ISSN:0975-9646

Kriti Priya Gupta / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (6) , 2014, 7430-7438

Genetic Algorithm Approach for Bandwidth Optimization in Near Video on Demand System

Kriti Priya Gupta
Symbiosis Centre for Management Studies, NOIDA
Faculty of Management Studies, Symbiosis International University

Abstract-In this paper, we employ Genetic Algorithm (GA) technique for minimizing the average bandwidth requirement in Near Video-on-Demand (NVoD) system. Three multicasting schemes are presented which require lesser bandwidth as compared to true Video-on Demand (VoD) systems. Scheme 1 is a double rate batching scheme in which the late arriving customers are served with unicast stream having double transmission rate until they get merged with the multicast stream. Scheme 2 is a client-buffering technique in which the unicast customers are allowed to concurrently buffer some part of the movie from the ongoing multicast stream. In scheme 3, the late arriving customers are served with bundled channels of incrementally increasing transmission rate. All the schemes are compared on the basis of the required bandwidth i.e. average number of I/O streams. The optimal batching time and the minimum streams required are determined by using GA. Numerical results are provided for verifying the analytical results with the GA results.

Keywords: Genetic Algorithm, Near video on demand, Multicasting, Bandwidth, Optimization.

1. INTRODUCTION

In the recent years, video-on-demand (VoD) has become a new source of interactive entertainment via computer communication networks. In a VoD system, customers can choose any movie from a distant video server just by using a remote control, to watch at any time they wish. A true VoD system provides a dedicated transmission stream to each customer. This type of system is quite infeasible to deal with a large number of customers, as it requires a very large bandwidth or a large number of I/O streams of the video server. Minimization of bandwidth is a big problem in VoD systems, which has attracted the attention of many researchers. The study in [14] provides some techniques for bandwidth resource optimization in VoD network architectures.

One way of efficient utilization of the bandwidth is near video-on-demand (NVoD) systems in which, customers requesting the same movie are grouped together in batches and then the movie is broadcasted to them using multicasting via a single transmission stream. NVoD is a cost effective solution for a large scale VoD system as it minimizes the bandwidth requirement by using multicast streams. The study in [22] discusses the advantages of NVoD systems over true VoD systems. The multicast streams are opened after a particular time, which is called as the batching time. When a customer arrives after the opening of the multicast stream, he is served via a dedicated unicast stream with faster transmission rate and as soon as the unicast stream comes in synchronization with the multicast stream, the customer is merged in the previous multicast stream. The selection of the batching time greatly affects the performance of such a system. The authors in [5] have suggested dynamic batching policies for VoD system. Several broadcasting schemes like Pyramid broadcasting [21], Harmonic broadcasting [10] and Skyscraper broadcasting [8] are proposed for metropolitan VoD systems to reduce the bandwidth requirement. Channel allocation problem in VoD system using batching adaptive piggybacking has been discussed in [11] but this approach is very complex, as it requires a replica of videos with different playout rates to be stored in the server in advance. A double-rate batching policy has been developed in [15], in which the customers arriving after the beginning of the multicast stream, are served by the unicast stream with double transmission rate. When the unicast stream comes in synchronization with the multicast stream, the unicast stream is released and the customer is merged in the multicast stream. But this policy is not scalable for very popular movies with high arrival rates. The study in [16] suggests an adaptive batching scheme, where the batching time changes according to the arrival rate of the customers. The fundamental limitations of multicast streaming algorithms in supporting interactive playback control have been investigated in [23] and a general solution is presented which can be applied to many of the existing multicast streaming algorithms to substantially improve their performance when interactive playback control was to be supported. The authors in [19] have studied optimal segment caching for peer to peer on-demand streaming. They have proposed a centralized heuristic to solve the segment caching optimization problem. They have also proposed a distributed caching algorithm in which each peer adaptively and independently replaced segments to minimize the popularity-supply discrepancy.

The NVoD users have to wait for some time before the required movies are actually displayed on their terminals or TV sets. This time is referred to as 'delay', which should be kept low in order to provide good quality of service. But, reducing the start up delay results in an increased bandwidth or increased number of multicasting streams. In such situations, bandwidth can be reduced by providing client (user) buffers. By using the buffer, a user can pre-fetch some parts of the movies to be used in future, from other channels. The study in [4] presents client-buffering techniques for scalable video broadcasting over broadband networks with low user delay. In these techniques the clients download the video data from an appropriate channel and then watch the movie by playback operation.

The studies in [24] and [9] integrate the fixed-delay pagoda broadcasting scheme to reduce client waiting time and buffer demand. A scalable binomial broadcasting scheme has been presented in [26] in which live videos are transferred using constant bandwidth, regardless of video length.

Many studies are proposed to broadcast segments over a single channel, such as PAS [25], and the reverse-order scheduling (ROS) scheme [3]. The basic concept behind these schemes is to partition a video into equal-sized segments, which are classified into several groups and transferred over a single channel according to a predefined arrangement.

Till now, the problems of optimizing the resource bandwidth for the video server have been solved by using the conventional linear programming or non-linear programming approaches depending on the nature of the objective function. But sometimes, finding a solution to these problems becomes a difficult task if the objective function is too intricate and analytical solution is difficult to obtain. Hence, non-traditional methods like Genetic Algorithms (GAs) based on evolutionary programming are coming up to deal with such situations. GAs are computerized search and optimization algorithms based on the mechanics of natural genetics and natural selection. So far, GAs have been applied to many optimization problems in different frameworks. Some applications of GAs in search and optimization can be found in [7] and [17]. Many researchers have used GAs for optimal design of reliable computer communication networks [12], [6]. The authors in [20] have used GA for optimal file placement on the video server in VoD systems. They have employed GA to determine the optimal number of copies of multimedia files and their corresponding disk locations on the video server. The investigation in [1] provides a hybrid GA for frequency assignment problem in radio communication systems.

