END-TO-END REQUEST HANDLING IN DISTRIBUTED VIDEO-ON-DEMAND SYSTEMS

Padmavathi Mundur, Robert Simon and Arun Sood
Department of Computer Science
George Mason University
Fairfax, VA 22030
{pmundur,simon,asood}@cs.gmu.edu

Abstract

In this paper we provide an end-to-end analysis of a distributed Video-on-Demand system consisting of a hierarchy of server clusters and networks. We develop an analytical framework for remote service by combining two critical subsystems, the storage and the network subsystems. We use this framework to analyze different request handling policies that maximize throughput. We show that the results of our end-to-end analysis are scalable and may be used to predict system performance when the demand changes and the system size is scaled up. The relative performance order of different request handling policies remains the same, with the policy that uses retrials at more than one resource exhibiting better performance than other policies.

Keywords: *Network and Resource Management, Distributed VoD.*

1 INTRODUCTION

The focus of this paper is to conduct an end-to-end analysis that provides insight into efficient operation of a hierarchical Video-on-Demand (VoD) system. Two issues dealt in this paper are that of remote service using real-time networks and managing clusters of multiple servers for balanced load. We provide an analytical framework for remote transfers that combines a server model with a network model. Using the framework we show how to analyze request handling policies that maximize throughput resulting in an efficient use of the multiple resources in the distributed VoD system. We also provide replication and video placement schemes to balance load on clusters of video servers.

Research in real-time networks, analyzing network conditions for guaranteed services has been discussed in [6], [4]. Load balancing schemes based on replication and placement techniques are discussed in [1], [10], [2]. In [10] replication and placement algorithms are discussed at the level of disks and disk groups. The placement algorithm follows a graph-theoretic approach and their replication technique is based on apportionment methods. Our replication scheme is similar and videos are replicated in proportion to their traffic intensity. In [1], the authors use the network element characterized by the bandwidth parameter in their analysis of a distributed VoD system. They present an r-ary tree like network topology; by assuming a balanced load, their placement scheme assigns movies to each level of the r-ary tree so that their expected demand adds up to the fraction of the traffic supported by the link at that level. In [2], network bandwidth partitioning is used in their load balancing scheme. Our paper is different from [1] and [2] in that load balancing is not our primary focus. Instead, we explicitly analyze the use of a real-time network for general purpose request handling in a hierarchical VoD system.

The rest of the paper is organized as follows: We discuss VoD system architecture in Section 2. In Section 3, we present an analytical model for the servers and the networks. In Section 4, we discuss cluster management. In Section 5, global performance metrics used to evaluate policies and simulation results are discussed. We conclude the paper with Section 6.

2 VOD SYSTEM ARCHITECTURE

The hierarchical VoD architecture consists of local and remote sites. With the hierarchical organization, if a request cannot be serviced from the local site, it will be directed to other remote sites. Each local site is characterized by a cluster of video servers serving users over a local distribution network, such as an ATM LAN. We assume that there are sufficient network resources at the local cluster to deliver videos to the users. The more popular videos are stored and served from the local video cluster, and the less popular videos from the remote sites. The remote site may be

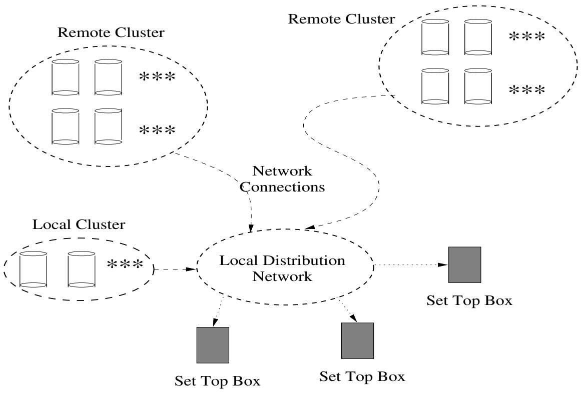

Figure 1. Hierarchical VoD Architecture

archival in nature, and can provide real-time video delivery service over high speed networks. A typical organization of local and remote clusters is shown in Figure 1. The set-top boxes at the client site provide the decoding and display functionality for the delivered video from the video servers.

