Comparative Study of Scalable Batching Policies in Disk-Array-Based Deterministic Video-on-Demand Servers *

Emmanuel L. Abram-Profeta
World Opponent Network
Bellevue, Washington 98007
Emmanuel.Abram@won.net

Kang G. Shin
Real-Time Computing Laboratory
EECS Department, University of Michigan
Ann Arbor, Michigan 48109–2122
kgshin@eecs.umich.edu

Abstract

Presented is a comparative study of various batching policies in deterministic video-on-demand (DVoD) servers under realistic conditions including customers' behavior, storage organization, and time variations in request arrival rates. DVoD servers provide video playback without any interaction support, and customers' QoS is simply expressed in terms of average waiting time to receive service and defection rate caused by long waits. We focus on disk-array-based DVoD servers, whose movie selection and channel capacity are determined by cost and the underlying striping scheme. We analyze the well-known lack of scalability of on-demand batching, and introduce a methodology to measure scalability. We then compare various scalable batching policies and show that even the simplest batching policy may vastly improve on-demand channel allocation. Finally, we study the feasibility of different storage organizations in which disks are partitioned into several clusters to reduce service disruptions during their reconfiguration. We also show that high service availability and scalable batching can be achieved in a cost-effective way.

1. Introduction

An immediate improvement over the traditional pay-per-view service offered by broadcast cable companies is to provide user access to VCR playback of rental programs. We call such a service "deterministic" video-on-demand (DVoD), because movies are of constant length, and hence, customers receive deterministic service. Similarly to the current cable TV, DVoD systems will experience variations in the request rate for videos on various timescales, for instance, on a daily or weekly basis. The ability to accommodate these possibly significant workload fluctuations and overloading situations, or *scalability*, is therefore highly desirable, given that economical viability of DVoD ultimately depends on customers' appreciation of the service delivered.

In this paper we consider scalability in customers' *admission* QoS, which, in DVoD servers, is simply expressed in terms of the waiting time to receive service (or *latency*) and the customers' *defection rate* due to long waits. Clearly, customers' admission QoS depends on (1) the number of concurrent channels supported by a DVoD server for isochronous delivery of video data, and shared among different movie titles; and (2) the DVoD server operation during the service scheduling phase. In the latter case, one possible way to improve scalability is to delay the DVoD server's response to requests *within customers' tolerance* and "batch" those requests made by many different viewers for the same movie within a *batching period* [4]. This method may potentially increase *throughput*, or average number of service requests granted per program, by serving multiple customers with a single channel (the throughput is inversely proportional to the defection rate). Furthermore, customers' average wait time may also be reduced since requests for the same movie that arrived within the same batching period do not have to compete for resources. Thus, the choice of a batching policy plays an important role in determining both customers' wait time and defection rate.

The simplest and most popular batching policy, on-demand batching, batches pending requests *whenever a channel becomes available*. However, lack of regulation in the channel allocation process causes a considerable scalability problem during extended periods of demand surge, thus leading to *congestion cycles*. The main intent of this paper is to comparatively evaluate *scalable* batching heuristics that a DVoD server can use in order to grant cus-

*The work reported in this paper was supported in part by the National Science Foundation under Grant MIP-9203895 and the Office of Naval Research under Grant N00014-94-1-0229.

tomers' requests with maximum QoS while achieving a high throughput. For this purpose, we will consider various factors for their possible impact on batching performance, including (1) non-static request arrival rates (referred to in the remainder of this paper as *nonstationarity*); (2) storage organization of the DVoD server which may consist of one or several disk arrays for high storage throughput; and (3) customers' willingness to wait (thus delaying their requests for batching), that is, *customers' patience behavior*.

The paper is organized as follows. Section 2 gives background on disk arrays and on the relationship among movie selection, channel capacity, and cost in a DVoD environment. Next, we identify the congestion problems associated with on-demand batching. We then review and compare several well-known scalable batching policies assuming infinitely or partially patient customers. We examine next the loss of customers' QoS that enables disk arrays to be clustered to provide customers with higher service availability. The paper concludes with some remarks in Section 7.

2. Disk-Array-Based VoD Servers

In order to support large-scale deployment at acceptable cost, a hierarchical VoD server architecture is recognized to be preferable [8], in which service providers manage their own set of video materials, disseminated from one or more archives to a set of local front-end video servers via a delivery network. Customers typically interact with local VoD servers via an access network, which may be based on different technologies. One of the most promising, hybrid fiber-coax (HFC), will eventually allow the existing cable plant to be upgraded to digital transmission and serve 200- to 1000-household neighborhoods with up to 500 digital channels, in addition to the existing capacity of 65 analog channels [6].

