Competitive Video on Demand Schedulers for Popular Movies[*][†]

Christos Bouras   Vaggelis Kapoulas   Grammati E. Pantziou
Paul G. Spirakis

Computer Technology Institute — CTI
P.O. Box 1122, GR-26110, Patras, Greece
E-mail: {bouras,kapoulas,pantziou,spirakis}@cti.gr

Abstract

In this paper we investigate the online video on demand problem, namely having to accept or reject a request for a movie without knowing the future requests. We present online movie-scheduling schemes that implement the principles of refusal by choice and delayed notification. A novel way to schedule movies that exploits the knowledge of the distribution of the preference of requests for movies, is shown to have a competitive ratio that outperforms all the previously known schemes in practical situations. In fact, our scheduler has a competitive ratio bounded above by a constant, independent of the number of the users, channels, or movies, in the case that a large fraction of the requests tends to concentrate in a small number of movies. We extend our approach by presenting an "adaptive" randomized scheduler which initially is not aware of the movie popularities but it adapts to it, and achieves a similar asymptotic competitive ratio.

Keywords: Video on demand, online scheduling algorithm, competitive ratio, probability distribution.

[*]A previous version of this work was accepted as a short paper in the 1996 ACM PODC.

[†]This research was partially supported by the European ESPRIT Long Term Research Projects GEPPCOM (contract No. 9072) and ALCOM–IT (contract No. 20244) and the Greek Ministry of Education

1 Introduction

Recent advances in computing and communication technology have made feasible video on demand systems. Usually, the movies are stored in a central video server (which may be connected to other servers by a high-bandwidth WAN). The video server is connected via a high-capacity fiber line to local distribution centers (hubs) from which coax cables are used to broadcast to the households. Major bottlenecks in this architecture are the limited number of broadcast channels (shared by many households) and/or the number of movies that the server can transmit concurrently (e.g. see [12]).

Different issues involved in the design of a video on demand system have been studied by different researchers the last few years. Architectural issues have been studied in [10, 15], physical storage organizations necessary for supporting video on demand systems in [5, 6, 13], and probabilistic models for the assignment of video data onto a storage hierarchy in [14]. The first attempt to tackle video on demand from an optimization perspective was done by Aggarwal, Garay and Herzberg in [1]. In that important work, the video on demand problem was studied in an online setting, where an online algorithm receives a sequence of requests for service. The performane of the online algorithm on a sequence of requests is compared to the performance of an optimal offline algorithm that services the *same* sequence of requests. Such an analysis of an online algorithm is referred to as *competitive* analysis. Aggarwal, Garay and Herzberg showed upper and lower bounds on the competitive ratio of online scheduling algorithms for certain scenarios, and also introduced the concept of refusal by choice with delayed notification and presented algorithms that exhibit under certain conditions, an asymptotically optimal behaviour.

Refusal by choice (and, in fact, by *random choice*) was used previously in the problem of admission control in fast networks (see [2, 3, 4, 9]). The admission control problem, first defined by Garay and Gopal in [8], is the problem of deciding online whether or not a network should accommodate a request for a large amount of data. The online video on demand research was complementary to the research for the admission control problem. While the research on admission control was mostly concerned with online allocation of network paths in a way that would minimize overlap with paths of future requests, the adaptive video on demand research focuses mostly on the issue of "revenue" of movie schedulers (over very simple networks). The revenue has to do with grouping requests so that a single transmission may serve many users requesting the same movie (popular movies). Also, the problem of online video on demand is related to the call control problem (see [3, 7]).

