Proxy Caching and Video Segmentation Based on Request Frequencies and Access Costs

Elias Balafoutis and Ioannis Stavrakakis
Department of Informatics & Telecommunications,
University of Athens, 15784 Athens, Greece
email:{balaf,istavrak}@di.uoa.gr

Abstract— **Partial caching of large media objects such as video £les has been proposed in recent proxy caching schemes, as the caching of entire videos can exhaust proxy storage resources within few requests. In this work the idea of segmenting videos into a number of chunks and replacement decisions be taken at the chunk level rather than based on the entire objects is adopted. In particular a video segmentation scheme called Variable Chunk Size is presented. This scheme uses the request frequencies to dynamically adjust the size of the chunk and it is shown to be capable of combining a high BHR with small response time to demand changes. Video segmentation is also considered as a mechanism to provide for caching differentiation based on access costs. By employing access cost dependent chunk sizes an overall access cost reduction is demonstrated.**

I. Introduction

Network caching has been widely used in the WWW as a solution for reducing bandwidth consumption and access latency, load balancing and improved data availability. The rapid growth in the deployment of internet based multimedia applications in conjunction with the large bandwidth requirements of video objects, have made video caching particularly important. Compared with the traditional web pages, video has several peculiarities, which are the reason that existing techniques for web caching are likely to be inappropriate when used for video. In the context of caching, the prominent characteristic of video, as recognized by recent related works, is it's large size. The large size of video makes the segmentation of video objects a sensible approach and inspired the development of partial caching techniques. Moreover, the structure of video gives several alternative choices in the segmentation of video.

Initial works on video caching have inherited the main characteristics of web caching schemes and have treated videos as single entities which are either cached completely or not at all [1], [2], [3]. More recent works have considered the video characteristics such as the associated rate variability, structure and large size, and have investigated the idea of partial video caching.

In [4] the initial frames of each video (called the pre£x) are cached in the proxy in order to improve the startup latency experienced by users. Additionally, smoothing is performed to reduce the peak bandwidth and increase the utilization in leased network channels that connect the proxy with the origin server. In a similar approach in [5], the bursty part of a VBR video stream is selected to be stored at the proxy while the remaining smooth part is retrieved directly from the repository, again reducing the peak bandwidth requirement in the backbone links. Both aforementioned schemes deal with the burstiness, which is inherent in VBR encoding algorithms.

In [6], [7] the pre£x is stored in the cache and the remaining part (the suf£x) is either explicitly requested from the video repository or retrieved through an ongoing multicast transmission which services a group of concurrent users; in the later case, a patch might be requested directly from the repository, so as to £ll the gap between the pre£x and the currently multicasted part of the suf£x. The focus of these works is to reduce the volume of traf£c that crosses the backbone by merging requests into multicast groups. Request merging is also proposed in [8], [9] in the form of window-based caching schemes. In particular, local proxies cache a sliding window of data, trying to merge requests for the same stream that arrive closely in time. This approach has the potential of reducing the number of concurrent connections to the server for the same video.

The caching of layered encoded video is studied in [10], [11]. In [10] an optimization algorithm determines which videos and which layers should be stored in the cache. In [11] the focus is on the maximization of the perceived quality for popular videos that are delivered over best-effort networks.

Unlike traditional web caching, most of the above video caching schemes, designed for Video on Demand systems, do not take into consideration the dynamic nature of caching according to which cache contents are dynamically updated based on demand, but rather focus on static replication strategies for some parts of the video.

The work that is more closely related to the one described here is [12]. In that work, video segmentation is considered and associated with a replacement algorithm which depends on (a) the object reference frequency and (b) the distance of each segment from the beginning of the video.

The work presented in this paper differs from the work in [12] in numerous ways. First, it proposes a different generalized segmentation scheme for the determination of the size of video segments. Second, video segmentation is examined in isolation and it is not associated with a specialized replacement algorithm. This allows for the derivation of conclusions that depend solely on the segmentation scheme irrespectively of the replacement policy. Third, our work considers additional aspects of performance such as the responsiveness to popularity changes and studies the trade-off between responsiveness and byte hit ratio (BHR). Additionally, it considers the case of different access costs for each video and proposes ways to integrate this information into the caching policy. Finally, it provides for a performance comparison of the proposed schemes.

