Analysis of Resource Sharing and Cache Management in Scalable Video-on-Demand

Bashar Qudah and Nabil J. Sarhan
Department of Electrical and Computer Engineering
Wayne State University
Detroit, MI 48202, USA
Email: {bqudah,nabil}@wayne.edu

Abstract

The required real-time and high-rate transfers for multimedia data severely limit the number of requests that can be serviced by Video-on-Demand *(VOD) servers. Resource sharing techniques can be used to address this problem. We evaluate through extensive simulation major resource sharing techniques, considering both the* True Video-on-Demand *(TVOD) and* Near Video-on-Demand *(NVOD) service models. Moreover, we propose a statistical approach for cache management and derive analytical models for optimal cache allocation to reduce the demands on the disk I/O when various resource sharing techniques are used.*

1 Introduction

Driven by the increasing expectations of customers for fully customizable and more convenient services, yet at low prices, *Video-on-Demand* (VOD) is expected to replace both broadcast-based TV programming and DVD movie rentals at stores. The market has already prepared for such a change, and some landmark steps have been taken.

Unfortunately, the number of concurrent video streams that can be supported is highly constrained by the required real-time and high-rate transfers. These requirements quickly consume server resources, including network bandwidth and disk I/O bandwidth. Resource sharing techniques face this challenge by utilizing the multicast facility. Resource sharing can be achieved by *stream merging* [5, 15, 4, 8, 20, 2], *periodic broadcasting* [10, 14, 16, 17, 23], and composite techniques [12]. Stream merging techniques combine streams when possible to reduce the delivery costs. These techniques include *Stream Tapping/Patching* [5, 15], *Transition Patching* [4], and *Earliest Reachable Merge Target (ERMT)* [8, 9]. Periodic broadcasting techniques, such as *Skyscraper Broadcasting* (SB) [16], *Greedy Disk-conserving Broadcasting* (GDB) [10], *Harmonic Broadcasting* (HB) [17], and *Fibonacci Broadcasting* (FB) [14], divide each supported video into multiple segments and broadcast them periodically on dedicated channels. Composite techniques, such as *Catching* [12] and *Selective Catching* [12], combine the advantages of stream merging and periodic broadcasting.

The literature lacks detailed comparative analysis of various resource sharing strategies. With the many available resource sharing techniques from different classes, it is unclear which one is the best to use in a target environment. For example, it is unclear how ERMT performs in comparison with Selective Catching and Transition Patching and how periodic broadcasting techniques perform compared with stream merging techniques. Moreover, only limited performance metrics, workloads, and service models were considered.

Furthermore, there is very little work on cache management for reducing the demands on the disk I/O when resource sharing techniques are applied. This is, however, an important issue for designing cost effective and scalable VOD servers, especially because of the widening gap between technology improvements in the capacities and speeds of hard disk drives [13].

This paper addresses those limitations. The main contributions of this study can be summarized as follows. (1) We evaluate through extensive simulation major resource sharing techniques under both the *True Video-on-Demand* (TVOD) and *Near Video-on-Demand* (NVOD) service models. (2) We develop analytical models for optimal tuning of design parameters for Transition Patching and Selective Catching. (3) We propose a statistical approach for cache management and derive analytical models for allocating cache space for various resource sharing techniques. (4) We introduce the load on the underlying storage subsystem as an additional dimension to the comparisons among various resource sharing techniques and examine the impact of

caching on reducing these requirements.

The performance evaluation considers four target environments: *mixed-video workload and TVOD*, *mixed-video workload and NVOD*, *hot-video workload and TVOD*, and *hot-video workload and NVOD*. The first two environments are suitable for analyzing only stream merging and composite techniques, whereas the other two can be used to compare all techniques, including periodic broadcasting.

The rest of this paper is organized as follows. Section 2 discusses major resource sharing techniques. The issue of cache management is discussed in Section 3. Section 4 shows how design parameters can be tuned optimally. Section 5 discusses the performance evaluation methodology and workload characteristics and presents the main results. Finally, conclusions are drawn in the last section.

2 Resource sharing

Let us now discuss the main resource sharing techniques analyzed in the paper: Stream Tapping/Patching, Transition Patching, ERMT, Selective Catching, FB, SB, GDB, and HB. To conduct fair comparisons and account for limitations in client bandwidth, we do not consider techniques that require client download bandwidth more than double the video playback rate, except for HB-based techniques for their effectiveness in reducing the server bandwidth requirements.

Stream Merging Techniques

Stream Tapping/Patching expands the Batching multicast tree dynamically to include new requests. A new request joins the latest *regular* (i.e., full) stream for the object and receives the missing portion as a *patch*. Hence, it requires two download channels (each at the video playback rate) and additional client buffer space. To avoid the continuously increasing patch lengths, regular streams are retransmitted when the required patch length exceeds a prespecified value called *regular window* (Wr). Transition Patching allows some patches to be sharable by extending their lengths. It introduces another multicast stream, called *transition patch*. The threshold to start a regular stream is Wr as in Patching, and the threshold to start a transition patch is called the *transition window* (Wt). ERMT is a near optimal hierarchical stream merging technique that requires two download channels (each at the playback rate). A new client or newly merged group of clients snoops on the closest stream that it can merge with if no later arrivals preemptively catch them.