None of the previous works took into account the fact that most requests tend to concetrate on (a few) popular movies. In this paper we present a novel movie-scheduling scheme which exploits the knowledge of the underlying distribution of movie requests, and achieves constant competitive ratio in the case that the number of popular movies is small (which is a realistic assumption). Our method follows the principle of refusal by choice with delayed notification. We also present a randomized movie-scheduling algorithm which does not need to know the distribution of the movie requests in advance but it is able to follow slow changes in this distribution, in an adaptive way that has a small transient behaviour. Our scheduler will adapt to such an unknown distribution quickly (statistically learns). This method also achieves constant competitive ratio in the case that the number of popular movies is small. This is due to the assumption on a distribution of the input requests (which restricts the oracles that would create worst-case behaviours in the lower bounds of [1]).

The rest of the paper is organized as follows. In section 2 we present the video on demand architectural model used, and some basic definitions. In section 3 we present the online movie scheduling algorithm S that knows the distribution $p()$ of the movie requests, and in section 4 we analyze its performance against an optimal offline algorithm. In section 5 we extend our approach by presenting an online scheduling algorithm R which is not aware of $p()$ but it adapts to it, and we discuss its performance.

2 The model and definitions

The video on demand model considered here follows [1], as far as the architecture is concerned. It consists of a video server which acts as a database of movies, and supports a fixed number of movie-streams (sessions). The users connect to the server via dedicated links and make movie-requests to it. The communication network used is equipped with a multicast facility. Thus, the same movie-stream can be sent to more than one users without causing any extra overhead to the server, and therefore, multiple users can participate in a single session. Let M be the set of movies stored in the server, U be the set of users making requests, and C be the set of channels in the system, and let m, u, and c be their cardinalities, respectively. The system is as in Figure 1.

The system has three time parameters:

T: The *duration* of a movie (we assume it is the same for all movies).

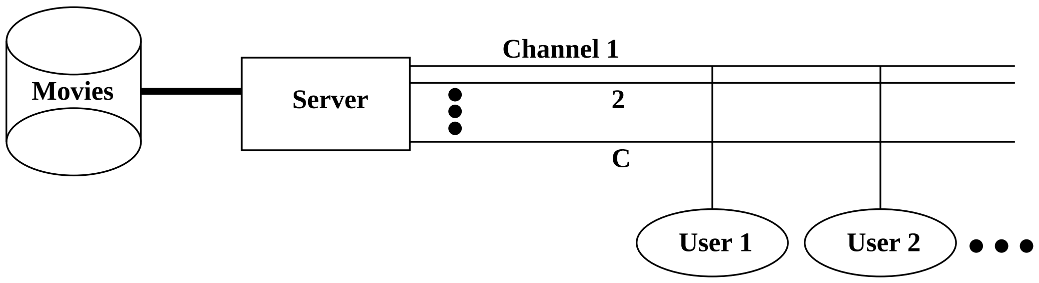

Figure 1: The model

τ: The maximum delay between a request and the start of the movie (if the request is accepted).

ν: The *notification time* (the maximum delay between a request and its response).

Obviously, $\nu \leq \tau \leq T$. In this paper we consider $\tau = \nu$. We assume users are "blocked" when seeing a movie, i.e., they cannot place other requests at that time. Also, we assume that each user can place at most one request in each interval of time T. Requests to the system are triples (q_time, $user$, $movie$) and responses are tuples (r_time, $user$, $movie$, $channel$, $servetime$), where r_time is a positive number in the interval $[q\_time, q\_time + \nu]$, $user \in U$, $movie \in M$, $channel \in C$, and $servetime$ is a real number in the interval $[r\_time, q\_time + \nu]$. By convention, a refusal to serve is a response with negative $servetime$.

A scheduling algorithm determines the responses of the system to the users at any moment, and allocates movies to channels based on the requests to the system up to the moment the scheduling is taking place. If $\nu = 0$ then a request must be responded as soon as is it is presented. In this case the scheduling algorithm performs "immediate notification". Otherwise (i.e., $\nu > 0$), the scheduling algorithm performs "delayed notification" and the response to the user is issued at most ν time units after the presentation of the request. A scheduling algorithm is said to "refuse by choice" if it has a response (t, u, m, c, s) with negative servetime s, while there exists a free channel at the reponse time t.