Recent advances in storage technology make it possible to completely share the channel capacity among movie titles, by using either fine-grained striping (FGS) or coarse-grained striping (CGS) of videos across disk arrays [10]. The first graph of Fig. 1 plots the relationship between maximum movie selection and the channel capacity supported by each striping scheme, and by the one-movie-per-disk scheme (OMPD) for the disk parameters corresponding to a commercially-available disk [9][1]. We considered *minimal-cost* configurations for constant bit rate (CBR) media data, determined using the *analytical* techniques elaborated on in [10]. These techniques rely on the calculation of the optimal retrieval cycle based on (1) the latency in accessing data on disk, which is the sum of seek, rotational, and settle latencies;[2] (2) round-robin placement of consecutive stripe

[1]For simplicity, the curves corresponding to each scheme are approximated from [2], but this does not affect our conclusion.

[2]Parameters usually provided by the manufacturer (cf.[9]).

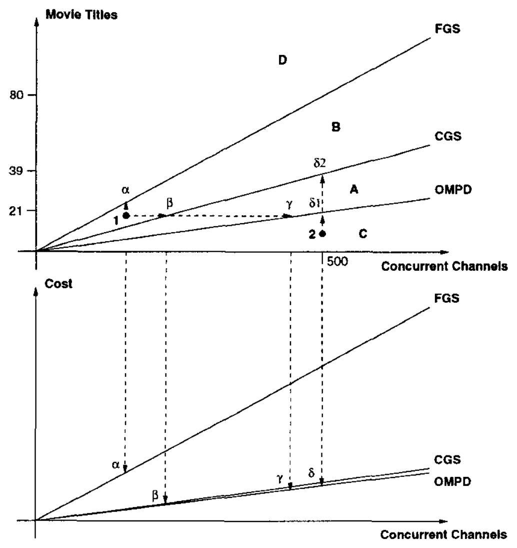

Figure 1. Choice of a storage organization.

units; (3) C-SCAN disk scheduling; and (4) number of disks and amount of RAM needed for hiccup-free continuous display. We assumed that all movies are constant length (of 100 minutes) and stored in the MPEG-1 format. As proved in [10], the retrieval cycle is much lower in CGS than in FGS, thus incurring a lower cost. This is illustrated in the second graph of Fig. 1).

Ideally, the storage organization of a DVoD server is chosen so a large collection of movie titles and concurrent channels are made available with the highest QoS at the lowest possible cost. In reality, a feasibility region must be determined to satisfy both constraints as closely as possible. This issue is made clearer in the first graph of Fig. 1. In constraint **1**, CGS provides extra storage space for new video materials or video replication. On the other hand, storing the required number of movies with FGS or OMPD is possible only if more channels than initially required are provided. As can be seen in the second graph of Fig. 1, cost ultimately determines the choice of a system. In fact, regardless of whether the system is disk-arm bound (areas **A** and **C**) or disk-space bound (areas **B** and **D**), CGS can be shown to be the most advantageous scheme since (1) the difference in cost between FGS and CGS increases almost twice faster than the difference in number of movie titles supported; (2) at approximately identical costs and channel capacity, CGS can support more movies than OMPD.

Note that constraints located in areas **B** and **D** are not economically rewarding since they correspond to vastly under-utilized storage-limited systems which provide a large selection of movies to a small user population. This type of constraint is typical of movie archives, for which tertiary storage technology may be better suited. Inversely, it is economically more viable for companies to design front-end servers of type **A** or **C** and serve the most frequently-requested movies to a subset of the total viewer population.

3. On-Demand Batching in DVoD Servers

In disk-array-based DVoD servers, new customers are served by transferring their requests from a request list to a service list according to a *scheduling policy*. It is shown in [4] that maximum factored queue length (MFQL) scheduling, which schedules the video with the largest factored queue length[3] ensures minimum waiting time for infinitely (in some cases, partially) patient customers, while LCFS is better suited to very impatient customers. Once a title has been selected, "on-demand" batching, services outstanding requests together whenever channels become available, without any control over the channel allocation process. Most movies are known to have playback lengths of about $90 - 100$ minutes, usually well-approximated by uniform distributions. Due to lack of variability, a large fraction of the channel capacity may be released in a short time, and then immediately reallocated. In case of sustained high request loads, such phenomenon, referred to as *channel clumping*, causes the DVoD server to remain unavailable for most of the time, and both customers' QoS and server throughput will degrade. This phenomenon is exacerbated by (1) the heavy-tailed Zipf-like localized access to these movies, and (2) the disparity between the number of individual movie titles usually held in most storage organizations and the much higher concurrent channel capacity allocated to these movies. This sensitivity of customers' QoS to channel clumping may also be found in more general "greedy policies" which allocate channels on an on-demand basis. An example of such policies is the *forced wait* policy in [5], in which the request at the head of the request queue is required to wait some pre-determined minimum amount of time before being serviced.