Work supported in part by the General Secretariat for Research and Technology of Greece under the Joint Greek-Italian Scienti£c and Technological Cooperation programme and the IST programme of the European Union under contract IST-2001-32686 (Broadway)

0-7803-7661-7/03/$17.002003 IEEE

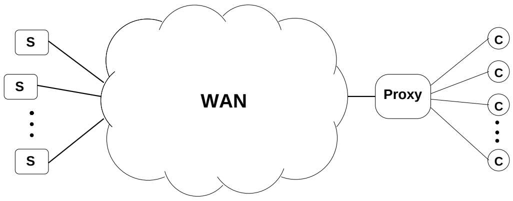

Fig. 1. Network topology: the origin servers (S) hold the available videos; clients (C) request videos and the proxy server services these request.

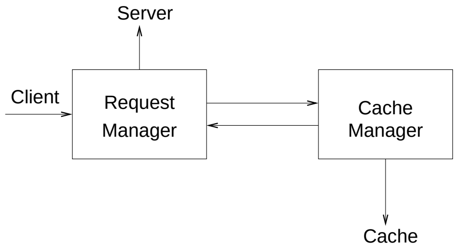

Fig. 2. The internal proxy architecture.

In a preliminary version of this work [13] an alternative segmentation scheme was presented which uses £xed size segments (chunks). This Fixed Chunk Size scheme was shown to provide increased Byte Hit Ratio (BHR) at the cost of responsiveness to demand changes. In this work a Variable Chunk Size scheme is proposed, which uses the request frequencies to dynamically adjust the size of the chunk. This scheme is shown to be capable of combining a small response time with high BHR. Moreover, segment-based caching is considered also as a mechanism to provide for caching differentiation based on access costs. By employing access cost dependent chunk sizes an overall access cost reduction is demonstrated.

The rest of the work is organized as follows. In section II the proxy server architecture is presented and the proposed caching techniques are introduced . The performance of the proposed schemes is evaluated via simulation in Section III. The work is concluded in Section IV.

II. Segment-Based Caching

A. System Architecture

Figure 1, illustrates the topology of the video distribution system under consideration. Videos are stored at geographically dispersed origin servers. A proxy server is installed at the same local area network with a number of clients. Requests for videos are directed to the proxy which services them either from its local cache or by contacting the origin servers over the wide area network. It is assumed that there is abundant bandwidth between the proxy and the clients to support video streaming. On the other hand, the transmission of videos from the origin servers over the wide area network is expensive as it consumes bandwidth, which is a scarce resource in the backbone. The proxy caches the videos trying to reduce the access to the servers and consequently the volume of data transmitted over the wide area network.

Figure 2 illustrates the internal architecture of the proxy server. The proxy consists of two major entities: the *request manager* and the *cache manager*. The request manager is responsible for the continuous streaming of video towards the clients. In general, its responsibility is to schedule the transmission of the pre£x to the clients and forward a request for the suf£x to the origin server. The main responsibility of the cache manager is to ef£ciently allocate the proxy's storage resources to the requested videos. This work focuses mainly on the functionality of the cache manager. To allow for the isolated study of the cache manager, only a simple request manager is considered. It is assumed that the request manager immediately initiates the transmission of the pre£x (if any) to the client and requests the suf£x from the origin server. In addition it is assumed that the suf£x can be (and is) delivered exactly when it is needed from the origin server.

The cache manager receives incoming data from the origin server and decides whether the new data will be cached or not. When the decision is to cache the newly retrieved video parts the cache manager also decides (a) how much space to dedicate to this video, (b) which of the missing parts of the video to hold and (c) which data to remove from the cache to make room for the new data.

The third decision is in essence the replacement policy. In traditional web-caching schemes and in some video caching schemes as well, the cache manager functionality is limited to the replacement decision. These schemes do not consider object segmentation and consequently they do not have to decide which parts of the object to cache. Instead they dedicate space equal with the size of the object. In contrast this work focuses on the £rst two functions ((a) and (b) above) of the cache manager which perform before the replacement function.

The £rst function, that is, the decision about the space allocated to each video, in essence controls the replacement granularity and as it will be shown in the sequel it affects signi£cantly the performance of the proxy.

The second function of the cache manager functionality, which is the selection of speci£c parts of a certain size, does not affect the performance in terms of the cache hit ratio since it does not affect the volume of data that is being cached. However it may affect the quality or delay of the perceived stream. For example, if layered encoding is assumed, the selected parts can be speci£c layers of the video and this may affect the perceived quality. Otherwise these parts can be consecutive frames at any distance from the beginning of the video and this selection may affect the delay (initial delay, or delay due to VCR operations) experienced by the users. In this work it is desirable the initial consecutive parts of a video to be cached than any other combination of same size data in order to reduce the initial delay, as neither layered encoding nor VCR operations are considered.