In the next section, an analytical model for remote transfers is described. There we use the size of a client's set-top box buffer to relate the server model with the network model. The buffer size at the client's set-top box is the amount of data retrieved at the server and sent over the network to the client. The data is transferred to the client periodically; that is, the requests are processed in *rounds*[7]. The round length itself is determined by the duration of the data in the client's buffer played back at a known rate, called the playback rate. Starvation at the client buffer occurs if the new data does not arrive before the end of the round length.

Servicing requests from remote clusters involves data transfer over the network. In addition to guaranteed bandwidth, the network has to provide bounded delays because of the need for periodic delivery of time sensitive data from the server. This prevents starvation at the client site. Rate-based scheduling techniques are available at the network that provide a maximum delay bound. Reservations are made for video connection requests on a per *link* basis. We assume *Weighted Fair Queueing* (WFQ) scheduling at the network [6], which provides a firm per packet end-to-end delay bound on a per link and per routing path basis and ensures that all transmitted packets will be able to meet this bound.

3 ANALYTICAL MODEL

In this section, we develop a model for remote transfers by relating the round length parameter at the server and the reserved rate at the network with the buffer size at the client. The minimum rate for the network connection is found using WFQ delay bounds. Admission control tests are defined

B	Buffer size at the receiver (set-top box)
R_{pl}	Playback rate per request
R_{nw}	Overall network bandwidth
R_d	Overall disk bandwidth
R_{res}	Reserved rate at the network per request
T_{round}	Round length
n	Maximum number of requests at the server
$n_{current}$	Number of requests being serviced at the server

Table 1. Model Parameters

at the server and the network, which decide whether an arriving request will be admitted into the system or rejected. The round length is the duration of the amount of data retrieved per request at the client site. The round length, T_{round}, for the server is determined by: $T_{round} = \frac{B}{R_{pl}}$.

The number of requests, n, serviceable at the server is derived as follows. The time to transfer the amount of data retrieved per round is bounded by the round length, that is, $\frac{nB}{R_d} \leq T_{round} \leq \frac{B}{R_{pl}}$. The maximum number of requests n, that can be serviced at the server is: $n \leq \left\lfloor \frac{R_d}{R_{pl}} \right\rfloor$

An admission control test at the server determines if the arriving request will be admitted into the system. The server can service a maximum of n requests in a round. The request will be admitted into the system if the following condition holds:

$$n_{current} + 1 \leq n \tag{1}$$

For videos served directly over the network, the minimum reserved rate at the network, R_{res}, required to prevent starvation at the client site is: $R_{res} \geq \frac{B}{T_{round}} = R_{pl}$.
The overall reserved bandwidth must be less than or equal to the overall bandwidth of each link on the path, that is, $nR_{res} \leq R_{nw}$. Therefore, the bounds on reserved rate at the network, R_{res}, are: $R_{pl} \leq R_{res} \leq R_{nw}$

Suppose a flow has a guaranteed rate of R_{res} and is routed on a path that has J links. The maximum delay bound per packet for that flow under WFQ scheduling[6] is given by

$$\bar{D} = \frac{B}{R_{res}} + \frac{(J-1)M}{R_{res}} + \sum_{j=1}^{J} \frac{M_{max}}{A_j} \tag{2}$$

where B is the size of the token bucket regulator, M is the maximum size of the packet for the flow, M_{max} is the maximum size of a packet permitted in the network (MTU), and A_j is the overall bandwidth on link j.

Using Equation 2, we now derive R_{res} corresponding to $\bar{D}$. This required minimum rate is used in the admission control test before admitting a new request on the network.

$$R_{res} = \frac{B + (J-1)M}{\bar{D} - \sum_{j=1}^{J} M_{max}/A_j} \tag{3}$$

Admission control tests for a new flow must be run on a link by link basis over the routing path. Suppose that link j has currently accepted T flows $1, 2, \ldots, T$. The i^{th} flow has a reserved rate of R^i_{res} computed using Equation 3 for a given delay bound. If the *new* flow requests a rate of R^{new}_{res}, it will be admitted only if the following relation holds and the new flow will be rejected if any of the links cannot provide the reserved rate, R^{new}_{res}:

$$\sum_{i=1}^{T} R^i_{res} + R^{new}_{res} \leq A_j \tag{4}$$

4 CLUSTER MANAGEMENT

In this section, we describe cluster management techniques at the local clusters in the hierarchical VoD architecture. Each cluster consists of several video servers. The number of videos to be placed on the servers is assumed to be large. Typically, no single video server has either the storage or the request stream handling capacity to store and deliver all videos. The varying popularity of the videos results in different access patterns at the servers. Concentration of only popular or unpopular videos on a single server must be avoided to prevent load imbalance. The imbalance problem is remedied using replication and proper video placement schemes. Two greedy algorithms for replication and placement are described next. The details of the algorithm are found in [5].

