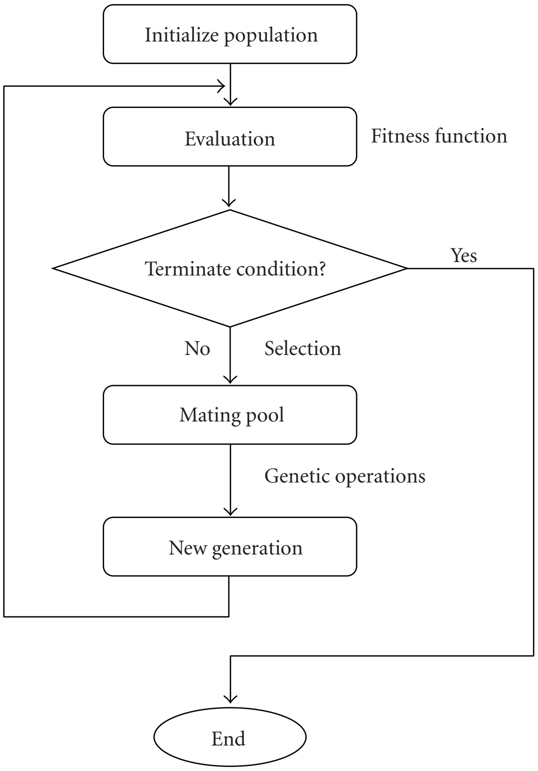


Figure 2: Flowchart of GA-based video coding and placement strategy.

4. Evolution Optimization

In this section, we exploit a GA-based approach to obtain a near optimal solution for the OPT problem in Section 3. We first briefly review the terminologies and operations of GA. Then, to solve the problem, the chromosome representation, the population size, and the fitness function for the OPT problem are discussed.

4.1. Genetic Algorithm. Genetic Algorithm (GA) is a population-based generic search method inspired by the *survival of the fittest* principal [28–30] that is derived from the mechanism of natural evolution context, where the stronger individual would likely be the champion in a competing world. The potential solution to the problem known as chromosome is constructed by a finite length of gene represented by a finite-length string over some finite alphabet (e.g., in a binary form). A pool of chromosomes forms a population, which is randomly generated at the beginning of the process. In each iteration, GA performs multidirectional stochastic search through a genetic evolution process by applying a number of genetic operators to the individual of the current population in order to produce individuals for the next generation. In general, a genetic operator known as *crossover* is used to combine two or more individuals from the pool to produce new individuals in the next generation. To introduce a genetic variation into the individual, *mutation* operator is applied to alter the value of each gene (i.e., allele) in an individual randomly with a small probability. Based on the fitness of the individuals in the current population, the individuals with a higher degree of fitness will be selected as a member of the population in the next generation through the *selection* process of GA. After a certain generation, it is expected that the best chromosome can be obtained which is reasonably close to the optimal solution. Figure 2 shows the general procedures of GA. The detailed working principle and implementation of GA can be found in [28–30]. GA has been successfully demonstrated as a powerful optimization

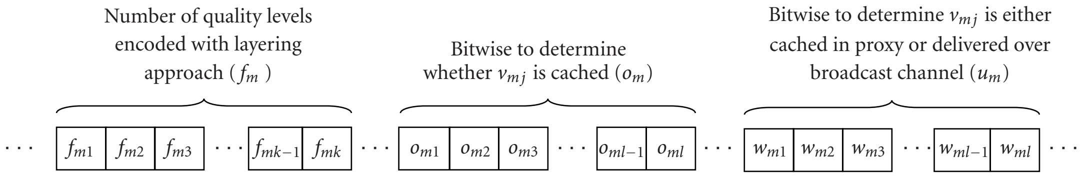
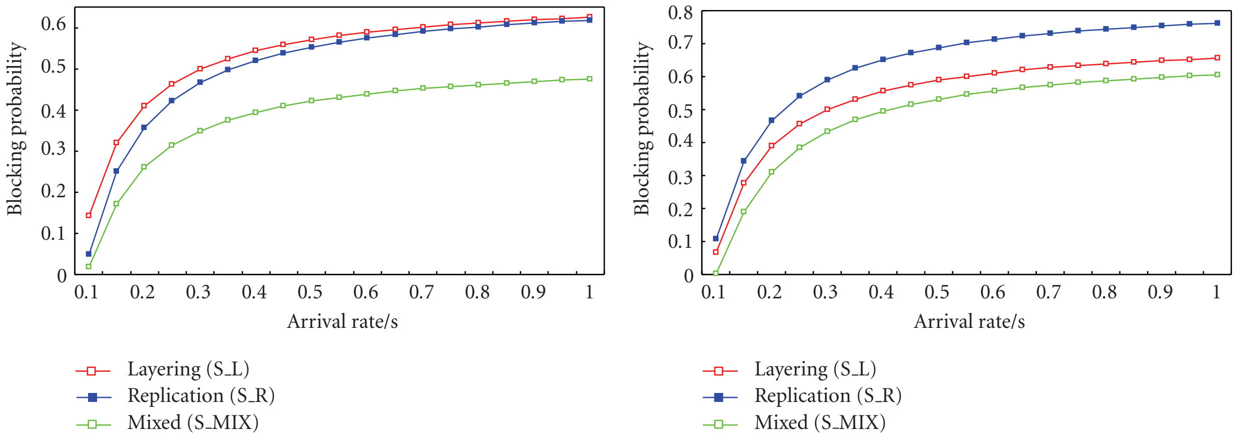