In this paper, we study three schemes for determining the optimal batching time thereby minimizing the average bandwidth i.e. the number of streams for NVoD systems. A simple GA is used for finding the minimum number of streams. The organization of the paper is as follows. Section 2 describes how GAs can be used to solve an unconstrained optimization problem. The methodology and working principles of GAs are described in detail. Firstly, the basic terms involved in the GAs are explained, then the formation of fitness function and the coding technique for the variables are discussed. The genetic operators i.e, reproduction, crossover and mutation are also conferred which are used for creating new population. In section 3, three multicasting schemes for minimizing the bandwidth requirement in NVoD systems are presented. Scheme 1 (S1) is based on the double rate batching policy in which the unicast customers are first served with double transmission rate and then merged with the multicast stream after getting synchronized with the same. Scheme 2 (S2) is a client buffering technique in which a late arriving customer is served via unicast stream while buffering some part of the movie, simultaneously. Scheme 3 (S3) is also a client-buffering technique in which, the server streams are grouped together into channels of increasing bandwidth. The beginning portion of the movies is transmitted via these channels so that the customers can be merged with an on-going broadcast stream quickly. Mathematical models for the number of streams required in each of the schemes are also presented. In section 4, the GA approach for determining the optimal number of streams for all the schemes is presented. Section 5 compares the analytical as well as GA results for all the schemes by using numerical illustrations. The bandwidth requirements for all the schemes are compared in terms of the number of streams required. S2 and S3 are compared on the basis of the buffer requirement also. Finally, the conclusion is drawn in the last section 6.

2. Genetic Algorithms: Preliminary Concepts

2.1 Basic Terminology

A genetic algorithm is a non-traditional optimization method, in which a string of numbers is manipulated in a manner similar to how chromosomes are altered in biological evolution. Each string of numbers is called a ‘*chromosome’* or an ‘*individual’*, and each number is referred to as a "*gene*." A set of chromosomes forms a ‘population’. A chromosome actually represents a variable, which is varied to optimize the ‘fitness function’. The fitness function corresponds to the objective function, which is to be optimized. There are three main genetic operators, ‘*reproduction’*, ‘*crossover’* and ‘*mutation’*, which are operated on the population to create a new population of points. The operation of GAs begins with an initial population of random chromosomes, which are encoded in some string structures. Each chromosome is then evaluated in terms of the fitness value. Then the genetic operators are operated on the population and a new population of chromosomes is formed. The new population is again evaluated and tested for termination. If the termination condition is not fulfilled, the population is iteratively operated by the genetic operators and evaluated subsequently. This procedure is continued until the termination criterion is met. One complete cycle of these operations and the evaluation procedure is known as a ‘*generation’*. More detailed description on GAs can be found in [13] and [2].

2.2 Coding of Chromosomes

The variables involved in the objective function are first coded in some string structures. Though coding of the variables is not absolutely necessary, it is good to follow a coding procedure. Generally, binary coding or base-2 representation is used for this purpose, in which, 1’s and 0’s are used for representing the strings. First, the variables are converted from base-10 representation in the real-world to a base-2 representation. To get the actual results, the base-2 strings are converted back into base-10.

2.3 Fitness Function

GAs follow the survival-of-the-fittest principle of nature to make a search process. For a particular optimization problem, the fitness function, F(x) is derived from the objective function f(x), and is used in successive genetic operations to evaluate the fitness value. For maximization problems, the fitness function can be considered to be the same as the objective function i.e. F(x)=f(x). For minimization problems, the fitness function can be obtained after converting the maximization problem into minimization problem by using a suitable transformation.

2.4 Genetic Operators

Below, we describe three main genetic operators, which are used for modifying the population in a GA.

(i) Reproduction

This is the first operator, which is applied on a population. It is also known as *selection* operator since it selects the good strings or chromosomes in a population and forms a mating pool by inserting multiple copies of them. By applying this operator, the above average strings are selected and bad strings are eliminated from the population. The two most commonly used reproduction operators are *Roulette wheel mechanism* and *Tournament selection mechanism.*

(ii) Crossover

This operator creates new strings by exchanging information among the strings of the mating pool. Two strings called parent strings are picked from the mating pool with a particular crossover probability and some portion of the strings are exchanged between them thereby forming new strings known as children strings. Generally, three crossover operators are used, *single-point crossover, two-point crossover and uniform crossover.*

(iii) Mutation

Like the cross over operator, mutation operator also searches the new strings in the population. It changes 1 to 0 and 0 to 1 in a string with a particular mutation probability. It provides a local search around the current solution.

Now, we present the working of a simple genetic algorithm in brief as follows:

Genetic Algorithm

1. Generate initial population randomly.
2. Code the chromosomes into base-2.
3. Evaluate the fitness value of all the chromosomes in the population by using the fitness function.
4. Repeat Steps 4.1 to 4.3 until the termination condition is met
   - 4.1 Reproduction
   - 4.2 Crossover
   - 4.3 Mutation
5. Decode the final chromosomes.
6. Stop.