Replication Algorithm

The replication algorithm computes the number of copies for videos by distributing the overall storage capacity of the cluster in proportion to their demand. The VoD system offers a video collection of V videos. Let λ_i be the arrival rate and a_i be the service time for the i^{th} video. Let d_i be the size of the i^{th} video in gigabytes(GB). Let total storage available at the cluster be $C = \sum_{j=1}^{N} C_j$ where C_j is the storage capacity of the j^{th} server in gigabytes and N is the number of servers in the cluster. The total traffic intensity from all videos is $\rho = \sum_{i=1}^{V} \rho_i$ where $\rho_i = \lambda_i a_i$. The number of copies of i^{th} video, K_i, is equal to :

$$K_i = \text{round}((C/d_i)(\rho_i/\rho))$$

The *round* function rounds K_i off to the nearest integer.

Video Placement Algorithm

The basic idea behind the placement algorithm is to match high traffic intensity videos with high request handling capacity servers. We achieve that by working with two ordered lists; one for the videos, from high to low traffic intensity and the other for the servers, from high to low request handling capacity. The videos are then matched to servers in a round-robin fashion, subject to storage constraint. Videos that have $K_i \geq 1$ are considered for placement on the local cluster. The videos with $K_i = 0$ will be served from the remote cluster. In the placement algorithm each replicated copy is treated separately. We now have $\widehat{V}$ videos, $\widehat{V} \geq V$, accounting for each replicated copy as a separate video. Each replicated copy will share the traffic intensity equally. Therefore, the traffic intensity ϕ_k for video k, $k \in \{1 \ldots \widehat{V}\}$, is given by $\phi_k = \rho_i / K_i$ where $K_i \geq 1$ and $i \in \{1 \ldots V\}$. Each video from an ordered list of videos, $1 \ldots \widehat{V}$, is mapped to a server from an ordered list of servers, $1 \ldots N$, subject to storage constraints.

5 END-TO-END ANALYSIS

This section describes request handling at the VoD system and metrics used in the performance analysis. The arriving request is routed to either the local cluster or the remote clusters. The decision to serve from local or remote is based on policy restrictions and availability constraints. A request can be serviced from the local cluster if the video is available at the cluster and if enough capacity is available at the server where the video resides. For a request serviced from the remote cluster, capacity at the server must be available and a guaranteed rate at the network as well as an acceptable bounded delivery delay must be provided. Since the local cluster is limited in storage space, the number of videos to be stored at the local cluster is limited. As a result, a large number of videos that are not frequently requested may not be stored on the local cluster. The demand for such videos is met from the remote clusters. Requests for videos not available at the local cluster are sent to remote clusters directly. A request that gets rejected by the admission control test is said to be *blocked*. The number of blocked requests forms the basis for developing the global performance metrics, *overall blocking probability* and *overall blocking rate*, described below.

The overall average request arrival rate at the VoD system is λ. Suppose that the probability of a request for i^{th} video is p_i. The arrival rate for i^{th} video is $\lambda_i = \lambda p_i$. The blocking probability, b_i for the i^{th} video is defined as the ratio of the total number of blocked requests to the total number of arrivals at the VoD system. The total number of blocked requests is the difference between total number of arrivals and the total number of requests serviced at all resources. Overall blocking probability, P_{vod} is the weighted average over all videos of the blocking probability per i^{th} video, $P_{vod} = \sum_i^V p_i b_i$.

The overall blocking rate, BR_{vod} is used as a measure of the performance of the VoD system as a whole. It indicates the number of arriving requests at the VoD system that will get blocked. The overall blocking rate is defined by: $BR_{vod} = \sum_i^V \lambda_i b_i$.

5.1 Simulation Results

The purpose of the simulation is to test the scalability of the end-to-end request handling policies for the hierarchical VoD architecture. We conducted the simulation using the CSIM simulation package [8].