B. The proposed Variable Chunk Size scheme

The aforementioned functions that make up the operation of the cache manager, work sequentially as follows. The cache manager uses a replacement unit, called a chunk; when a video that is not in the cache is requested it is fetched from the server and its initial segments with the size of a chunk are stored in the proxy. In case there is not suf£cient space in the cache the replacement algorithm selects a video for removal. The last chunk of the selected video is removed. Each additional request for the same video results in the caching of an additional (consecutive) chunk. This guarantees that only the pre£x (initial consecutive parts) of each video is cached.

The size of the chunk, that is, the granularity of the replacement procedure, has a great impact in the performance of the cache and it is determined by video segmentation schemes as the proposed Variable Chunk Size (VCS) schemes that is presented in the sequel.

Under the Variable Chunk Size (VCS) scheme, the size of the chunk is a function of the cached part of the video at the time the request is made. This means that the size of the chunk is (a) typically different for each video and (b) not the same for all requests of a speci£c video. More speci£cally, the size of the chunk for a requested video i when there are k units of video i cached is: $chunk(i,k) = g * k$, $k \neq 0$, where $g > 0$, will be referred to as the *acceleration factor*. If the requested video is initially missing from the cache, that is k = 0, then a few segments are cached as the £rst chunk, $chunk(i,0)$. For subsequent requests the size of the chunk is equal to a multiple of the cached segments of video i at the time instant that the request was made. The motivation under this scheme is (a) to provide large chunk size to popular objects and consequently to increase the probability that a greater portion of these objects will be held in the cache and (b) to give the opportunity to new popular videos that are missing from the cache to increase their cached portion faster (within fewer requests) by increasing the chunk size in each request.

For comparison reasons the Fixed Chunk Size (FCS) scheme which was introduced in [13] is also considered. Under FCS the size of the chunk is a tunable parameter which can vary from a few frames to the complete video but once it has been selected, it remains the same for all videos and for all requests[1].

C. *Chunk Size Determination Based on the Access Costs*

Replacement algorithms such as Least Recently Used (LRU) or Least Frequently Used (LFU), base the replacement decisions on the objects' popularity. In most practical cases, there are other issues as well to be considered, such as a different access cost for each object or the case in which some content providers are willing to allow for their content to be preferentially treated by network caches. In these cases there is a need for the caching system to be able to take such factors into consideration.

In this work, a different access cost for each video is considered. In this case the quantity that contributes to the overall cost is the product of the request probability and the access cost for the requested video. In the following two alternative approaches are proposed to account for the different access costs.

The £rst approach provides the differentiation based on the access costs by properly determining the chunk sizes. In particular a video segmentation scheme referred to as *Chunk Based Differentiation* is proposed which is shown to reduce the overall access cost. Under this scheme, an appropriate selection of the chunk size for each video is made based on its access cost. In particular, the largest possible chunk size (a complete video) is assigned to the object with the highest access cost and proportionally (based on the access cost) smaller chunk sizes for the remaining objects. In a sense, this scheme is a combination of the FCS and VCS schemes proposed earlier, as the size of the chunk is different for each video (it is proportional to its access cost) but it is the same for all requests for the same video.

The second approach uses the FCS scheme in combination with *probabilistic LRU* and leaves the replacement algorithm to account for the different access costs. Probabilistic LRU has been proposed for web-caching (caching of entire objects) in [14]. Under that scheme, when a new request for a video arrives the least recently requested item is removed from the cache with a probability which is proportional to its access cost and it remains in the cache (no replacement takes place) with the supplementary probability. The two approaches are compared with each other in section III and the bene£ts of each scheme are presented.

[1] When the size of the chunk is equal to a complete video this scheme decays to web-like caching schemes

III. Performance Study

A. *Performance Metrics*

As mentioned in Sect. II-A, the main responsibility of the cache manager is to ef£ciently manage the proxy's storage resources so as to reduce the volume of the data that are fetched from the origin servers. This performance aspect is captured by the Byte Hit Ratio (BHR), which is the fraction of data that can be served directly from the cache's local storage. In systems where partial caching is applied a hit in most cases is partial and should capture the portion of data currently on the proxy. Consequently, the BHR for a single request for video i is de£ned as:

$$BHR_i = \frac{\text{Size of the cached portion of the requested video i}}{\text{Size of the complete video i}}$$

BHR_i takes values between 0 and 1; 0 for a complete miss, and 1 for a complete hit. The average BHR of all requested videos over an interval x is denoted as $BHR(x)$; the interval x can be a time interval (e.g., a day) or a number of requests. The steady state BHR (ss-BHR) is determined from $BHR(x)$ as x tends to in£nity and assuming that no popularity changes occur. That is, the probability of a video is assumed to remain unchanged over the interval x.