3.1. Congestion Cycles

Channel clumping occurs at unpredictable times, and then disrupts customers' QoS for extended periods of time. To illustrate this point, we consider a CGS disk-array that provides access to N different movie titles whose length is either constant, or uniformly distributed between 80 and 120 minutes, with an average length of $L = 100$ minutes in both cases. We assume that the request arrival rate is Poisson with parameter λ. With a channel capacity of s channels, the traffic intensity is given by $\rho = \frac{\lambda}{s\mu}$. Movie popularity can be defined as a vector of access probability $\bar{p} = [p_1, p_2, \cdots, p_N]$ such that $\sum_{m=1}^{N} p_m = 1$ [8]. We adopt Zipf's law [4] as the movie-popularity model.

[3]the queue length for a video divided by the square root of its popularity

Note that from a customer point of view, the average movie length can be defined as $L_c = \frac{\sum_{i=1}^{N} L_i}{N}$, whereas from the DVoD server point of view, the average duration of a channel is not known a priori (it is $L_s = \sum_{i=1}^{N} p_i L_i$ if channels are fairly shared among titles).

General arbitrary models of nonstationarity have been proposed in [5]. However, to the best of our knowledge, there are no published realistic models of customers generating non-static requests in a real VoD system. Hence, we chose a model which would, ideally, reproduce a realistic *dynamic* demand on the DVoD server while displaying the shortcomings of on-demand batching. Commonly observed variations in the request rate for TV programs alternate between "prime time" (e.g., circa 8 p.m.) and "off-hours" (e.g., early morning). We thus considered an arrival model, denoted by $(\bar{\rho} = [\rho_h, \rho_m, \rho_l], RP, T)$, of period T, alternating between plateaus of high and medium traffic intensity (ρ_h and rho_m), interleaved with plateaus of traffic intensity randomly distributed between 0 and ρ_l. Variations between consecutive plateaus are linear, and the plateau width is randomly distributed between $RP \cdot \frac{T}{4}$ and $\frac{T}{4}$. With very few parameter adjustments, a wide array of nonstationary demand patterns can be generated as successions of peak periods with uneven durations and amplitudes and continuous variations in between. This model also captures *continuous* customers' cyclic behavior. Plateaus of low traffic intensity, during which few channels are requested, make the system prone to channel clumping and congestion cycles.

Fig. 2 represents, over a one-week period, a $([10.0, 5.0, 1.0], 0.8, 24)$ workload with a uniform plateau distribution. For readability, the averaging period was set to 10 minutes. We also assumed that at $t = 0$, all channels are free, and only allocated one after another at regularly spaced intervals of duration $\frac{L_c}{s}$ over the first period of length L_c. Clearly, this initial allocation pattern is not enough to prevent congestion cycles from building up after extended period of pseudo-stability (similar observations were made for a $([20.0, 10.0, 1.0], 0.8, 24)$ workload). These cycles span over several periods, then fade away before starting again. As documented in [5] on the timescale of a burst (e.g.1 hour), channel clumping during high loads also causes wide oscillations in latency, which occur with the same intensity during both peak and light load periods. More importantly however, the *cyclic long-range* behavior is also independent of the traffic intensity during plateaus.

3.2. Burst Absorption Capability Analysis

Ideally, a scalable batching policy should always prevent the degradation of customers' QoS and server throughput. Performance should thus be as *self-similar* as possible, that is, remain approximately constant and acceptable on various timescales, regardless of the duration and amplitude of

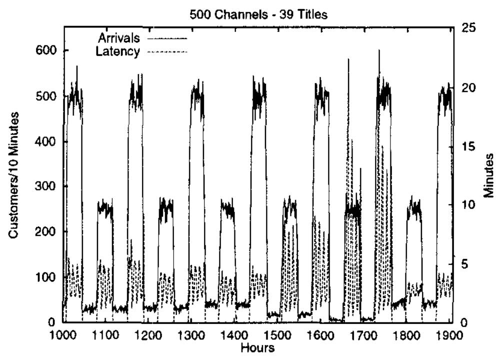

Figure 2. Congestion cycles in on-demand batching for $([10.0, 5.0, 1.0], 0.8, 24)$.