Periodic Broadcasting Techniques

SB, GDB, and FB divide each video into a number of non-equal segments and broadcast each segment periodically on a dedicated channel at the video playback rate. The client waits until the beginning of the next broadcast of the first segment and then receives data concurrently from two broadcast channels. The relative length of the n^{th} segment compared to the first segment is determined using a partitioning series. Each protocol has a different series as shown in Table 1. On the other hand, HB divides each video into a number of equal segments and broadcasts them periodically on channels with decreasing bandwidth. The bandwidth for channel i is b/i, where b is the playback rate. The client has to receive data concurrently from all broadcast channels allocated for the requested video. HB has a continuity problem, which was resolved by Cautious Harmonic Broadcasting (CHB) and Quasi Harmonic Broadcasting (QHB) [19]. Both still require clients to receive data from all broadcast channels.

Table 1: Segment Partitioning in Periodic Broadcasting

Technique	Partitioning Series
SB	1, 2, 2, 5, 5, 12, 12, 25, 25, 52, 52, 105, 105,...
FB	1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377,...
GDB	1, 2, 2, 5, 5, 12, 12, 25, 25, 60, 60, 125, 125,...
Modified GDB	1, 1, 1, 2, 2, 5, 5, 12, 12, 25, 25, 60, 60,...

Composite Techniques

Catching uses a modified version of GDB (whose series is shown in Table 1) to deliver the hot videos, but instead of making the client wait until the beginning of the next broadcast time, the client receives the small missed portion by a patch. A client initially listens to one broadcast channel and its own patch, and then to two broadcast channels. Selective Catching extends Catching by delivering the cold videos using *Controlled Multicast* [11], which works essentially the same as Patching.

3 Cache Management

The goal of cache management is to reduce the load on the storage subsystem and thus reduce the cost of the overall solution by caching selected data in the main memory. For VOD servers, a caching technique, called *Interval Caching* [6], was proposed. This technique exploits the locality of reference by caching intervals between successive streams. It is effective in reducing the I/O load, but unlike other resource sharing techniques, it does not address the problem of the limited network bandwidth. In fact, cache management can be used with any resource sharing technique, but there has been very little work on that. In [3], the impact of caching was reported by actual measurements for only limited caching schemes and limited resource sharing techniques.

We investigate here a statistical approach for cache management. With this approach, the server periodically computes video access frequencies and determines the data to be cached based on these statistics. Besides being performance effective, this approach is easy to implement and

incurs small overhead as updates are triggered only when the workload varies considerably. This work differs significantly from previous work on proxy caching [18] (and references within), which has different objectives and factors affecting the allocation decisions. In addition, our approach differs from low-level caching techniques, such as *Least Recently Used* (LRU), *Least Frequently Used* (LFU), and *Interval Caching* [6, 21], whereby the content of the cache changes frequently, incurring significant overheads.

In this section, we first derive the access distribution equations for videos served by various techniques and then use these equations to determine the optimal cache allocation. For ease of reference, Table 2 describes the main parameters used in the subsequent analysis.

Table 2: Main Parameters

Parameter Name	Symbol
Request Arrival Rate for the j^{th} Video (Req/s)	λ_j
Inter Arrival Time for the j^{th} Video (s) = $1/\lambda_j$	Δ_j
Access Frequency Distribution for j^{th} Video (Req/s)	$F_j(t)$
Cache Size (s)	S
Size of Cached Portion from the j^{th} Video (s)	S_j
Number of Videos	V
j^{th} Video Duration (s)	D_j
Regular Window for the j^{th} Video (s)	Wr_j
Transition Window for the j^{th} Video (s)	Wt_j
Number of Requests per Video Length for the j^{th} Video $= \lambda_j \times D_j$	N_j

3.1 Video Access Distributions

We derive next the access distribution equations for each playback point in a video delivered using various techniques. We assume here a TVOD model where every request is served immediately (except for Periodic Broadcasting). Hence, the average service rate of video j is the same as its average request arrival rate (λ_j). We also assume that $F_j(t)$ is the rate (frequency) by which the t^{th} point of time of video j is requested from the disks when there is no cache. The derived equations were validated by simulation.