3. MULTICASTING SCHEMES FOR NVOD SYSTEMS

In this section, we describe the three multicasting schemes for minimizing the average bandwidth or average number of streams in a NVoD system. All the schemes are explained one by one alongwith the corresponding mathematical models later in this section. The following notations are used for the mathematical modelling of the schemes:

N_i Number of streams required for ith scheme (i=1,2,3)
D_{max} Maximum start-up delay for the customers
B Buffer requirement for the users
λ Mean arrival rate of the customers according to Poisson distribution
L Length of the movie
T_b Batching time (in minutes)
C Bandwidth for a transmission line or stream (bits/minute)
β Average number of customers arriving within batching time T_b
B_r Bandwidth requirement (in minutes) for one multicast group for the whole movie

Now we describe the three schemes along with the corresponding mathematical models as follows:

3.1 Scheme 1: Multicasting without user buffer (S1)

This is a multicasting scheme, in which, total movie length is divided into an interval of T_b minutes and a multicast stream is opened at each interval. For providing good quality of service, the customers arriving after the beginning of the multicast stream are served via unicast streams with double transmission rate. When the unicast stream gets synchronized with the multicast stream, the unicast stream is released and the customers are merged in the multicast stream. The mean number of customers arriving within the batching time T_b can be approximated as $\beta = \max(\lfloor \lambda . T_b \rfloor, 1)$, where $\lfloor * \rfloor$ denotes the greatest integer less than *. Hence, for one multicast group, the bandwidth demand for the whole movie can be obtained as

$$B_r = x(2C) + (2x)(2C) + \ldots + (\beta x)(2C) + (L - 2x)(C)$$
$$= Cx(\beta)(\beta+1) + (L-2x)(C) \quad \ldots(1)$$

The first β customers are the unicast customers who require 2C bandwidth in order to double the transmission rate. It should be noted that in the last term, C is multiplied by $L - 2x$ because the multicast stream is started when the first customer joins it. Assuming Poisson process, the distribution f(x) of inter-arrival time is given by $f(x) = \lambda e^{-\lambda}$. Hence the average number of streams for S1, can be computed as follows [18]:

$$N_1 = \frac{\int_0^{T_b - 1} B_r . L . f(x) dx}{C . T_b}$$
$$= \frac{\beta(\beta+1)}{T_b}\left[\left(1 - \frac{1}{\lambda} - T_b\right) e^{-\lambda(T_b - 1)} + \frac{1}{\lambda}\right] + \frac{L}{T_b}\left[1 - e^{-\lambda(T_b-1)}\right] - \frac{1}{T_b}\left[\left(1 - \frac{1}{\lambda} - T_b\right) e^{-\lambda(T_b-1)} + \frac{1}{\lambda}\right] \quad \ldots(2)$$

By differentiating the above equation w.r.t. T_b and equating it to zero, we can obtain the optimal batching time T_b^* and optimal number of streams N_1^* for S1.

3.2 Scheme 2: Multicasting with user buffer (S2)

Similar to S1, in this scheme also, a movie is broadcasted at a regular interval of T_b minutes. But this scheme provides buffering also at the customer's end. If a customer sends the request of a movie in less than D_{max} minutes before the start of a multicast stream, he waits till the start of a multicast stream. On the other hand, if he requests the movie after D_{max} minutes, then he is served via a unicast stream with C bandwidth and simultaneously buffers the movie from the ongoing closest multicast stream. As soon as the customers come in a position to start retrieving the movie from his own buffer, the unicast stream is released and the customer watches the rest of the movie form his buffer. If x is the arrival time of a customer's request, then the average number of streams for S2 is given by [4]

$$N_2 = \lambda \int_0^{T_b - D_{max}} \frac{x}{T_b} dx + \frac{L}{T_b}$$

$$= \frac{\lambda (T_b - D_{max})^2}{2T_b} + \frac{L}{T_b} \qquad \ldots(3)$$

By differentiating Eq. (3) w.r.t T_b and equating it to zero, we get the optimal batching and minimum number of streams as

$$T_b^* = \sqrt{D_{max}^2 + \frac{2L}{\lambda}} \qquad \ldots(4)$$

$$\text{and } N_2^* = \lambda \left(\sqrt{D_{max}^2 + \frac{2L}{\lambda}} - D_{max} \right) \qquad \ldots(5)$$

The optimal buffer requirement for S2 is given by

$$B^* = T_b^* - D_{max} \qquad \ldots(6)$$

3.3 Scheme 3: Stream-bundling multicasting with user buffer (S3)

In this scheme, the server streams are bundled together into multicast channels of increasing bandwidth with an increment of C bits/min for serving the customers more quickly. The total transmission time of a movie is divided into the slots of T_b minutes and each slot is further subdivided into mini-slots of D_{max} minutes. If a customer arrives after D_{max} minutes, he is served via a bundled channel of C bandwidth. If the customer arrives after 2*Dmax minutes, then he is served via a bundled channel of 2C bandwidth and so on. Hence the number of bundled channels for the movie is (T_b / Dmax)-1. While getting served by these high-speed channels, the customer concurrently buffers the movie from the ongoing multicast stream, which was started at the beginning of the slot.

For this scheme, the average number of streams is given by

$$N_3 = \sum_{i=1}^{T_b / D_{max} - 1} \frac{iD_{max}}{T_b} + \frac{L}{T_b}$$

$$= \frac{T_b}{2D_{max}} - \frac{1}{2} + \frac{L}{T_b} \qquad \ldots(7)$$

The optimum values of T_b and N_3 are obtained as

$$T_b^* = \sqrt{2LD_{max}} \qquad \ldots(8)$$

$$\text{and } N_3^* = \sqrt{\frac{2L}{D_{max}}} - \frac{1}{2} \qquad \ldots(9)$$

The optimal buffer requirement for S3 is given by

$$B^* = T_b^* - D_{max} \qquad \ldots(10)$$

4. Optimal Number Of Streams Using GA

In this section, we discuss the GA approach for finding the optimal number of streams for the schemes discussed in previous section. Since the objective is to minimize the average number of streams, the fitness function for the GA is taken as

$$F(x) = 1/(1+f(x)) \qquad \ldots(11)$$

where $f(x) = N_i$, i=1,2,3 corresponding to S1, S2 and S3 respectively.