Number of quality levels encoded with layering approach (f_m)
Bitwise to determine whether v_{mj} is cached (o_m)
Bitwise to determine v_{mj} is either cached in proxy or delivered over broadcast channel (u_m)
$\cdots$ f_{m1} f_{m2} f_{m3} $\cdots$ f_{mk-1} f_{mk} $\cdots$ o_{m1} o_{m2} o_{m3} $\cdots$ o_{ml-1} o_{ml} $\cdots$ w_{m1} w_{m2} w_{m3} $\cdots$ w_{ml-1} w_{ml} $\cdots$

Figure 3: Chromosome.



(a) Blocking Probability (SCENARIO(A))

(b) Blocking Probability (SCENARIO(B))

Figure 4: System performances against arrival rate.

tool for solving various real-world complex problems [20] and has been deployed in some applications, such as those mentioned in [21, 22].

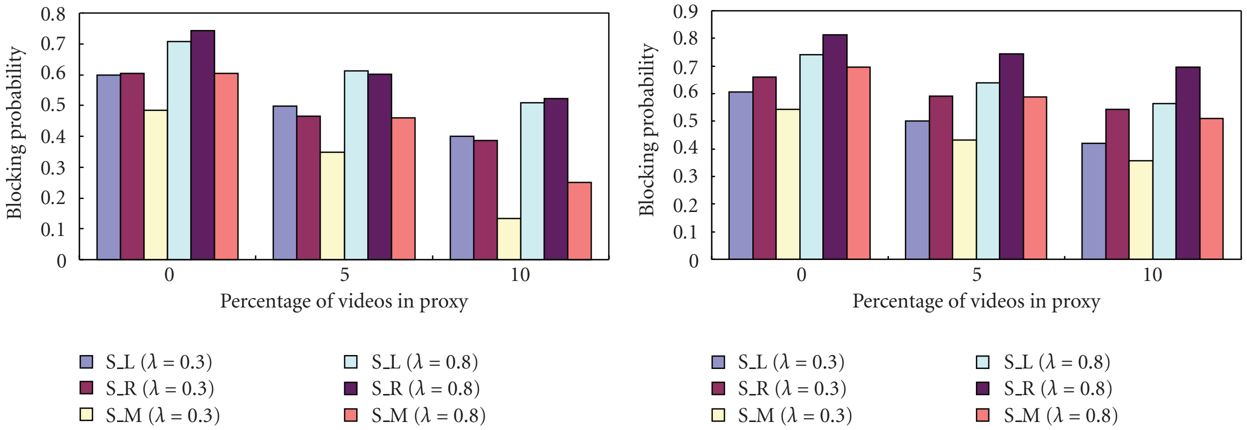
4.2. Chromosome Representation. To represent the coding strategy and the caching mechanism of v_{mj}, 3 vectors are defined. Let vector $\mathbf{u_i} = (w_{i1}, w_{i2}, \ldots, w_{il})$ and w_{ij} is set to 1 if v_{ij} is delivered over broadcast channel as mentioned. Then, vector $\mathbf{o_i} = (o_{i1}, o_{i2}, \ldots, o_{il})$, that is, $\mathbf{o} = (o_1, o_2, \ldots, o_M)^T$, is defined (it is reminded that $o_{mj} = b_{mj} \mid w_{mj}$). In addition, let $f_i = (f_{i1}, f_{i2}, \ldots, f_{ik})$ be the binary form of e_i for video i (note that f_{i1} is MSB while f_{ik} is LSB (MSB means most significant bit, LSB means least significant bit)). Since the highest value of e_i is l, the number of bits required for representing e_j is given by $k = \lceil \log_2 l \rceil$ for all j. Therefore, the chromosome can be represented in the form of binary string $I = \{\{f_1, f_2, \ldots, f_M\}, \{o_1, o_2, \ldots, o_M\}, (w_1, w_2, \ldots, w_M\}\}$ as depicted in Figure 3 and the allele space of each gene is $\{0, 1\}$. The total number of bits required for the chromosome can then be expressed by $G = M(k + 2l)$ and thus the searching space includes 2^G possible solutions.