Patching and Controlled Multicast

Because of having two types of streams in Patching (and Controlled Multicast), and having a limit on the maximum patch length (Wr_j), the video can be divided into two regions: $[0, Wr_j]$ and $[Wr_j, D_j]$. Since every regular stream delivers the full video data, all points in the second region will be requested with an equal rate: $1/Wr_j$. The situation is different for the first region because the patches vary in length. The closer the point to the beginning of the video, the more likely it will be missed and requested by patches. The access rate approaches the video request rate (λ_j) as the point becomes closer to the beginning. Consequently, the access distribution can be formulated as follows:

$$F_j(t)=\begin{cases}\lambda_j - (\frac{\lambda_j}{Wr_j} - \frac{1}{(Wr_j)^2})t & 0 \le t < Wr_j, \\ \frac{1}{Wr_j} & Wr_j \le t \le D_j.\end{cases} \quad (1)$$

Transition Patching

Transition Patching has three types of streams: patches with maximum length of Wt_j, transition patches with lengths between $3Wt_j$ and $Wr_j + 2Wt_j$, and the regular streams with full video length. Therefore, the region $[Wr_j + 2Wt_j, D_j]$ is delivered by only regular streams at an access rate of $1/Wr_j$. As in Patching, the rate for region $[0, Wt_j]$ decreases linearly from λ_j to $1/Wt_j$. The region $[Wt_j, 3Wt_j]$ is delivered by every transition patch and regular stream, and thus the rate is $1/Wt_j$. The region $[3Wt_j, 4Wt_j]$ is delivered by every transition patch and regular stream except the first transition patch in every regular window (Wr_j). The region $[4Wt_j, 5Wt_j]$ is delivered by every transition patch and regular stream except the first two transition patches in every regular window. The pattern continues up to the region $[Wr_j + Wt_j, Wr_j + 2Wt_j]$, which is delivered by only regular streams and the last transition patch in every regular window, leading to a stair shape curve. Consequently, the access distribution can be formulated as follows:

$$F_j(t)=\begin{cases}\lambda_j - (\frac{\lambda_j}{Wt_j} - \frac{1}{(Wt_j)^2})t & 0 \le t < Wt_j, \\ \frac{1}{Wt_j} & Wt_j \le t < 3Wt_j, \\ \frac{1}{Wt_j} - \frac{n}{Wr_j} & 3Wt_j \le t < Wr_j + 2Wt_j, \\ \frac{1}{Wr_j} & Wr_j + 2Wt_j \le t \le D_j,\end{cases} \quad (2)$$

where $n = \lfloor \frac{t}{Wt_j} \rfloor - 2$.

ERMT

Under a uniform arrival process, where the inter-arrival time between successive requests for video j is Δ_j, the length of streams in ERMT have the following pattern: $N_j/2$ streams have the length of Δ_j, $N_j/4$ streams have the length of $4\Delta_j$, $N_j/8$ streams have the length of $10\Delta_j$, etc. Generally, $N_j/2^i$ streams have length $L(i) \times \Delta_j$, up to $i = \log_2(N_j)$, where $L(i) = 2L(i-1) + 2$, and $L(1) = \Delta_j$. In reality, a customer does not arrive every exactly Δ_j, but this will be the average behavior over a long period. Hence, the first part of the video from the beginning to Δ_j is delivered (i.e. requested) by every stream, with a request frequency equal to λ_j. The second part from Δ_j to $4\Delta_j$ is requested by $N_j - N_j/2$ streams. The third part from $4\Delta_j$ to $10\Delta_j$ is requested by $N_j - N_j/2 - N_j/4$ streams, and so on. To generalize, the access frequency for the t^{th} point of time in the j^{th} video is given by $F_j(t) = \frac{\lambda_j}{2^{(1+\lfloor \log_2((t/\Delta_j+2)/3) \rfloor)}}$.

Catching and Selective Catching

In Catching, each video is divided into K_j segments, and each segment is broadcast periodically on a dedicated channel. Thus, the access frequency for any point in a segment is

equal to the inverse of its length. Catching also uses patch streams to service requests immediately. Patches produce additional accesses to the data points of the first segment. The request frequency due to patches decreases linearly up to the last point in the first segment. Consequently, the access distribution is given by

$$F_j(t) = \begin{cases} \lambda_j + \frac{h(K_j)}{D_j} - \lambda_j \frac{h(K_j)}{D_j} t & 0 \le t \le \frac{D_j}{h(K_j)}, \\ \frac{h(K_j)}{f(n) D_j} & \frac{D_j}{h(K_j)} < t \le D_j, \end{cases} \quad (3)$$

where $h(m) = \Sigma_{i=1}^{m} f(i)$, $f(i)$ is the i_{th}-segment's relative size compared to the first segment in the modified GDB partition series, and n can be found using the inverse of function $h(m)$ as $n = h^{-1}(\frac{h(K_j)}{D_j} t)$. For Selective Catching, Equation (3) can be used for hot videos while Equation (1) can be used for cold videos.

Periodic Broadcasting

The access distribution function $F_j(t)$ is very simple in the case of periodic broadcasting. For each video segment, it is the inverse of the segment repeat time. The segment repeat time is simply the segment length with GDB, SB, and FB and the segment length divided by the segment allocated bandwidth with HB-based techniques.