B. *Simulation Model*

The request distribution of the videos was assumed to follow a zipf-like distribution similar to that observed for web pages [15]. Typically, the request pattern in a proxy server exhibits temporal locality. However due to the lack of real video server traces, and the dif£culty of producing a synthetic workload that does exhibit temporal locality while the video popularity follows the zipf's law [11], [16], the independent reference (IR) model is assumed. According to the IR model a video i is requested with probability p_i independently of previous requests. The fact that several results obtained with real traces are qualitatively similar to those obtained with the simulation using the IR model [15] indicates that the IR model is suf£cient to predict the relative performance of the proposed caching algorithms.

The simulation model consists of a video server with N videos, and a proxy server with a storage capacity of K complete videos; the ratio K/N captures the relative cache size of the proxy. For simplicity it is assumed that all videos are constant bitrate encoded and are of equal length L units. Under the constant bitrate assumption video length units can be time or storage units. p_i denotes the request probability of video i which is also refered to as the popularity of video i. The video popularity follows a Zipf distribution; that is, the request probability for video i is $p_i = C/i^a$, where $C = (\sum_{i=1}^{N} \frac{1}{i^a})^{-1}$ and a is the Zipf parameter determining the skewness of the distribution. The parameters of the simulation study are summarized in Table I.

It is assumed that request arrivals follow a Poisson process with mean rate λ. The popularity changes considered in the simulation were implemented using the following setting: whenever a

TABLE I

SIMULATION PARAMETERS

Notation	*System Parameters*	*Default Values*
L	Video Size / Duration	1000 units / 1hour
K	Cache Size	100 videos
N	Number of Videos in the repository	1000
K/N	Relative Cache Size	10%
a	Zipf parameter	0.8
λ	Mean request arrival rate	30 req/hour

popularity change is about to occur, we transpose the popularities of videos, i.e., popular videos become unpopular and vise versa. Under the new popularity distribution, unpopular videos that were missing from the cache appear as new hot videos and eventually capture a signi£cant part of the cache. Previously popular videos are made unpopular and are eventually pushed out of the cache.

In the simulations the LRU replacement policy is used with a slight modi£cation that prevents the replacement of chunks belonging to "active videos" (videos that are currently streamed), i.e., the least recently used inactive video is selected for the replacement. LRU is chosen as it is the most widely studied replacement policy and is often used as a comparison standard.

C. Simulation Results

1) Video Segmentation: The Byte Hit Ratio

Fig. 3 illustrates the effect of the relative cache size on ss-BHR, for several cases. As expected the BHR increases with the relative cache size since a greater number of chunks £t in the cache. Similar observations apply to Fig. 4 that depicts the ss-BHR as a function of the Zipf parameter. The VCS scheme outperforms FCS especially for small caches sizes and highly skewed request distributions. This results are reasonable since popular objects are treated preferentially by the VCS scheme which allocates to them a larger chunk size.

Fig. 5 illustrates the ss-BHR as a function of the chunk size for the FCS scheme, under a non-changing popularity distribution and for the system parameters presented in table I. It is observed that as the chunk size increases, the BHR reduces initially fast and then slowly converges to a value which is the BHR of a web-like video caching scheme. Similar observations apply to Fig. 6 which illustrates the ss-BHR as a function of the acceleration factor for the VCS scheme. Generally a large value of the acceleration factor leads to large chunk sizes and consequently to the reduction of the BHR. Note - by observing the different scale in the axis of these two £gures - that the impact of the acceleration factor in the VCS scheme, is rather small comparatively with the impact of the chunk size on the FCS scheme.