4.3. Population Size. Population size is a critical factor affecting the performance of GA. Basically, a large population size requires a high computational cost while a small population size increases the chance of premature convergence. Other than randomly choosing initial populations, Reeves [31] proposed the principle of *minimum population sizes for τ-ary alphabets* to decide an appropriate value. The author suggested a preferable property of an initial population such that "every possible point in the search space should be reachable from the initial population by crossover only." This property can be satisfied only if there is at least one instance of every allele at each locus in the whole population of chromosomes [31]. Given the population size Z, the length of chromosome G and the cardinality τ of the gene at each locus, the probability that at least one allele is presented at each locus in the initial population (ψ) can be computed by

$$\psi = \left(\frac{\tau! S(Z, \tau)}{\tau^Z} \right)^G, \tag{7}$$

where $S(Z, \tau)$ is the Stirling number of the second kind. Equation (7) provides a guideline to choose a suitable Z so that it is large enough to ensure a high probability ψ in the initial population. For example, to achieve $\psi \geq 0.999$, the minimum value of Z should be 21 given $M = 50$, $l = 7$, and $\tau = 2$.

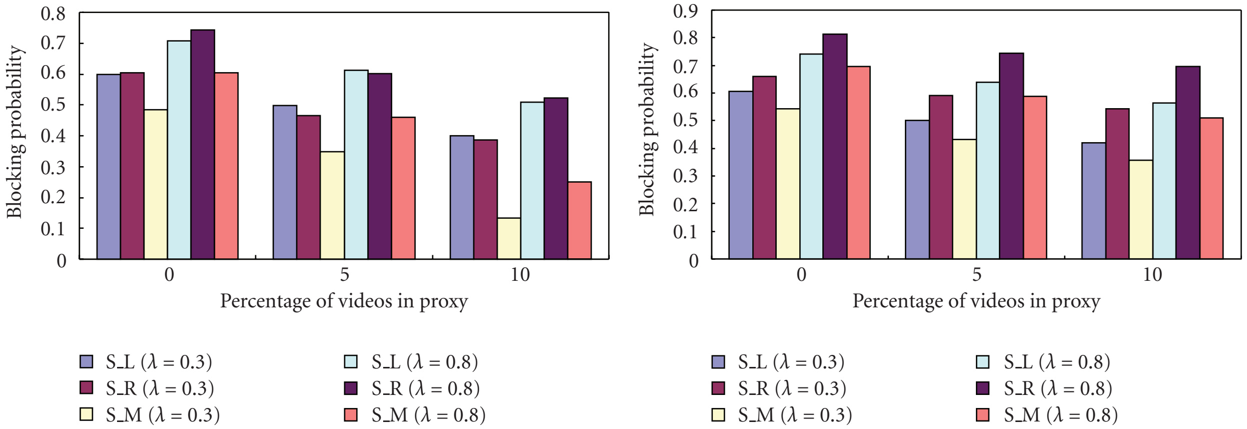
4.4. Fitness Function. In GA, the fitness function is used to evaluate the goodness of a chromosome for the problem. The fitness function F of a chromosome is closely related to the output of the objective function (i.e., OPT) by this chromosome. Note that v_{mj} can be either cached in the proxy server or delivered over the broadcast channels if o_{mj} is set. However, it is obvious that the proxy capacity required for caching and the bandwidth required for broadcasting may exceed the limitations and the constraints in OPT are