3.2 Cache Allocation Model

The derived access distribution models in the previous subsection exhibit the following property: $F_j(t_1) \ge F_j(t_2)$ for all $t_1 < t_2$. If the optimal cached size for the j^{th} video is found to be S_j seconds, then we simply cache from the beginning of the video till the ${S_j}^{th}$ second. To allocate the available cache optimally among all videos under such property, we simply need to solve two equations:

$$S = \sum_{j=1}^{V} S_j \quad and \quad F_i(S_i) = F_j(S_j),$$

for all $i, j \in VideosSet$ with S_i and $S_j > 0$. The second equation means that for all videos with cached portions greater than zero, the request frequency of the last cached frame for every video must be equal (or very close).

4 Tunings of Design Parameters

In this section, we derive equations for determining optimally the values of W_r and W_t for Transition Patching. For Selective Catching, we present an approach for deciding on what videos to serve with Catching and what videos with Controlled Multicast when the server resources are limited. No approach was provided in [12] for such situations.

Transition Patching

The objective here is to derive simple equations for Wr and Wt, which minimize the number of required server channels. The number of required channels (ch_j) with any technique is given by $ch_j = \int_0^{D_j} F_j(t)dt$, where $F_j(t)$ is the request distribution for the chosen technique. For Transition Patching, $ch_j = \frac{1}{2}\lambda_j Wt_j + \frac{1}{2}\frac{Wr_j}{Wt_j} + \frac{D_j}{Wr_j} - 2\frac{Wt_j}{Wr_j} + 1$. By optimizing the number of channels with respect to Wr_j and Wt_j, and assuming $Wt_j << D_j$, we obtain:

$$Wr_{j_{opt}} = \sqrt{2Wt_j(D_j - 2Wt_j)}$$

and

$$Wt_{j_{opt}} \approx \sqrt[3]{2\Delta_j^2 D_j} / \sqrt[3]{1 - 2\sqrt[3]{2\Delta_j^2 / D_j^2}}.$$

Catching and Selective Catching

In [11] and [12], the optimal number of server channels required for Controlled Multicast/Patching and for Catching were shown to be $ch_j = \sqrt{2\lambda_j D_j + 1} - 1$ and $ch_j = K_j^* + \lambda_j FS_j/2$, respectively. Where FS_j is the first segment length of video j and K_j^* is the optimal number of broadcast channels. By comparing the required channels by both, the technique that results in smaller value is chosen for service [12]. Unfortunately, in many situations the number of server channels is not sufficient to satisfy the Catching requirement for every hot video. In that case we need to determine the subset of the hot videos to be served by Catching. We introduce the following *additional* test: video j is served by Catching if $ServerChannels \times \frac{\lambda_j}{\lambda} \ge ch_j$. The idea here is to serve by Catching, among the hot videos, only the videos whose numbers of required channels are not greater than their fair share, determined by their contributions to the total number of requests.

5 Performance Evaluation

5.1 Evaluation Methodology

We have developed a simulator for VOD that models various resource sharing techniques and captures the derived analytical models for cache management. We consider two service models: *True Video-on-Demand* (TVOD) and *Near Video-on-Demand* (NVOD). We also consider two video workloads: *mixed-video* and *hot-video*. The mixed-video workload contains videos with popularities varying from cold to hot, and the hot-video workload contains only hot videos. The two service models and the two video workloads lead to four target environments: *mixed-video workload and TVOD*, *mixed-video workload and NVOD*, *hot-video workload and TVOD*, and *hot-video workload and NVOD*. The first two environments are suitable for analyzing stream merging and composite techniques but not periodic broadcasting techniques, which are suitable only for

hot videos. By contrast, the third and the fourth environments can be used to compare all techniques.

In the TVOD model, we compare the server resources required to service all requests immediately. In the NVOD model, however, we fix the server resources and compare the *customer defection probability* (defined as the probability that customers leave the server without being serviced because of waiting times exceeding their tolerance) and the *average waiting time*. Two types of server resources are considered: *server channels* and *disk channels*. Server channels represent the requirement in network bandwidth, whereas disk channels represent the disk I/O bandwidth requirement. A channel is capable of supporting only one stream at a time.

The results for interval caching are not shown to maintain the clarity of the figures because it (combined with Batching) performs very poorly compared with stream merging techniques.

5.2 Workload Characteristics

In accordance with most prior studies, we assume that the arrival of requests follows a Poisson Process with an average arrival rate λ and that the accesses to videos follow a Zipf-like distribution with skewness parameter θ. Table 3 shows the default values of the input parameters.

We consider two models of customer waiting tolerance. For the mixed-video workload, we assume as in prior work that the tolerance follows the exponential distribution with a mean value of 2 minutes. For the hot-video workload, however, we borrow the *Time of Service Guarantee* (TSG) model [22, 24], which was used in a different context, since broadcasting techniques can provide time of service guarantees by upper limiting the waiting times by the length of the first segment. With TSG, customers who receive time of service guarantees shorter than 2 minutes will wait to be serviced, whereas the waiting tolerance of all other customers follows the exponential distribution with a mean value of 2 minutes.