Let r be a request and e a movie. We assume that the request r will be for movie e with probability $p(e)$, i.e., we assume that requests are "independently" created according to the distribution $p()$ which indicates the movie popularities. Smart schedulers accumulate arriving requests into queues Q_j one for each movie e_j. The idea is to serve for each j, all the requests for movie e_j accumulated into Q_j with a single transmission (over a single channel). For an input distribution D (and a randomized approach) the revenue of a sequence s of requests, for an online scheduler A, is a random variable showing how many requests were served in the run. Let *its expected value* be $x = E(rev(A(s)))$. Let OPT be an off-line algorithm that plays the role of an adversary whose goal is to produce a request sequence that would force the player (the online algorithm A) to perform poorly. Let $y = E(rev(OPT(s)))$ be the expected revenue of OPT. The *competitive ratio* of the scheduler A is then

$$C(A) = \sup_{OPT, s \in D} \frac{y}{x}$$

We of course assume that A, OPT follow the restrictions of T, τ, ν and also that the distribution $p()$ is respected in all sequences.

3 The online movie-scheduling algorithm S

In this section we present an online movie-scheduling shceme S for the video on demand problem. We assume that the distribution $p()$ is known, i.e., the scheduling algorithm S is aware of the movie popularities. Then, S uses this information to divide the set C of channels into subsets $C_1, C_2, \cdots, C_k$, so that channels in partition C_i, $i \in \{1, \cdots, k\}$, will be used for movies to be seen by at least $h(i)$, $i \in \{1, \cdots k\}$, users on average, where h is an increasing function from $\{1, \cdots, k\}$ to $\{1, \cdots, u/c\}$ (the way that the integer k and the function h are determined, is discussed in the sequel of this section). The goal is to allocate less channels to more popular movies, in order to optimize channel revenue and reduce unused channels. The sets C_i change dynamically, i.e., each channel may belong to different classes at different time periods of the execution.

The scheduling algorithm S employs m queues Q_j, $j = 1, \cdots, m$, one for each movie e_j, $j = 1, \cdots, m$. When a request for movie e_j is done, it is inserted into Q_j. Each Q_j has a "start-time" $start_j$ which is the time when the earliest request for that movie arrived. At time $start_j + \nu$ the scheduler decides whether to serve the requests in Q_j. If there is a free channel in a set C_i with $h(i) \leq |Q_j|$, then all movie requests in Q_j are served on that channel by a single transmission. If, however, no such channel is available, then S rejects only those requests in Q_j made at time $start_j$ and resets $start_j$ to the time of the earlier request now in Q_j. When a channel is freed it is chosen to be placed to a set C_i with probability f_i'/F where $F = \sum_{i=1}^{k} f_i'$.

The f_i''s, k and h are estimated as follows: If n users place " overlapping requests" (i.e., requests that can be served concurrently), then

$$P_i^e = Prob\{\text{movie } e \text{ will be selected by } i \text{ users}\} = \binom{n}{i} p^i(e)(1-p(e))^{n-i}$$

where $p(e)$ is the probability of the movie e. Recall that each user can place at most one request. Let

$$Q_i^e = \sum_{j=i}^{n} P_j^e = Prob\{e \text{ will be selected by at least } i \text{ users}\}$$

If $y_i^e = 1$ with probability Q_i^e and 0 else, then, $\sum_e y_i^e$ equals *the number of movies that will be selected by at least i users to be seen "concurrently".*

Clearly, $E\left(\sum_e y_i^e\right) = \sum_e Q_i^e$ by linearity of expectation. Let $f_i = \sum_e Q_i^e$. (We may use $n = u$ for the worst case demands. Actually, one may adaptively use n equal to u minus the number of users that are currently seeing a movie.)