(a) Blocking Probability (SCENARIO(A))

(b) Blocking Probability (SCENARIO(B))

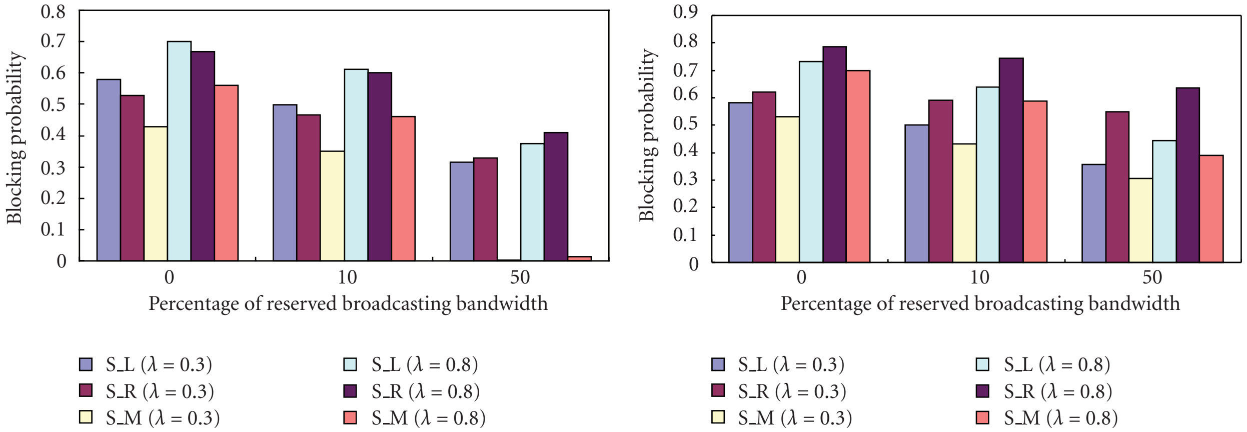
Figure 5: System performances against proxy size.



(a) Blocking Probability (SCENARIO(A))

(b) Blocking Probability (SCENARIO(B))

Figure 6: System performances against reserved broadcasting bandwidth.



(a) Blocking Probability (SCENARIO(A))

(b) Blocking Probability (SCENARIO(B))

Figure 7: System performances against skew parameter.

violated. A penalty scheme is thus applied to those chromosomes violating these constraints. Hence, we transform OPT to an unconstrained form to produce the fitness function:

$$F = P_O + \phi_1(y) + \phi_2\big(\chi\big), \tag{8}$$

where ϕ_i is the penalty function. To reflect the condition of the low performers, we square the violation of the constraints $\phi_i(y) = y^2$ [29].

5. Experimental Results

In our experiment, GAlib [32], which is a set of C++ genetic algorithm objects to perform optimization, is used to solve the OPT problem. It is assumed that there are 50 videos in the system and each of them is fixed as 90 minutes long and is encoded into seven quality levels. The client requests are modeled as the Poisson arrival process and the video popularity is followed by Zipf's distribution with the skew parameter $\theta = 0.271$. Assume that the streaming rate of the base layer of all videos is $\eta^L_{m1} = 56$ Kbps and all layers that have the same rate [33], that is, $\eta^L_{mj} = \eta^L_{m1}$. As the backbone bandwidth is fixed, the proportion of bandwidth, p_{rsv}, is reserved for video broadcasting, that is, $B_{rsv} = p_{rsv}B$. The results in [8] showed that less than 10 broadcasting channels are sufficient to provide delay insensitive VoD services. Hence, H^x is set to 10 for the following experiments. As reported in [23], the amount of overhead incurred by the layered encoded videos is varied from 0 to 30%. To analogize the heterogeneity of network environments, two requesting patterns, namely, "SCENARIO(A)" and "SCENARIO(B)", are defined in our experiment [19]. "SCENARIO(A)" is to model the less heterogeneity environment where the system only serves two types of clients (e.g., modem and Ethernet), that is, $r_2 = r_7 = 0.5$ but $r_1 = r_3 = r_4 = r_5 = r_6 = 0$. "SCENARIO(B)" focuses on the high heterogeneity environment that all the qualities of a video are requested uniformly, that is, $r_j = 1/l$, for all j. Table 2 summarizes the parameters used in the experiment.