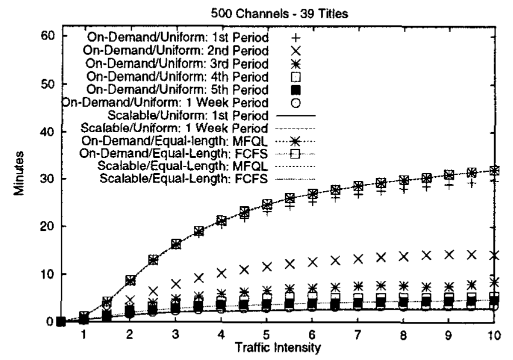

Figure 3. Burst absorption capability of on-demand batching.

high load periods. This property guarantees a congestion-free batching policy, devoid of short- and long-term oscillation cycles. Our methodology to evaluate scalability consists of studying the *burst absorption capability* (BAC), which is defined as the time it takes for an initially-empty DVoD server to recover from a sudden burst of requests at an arbitrary rate ρ, and of infinite duration. In practice, the BAC is measured by comparing customers' QoS averaged over successive periods of time.

As an example, we considered in Fig. 3 a DVoD server with 39 videos of equal or uniformly-distributed durations, in a CGS disk array of channel capacity $s = 500$. We evaluated the BAC by measuring the admission latency averaged over successive periods of duration L_c each. In the case of on-demand batching of equal-length movies, customers' latency is always unacceptably high, regardless of the averaging period, and increases with ρ during the burst. (It is also interesting to note that although MFQL is superior in theory, this scheduling policy only yields a marginal improvement as compared to FCFS in the case of equal-length movies.) With uniformly-distributed movie lengths on the other hand, the variability in movie lengths allows the system to recover from the burst in five periods or approximately 8 hours. Lastly, when channels are sourced at fixed-length intervals of duration $\frac{L_c}{s}$ each, the latency remains low, regardless of the averaging period and ρ. Clearly, the BAC study showed the deficiencies of on-demand batching and identified a scalable batching policy. This policy will be elaborated on in the next section.

4. Scalable Batching in Disk Arrays

4.1. Near Video-on-Demand

Batching in NVoD is to determine a program *schedule* according to which the same material is sourced at almost equally-spaced intervals, called *phase offsets*, if requests were placed during the last $\frac{L}{s_m}$ time units. Given an arbitrary number of channels s, a schedule corresponds to a partition $[s_1, \cdots, s_N], \sum_{m=1}^{N} s_m = s$ such that the phase offsets are given by $[\frac{L_1}{s_1}, \cdots, \frac{L_N}{s_N}]$. A variation of NVoD consists in completely sharing the channel capacity s among all movie titles and sourcing scheduled video programs every $\frac{b=\bar{L}}{s}$ units of time, where $\bar{L}$ is the average length of a program. Such a system is said to be *heterogeneous* NVoD (NVoD-h). In the case of equal-length movies, $\bar{L} = L$. In the case of uniformly-distributed movie lengths, $\bar{L}$ is calculated such that b corresponds to the average length of a busy period, that is, the average time interval between the availability of two consecutive free channels, when the VoD server is congested.

4.2. Rate Control

Rate control (RC) [5] is an ad-hoc technique to reduce congestion. RC was shown to perform better than more general "greedy"(e.g., on-demand) batching policies. The basic idea behind RC is to allocate channels as uniformly as possible by specifying a control interval of T_{int} minutes during which the total number of channels allocated is upper-bounded by $s_{max} = \lfloor \frac{T_{int} \cdot s}{L} \rfloor$. Then, channel allocation is controlled by enforcing a minimum interval Δ from one channel allocation to the next, which is computed at run-time as $\Delta = \frac{t_{left}}{s_{max} - s_{allocated}}$, where t_{left} is the time left until the next control interval, and $s_{allocated}$ the number of channels already allocated. Ideally, because s_{max} is an integer, T_{int} should be chosen in multiples of $\frac{L}{s}$ for optimal results. In a DVoD system with equal-length movies, if T_{int} is not chosen in multiples of $\frac{L}{s}$, channels are allocated at a faster rate, thus resulting in moderate congestion and higher latency.

$T_{int} = \frac{L}{s}$ corresponds to NVoD-h. One potential advantage of RC over NVoD-h is that if T_{int} corresponds to

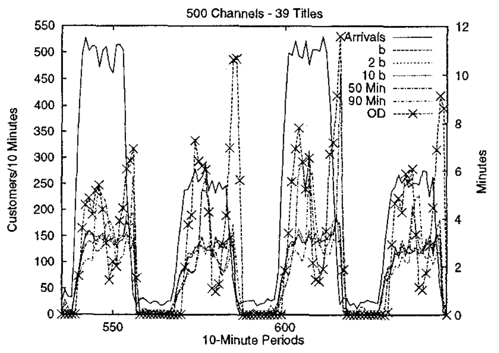

Figure 4. Choice of a control interval.