Let $f_1' = f_1$. Let $f_2' = \sum_{i=2}^{j_2} f_i$ for a $j_2 \geq 2$ such that

$$\frac{f_1}{2} \leq f_2' < f_1$$

In general, let $f_l' = \sum_{i=j_{l-1}}^{j_l} f_i$, where j_{l-1} is such that $f_{l-1}' = \sum_{i=j_{l-2}}^{j_{l-1}} f_i$, and j_l is such that

$$\frac{f_1}{l} \leq f_l' < \frac{f_1}{l-1}$$

Finally, let k be such that $f_k' = \sum_{i=j_{k-1}}^{n} f_i$ and

$$\frac{f_1}{k} \leq f_k' < \frac{f_1}{k-1}$$

Let $h(1) = 1$ and

$$h(l) = l\frac{u}{ck} \quad for \quad l = 2, \cdots, k$$

Definition 1 *The distribution $p()$ is called k-skewed if the cummulative statistic f_i is such that there exist f_i', $i = 1, \cdots, k$, as they are defined above.*

Let g_i be the expected number of channels currently in set C_i, $i = 1, \cdots, k$. Note that

$$g_i = \frac{cf_i'}{F}, \qquad i = 1, \cdots, k$$

where $F = \sum_{i=1}^{k} f_i'$. Any transmission made on a channel in set C_i must serve at least one request for $i = 1$, and at least $lu/(ck)$ requests for $l = 2, \cdots, k$.

4 The performance of S

As in [1], *the saturation level at instant* t is the highest i such that all channels in sets $C_1, C_2, \ldots, C_i$ are occupied at time t. The saturation level of an interval of time is the highest saturation level achieved during the interval.

We divide executions into intervals I_0, I_1, ...of time T each, i.e., $I_j = [jT, (j+1)T)$.

For all j, let $A(j)$ be the number of requests accepted by S at interval I_j in response to a request sequence, and $R(j)$ the number of requests rejected by S but accepted by the offline algorithm OPT in its execution in response to the same request sequence. Let σ_j the saturation level of interval I_j.

Thus, there is some $t \in I_j$ such that all channels in sets C_1, C_2, $\ldots C_{\sigma_j}$ are occupied. Since such requests must have been scheduled to run no earlier than $t - T$ we have $\forall j$,

$$A(j-1) + A(j) \geq h(1)g_1 + h(2)g_2 + \cdots h(\sigma_j)g_{\sigma_j} = \frac{c(h(1)f'_1 + h(2)f'_2 + \cdots + h(\sigma_j)f'_{\sigma_j})}{F} \geq$$

$$\geq \frac{c + u/k(\sigma_j - 1)f_1}{f_1(1 + H_{k-1})} \geq \frac{c + u/k(\sigma_j - 1)}{1 + H_{k-1}} \tag{1}$$

where $H_{k-1} = 1 + \frac{1}{2} + \cdots + \frac{1}{k-1}$. (Note that $F = \sum_{i=1}^{k} f'_i < f_1 + f_1 + f_1/2 \cdots + f_1/(k-1) = f_1(1 + H_{k-1})$.)

Now, all requests in $R(j)$ were rejected by S during I_j, so each such request was made in the interval $[jT - \nu, (j+1)T - \nu)$. The offline algorithm OPT could thus serve these requests anytime in $[jT - \nu, (j+1)T - \nu)$. Since each channel is freed after T time units, OPT can utilize *each* channel twice in this interval.

To bound the number of requests (for the same movie) that such a transmission by OPT could serve we distinguish cases depending on the value of σ_j.

Suppose first that $\sigma_j < k$. Then, any offline transmission serving more than $h(\sigma_j)$ requests would also be served by S. Therefore, $\forall j\ R(j) \leq 2h(\sigma_j)c = 2u/k\sigma_j$.

Let $\overline{A}(j)$, $\overline{R}(j)$ be the expected values of $A(j)$ and $R(j)$, respectively. Let also $\overline{A} = \sum \overline{A}(j)$ be the expected total number of requests accepted by S, and $\overline{R} = \sum \overline{R}(j)$ the expected total number of requests rejected by S but accepted by the offline algorithm OPT.