For scheduling, we use *Maximum Queue Length* (MQL) [7] and *First Come First Serve* (FCFS) [7]. The results for *Maximum Factored Queue Length* (MFQL) [1] are not shown because it performs poorly in the stream merging environment (despite its relatively good performance with Batching).

5.3 Result Presentation and Analysis

5.3.1 Environment I: Mixed-Video and TVOD

This environment helps in comparing stream merging and composite techniques in terms of the number of server and disk channels required to achieve TVOD (i.e. zero defection and waiting time). Let us start by varying the arrival rate

Table 3: Default Parameter Values

Parameter	**Default Value(s) (Mixed-Video)**	**Default Value(s) (Hot-Video)**
Arrival Rate (Req/min)	30	upto 2400
Number of Videos	120	$1, 10, 20$
Video Duration (min)	120	120
Video Skewness (θ)	0.271	0.271
Scheduling Policy	MQL	MQL, FCFS
Waiting Tolerance Model	Exponential	TSG

while fixing the cache size at 5% of the total size of videos. Figures 1 and 2 plot the average value results, whereas Figure 3 plots the maximum (actual) required disk I/O bandwidth. The average results represent the average requirements over long periods, while the maximum results represent the actual required number of channels for providing TVOD for all requests. As expected, the gaps among the techniques widen as the arrival rate increases because the workload offers increasingly higher degrees of resource sharing, which are exploited to different levels by different techniques. Generally, ERMT has the lowest requirements, and Patching the highest. Whereas ERMT requires considerably less average disk bandwidth than Transition Patching, they both require comparable disk I/O bandwidth to provide TVOD all the time. Interestingly, Selective Catching does not perform relatively well. The figures also illustrate that resource sharing can achieve significant reductions in network bandwidth requirements. Compared with the simple unicast scheme, the reductions with ERMT approximately range from a factor of 3 to a factor of 15.

Let us now discuss the effectiveness of caching in reducing the disk I/O requirements and examine how various techniques benefit from caching. Figure 4 depicts the actual numbers of disk channels that must be available to provide TVOD for all requests. Whereas ERMT requires the smallest number of disk channels with no cache, the situation changes when the cache is introduced. With a cache size greater than 5% of the total size of videos, Transition Patching starts requiring fewer disk channels than ERMT. With a cache size greater than 12%, all the other techniques require fewer disk channels than ERMT. ERMT benefits the least from the cache and Patching benefits the most because of their access distributions in Subsection 3.1. In particular, as the cache size increases from 1% to 2%, ERMT reduces the average disk channel requirement by approximately 7% to 10% and the actual disk channels required to provide TVOD by 4% to 5%, compared with 11% to 16% and 12% to 16%, respectively, with Patching.

5.3.2 Environment II: Mixed-Video and NVOD

By varying the number of server or disk channels, we can use this environment to compare stream merging and composite techniques in terms of the achieved average customer defection probability and waiting times. Figures 5 and 6 plot these two metrics versus the number of server channels for various techniques. The results here demonstrate once again that ERMT performs the best and Patching the worst.

The results are different when limited resource is the number of disk channels. Figures 7 and 8 show that Transition Patching performs the best and ERMT performs the worst, when the limited resource is the disk bandwidth, in terms of both the defection probability and the waiting times, except when the cache size is very limited and less than 2% of the total size of videos. Interestingly, with no cache, Transition Patching requires about 700 disk channels while ERMT requires only 580 disk channels to provide a TVOD service. With 12% of the video data in the cache, however, Transition Patching can provide TVOD with the 400 available disk channels, whereas ERMT needs more than double that cache. These results are consistent with the results in Figure 4. The reason is that even with a small cache, Transition Patching (and the other two patching-based techniques: Patching and Selective Catching) can serve a large fraction of the requests fully from the cache. With ERMT, however, disk channels will be required by long streams and also indirectly by a large number of short streams because they will likely cause previous streams in the hierarchy to get extended. As discussed earlier, Patching benefits the most from the cache, but its stream merging ability is lower than Transition Patching, and thus its overall performance is worse.

5.3.3 Environment III: Hot-Video and TVOD

In this environment, we compare the performance of various periodic broadcasting techniques (SB, GDB, and FB) and the best performer in Environment I (ERMT). Since periodic broadcasting cannot provide a zero waiting time, we assume that a maximum waiting time below 2 seconds constitutes a TVOD service.

Figure 9 plots the average and maximum required server channels per video to provide TVOD for the different techniques. Note that unlike periodic broadcasting, the number of server channels per video with ERMT depends on the arrival rate and the number of supported videos. Whereas periodic broadcasting techniques reserve channels for each video, ERMT deals with all the resources as one pool and distributes them dynamically to the requests. While the average helps in understanding the overall system load, the maximum indicates the actual bandwidth to be reserved to achieve TVOD. The distinction between the two metrics was ignored in previous studies, where only the average results were usually reported. The results demonstrate that ERMT scales very well with the request rate even in terms of both the average and actual (maximum) required bandwidth. For a workload of 20 2-hour videos, the request rate must be constantly over 900 requests/minute for the best periodic broadcasting technique (FB) to be a better choice than ERMT. Even when that is the case, while ERMT requires more guaranteed bandwidth, it does not consume it all the time, which may enable the system to face short periods of drops in server capacity. Since the required bandwidth with ERMT is a function of the number of requests per video length (i.e., $N = \lambda D$), the breakpoint is higher than 900 requests/minute when the videos are shorter.