several busy periods, channels can be allocated on-demand, thus without control, during very small bursts in a non-saturated system, and customers are not forced to wait unnecessarily. The choice of an optimum control interval thus depends on the expected level of nonstationarity. This feature is, however, of limited use since T_{int} should actually be chosen as small as possible when peak periods are of unpredictable duration and high intensity, in order to provide a short response time before saturation. Depending on the accuracy of $\bar{L}$, T_{int} should thus be equal to a few estimated busy periods. This is illustrated in Fig. 4 for a $([10.0, 5.0, 1.0], 0.8, 10)$ workload and uniformly-distributed movie lengths: RC degenerates to on-demand batching when T_{int} is chosen beyond a certain limit. The reason for this is that peak periods may happen right before the end of an interval of duration T_{int}, during which very few channels have been allocated prior to the sudden burst of requests. $T_{int} = n\frac{L_c}{s}, n = 2, 10$, on the other hand, provides results similar to NVoD-h ($n = 1$).

4.3. Quasi Video-on-Demand

An attractive alternative to NVoD and NVoD-h is to allocate channels based on a threshold on the number of pending requests. Such class of VoD systems is called *Quasi-VoD* (QVoD) and regulates channel allocation according to the underlying traffic intensity. NVoD can be upgraded to a QVoD system (called *"QVoD-enhanced" NVoD*, or QNVoD), by (1) partitioning the channel capacity s into $[s_1, \cdots, s_N]$ as in NVoD; and (2) defining a vector of optimal thresholds $\overline{K}^{opt} = [K_1^{opt}, \cdots, K_N^{opt}]$. It is shown in [3] that QNVoD may yield a dramatic improvement over NVoD in terms of customers' latency and throughput (in case of partially-patient customers). If the channel capacity s is completely shared by all titles, channels may source scheduled movies only as soon as a certain threshold on the number of pending requests has been reached, and regardless of the titles requested. Such a system is said to be *heterogeneous* QVoD (QVoD-h). Note that on-demand batching is a special case of QVoD-h with $K = 1$.

Determining the best thresholds is usually intractable as they vary with the system operating point (e.g., traffic intensity, customers' patience). When customers are partially patient, the choice of an optimal threshold depends on whether customers' latency or defection rate is more valued by the DVoD server. In general, however, servicing a large number of customers is more important than providing a low-latency service at the cost of a high defection rate. It is therefore reasonable to choose a threshold that minimizes the defection rate. This issue is elaborated on [2], in which it is shown that K^{opt} varies *linearly* with the traffic intensity for both (1) partially patient customers (assuming the patience model presented in the next section), and (2) infinitely patient customers. In practice, we can dynamically adjust K to K_{opt} corresponding to the instantaneous arrival rate of requests.

In summary, the various scalable batching policies can be classified in terms of complexity, from QNVoD and QVoD-h which require significant tuning, to NVoD-h which requires no tuning at all. To a lesser extent, NVoD also requires very little adjustment considering that the various partitioning heuristics presented in [3] run in linear time.

5. Performance Comparison

Here, we consider partially-patient customers, who may leave the system after waiting a certain period of time. For a VoD service provider, the patience factor is of particular importance since the server throughput will determine the cost per stream of a configuration, and therefore economical viability. In our patience model, the customer will agree to wait τ time units or more with probability $e^{-\frac{\tau}{\bar{\tau}}}$, where $\bar{\tau}$ is the average time customers had agreed to wait. We define the *relative patience factor* as $\beta = \frac{\bar{\tau}s}{\bar{L}}$. This metric which indicates customers' average willingness to wait for service *relative to the server average busy period*, is representative of customers' patience relative to the VoD server channel capacity, since defections only occur during busy periods.

In addition to NVoD-h, QVoD-h, and on-demand batching, we considered the following NVoD and QNVoD systems:

1. An NVoD server with $[s_1, \cdots, s_N]$ minimizing the average NVoD phase offset; this configuration, called *NVoD EW-OPT*, also yields minimum latency in the case of infinite patience, by allocating channels more evenly among movie titles.
2. A QNVoD server with the same channel allocation vector $[s_1, \cdots, s_N]$ defined by NVoD EW-OPT, used in conjunction with $\overline{K}^{opt}$ which minimizes the defection rate for each movie title. This configuration is called *QNVoD EW-OPT*.

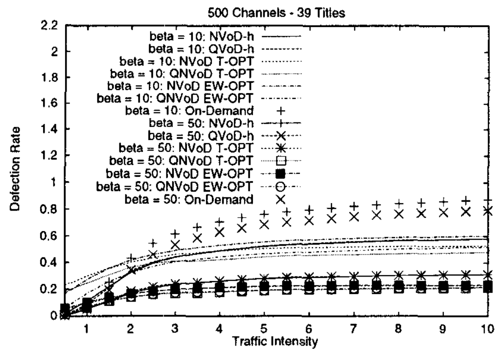

Figure 5. Defection rate with partially-patient customers for 500 channels.