Response Time

In the results presented above it was assumed that the popularity of videos does not change, that is, the underlying request probability remains the same. Upon a change of popularities, the BHR is expected to decrease for some period and then converge to the new steady-state value. It should be noted that the new ss-BHR is not necessarily the same with the old one as it depends on the skewness of the popularity distribution. Responsiveness can be qualitatively de£ned as the ability of the system to adapt to changes in popularities. In order to capture this performance

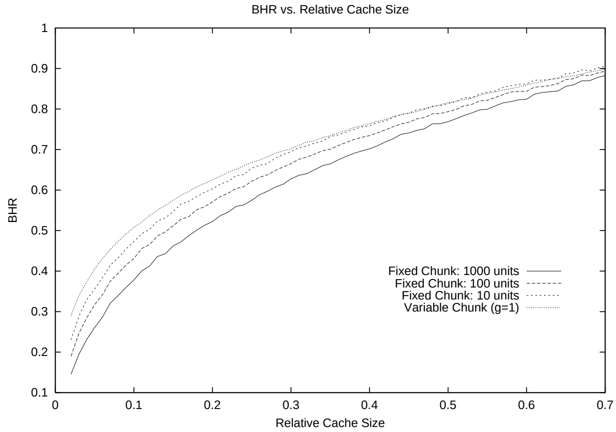

Fig. 3. The ss-BHR against relative cache size for several cases

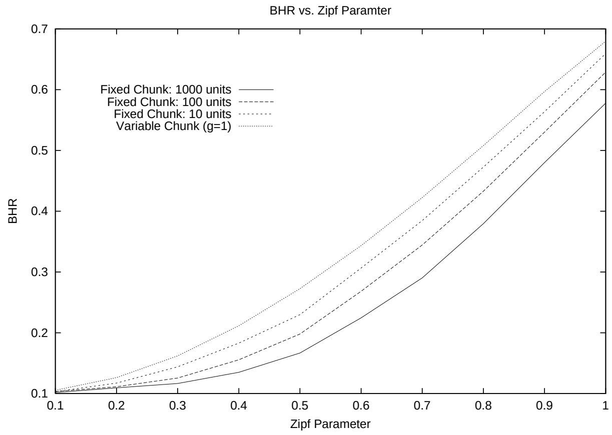

Fig. 4. The ss-BHR against Zipf parameter a, for several cases

aspect, we ¤ush the cache[2] and measure the time needed for the BHR to reach 90% of its steady-state value. In Fig. 7 and Fig. 8 the response time is illustrated for FCS and VCS respectively. For the FCS scheme, the response time is smaller for large chunk sizes and increases rapidly as the chunk size decreases. The same happens in the VCS scheme with the acceleration factor. What is worth noting in these £gures is the relatively small response time under the VCS scheme even for very small values of the acceleration factor.

Comparing £gures 5, 7 with 6, 8 it can be concluded that the VCS scheme achieves a more effective compromise in the trade-off between BHR performance and responsiveness compared to the FCS scheme. For example, a BHR of 50% can be achieved with a £xed chunk size of 10 units (Fig. 5) at the cost of a huge response time of 1000 hours (Fig. 7). The same BHR can be achieved under the proposed VCS scheme with accelerator factor 2 (Fig. 6) at the cost of response time of 10 hours (Fig. 8).

The responsiveness of the caching system affects signi£cantly the overall performance especially when demand changes occur often. When frequent changes of popularity occur, the ss-BHR may practically never be reached if the caching system requires a long adaptation period. This could lead a system that appears to have a high ss-BHR to a worst long term average BHR since transient periods dominate the long term performance. This is illustrated in Figures 9 and 10.

Fig. 9 shows the BHR(24h) versus time for the FCS with chunk

[2]This is an extreme case of popularity change since it is equivalent to a cache full with totally unpopular videos

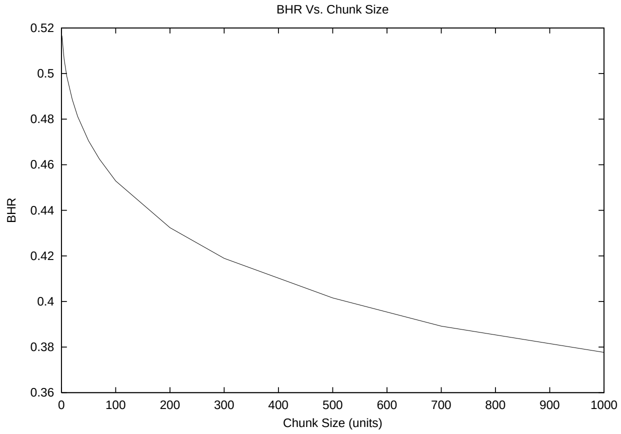

Fig. 5. The ss-BHR for different chunk sizes, for the FCS scheme.