To highlight the effect of the difference between the average and actual required number of channels, Figure 10 plots the average and maximum customer waiting times if only the average required bandwidth is provided. Therefore, the number of provided channels varies with the request rate and with the number of videos as well. The service is obviously not TVOD, especially for the relatively low request rates since some customers had to wait few minutes for service. Since scheduling policies can impact the performance of ERMT compared with periodic broadcasting, both MQL and FCFS are examined. MQL achieves higher throughput and shorter average waiting time, whereas FCFS is fairer and can significantly reduce the maximum waiting time.

5.3.4 Environment IV: Hot-Video and NVOD

In this environment, we can also compare the performance of various periodic broadcasting techniques with ERMT. Figures 11 and 12 compare various techniques in terms of the defection probability and average waiting time. The results demonstrate that broadcasting techniques can achieve much lower defection probabilities than ERMT, especially when the available server bandwidth is not very limited. With only 8 server channels per video, all broadcasting techniques cause no defections, whereas ERMT causes more than 20% defections. However, ERMT achieves shorter average waiting time for those clients who were serviced, partially due to the high defection percentage.

Finally, let us compare the techniques with Cautious Harmonic Broadcasting (CHB). Despite its high effectiveness in reducing server bandwidth, CHB requires high client bandwidth. The required number of client channels is equal to the number server channels allocated for the video. In contrast, FB, SB, GDB, and ERMT always require only two channels. Moreover, CHB partitions each supported video into a very large number of segments.

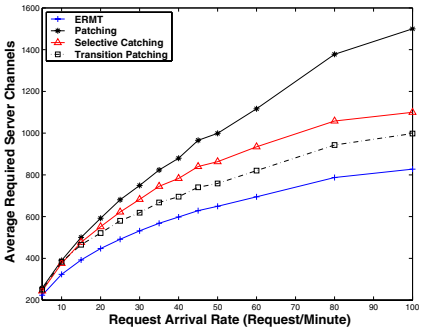

Figure 1: Effect of Request Rate on Average Server Channels [Env I]

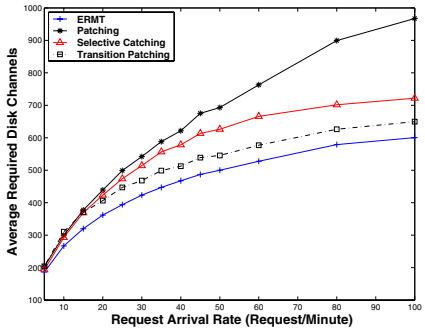

Figure 2: Effect of Request Rate on Average Disk Channels [Env I]

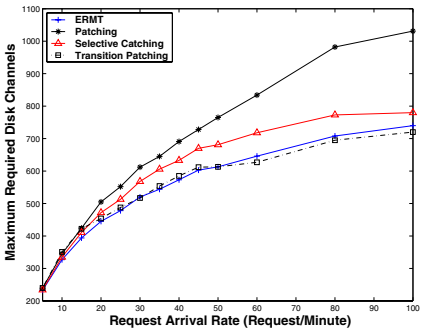

Figure 3: Effect of Request Rate on Maximum Disk Channels [Env I]

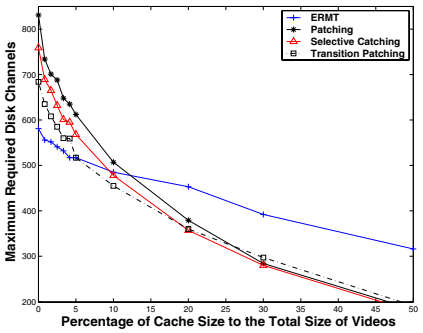

Figure 4: Effect of Cache Size on Maximum Disk Channels [Env I]

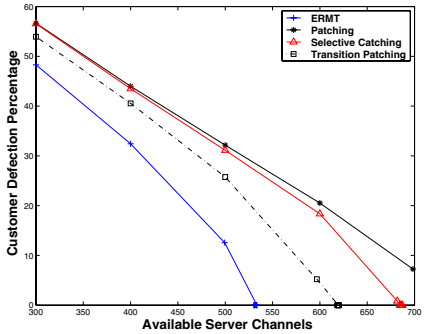

Figure 5: Effect of Server Channels on Defection Probability [Env II]

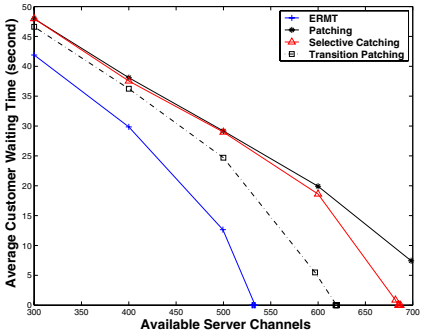

Figure 6: Effect of Server Channels on Average Waiting Time [Env II]