The variable T_b is taken as a chromosome whose best-fit value is to be found using the GA. The population of 6 chromosomes is used for searching the optimal solution. The chromosomes are converted into binary strings using binary coding. The length of each of the strings is taken as 32 bits. The following linear mapping is used for transforming the variables into binary strings and vice-versa:

$$c_i = c_i^L + \frac{c_i^U - c_i^L}{2^{32} - 1} s_i \qquad \ldots(12)$$

where we denote

c_i : ith chromosome (i=1,2,…,6)

c_i^U : upper bound for the ith chromosome

c_i^L : lower bound for the ith chromosome

s_i : coded string for of the ith chromosome

The value of each chromosome lies between the upper and lower bounds, i.e.

$c_i^L \le c_i \le c_i^U$ for i=1,2,…,6

The Roulette wheel mechanism is used for the reproduction operator, where a string is selected with a probability proportional to its fitness. The probability of selecting the ith string is taken as

$$p_i = \frac{F_i}{\sum_{j=1}^{6} F_j}, \text{ i=1,2,..,6} \qquad \ldots(13)$$

The single point crossover is used for exchanging the parent strings from the mating pool. The parent strings are chosen with a crossover probability p_c and the crossover points are selected at random. Random mutation is done on the population and each individual (string) in the population is mutated with a mutation probability p_m. The GA is run for a maximum of 50 generations.

5. NUMERICAL EXPERIMENTS

In this section, we present numerical illustration for verifying the GA results with the analytical results for the three schemes mentioned in section 3. The GA is coded in MATLAB and is run on Pentium II. For scheme 1, the minimum number of streams is determined numerically by using the MATLAB function 'fmin'. For illustration purpose, we assume D_{max} = 2 minutes and L=100 minutes. For all the schemes, the optimal batching time and minimum number of streams are obtained using the GA with parameters as provided in table 1.

GA parameter	Value
Maximum Generations	50
Population Size	6
No. of bits for encoding the chromosome	30
Mutation probability (p_m)	0.03
Crossover probability (p_c)	1

Table 1: GA parameters for numerical illustration

Fig. 1 shows the variation of N with respect to T_b for all the three schemes, by taking λ=18/min. The figure shows that N first decreases and then increases with T_b for the schemes S1 and S2 whereas in S3, it tends to constant after having decreased in the beginning. Hence, minimum of N exists for all the schemes for some particular values of T_b. Fig.2, represents the optimal number of streams for all schemes by varying λ. It is noted that for very small values of λ, S2 requires the lesser number of streams while S3 requires the larger number of streams as compared to S1. For intermediate values of λ, S1 gives the worst results in terms of the required number of streams and S2 gives the best results. Also, for very large arrival rates, S3 is the best scheme, as it requires the least number of streams. We also note that the number of streams in S1 and S2 increase with λ and for S3, it remains constant, which is quite obvious.

Fig. 3 depicts the variation of N* with D_{max} by taking λ=0.2/min. Clearly, S2 is the best scheme in terms of bandwidth requirement for low D_{max} , and S3 is the worst scheme. For intermediate values of D_{max}, S1 requires largest number of streams and if the start-up delay is very large, then S3 requires the m inimum number of streams. In fig. 4, the optimal buffer requirement (B*) for all the schemes is shown by varying λ. For higher arrival rates, S2 requires lesser buffer as compared to S3 whereas for lower arrival rates, S3 requires lesser buffer that S2. Further, as the arrival rate increases, the optimal buffer requirement for S2 decreases, while it remains constant for S3.

Figures 5(a)-7(a) illustrate the minimum and maximum values and mean values along with the standard deviation of the fitness functions in each generation of the GA for S1, S2 and S3 respectively. In figures 5(b)-7(b) the mean values along with the standard deviation of the fitness functions in each generation of the GA are shown for S1, S2 and S3 respectively.

Table 2 provides the analytical as well as GA values of optimal batching time for different values of D_{max} by taking λ=10. It is observed that for S1 and S3, the batching time increases with D_{max}, which implies that low start-up delay requires lesser batching time and high start-up delay requires larger batching time. Also, the batching time for S2 is independent of the start-up delay. Further, for higher arrival rates (i.e for very popular movies), the batching time is very small. This indicates that multicast streams should be opened very frequently if the arrival rate is very high. Table 3 shows the optimal batching time for all the schemes by taking several values of λ.

In tables 4 and 5, the values of the chromosomes (strings) obtained by the GA are shown for first and last generations. Table 4 gives the values of the chromosomes for N1 and N2 for different values of λ. In table 5, the chromosome values for different values of D_{max} are provided for N2 and N3.

Overall, we conclude that for higher arrival rates, the client buffering schemes (S2 and S3) perform better as far as the bandwidth requirement is concerned. Also, the GA results are quite closer to the analytical results. Also, in S1, where the analytical results are difficult to obtain, the GA results provide easy solution, which is at par with the numerical results.