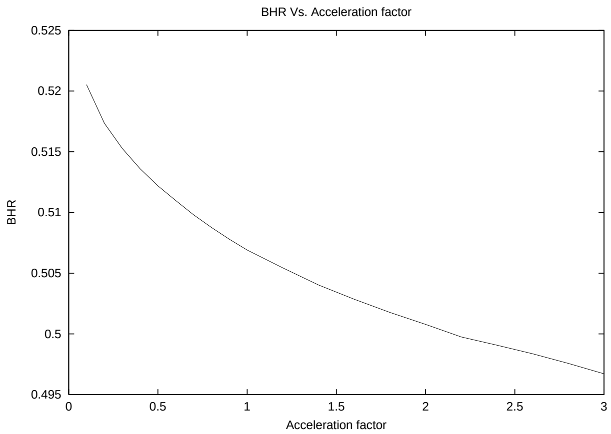

Fig. 6. The ss-BHR for different values of the acceleration factor, for the VCS scheme.

size of 30 units and the VCS scheme with $g = 1$[3]. The BHR initially oscillates around the steady-state value for both schemes. At time instant 500 a demand change occurs. In the period that follows this change, the BHR reduces for both schemes and then again converges (in an oscillatory fashion)to the steady state value. This reduction occurs due to the fact that the content of the cache during the transient period does not match the optimal one under the new demand distribution. It is observed that the VCS scheme with $g = 1$ achieves higher BHR than the FCS scheme with chunk size of 30 units during both the steady-state and the transient period, resulting in a higher long term average BHR.

Similar observations apply to Fig 10 where the VCS scheme with $g = 10$ is compared with the FCS with chunk size of 100 units. These results seem to suggest that for any £xed chunk size of the FCS scheme, an appropriate value of the acceleration factor (g) of the VCS exists such that the VCS scheme results in a higher BHR than the FCS scheme during the steady-state and no worse during the transient period, thus achieving a higher long term average BHR.

2) The Chunk Based Differentiation scheme: In this section some results are presented illustrating the ability of the Chunk Based Differentiation scheme to reduce the overall cost when different access costs for each video is assumed. In order for the results to be comparable with those in the previous section it is assumed that all objects are requested with probability $p_i = 1/N$ and the access costs follow a zipf-like distribution similar to the popularity distribution in the previous section. Equivalently with

[3]In order to achieve a fast convergence to the steady state, the cache is considered initially full with the most popular videos that £t in the cache.

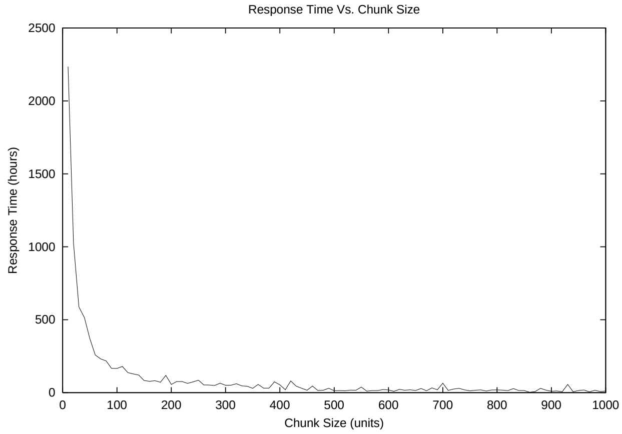

Fig. 7. Response time for the FCS scheme

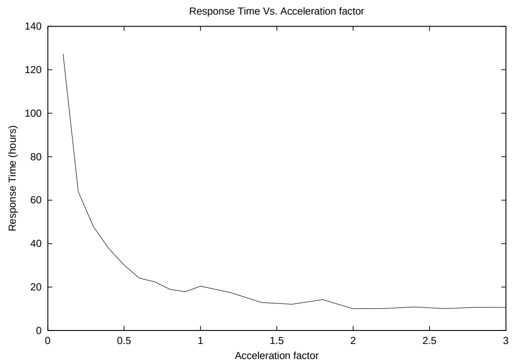

Fig. 8. Response time for the VCS scheme

the BHR the performance metric of interest that is considered is the *cost reduction ratio*

$$CRR = \frac{\sum_{i \in R} k_i * c_i}{\sum_{i \in R} L_i * c_i}$$

where, R is the set of video indexes which correspond to the requests made to the proxy and k_i, c_i and L_i are the size of the cached portion, the access cost, and total size, respectively, of the video associated with request i.