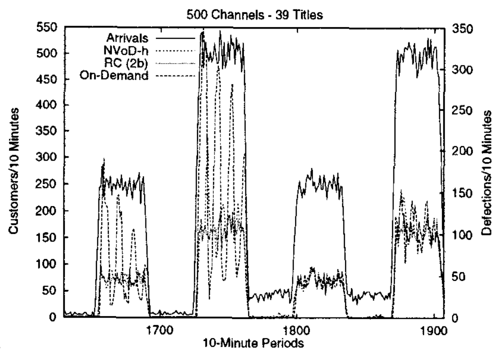

Figure 6. Defection rate in scalable batching policies for $([10.0, 5.0, 1.0], 0.8, 24)$.

3. An NVoD server with the partition $[s_1, \cdots, s_N]$ minimizing the defection rate. This configuration, called *NVoD D-OPT*, indicates the maximum throughput achievable by an NVoD server. NVoD D-OPT tends to allocate all channels to the most popular movies as the patience factor decreases.
4. A QVoD-enhanced NVoD server with the same partition $[s_1, \cdots, s_N]$ determined by NVoD D-OPT. This configuration is called *QNVoD D-OPT*.

Schedules for these policies are determined using the heuristics presented in [3]. They determine the range of defection rates and latencies achievable by NVoD and QNVoD systems with partially-patient customers. In practice, it has been shown in [3] that other partitioning heuristics such as allocating channels proportionally to the popularities, perform somewhere in between D-OPT and EW-OPT, thus achieving an acceptable tradeoff between defection rate and phase offset.

In Fig. 5 we compare the BAC in the defection rate for equal-length movies. We considered $\beta = 10$ and 50, which, for $s = 500$ and $L = 100$ minutes, correspond to customers who are willing to wait an average of 2 minutes (thus relatively impatient) and 10 minutes (thus moderately patient). Note that these results are independent of the averaging period since all policies are scalable, with the exception of on-demand batching. Although all these batching policies seem to be quite similar to each other, we actually found that QNVoD policies increasingly outperform all other systems as channel capacity and movie selection decrease, but at the cost of extensive tuning. More details about this issue may be found in [2, 3]. In the case of impatient customers ($\beta = 10$), our results indicate that scalable batching policies fail to prevent the high defection rate and wild oscillations observed in on-demand batching. This case, however, is not representative since high defection rates should be avoided at all cost. Even though it is not possible to detail here each particular case of channel capacity, movie selection, and customers' patience, all our experiments confirmed that NVoD-h with MFQL scheduling provides a remarkably simple way to achieve scalability, low latency and defection rate for realistic storage parameters. This is illustrated in Fig. 6, which represents short-term defection rate for a $([10.0, 5.0, 1.0], 0.8, 24)$ workload and $\beta = 50$ (similar observations were made for the latency).

6. Batching in Clustered Disk Arrays

6.1. Motivation

In a DVoD server, reconfigurations typically occur after disk failures, and when video material is updated. At present, the mean time to failure (MTTF) of a single disk is in the order of $600,000$ hours [11]. In local front-end servers containing less than 100 such disks, an aggregate MTTF greater than $6,000$ hours (approximately 250 days) is negligible in comparison with the much higher frequency of video material renewal, which threatens the availability of the entire array by requiring all data to be reassigned to disks. Since VoD service will compete with — and eventually replace — the existing video rental facilities, the most popular movies held by front-end VoD servers will be renewed on a weekly basis. Furthermore, insertion of advertisements or movie previews may subsidize VoD service providers on a daily basis in the same way as traditional video and pay-per-view services. In all these cases, the limitations of monolithic disk arrays are addressed by using *narrow striping* across multiple independent *clusters*, which will serve as a repository for a subset of all movie titles. Customers' QoS can now be seen as a combination of (1) scalability, and (2) *service availability* during reconfigurations. The latter property is critical in a multicast VoD server, in which, typically, a large number of customers may be affected by service disruptions.

6.2. Partitioning a Disk Array

We consider the problem of partitioning a CGS disk-array into equal-size clusters. The more general issue of unequal-size clusters is left as future work. It should be noted, however, that versatility is difficult to achieve with heterogeneous configurations, because they complicate (1) assignment of videos to clusters, and (2) video migration among clusters, both of which are needed to maintain a balanced, efficient system as movie popularities change.