Dmax	Tb*					
	S1		S2		S3	
	Numerical	GA	Analytical	GA	Analytical	GA
1	3.1607	3.1615	4.5826	4.3996	14.1421	13.9889
2	3.1607	3.1615	4.899	4.7822	20	21.2774
3	3.1607	3.1615	5.3852	5.3625	24.4949	23.2774
4	3.1607	3.1615	6	5.4271	28.2843	27.4127
5	3.1607	3.1615	6.7082	6.6976	31.6228	31.3864
6	3.1607	3.1615	7.4833	7.5872	34.641	35.2361
7	3.1607	3.1615	8.3066	8.4491	37.4166	38.2899
8	3.1607	3.1615	9.1652	9.7223	40	39.784
9	3.1607	3.1615	10.0499	10.2781	42.4264	43.2215

Table 2: Optimal batching time (Tb*) for various values of Dmax by taking l=10

λ	Tb*					
	S1		S2		S3	
	Numerical	GA	Analytical	GA	Analytical	GA
0.1	19.5449	18.2464	44.7661	45.1411	20.000	19.1749
0.2	16.8287	16.8513	31.686	32.9849	20.000	19.1749
0.3	15.5715	16.5248	25.8972	24.5172	20.000	19.1749
0.4	14.4181	14.0189	22.4499	21.8687	20.000	19.1749
0.5	13.3531	10.742	20.0998	21.7643	20.000	19.1749
5	4.468	4.403	6.6332	6.627	20.000	19.1749
10	3.161	3.162	4.899	4.843	20.000	19.1749
20	2.141	2.230	3.7417	3.735	20.000	19.1749
30	1.825	1.804	3.266	3.253	20.000	19.1749
40	1.581	1.079	3	3.052	20.000	19.1749

Table 3: Optimal batching time (Tb*) for various values of l by taking Dmax=2		λ=.1		λ=.2		λ=.3		λ=.5		λ=1		λ=5		λ=10		λ=20		λ=30	
	Pop	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen
N1	1	28.2883	30.3329	26.1112	16.9701	33.9497	21.6342	42.1584	10.742	15.5801	4.0493	0.8724	5.3161	1.9488	3.4736	8.0943	2.6903	1.8963	1.852
	2	42.8321	43.244	40.8408	16.9907	46.4064	24.2556	29.375	23.1184	19.744	16.412	5.6458	5.4809	7.6939	4.8591	4.9366	3.9403	9.1914	1.5395
	3	21.9108	42.4423	13.8514	13.8657	47.2251	24.7684	23.1184	23.1199	37.3833	4.0493	4.2283	5.4736	7.5099	4.0974	9.264	3.9403	9.9862	0.6222
	4	17.5406	39.6583	23.2951	13.0704	35.1604	18.0178	6.8234	23.1184	14.7714	16.4116	0.8463	6.5722	5.8478	3.448	6.4961	2.6903	7.2157	9.352
	5	25.0285	18.2464	20.7375	13.8658	16.5248	12.2561	17.657	24.6808	23.8375	16.5434	4.5131	6.5722	0.5821	3.7617	9.5303	2.6879	6.454	1.852
	6	15.0685	40.5128	35.1052	16.9766	24.7561	24.7592	37.2185	26.2509	40.8167	16.5426	5.6543	1.5649	4.7029	8.476	9.1531	2.6903	7.865	1.852
N2	1	31.9981	26.2823	38.7575	33.528	50.9453	24.2611	57.1147	18.2321	46.6713	18.724	0.9905	5.3978	2.5273	9.3459	1.8511	5.0077	2.7199	2.8731
	2	62.9143	65.1114	2.8529	29.721	32.1274	21.9879	43.8668	21.0843	14.8622	20.7764	4.1944	7.8612	4.8364	0.4263	2.8683	5.0923	0.7748	2.8778
	3	37.7157	69.7589	62.4936	32.9849	57.2619	21.9882	3.5843	17.6852	52.1859	2.6921	4.2655	5.4363	7.3446	8.0959	5.3043	5.394	7.6833	2.8774
	4	30.5369	61.5558	4.7059	32.9849	23.605	21.9197	53.2266	56.9997	41.8841	27.6112	1.7483	5.3978	4.3366	9.9711	7.0189	5.0142	9.979	2.8781
	5	17.533	45.1411	33.0023	37.7278	55.1018	21.9198	61.6146	66.4232	66.7108	1.2245	6.5202	5.3978	5.3077	5.5961	5.0137	5.1509	8.4148	9.995
	6	28.9231	65.3839	30.3109	33.0023	11.0108	21.9197	21.1455	21.0838	19.7885	20.1836	6.6015	7.8612	3.7133	9.9721	1.8097	6.3424	4.3694	2.8775

Table 4: First and last generation values of the chromosomes of N1 and N2 for various values of λ