We first evaluate the performance impact of various arrival rates to the blocking probability and compare the proposed system with the system using either the layering (S_L) or replication (S_R) approach (i.e., the system only uses layering or replication [19]). In Figure 4, as expected, the blocking probability is increasing when the arrival rate is increased under various configurations. It can be seen that the system with both layering and replication (S_MIX) can perform better than S_L and S_R in both scenarios. It can be found in Figure 4(a) that S_MIX can have a significant improvement in less heterogeneity environment. When the arrival rate is 0.1 req/s, the blocking probability of S_MIX is reduced to 0.018 (S_R is 0.048 and S_L is 0.143). Note that S_MIX can still obtain up to about 20% reduction of blocking probability if the arrival rate is increased to 1 req/s. In SCENARIO(B), it can be observed that S_MIX can have an improvement up to 8% as shown in Figure 4(b).

To investigate how the system can be improved by S_MIX approach, we first look at how the coding and cache strategy for different quality levels of videos in S_MIX is organized by GA as compared with that in S_L and S_R. Tables 3 depict the coding scheme and proxy-broadcast map for different system configurations. In the table, the coding and cache strategy for a specific quality level of video is represented by "(x, y)", where "$x = o_{mj}$" and "y = coding scheme (R = Replication, L = Layering)". "—" represents that the corresponding quality level is not required. We only show the configuration of the first 25 videos as the configuration of the rest videos are the same as the 25th one. In Table 3(a), it can be observed that all quality levels of the videos should be encoded by layering in S_L and only two quality levels are needed if replication is used in S_R. In S_MIX, it can be seen that the quality levels are encoded by the layering approach only if the upper quality levels of the corresponding video is cached in proxy or delivered over the broadcast channels. On the other hand, replication is used when the video is not cached or broadcast. As layering is suitable for caching and replication is favor to end-to-end transmission, S_MIX takes the benefits from both approaches. Unlike S_R and S_L that videos are cached according to the videos, S_MIX takes both coding strategy as well as the bandwidth usage into account. It is found that S_MIX allocates the cache space to most of the 2nd quality level of layered-encoded videos. Although the 1st quality level of the corresponding videos is required to be transmitted over the dedicated channel when the users request for the 2nd quality level of the videos, the server bandwidth requirement of S_MIX is still less than that of S_R because part of the video data can be obtained from the proxy server or the broadcast channels directly. Similar observations can been found in "SCENARIO(B)." Only cached or broadcast videos are layered-encoded and the others use replication so that more videos can be served by the proxy server or the broadcast channels as compared to S_R and fewer server bandwidth are required as compared to S_L.

In order to have a close look on the effectiveness of S_MIX, Figures 5 and 6 show the blocking probability of the systems when these parameter are varied. We first investigate the impact of the proxy size. Figure 5 illustrates the system blocking probability as the proxy size is changed. Increasing the proxy size results in fewer video requests to the central server and thus the blocking probabilities are decreasing. It can be seen that S_MIX can perform better than S_L and S_R in both requesting patterns, especially at low arrival (i.e., 0.3 req/s) and large proxy size. In Figure 5(a), S_MIX can have significant improvement but S_L and S_R only have a linear improvement when the proxy size is changed. When K is set to 10%, S_MIX obtains up to about 65% reduction of blocking probability. When the arrival rate is increased to 0.8 req/s, the system can still achieve 50% improvement compared to S_L. When the proxy size is increased, more layered-encoded videos with lower quality levels are assigned to the proxy server in S_MIX. Thus, more videos with less popularity can also be served by the proxy server directly. The similar trend can be observed in "SCENARIO(B)" which is shown in Figure 5(b). The results show that the blocking probability of S_MIX can be less than that of S_L up to 10%.

Figure 6 shows the blocking probability when the proportion of bandwidth reserved for broadcasting is changed.

It can be seen that the system performance is greatly improved in S_MIX compared to S_L and S_R when p_{rsv} is increased, especially in less heterogeneity network environment. Although the system blocking probability can be further reduced when p_{rsv} is increased, the system will suffer from a problem that the remaining bandwidth is not sufficient for the less popular videos.