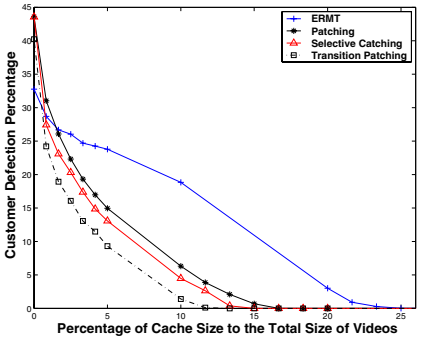

Figure 7: Effect of Cache Size on Defection Probability [Env II, 400 Disk Channels]

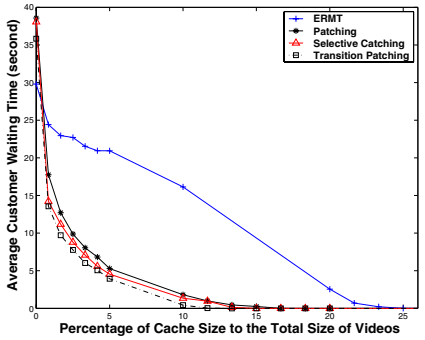

Figure 8: Effect of Cache Size on Average Waiting Time [Env II, 400 Disk Channels]

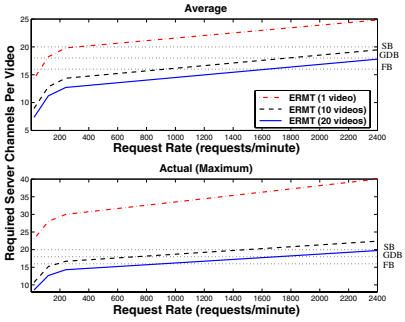

Figure 9: Effect of Request Rate on Required Server Channels [Env III]

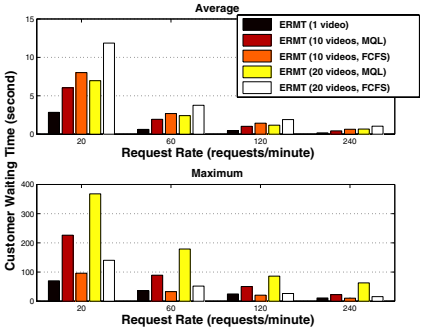

Figure 10: Effect of Request Rate on Waiting Time [Env III, Average Required Channels Provided]

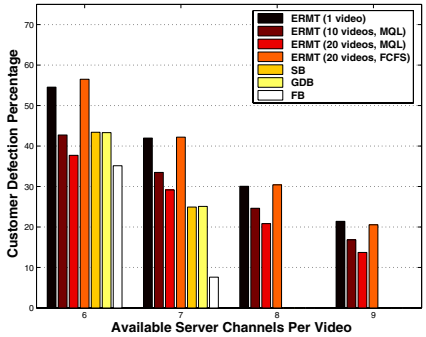

Figure 11: Effect of Server Channels per Video on Defection Probability [Env IV, TSG, 240 requests/min]

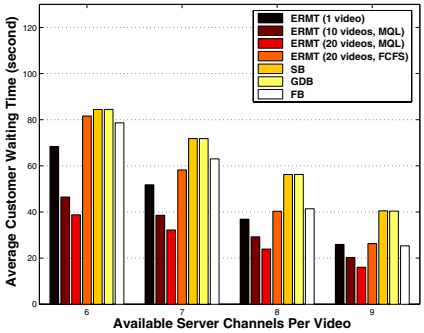

Figure 12: Effect of Server Channels per Video on Average Waiting Time [Env IV, TSG, 240 requests/min]

6 Conclusions

This study has provided three main contributions. First, we have conducted a detailed analysis of major resource sharing techniques, spanning different classes, and have developed analytical models for optimal tuning of design parameters for some techniques. Second, we have proposed a statistical approach for cache management and have derived analytical models for optimal cache allocation to reduce the demands on the disk I/O when various resource sharing techniques are used. Third, we have introduced the load on the underlying storage subsystem as an additional dimension to the comparisons among various techniques and have examined the impact of caching on reducing these requirements. The main results can be summarized as follows: (1) The proposed statistical cache management approach is very effective in reducing the disk I/O bandwidth requirements further: with a cache size of only 2% of the total size of videos, these requirements can be reduced by up to 16% for systems with moderate request rates. The reduction can be much higher for systems with high request rates. (2) ERMT is the best candidate for a TVOD service model. (3) In a NVOD service model, ERMT can lose that lead to Transition Patching when the limited resource is the disk I/O bandwidth and to FB when the server capacity is very limited or the number of videos in the workload is very small and all videos are extremely popular.