		Dmax=1		Dmax=2		Dmax=3		Dmax=4		Dmax=5		Dmax=6		Dmax=7		Dmax=8		Dmax=9	
	Pop	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen	fgen	lgen
N2	1	3.0028	4.0813	1.7288	5.4788	5.6362	5.689	5.6173	9.7619	2.0158	6.1481	7.0386	7.0386	1.4296	7.1497	1.8177	9.1149	8.2321	4.7832
	2	0.255	4.3273	6.8553	5.4791	9.8475	5.3222	7.3749	1.9339	3.4693	9.8883	0.7994	7.0483	6.9944	6.017	4.2121	9.1046	0.0454	9.4851
	3	3.3963	4.7048	4.1144	5.4849	6.7947	5.6737	4.3475	4.7625	4.7433	7.3883	7.0432	7.0603	6.5135	9.6498	4.1864	7.8563	9.4854	9.4854
	4	5.5307	4.4921	5.4844	5.4986	2.5485	7.0171	3.3017	3.5845	1.4949	8.57	9.8266	7.0483	5.2443	7.1566	5.7652	9.1144	3.7429	9.4854
	5	4.4103	4.0966	1.3876	5.4788	1.8763	7.8221	6.7399	4.7619	6.4095	6.1481	3.3443	7.0383	6.0403	1.0169	7.5111	7.8644	4.1093	7.315
	6	9.4849	4.1711	8.2412	5.5178	4.2791	6.9214	8.6432	1.9339	0.1561	8.4918	1.0887	2.0373	9.8322	7.1498	8.9963	7.862	0.156	9.4854
N3	1	34.6148	15.2195	21.0646	13.6727	27.2578	35.0153	38.4621	31.9366	16.5063	26.5943	21.7616	35.4417	32.837	17.849	10.7702	13.9691	34.8274	23.9784
	2	33.9004	15.2194	20.5239	38.9877	43.9203	40.5023	5.2719	37.394	12.3283	27.9153	35.2546	35.4413	11.5209	34.0209	16.5707	16.7478	9.4857	24.8573
	3	12.5831	22.4155	21.7645	20.7065	22.2548	29.3868	33.7157	26.2999	27.773	27.8385	27.536	35.4413	43.128	12.9272	1.6338	16.7705	5.9766	23.4504
	4	16.3781	13.9892	24.1791	38.9887	30.3022	30.6647	38.0969	40.5417	20.677	28.1103	0.2137	35.9577	25.7715	34.724	16.5659	16.7706	6.7834	27.1425
	5	34.4649	15.3966	37.6611	13.6985	19.2328	17.4591	4.0765	37.383	26.5198	37.7587	5.8462	36.1668	40.1493	12.2227	15.197	16.7486	35.3168	22.7483
	6	29.7076	36.4892	7.9271	13.6766	39.6782	28.9036	40.198	31.9044	34.4442	28.0018	10.8567	35.2532	26.8191	17.6732	1.1429	16.7705	27.529	24.6815

Table 5: First and last generation values of the chromosomes of N2 and N3 for various values of D_{max}