The skew parameter against the blocking probability is plotted in Figure 7. As expected, the blocking probability is increasing with the skew parameter. The performance of S_MIX is superior to that of S_L and S_R even if the popularity of all quality levels of all the videos are uniformly distributed, that is, $\theta = 1.0$. In "SCENARIO(A)", the blocking probability of S_MIX is reduced to 0.5 (S_R is 0.696 and S_L is 0.686) when $\lambda = 0.3$ and $\theta = 1.0$. For high arrival rate, S_MIX can still achieve up to about 18% reduction of the blocking probability.

6. Conclusion

In this paper, we investigate a feasible enhancement solution to a hierarchical VoD system using proxy caching coupled with video broadcasting and appropriate coding schemes in a heterogeneous environment. In the proposed framework, different quality levels of video can be encoded by either replication or layering approach. Each of them is then either cached in proxy server or delivered over video broadcast channels/or unicast channels. The objective of this work is to determine the appropriate coding strategy as well as the suitable delivery mechanism to a specific quality level of video such that the overall system blocking probability is minimized. To solve this complex problem, an evolutionary approach based on a genetic algorithm (GA) is used for finding a near-optimal solution for this difficult video assignment problem. From the results, it can be seen that the system performance can be significantly enhanced by efficiently coupling the various techniques. In this paper, we focus on videos coded with MPEG2 with different coding layers. Recently, the new scalable video coding (SVC) extension of H.264/AVC standard [34] provides network-friendly scalability at a bit stream level has been proposed. We are going to investigate the performance of the system with this coding technique in our framework in the future.

References

[1] F. Thouin and M. Coates, "Video-on-demand networks: design approaches and future challenges," *IEEE Network*, vol. 21, no. 2, pp. 42–48, 2007.

[2] A. Dan, P. Shahabuddin, D. Sitaram, and D. Towsley, "Channel allocation under batching and VCR control in Video-on-Demand systems," *Journal of Parallel and Distributed Computing*, vol. 30, no. 2, pp. 168–179, 1995.

[3] K. A. Hua, M. A. Tantaoui, and W. Tavanapong, "Video delivery technologies for large-scale deployment of multimedia applications," *Proceedings of the IEEE*, vol. 92, no. 9, pp. 1439–1451, 2004.

[4] J. W. Wong, "Broadcast delivery," *Proceedings of the IEEE*, vol. 76, no. 12, pp. 1566–1577, 1988.

[5] K. A. Hua and S. Sheu, "Skyscraper broadcasting: a new broadcasting scheme for metropolitan video-on-demand systems," in *Proceedings of the Conference on Communications Architectures, Protocols and Applications (SIGCOMM '97)*, pp. 89–100, 1997.

[6] L.-S. Juhn and L.-M. Tseng, "Harmonic broadcasting for video-on-demand service," *IEEE Transactions on Broadcasting*, vol. 43, no. 3, pp. 268–271, 1997.

[7] W. C. Liu and J. Y. B. Lee, "Constrained consonant broadcasting—a generalized periodic broadcasting scheme for large scale video streaming," in *Proceedings of the IEEE International Conference on Multimedia & Expo*, Baltimore, Md, USA, July 2003.

[8] E. M. Yan and T. Kameda, "An efficient VoD broadcasting scheme with user bandwidth limit," in *Proceedings of the SPIE/ACM Conference on Multimedia Computing and Networking*, vol. 5019, pp. 200–208, Santa Clara, Calif, USA, 2003.

[9] J. Liu and J. Xu, "Proxy caching for media streaming over the internet," *IEEE Communications Magazine*, vol. 42, no. 8, pp. 88–94, 2004.

[10] R. Tewari, H. M. Vin, A. Dan, and D. Sitaram, "Resource-based caching for web servers," in *Proceedings of the Multimedia Computing and Networking (MMCN '98)*, pp. 191–204, San Jose, Calif, USA, January 1998.

[11] S. Sen, J. Rexford, and D. Towsley, "Proxy prefix caching for multimedia streams," in *Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societie (INFOCOM '99)*, pp. 1310–1319, New York, NY, USA, March 1999.

[12] S. Chen, B. Shen, S. Wee, and X. Zhang, "Designs of high quality streaming proxy systems," in *Proceedings of the IEEE Conference on Computer Communications (INFOCOM '04)*, pp. 1512–1521, Hong Kong, March 2004.