6.2.1. Eligible Configurations

Let's consider a 500-channel disk array with CGS, comprising 21 2-GB disks of the type considered in Section 2 (cf.Fig. 1), and supporting a maximum of 39 MPEG-1 titles of 100 minutes each. Assuming that less than 39 titles have to be stored, partitioning these 21 disks at constant cost leads to only two *eligible configurations*: 3 clusters of 7 disks each, or 7 clusters of 3 disks each. If the storage cost is allowed to vary within a given range, corresponding, for instance, to the cost of 1 or 2 disks and the RAM needed for full utilization, more configurations are eligible and may offer a better tradeoff.

Fig. 7 represents the variations of the total cost, number of movie titles, and number of disks as a function of the number C of clusters, assuming a minimum channel capacity of 500 and constraints on the required channel capacity and movie selection located in area **A** or **C** of Fig. 1. To make the graph more readable, various constraints on the number of movie titles are also indicated: (1) 39 movie titles; (2) 30 movie titles, corresponding to constraint **A**; and (3) 10 movie titles (constraint **C**). We also indicated the maximum cost allowed, arbitrarily set to the cost of a 1-cluster system *plus* 10%.

For a fixed number of disks per cluster, disk bandwidth is increasingly under-utilized as the number of cluster grows, since each cluster is used to support less channels, and beyond a certain number of clusters, the cost constraint is violated. In the case of constraint **C**, the required movie selection is low enough to be supported by most configurations, even by those corresponding to OMPD. The set of eligible configurations is then fully determined by the cost constraint, and corresponds to configurations with $C \in \{1 \rightarrow 6, 8, 11 \rightarrow 13, 21 \rightarrow 27\}$. Constraint **A** is more restrictive since configurations that are acceptable in cost do not necessarily provide enough storage space to store 30 titles. So both cost and selection constraints have to be considered, making OMPD configurations ineligible. In the limited case of 39 movie titles, only few configurations are eligible ($C = 1, 3, 4, 5, 6, 8, 13$).

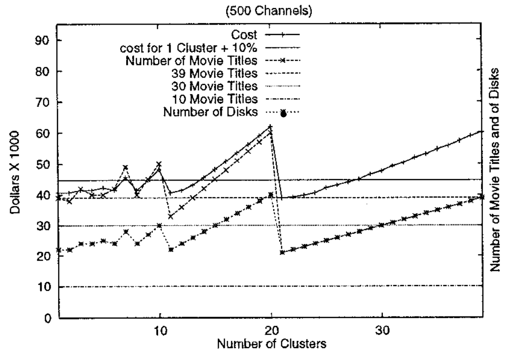

Figure 7. Storage cost, number of movies titles supported and disks for various number of clusters: constraints A and C.

6.2.2. The Video Allocation Problem

Once an eligible configuration has been chosen, movie titles have to be replicated and assigned to different clusters so that a maximum number of sessions may be supported, and customers' admission latency may be minimized. These goals are achieved in a balanced system by assigning movies to clusters so as to make requests uniformly-distributed over different clusters [8].

The total storage capacity measured in number of movie titles is compared to the constraint on movie selection. If space is available for replication (e.g., 39 slots for 30 movie titles), unused slots are allocated to videos according to some replication policy. A control server will then keep track of the load on each cluster to ensure all the copies to be accessed uniformly. Each video is assigned a *partial popularity*, which is equal to its initial popularity divided by the number of its copies. Assuming C clusters and s_C slots per cluster, we want to find an assignment from the list of all $C \cdot s_C$ instances of all movie titles and their corresponding partial popularity to the C clusters so the highest access frequency among all clusters is minimized.

Theorem: *An optimum assignment can be found in* $O((C^2 s_C^2 + C^3) \log(C \cdot s_C))$ *time.* □

The proof is presented in [2]. In summary, we show that our assignment problem is reducible to a binary search, in which each iteration involves solving a *max-flow problem* in a directed network [7], thus resulting in a polynomially-bounded complexity. It is found in [2] that the optimum assignment improves the commonly-used greedy allocation, although the greedy assignment, used in conjunction with simple duplication, yields near-optimal assignments.

6.3. Simulation Results

Service availability is measured by the proportion of customers which remain unaffected by the failure of a single cluster. Assuming that viewers of a replicated video can be shifted to an alternate cluster, availability A_v depends only on the access frequency to non-replicated videos and is given by $A_v = 1 - \frac{1}{C} \sum_{non\ replicated} p_i$. A_v is lower-bounded by $1 - \frac{1}{C}$, corresponding to the configuration with no replicated movie titles (39 titles). In constraint **A**, skewed access and moderate replication of the most popular videos in a 2- or 3-cluster configuration is sufficient to ensure $A_v \geq 85\%$. In the case of constraint **C**, a 2-cluster configuration is sufficient to provide near-maximum availability, since most or all of videos can be held within each cluster.