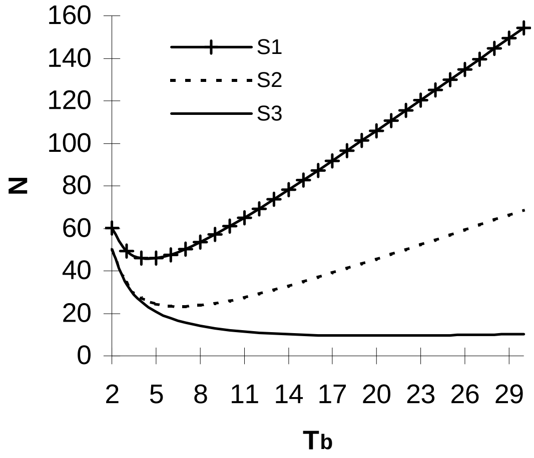

Fig. 1: Number of streams (N) by varying Tb

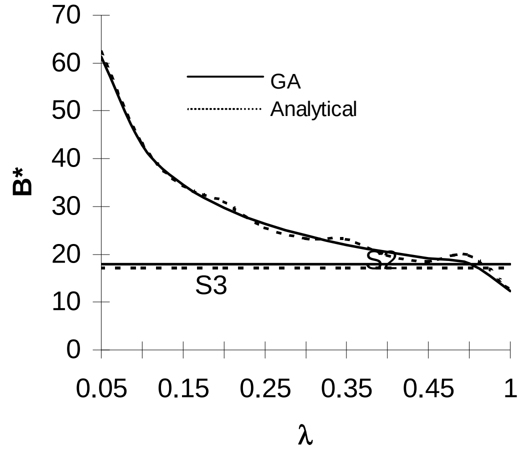

Fig.4: Optimal buffer requirement B*by varying λ

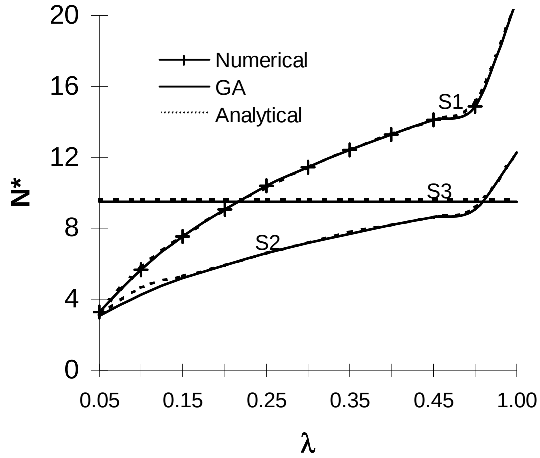

Fig. 2: Optimal number of streams N*by varying λ

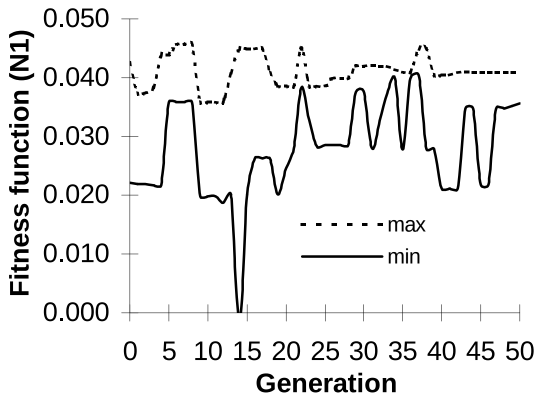

Fig. 5(a): Maximum and Minimum values of fitness function for N1

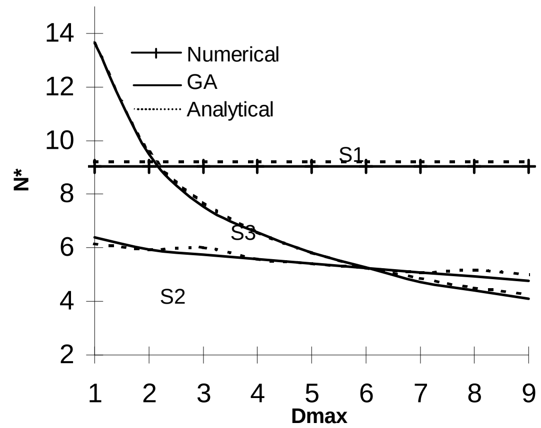

Fig. 3: Optimal number of streams N* by varying Dmax

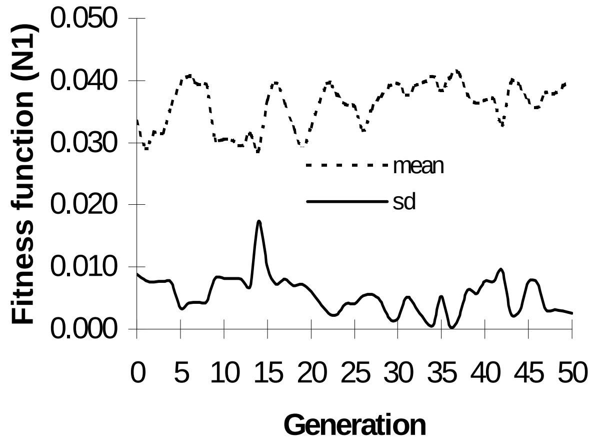

Fig. 5(b): Mean and Std. Deviation of fitness function for N1

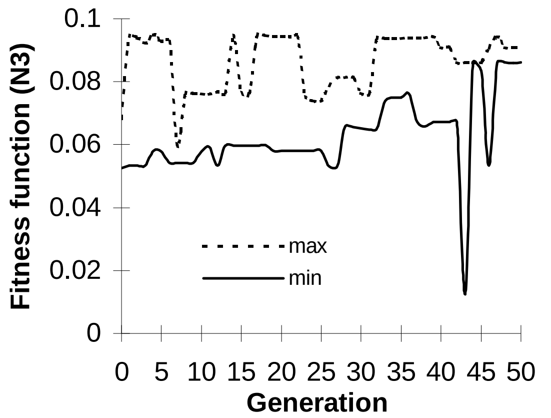

Fig. 6(a): Maximum and Minimum values of fitness function for N2

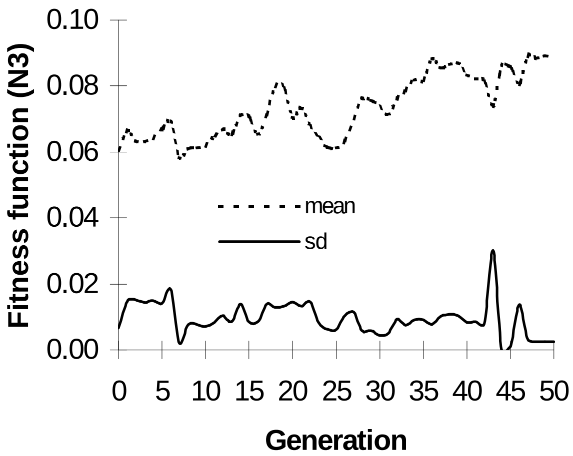

Fig. 6(b): Mean and Std. Deviation of fitness function for N2

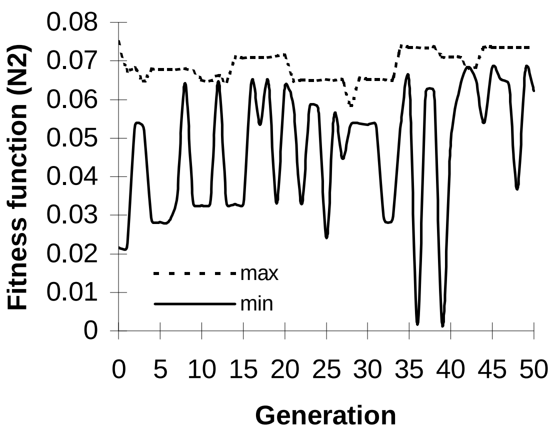

Fig. 7(a): Maximum and Minimum values of fitness function for N3

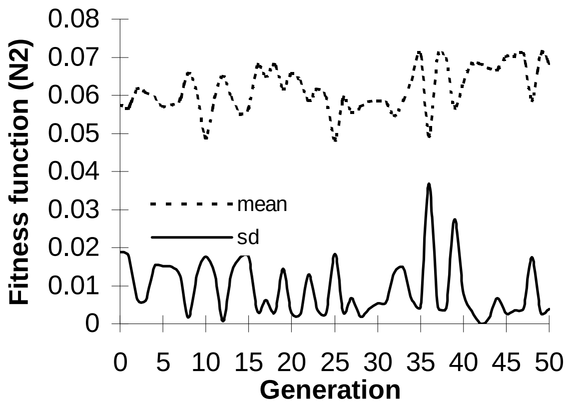

Fig. 7(b): Mean and Std. Deviation of fitness function for N3

6. Conclusion

The problem of minimizing the bandwidth or average number of streams required in a NVoD system is studied. Three multicasting schemes are considered and the optimal batching time is determined using a simple Genetic Algorithm (GA) thereby minimizing the average number of streams. All the schemes are compared on the basis of the bandwidth requirement. Scheme 1 is a double rate batching scheme which requires largest number of streams. Scheme 2 is a client buffering scheme which offers buffering of movie to the users. This scheme performs better than scheme 1, as it requires less number of streams. Scheme 3, which is also a client buffering technique, provides channel bundling or grouping such that the streams are grouped together into channels of incrementally increasing bandwidth. These high bandwidth channels are used to deliver the starting portion of the movie while the users concurrently buffer the later part of the movie from the ongoing multicasting stream. We conclude that the client-buffering techniques substantially reduce the average number of streams, for higher arrival rates or for popular movies. However in case of low arrival rates, scheme 1 is better than scheme 3. The genetic algorithm suggested, provides very accurate results for the optimal batching time for all the schemes. In scheme 1, where analytical results are difficult to obtain, GA facilitates an easy solution technique of minimizing the average number of streams.