Then

$$C(S_1) \leq \frac{\overline{A} + \overline{R}}{\overline{A}} = 1 + \frac{\overline{R}}{\overline{A}} \tag{2}$$

i.e.,

$$C(S_1) \leq 1 + \frac{2u/kE(\sigma_j)}{u/kE(\sigma_j - 1)/(1 + H_{k-1})} = 1 + 2(1 + H_{k-1}) = 3 + 2H_{k-1} \tag{3}$$

Consider now the case where $\sigma_j = k$. Note that $R(j) \leq 2h(k)c = 2u$ while

$$A(j) \geq \frac{c+u}{1 + H_{k-1}}$$

yielding the same result. Therefore, we have the following theorem.

Theorem 1 *If $p()$ is k–skewed then the competitive ratio of the shceduler S is asymptotically bounded above by $3 + 2H_{k-1}$, where H_{k-1} is the $(k-1)$ harmonic number (i.e. $H_{k-1} = 1 + \frac{1}{2} + \ldots + \frac{1}{k-1}$).*

Note that k can be independent of u, c, m. The actual value of k depends on the way that the sequence f_1, $f_2, \cdots$ decreases. If the sum $\sum_j f_j$ converges then the value of k is constant. For example, if the sequence $f_1, f_2, \cdots$ decreases as the sequence $1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \cdots$, then $k = 4$.

Note that our assumption that k is a small number is a realistic one given that the number of popular movies is small. Even if the distribution $p()$ is not good enough so that the cummulative statistic f_i to be such that f_i' can be defined, the sequence f_i is still rapidly decreasing. In this case, we may redefine the sequence f_i' so that that it adapts to the way f_i is decreasing. This means that we might not be able to use the "harmonic" way that scheduler S employs to group consecutive elements of the sequence f_i into an element of f_i' but still we may group them in a non-canonical way so that the total number of elements in f_i' (i.e., the value of k) is small. Therefore, even in this case we have an online movie-scheduling algorithm that achieves competitive ratio against any offline algorithm similar to that of S.

5 Work in progress

In this section we present an *adaptive* online movie-scheduling scheme R which is not aware of the movie popularities, i.e., the distribution $p()$ in not known. R uses an intitial partition of the channels in C into classes $C_1, C_2, \cdots, C_\lambda$, and a mechanism to dynamically reallocate channels to the sets C_i, and *adjusts* to the initially unknown distribution $p()$ and therefore, achieves an asymptotic competitive ratio against any offline algorithm similar to the competitive ratio of S.

The adaptive scheduler R partitions the channels into λ classes $C_1, \ldots, C_\lambda$ where $\lambda = \lceil \frac{u}{c} \rceil$, with $|C_i| = \lceil \frac{c}{iH_i} \rceil$ (H_i is the i^{th} Harmonic number), i.e., the initial allocation is the full

Harmonic allocation. R attempts to serve requests as in S, with the following difference: At time $start_j$ of the queue Q_j the scheduler looks for a free channel in a set C_k, $h(i) \leq |Q_j|$ (initially, $h(i) = i$). If such a channel does not exist it rejects the head of the queue. If such a channel exists, then it serves the queue *with probability* $\frac{1}{2}$. Else the (candicate) head of Q_j is rejected and *the free channel* is *reallocated* to a set C_i with probability f'_i/F, where $F = \sum_i f'_i$. The f'_i's are estimated as follows. Let f_i be the number of queues with at least i requests, for $i = 1, \cdots, n$. We use the sequence f_i to define the sequence f'_i and the function h in a way similar to that of S. Also, when a channel is freed, it is *reallocated* according to the estimates of $\frac{f'_i}{F}$. Assume that the sequences of movie request over time are ergodic i.e., time average are close to ensemble averages. Also, assume that the number of requests generated in a period of ν time units (request generation rate) is high (at least $\log m$). Then, by assuming a long enough run, the estimator is very close to the actual statistic, i.e., R *adjusts* to the (unknown) $p()$ distribution by its dynamic reallocation mechanism. So, we claim the following result.