Availability is somewhat misleading since whenever a cluster is affected by a reconfiguration or a disk failure, the requests received by the remaining clusters will increase by a factor of up to $1 - \frac{1}{C}$. (This upper-bound may be reached under some constraints of type **C**, depending on the number of clusters). Replicated videos will thus be accessible with a higher latency, and hence, with a very few clusters, the VoD server may only benefit from such high availability for a restricted range of request rate. Excessive partitioning, on the other hand, tends to cause additional defections since customers become more impatient relatively to the average busy period in each cluster. Fortunately, in the case of moderate patience, Fig. 8 indicates that as the number of clusters increases, the additional defection rate in NVoD-h is insignificant in view of its high availability (similar results are presented in [2] for the latency). In addition, scalability ensures that only those customers requesting non-replicated movies are affected.

7. Conclusion

In this paper, we presented a systematic and realistic way of comparing various batching schemes by considering such factors as customers' behavior, storage organization, and non-static request arrival rates. We focused on disk-array-based DVoD servers, whose movie selection and channel capacity are determined by cost and the striping scheme used. We presented on-demand batching policies and identified the lack of scalability, which causes short- and long-term degradation of customers' QoS and server throughput. So, we comparatively evaluated various scalable batching policies, showing that the simplest scalable batching policy, NVoD-h, may vastly improve on-demand channel allocation. Finally, we extended the results to clustered disk arrays, and showed that high service availability and scalable batching can be achieved in a cost-effective way.

It is worth mentioning that one well known problem with batching is the support of VCR functionality. This issue is dealt with in [1, 2], in which it is shown that (1) decentralized caching of portions of videos in a buffer located in customers' premise equipment is an interesting alternative to

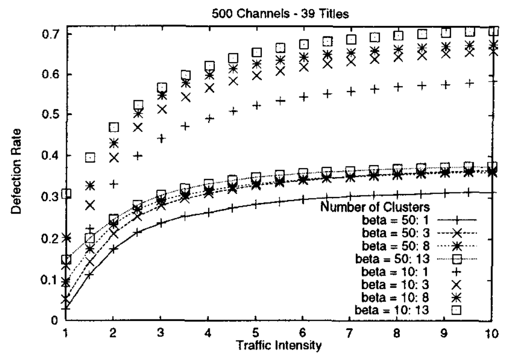

Figure 8. Defection rate in NVoD-h for $\beta = 50$.

distribute the cost of providing *unrestricted* VCR functions without compromising the VoD service scalability achieved through batching; and (2) the blocking probability of VCR actions is usually independent of the batching policy, provided that the latter is chosen among the NVoD-based systems presented in this paper.

References

[1] E. L. Abram-Profeta and K. G. Shin, "Providing Unrestricted VCR Functions in Multicast Video-on-Demand Servers," *Proc. IEEE International Conference on Multimedia Computing and Systems*, pp. 66-75, June-July 1998.

[2] E. L. Abram-Profeta, *Support for Service Scalability in Video-on-Demand End-Systems*, Ph.D. Thesis, University of Michigan, May 1998.

[3] E. L. Abram-Profeta and K. G. Shin, "Scheduling Video Programs in Near Video-on-Demand Systems," *Proc. ACM Multimedia'97*, pp. 359-369, November 1997.

[4] C. C. Aggarwal, J. L. Wolf and P. S. Yu, "On Optimal Batching Policies for Video-on-Demand Storage Servers," *Proc. ACM Multimedia'96*, pp. 253-258, 1996.

[5] K. C. Almeroth, A. Dan, D. Sitaram, and W. H. Tetzlaff, "Long Term Resource Allocation in Video Delivery Systems," *Proc. IEEE INFOCOM'97*, April 1997.

[6] B. Furht, D. Kalra, and A. A. Rodriguez, "Interactive Television Systems," *Multimedia Tools and Applications*, pp. 235-277, Kluwer Academic Publications, Inc., 1997.

[7] J. Korst, "Random Duplicated Assignment: An Alternative to Striping in Video Servers," *Proc. ACM Multimedia'97*, 1997, pp. 219-226.

[8] T. D. C. Little, D. Venkatesh, "Popularity-based assignment of movies to storage devices in a video-on-demand system," *Multimedia Systems*, Vol. 2, No. 6, pp. 280-287, 1995.

[9] *Seagate Product Overview*, October 1993.

[10] B. Özden, R. Rastogi, A. Silberschatz, "Disk Striping in Video Server Environment," *Proc. IEEE International Conference on Multimedia Computing and Systems*, pp. 580-589, 1996.

[11] R. Zimmermann, S. Ghandeharizadeh, "Continuous Display Using Heterogeneous Disk-Subsystems," *Proc. ACM Multimedia'97*, 1997, pp. 227-238.