References

[1] C. C. Aggarwal, J. L. Wolf, and P. S. Yu. The maximum factor queue length batching scheme for Video-on-Demand systems. *IEEE Trans. on Computers*, 50(2):97–110, Feb. 2001.

[2] A. Bar-Noy, G. Goshi, R. Ladner, and K. Tam. Comparison of stream merging algorithms for Media-on-Demand. *Multimedia Systems Journal*, 9:211–223, 2004.

[3] M. Bradshaw, B. Wang, S. Sen, L. Gao, J. Kurose, P. Shenoy, and D. Towsley. Periodic broadcast and patching services: Implementation, measurement and analysis in an Internet streaming media testbed. In *Proc. of ACM Multimedia*, pages 280–290, Oct. 2001.

[4] Y. Cai and K. A. Hua. An efficient bandwidth-sharing technique for true video on demand systems. In *Proc. of ACM Multimedia*, pages 211–214, Oct. 1999.

[5] S. W. Carter and D. D. E. Long. Improving Video-on-Demand server efficiency through stream tapping. In *the International Conference on Computer Communication and Networks (ICCCN)*, pages 200–207, Sept. 1997.

[6] A. Dan, D. M. Dias, R. Mukherjee, D. Sitaram, and R. Tewari. Buffering and caching in large-scale video servers. In *Digest of Papers. IEEE Int'l Computer Conf.*, pages 217–225, March 1995.

[7] A. Dan, D. Sitaram, and P. Shahabuddin. Scheduling policies for an on-demand video server with batching. In *Proc. of ACM Multimedia*, pages 391–398, Oct. 1994.

[8] D. L. Eager, M. K. Vernon, and J. Zahorjan. Optimal and efficient merging schedules for Video-on-Demand servers. In *Proc. of ACM Multimedia*, pages 199–202, Oct. 1999.

[9] D. L. Eager, M. K. Vernon, and J. Zahorjan. Minimizing bandwidth requirements for on-demand data delivery. *IEEE Trans. on Knowledge and Data Engineering*, 13(5):742–757, Sept. 2001.

[10] L. Gao, J. Kurose, and D. Towsley. Efficient schemes for broadcasting popular videos. In *Proc. of the Int'l Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV)*, July 1998.

[11] L. Gao and D. Towsley. Supplying instantaneous Video-on-Demand services using controlled multicast. In *Proc. of IEEE Multimedia Computing and Systems*, pages 117–121, June 1999.

[12] L. Gao, Z.-L. Zhang, and D. F. Towsley. Catching and selective catching: efficient latency reduction techniques for delivering continuous multimedia streams. In *Proc. of ACM Multimedia*, pages 203–206, Oct. 1999.

[13] J. Gray and P. Shenoy. Rules of thumb in data engineering. In *Proc. of the IEEE International Conference on Data Engineering*, Feb 2000.

[14] A. Hu. Video-on-Demand broadcasting protocols: A comprehensive study. In *Proc. of IEEE INFOCOM*, April 2001.

[15] K. A. Hua, Y. Cai, and S. Sheu. Patching: A multicast technique for true Video-on-Demand services. In *Proc. of ACM Multimedia*, pages 191–200, 1998.

[16] K. A. Hua and S. Sheu. Skyscraper broadcasting: A new broadcasting scheme for metropolitan Video-on-Demand system. In *Proc. of ACM SIGCOMM*, pages 89–100, Sept. 1997.

[17] L. Juhn and L. Tseng. Harmonic broadcasting for Video-on-Demand service. *IEEE Trans. on Broadcasting*, 43(3):268–271, Sept. 1997.

[18] J. Liu and J. Xu. Proxy caching for media streaming over the Internet. *IEEE Communications Magazine*, 42(8):88–94, Aug. 2004.

[19] J.-F. Pâris, S. W. Carter, and D. D. E. Long. Efficient broadcasting protocols for video on demand. In *Proc. of the Int'l Symp. on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 127–132, July 1998.

[20] M. Rocha, M. Maia, I. Cunha, J. Almeida, and S. Campos. Scalable media streaming to interactive users. In *Proc. of ACM Multimedia*, pages 966–975, Nov. 2005.

[21] N. J. Sarhan and C. R. Das. Caching and scheduling in NAD-based multimedia servers. *IEEE Trans. on Parallel and Distributed Systems*, 15(10):921–933, Oct. 2004.

[22] N. J. Sarhan and C. R. Das. A new class of scheduling policies for providing time of service guarantees in Video-On-Demand servers. In *Proc. of the 7th IFIP/IEEE Int'l Conf. on Management of Multimedia Networks and Services*, pages 127–139, Oct. 2004.

[23] Y.-C. Tseng, M.-H. Yang, and C.-H. Chang. A recursive frequency-splitting scheme for broadcasting hot videos in VOD service. *IEEE Transactions on Communications*, 50(8):1348–1355, Aug. 2002.

[24] A. Tsiolis and M. K. Vernon. Group-guaranteed channel capacity in multimedia storage servers. In *Proc. of ACM SIGMETRICS*, pages 285–297, 1997.