In Fig. 11 the performance of the proposed algorithm for several cache sizes is illustrated. It is observed that although FCS with probabilistic LRU performs better, the general behavior of the Chunk Based Differentiation scheme, is similar to that of FCS with probabilistic LRU when no demand changes occur. However since under the Chunk Based Differentiation scheme the chunk sizes are different for each video, the responsiveness of this scheme would be closer to that of the VCS scheme presented in Fig 8. In contrast, the responsiveness of the FCS with probabilistic LRU would be far worst than that of FCS with plain LRU presented in Fig. 7, since under the probabilistic LRU, only a small fraction of requests is taken into account. Based on the £ndings about the trade-off between ef£ciency and adaptability to demand changes discussed in £gures 9, 10 and more analytically in [13], the difference in the response time and the relatively small difference in the cost reduction ratio under no demand changes, make the Chunk Based Differentiation scheme an attractive solution when demand changes occur.

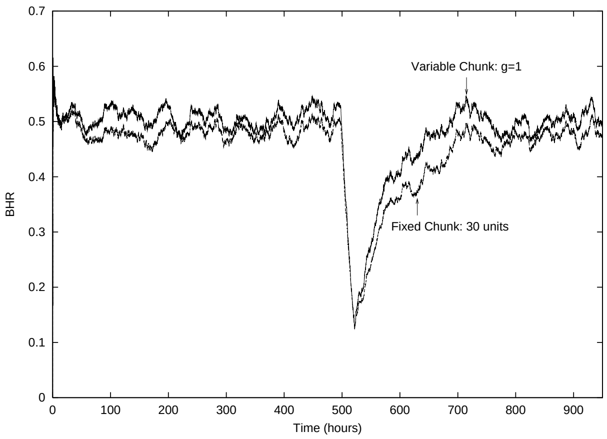

Fig. 9. The BHR(24h) versus time for the FCS with chunk size of 30 units and VCS with g = 1. A sudden change in the demand pattern occures at time instant 500.

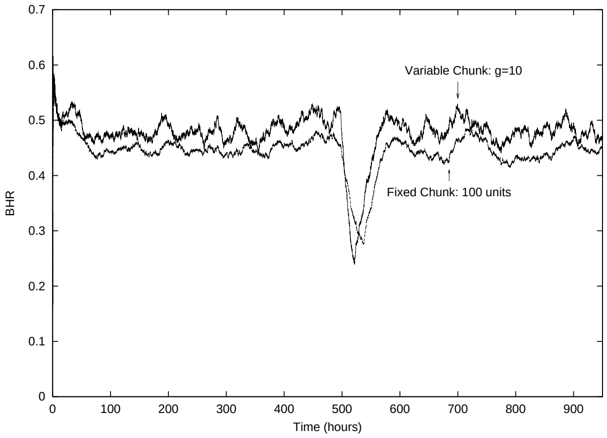

Fig. 10. The BHR(24h) versus time for the FCS with chunk size of 100 units and VCS with g = 10. A sudden change in the demand pattern occures at time instant 500.

IV. CONCLUSIONS

For the caching of large media objects such as video £les, partial caching has been proposed. Recently developed replacement algorithms for partial caching schemes, propose that videos be segmented into chunks and replacement be performed at the level of a chunk. In this paper a video segmentation scheme, called Variable Chunk Size segmentation scheme, which dynamically determines the size of the chunk for each video based on the request frequencies is presented. The performance of the proposed scheme is studied with respect to the byte hit ratio and the responsiveness (the ability to adapt to demand changes) and it is shown that it is capable of combining high BHR and small response time effectively.

The bene£t of segment-based caching is also demonstrated for the case where different access cost for each video is considered. Two schemes have been proposed on this direction: The Chunk Based Differentiation scheme where the size of the chunk is based on the access cost of each video and and the combination of the FCS scheme with probabilistic LRU. Under £xed demand distribution it is shown that the later scheme achieves a higher cost reduction than the Chunk Based Differentiation scheme. Nevertheless it is argued that due to its good cost reduction ratio and better response to demand changes Chunk Based Differentiation may be particularly effective under a demand changing environment in which the FCS scheme and the probabilistic LRU respond rather slowly.