References

1. Alabau, M., Idoumghar, L. and Schott, R., “New hybrid genetic algorithms for the frequency assignment problem”, IEEE Trans. Broadcasting, 48 (1), pp. 27-34, March, 2002.
2. Aminzadeh, F., “Soft computing”, PTR Prentice Hall, New Jersey, 1994.
3. B. S. Wu, C. C. Hsieh, and Y. W. Chen, “A reverse-order scheduling scheme for broadcasting continuous multimedia data over a single channel,” IEEE Transactions on Broadcasting, 57 (3), pp. 721–728, 2011.
4. Chan. G.S.H. and Yeung, I.S.H., “Client buffering techniques for scalable video broadcasting over broadband networks with low user delay”, IEEE Trans. Broadcasting, 48 (1), pp. 19-26, March, 2002.
5. Dan, A., Sitaram, D. and Shahabuddin, P., “Dynamic batching policies for an on demand video server”, ACM/Springer Multimedia Syst., 4, pp. 112-121, June 1996.

6. Dengiz, B., Altiparmak, F. and Smith, A.E., “Local search genetic algorithm for optimal design of reliable networks”, IEEE Trans. Evolutionary Computation, 1(3), pp. 179-188, Sept., 1997.
7. Goldberg, D.E. and Holland, J.H., “Genetic algorithms in search, optimization and machine learning”, *Addison Wesley*, 1989.
8. Hua, K.A. and Sheu, S., “Skyscraper broadcasting: A new broadcasting scheme for metropolitan video-on-demand systems”, ACM Computer Commun. Rev., 27, pp. 89-100, Oct. 1997.
9. J. F. Paris, “A fixed-delay broadcasting protocol for video-on demand,” Proceedings of the 10th International Conference on Computer Communications and Networks (ICCCN ’01), pp. 418–423, Scottsdale, Ariz, USA, October 2001.
10. Juhn, L.S. and Tseng, L.M, “Harmonic broadcasting for video-on-demand service”, IEEE Trans. Broadcasting, 43, pp. 268-271, Sept., 1997.
11. Kim, H.J. and Zhu, Y., “Channel allocation problem in VOD system using batching adaptive piggybacking”, IEEE Trans. Consumer Electronics, 44 (3), pp. 969-976, 1998.
12. Kumar, A., Pathak, R.M. and Gupta, Y.P., “Genetic algorithm based reliability optimization for computer network expansion”, IEEE Trans. Reliability, 44 (1), pp. 63-72, 1995.
13. Michalewicz, Z., “Genetic algorithms + data structures = evolutionary programs”, Springer Verlag, 1992.
14. Petit, G.H., Deloddere, D. and Verniest, W., “Bandwidth resource optimization in video-on-demand network architectures”, Proc. Ist IEEE International Workshop on Community Networking, New York, NY, pp. 91-97, July 1994.
15. Poon, W.F. and Lo K.T., “New batching policy for providing true video-on-demand (T-VoD) in multicast system”, Proc. IEEE Int. Conf. Communications (ICC’99), 2, pp. 983-987, June 1999.
16. Poon, W.F., Lo, K.T. and Feng, J., “Adaptive batching scheme for multicast video-on-demand systems”, IEEE Trans. Broadcasting, 47 (1), pp. 66-69, March 2001.
17. Reeves, C.R., “Genetic algorithms, modern heuristic techniques for combinatorial problems”, Blackwell Scientific Publications, 1993.
18. Shachnai, H. and Yu, P.S., “Exploring wait tolerance in effective batching for video-on-demand scheduling”, Proc. Eighth Israeli Conf., pp. 67-76, 1997.
19. Tang, H.-S., Chan, S.H.G., Li, H., “Optimizing segment caching for peer-to-peer on-demand streaming”, Proc. IEEE International Conference on Multimedia and Expo, ICME 2009, pp. 810-813, 2009
20. Tang, K.S., Ko K.T., Chan, S. and Wong, E.W.M., “Optimal file placement in VOD system using genetic algorithm”, IEEE Trans. Industrial Electronics, 48(5), pp. 891-897, Oct.2001.
21. Viswanathan, S. and Imielinski, T., “Metropolitan area video-on-demand service using pyramid broadcasting”, Multimedia Syst., 4, pp. 197-208, Aug. 1996.
22. Wong, W.T., Zhang, L. and Pang, K.K., “Video on demand service policies”, Proc. IEEE Singapore Int. Conf. Networks, pp. 560-564, July 1995.
23. Wong, Y. W.,Lee, Jack Y. B, Li, Victor O. K.,Chan, Gary S. H.: Supporting interactive video-on-demand with adaptive multicast streaming, IEEE Transactions on Circuits and Systems for Video Technology, 17(2), pp. 129-142, Feb. 2007.
24. Y. N. Chen and L.M. Tseng, “An efficient periodic broadcasting with small latency and buffer demand for near video on demand,” International Journal of DigitalMultimedia Broadcasting, Article ID 717538, 7 pages, 2012.
25. Y.-W. Chen and C.-Y. Chen, “PAS: a new scheduling scheme for broadcasting a video over a single channel,” IET Communications, 5(7), pp. 951–960, 2011.
26. Z.-Y. Yang, Y.-M. Chen, and L.-M. Tseng, “A seamless broadcasting scheme with live video support,” International Journal of Digital Multimedia Broadcasting, Article ID 373459, 8 pages, 2012.