Result 1 *If $p()$ is k–skewed and the movie request sequence is ergodic then the* asymptotic *competitive ratio of the adaptive scheduler R is asymptotically bounded above by $3 + 2H_{k-1}$, where H_{k-1} is the $(k-1)$ harmonic number.*

Acknowledgments

The authors wish to thank J. Garay for motivating discussions during ESA '95, and the anonymous referees for their constructive comments on previous versions of this work.

References

[1] S. Aggarwal, J. A. Garay and A. Herzberg, "Adaptive Video on Demand", in the *Proc. of the 3rd Ann. European Symposium on Algorithms (ESA'95)*, LNCS 959, pp. 538–553, Springer-Verlag.

[2] B. Awerbuch, R. Gawlick, T. Leighton and Y. Rabani, "On-line Admission Control and Circuit Routing for High Performance Computing and Communication", in the *Proc. of IEEE Symp. on Foundations of Computer Science (FOCS'94)*, pp. 412-423, 1994.

[3] B. Awerbuch, Y. Bartal, A. Fiat and A Rosén, "Competitive non-preemptive call control", in the *Proc. of the 5th Ann. ACM-SIAM Symp. on Discrete Algorithms (SODA'94)*, 1994.

[4] B. Awerbuch, Y. Azar and S. Plotkin, "Throughput competitive on-line routing", in the *Proc. of the 34th Ann. Symp. on Foundations of Computer Science (FOCS'93)*, 1993.

[5] H.J. Chen and T.D.C. Little, "Physical Storage Organizations for Time-Dependent Multimedia Data", in Proc. *ACM FODO*, (1993).

[6] S. Christodoulakis and C. Faloutsos, "Design and Performance Considerations for an Optical Disk-Based, Multimedia Object Server ", *Computer 19*, (1986), pp.45-56.

[7] J. A. Garay, I. Gopal, S. Kutten, Y. Mansour and M. Yung, "Efficient On-line Call Control Algorithms", in the *Proc. of the 2nd Israeli Symp. on Theory of Computing and Systems*, pp. 285–293, June 1993.

[8] J. A. Garay and I. S. Gopal, "Call Preemption in Communication Networks", in the *Proc. of INFOCOM '92*, Florence, Italy, pp. 1043-1050, 1992.

[9] V. Kapoulas and P. Spirakis, "Randomized Competitive Algorithms for Admission Control in General Networks", in the *Proc. of the 14th Ann. ACM Symp. on Principles of Distributed Computing (PODC'95)*, August 1995.

[10] S. Loeb, "Delivering Interactive Multimedia Documents over Networks", *IEEE Communications Magazine 30*, (1992), pp.52-59.

[11] R. Motwani and P. Raghavan, "Randomized Algorithms", *Cambridge Univ. Press*, pages 318 and 333, 1995.

[12] S. M. McCarthy, "Integrating Telco Interoffice Fiber Transport with Coaxial Distribution", in the *Proc. of SPIE - Int. Soc. Opt. Eng.*, 1786:23–33, November 1993.

[13] P.V. Rangan, H.M. Vin, S. Ramanathan, "Designing an On-Demand Multimedia Service", *IEEE Communications Magazine 30*, (1992), pp.56-64.

[14] R. Ramarao, V. Ramamoorthy, "Architectural Design of On-Demand Video Delivery Systems: The spatio-Temporal Storage Allocation Problem", in Proc. *ICC* (1991).

[15] W.D. Sincoskie, "System Architecture for a Large Scale Video-On-Demand Service", *Computer Networks and ISDN Systems 22*, (1991), pp. 155-162.