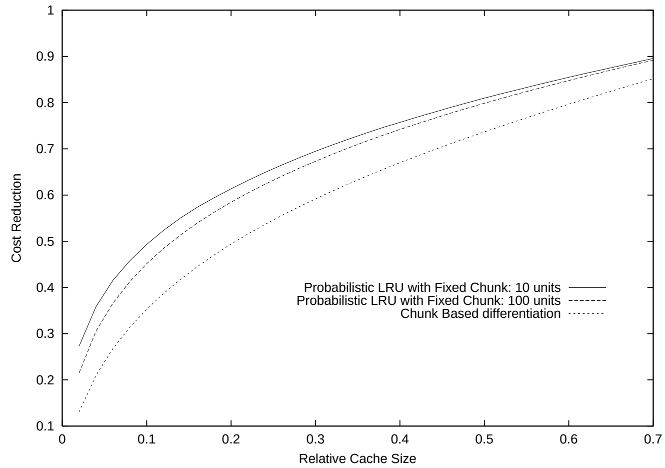

Fig. 11. Cost reduction ratio of FCS with probabilistic LRU and Chunk Based Differentiation for several cache sizes

REFERENCES

[1] Scott A. Barnett, Gary J. Anido, and H.W. Beadle, "Caching policies in a distributed video on-demand system," in *Australian Telecommunication Networks and Applications Conference*, Sydney, Australia, 1995.

[2] J.-P. Nussbaumer, B. V. Patel, F. Schaffa, and J. P. G. Sterbenz, "Networking requirements for interactive video on demand," *IEEE Journal on Selected Areas in Communications*, vol. 13,5, pp. 779–787, 1995.

[3] Christos Papadimitriou, Srinivas Ramanathan, P Venkat Rangan, and Srihari Sampathkumar, "Multimedia information caching for personalized video-on-demand," *Computer Communications*, vol. 18, no. 3, Mar. 1995.

[4] Subhabrata Sen, Jennifer Rexford, and Don Towsley, "Proxy pre£x caching for multimedia streams," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, New York, Mar. 1999.

[5] Zhi-Li Zhang, Yuewei Wang, D. H. C. Du, and Dongli 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, Aug. 2000.

[6] Bing Wang, Subhabrata Sen, Micah Adler, and Don Towsley, "Proxy-based distribution of streaming video over unicast/multicast connections," Technical Report UMASS TR-2001-05, University of Massachusetts, Amherst, 2001.

[7] S. Ramesh, I. Rhee, and K. Guo, "Multicast with cache (mcache): An adaptive zero-delay video-on-demand service," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Anchorage, Alaska, Apr. 2001.

[8] S.-H. Gary Chan and Fouad A. Tobagi, "Caching schemes for distributed video services," in *Proceedings of the IEEE International Conference on Communications (IEEE ICC)*, Vancouver, Canada, June 1999.

[9] Markus Hofmann, T.S. Eugene Ng, Katherine Guo, Paul Sanjoy, and Hui Zhang, "Caching techniques for streaming multimedia over the internet," Tech. Rep., Bell Laboratories, May 1999.

[10] Jussi Kangasharju, Felix Hartanto, Martin Reisslein, and Keith W. Ross, "Distributing layered encoded video through caches," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Anchorage, Alaska, Apr. 2001.

[11] Reza Rejaie, Haobo Yu, Mark Handley, and Deborah Estrin, "Multimedia proxy caching mechanism for quality adaptive streaming applications in the internet," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, Tel Aviv, Israel, Mar. 2000.

[12] Kun-Lung Wu, Philip S. Yu, and Joel L. Wolf, "Segment-based proxy caching of multimedia streams," in *WWW 2001*, Hong Kong, China.

[13] Elias Balafoutis, Antonis Panagakis, Nikolaos Laoutaris, and Ioannis Stavrakakis, "The impact of replacement granularity on video caching," in *Proceedings of IFIP Networking 2002*, Pisa, Italy, May 2002, pp. 214–225.

[14] David Starobinski and David Tse, "Probabilistic methods for web caching," *to appear in Performance Evaluation (Special Issue of SAPM 2000 workshop).*

[15] Lee Breslau, Pei Cao, Li Fan, Graham Phillips, and Scott Shenker, "Web caching and zipf-like distributions: Evidence and implications," in *Proceedings of the Conference on Computer Communications (IEEE Infocom)*, New York, 1999.

[16] Barford Paul and Crovella Mark, "Generating representative web workloads for network and server performance evaluation," in *Measurement and Modeling of Computer Systems*, 1998, pp. 151–160.