[13] Z.-L. Zhang, Y. Wang, D. H. C. Du, and D. Su, "Video staging: a proxy-server-based approach to end-to-end video delivery over wide-area networks," *IEEE/ACM Transactions on Networking*, vol. 8, no. 4, pp. 429–442, 2000.

[14] T. Jiang, M. H. Ammar, and E. W. Zegura, "Inter-receiver fairness: a novel performance measure for multicast ABR sessions," in *Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, vol. 26, pp. 202–211, Madison, Wis, USA, June 1998.

[15] R. Rejaie and J. Kangasharju, "Mocha: a quality adaptive multimedia proxy cache for Internet streaming," in *Proceedings of the 11th IEEE International Workshop on Network and Operating System Support for Digital Audio and Video*, pp. 3–10, January 2001.

[16] J. Kangasharju, F. Hartanto, M. Reisslein, and K. W. Ross, "Distributing layered encoded video through caches," *IEEE Transactions on Computers*, vol. 51, no. 6, pp. 622–636, 2002.

[17] T. Kim and M. H. Ammar, "A comparison of layering and stream replication video multicast schemes," in *Proceedings of the IEEE International Workshop on Network and Operating System Support for Digital Audio and Video*, pp. 63–72, New York, NY, USA, 2001.

[18] F. Hartanto, J. Kangasharju, M. Reisslein, and K. W. Ross, "Caching video objects: layers vs. versions," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '02)*, vol. 2, pp. 45–48, Lausanne, Switzerland, August 2002.

[19] K.-M. Ho, W.-F. Poon, and K.-T. Lo, "Performance study of large-scale video streaming services in highly heterogeneous environment," *IEEE Transactions on Broadcasting*, vol. 53, no. 4, pp. 763–773, 2007.

[20] J. H. Holland, *Adaptation in Natural and Artificial Systems*, The MIT Press, Cambridge, Mass, USA, 1975.

[21] K.-S. Tang, K.-T. Ko, S. Chan, and E. W. M. Wong, "Optimal file placement in VOD system using genetic algorithm," *IEEE Transactions on Industrial Electronics*, vol. 48, no. 5, pp. 891–897, 2001.

[22] W. K. S. Tang, E. W. M. Wong, S. Chan, and K.-T. Ko, "Optimal video placement scheme for batching VOD services," *IEEE Transactions on Broadcasting*, vol. 50, no. 1, pp. 16–25, 2004.

[23] J. I. Kimura, F. A. Tobagi, J. M. Pulido, and P. J. Emstad, "Perceived quality and bandwidth characterization of layered MPEG-2 video encoding," in *International Symposium Voice, Video and Data Communications*, vol. 3845 of *Proceedings of SPIE*, pp. 308–319, Boston, Mass, USA, 1999.

[24] A. Ganjam and H. Zhang, "Internet multicast video delivery," *Proceedings of the IEEE*, vol. 93, no. 1, pp. 159–170, 2005.

[25] C. Costa, I. Cunha, A. Borges et al., "Analyzing client interactivity in streaming media," in *Proceedings of the 13th International World Wide Web Conference Proceedings (WWW '04)*, pp. 534–543, May 2004.

[26] G. Zipf, *Human Behavior and the Principle of Least Effort*, Addison-Wesley, Reading, Mass, USA, 1949.

[27] J. Medhi, *Stochastic Process*, Wiley InterScience, New York, NY, USA, 2nd edition, 1994.

[28] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Program*, Springer, Berlin, Germany, 3rd edition, 1996.

[29] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, London, UK, 1989.

[30] K. F. Man, K. S. Tang, and S. Kwong, *Genetic Algorithm: Concepts and Designs*, Springer, London, UK, 1999.

[31] C. R. Reeves, "Using genetic algorithms with small populations," in *Proceedings of the 5th International Conference on Genetic Algorithms (ICGA '93)*, pp. 92–99, 1993.

[32] GAlib, http://lancet.mit.edu/ga/.

[33] R. Rejaie, M. Handley, and D. Estrin, "Layered quality adaptation for Internet video streaming," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 12, pp. 2530–2543, 2000.

[34] H. Schwarz, D. Marpe, and T. Wiegand, "Overview of the scalable video coding extension of the H.264/AVC standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1103–1120, 2007.