The specific hierarchical architecture simulated consists of three levels: local cluster, remote cluster1 via network1, and remote cluster2 via network2. Each local cluster server is modeled after a 16 Gigabyte(GB) RAID-3 which can simultaneously serve 22 request streams of 4 megabits per second (Mbps)[9]. It is assumed that adequate storage and stream handling capacities are present at the servers in the remote clusters and that requests are not blocked. Notice that this is not a limitation of our model, but rather done to simplify the presentation of results. The two remote clusters are archival in nature and store all videos. They also provide direct real-time video delivery service to arriving requests.

Network related data such as link capacities and the maximum packet sizes are taken from [4]. The burst size of 100 kilobytes (kB) for the token bucket regulator at the servers is also the buffer size at the clients' set-top boxes. The number of hops to remote cluster1 and remote cluster2 is assumed to be 3 and 6, respectively. The network MTU and the maximum packet size are 1500 bytes. The end-to-end delay for stored video is 300 ms. The playback rate is 4 Mbps. The reserved rate at the network corresponding to the input delay is calculated using Equation 3. The reserved rate is at least as large as the playback rate.

The access patterns to each video depends on the popularity of the video. We use a Zipf distribution to characterize the probability of choosing n^{th} video from a collection of V videos [3]. The probability of choosing the n^{th} most popular video from a collection of V videos, is determined by Z/n where $Z = 1/(1 + 1/2 + 1/3 + \ldots + 1/V)$. The order in which they appear is also an indication of their popularity, the first video being the most popular and the last video the least popular. Using the Zipf distribution, if the overall arrival rate of request to the VoD system is λ, the arrival rate of i^{th} video is determined by $\lambda_i = \lambda p_i$ where $p_i = Z/i$. We use the arrival rate to determine the traffic intensity for each video. We in turn use the traffic intensity in our replication algorithm to determine the number of copies per video. The greedy placement algorithm from Section 4 is used to place the videos on the servers with balanced load as our goal. The number of videos in the video collection is 15. The size of videos is in gigabytes (GB) ranging between 1.23 to 3.66 computed using a playback rate of 4 Mbps. The service time in hours ranges from 0.68 to 2.03.

The specific request handling policies evaluated are:
Redirect - Blocked requests at local cluster are redirected to remote cluster1 and blocked requests from remote cluster1 are redirected to remote cluster2.

Split-redirect - The incoming request traffic is first split between local and remote clusters for videos common to local and remote clusters. The remote cluster traffic includes redirected traffic from local cluster. This traffic is split between remote cluster1 and remote cluster2. For example, under policy *Split-Redirect45* incoming traffic for common videos between local and remote is split in the proportion $40 - 60$, 40% to local and 60% to remote and the number 4 in the policy label refers to this proportion. All remote traffic is split between remote cluster1 and remote cluster2 in equal proportion and the number 5 in the policy label refers to this split. All blocked requests from local cluster are redirected to remote cluster1. Blocked traffic from remote cluster1 is not redirected to remote cluster2.

Split - Similar to Split-redirect, but the blocked requests at local cluster are not redirected to remote cluster1.

Two sets of experiments are conducted. The total number of hours simulated for each run is 10000. In the first set, the capacity of the local cluster is kept constant at 10 servers and the network bandwidth at 622 Mbps. The arrival process for each request is assumed to be a Poisson process with exponential interarrival times. The range of values for arrival rate, λ, is between 200 to 380 requests per hour. The purpose of this experiment is to test the performance of the request handling policies mentioned above. We also reverse the order of videos and evaluate the policies to show that they are insensitive to service times.

Second set of experiments include varying the local cluster capacity from 2 servers to 20 servers. The network bandwidth is also varied from 124 Mbps to 1244 Mbps. The arrival rates are proportionately increased or decreased as the capacity is varied. For instance, at 2 servers and 124 Mbps network bandwidth, the arrival rate is 64 requests per hour and at 20 servers and 1244 Mbps network bandwidth, the arrival rate is at 640 requests per hour. The purpose of this experiment is to show that the results of our end-to-end analysis are scalable.

5.2 Performance Evaluation

We show in the following analysis these major results:

- As system size scales up, the overall blocking probability remains the same for increasing demand, indicating that the results of end-to-end analysis are scalable.
- The relative order of performance for different request handling policies also remains the same as the VoD system size is varied; redirect policy doing strikingly better than either the split-redirect or the split policies.

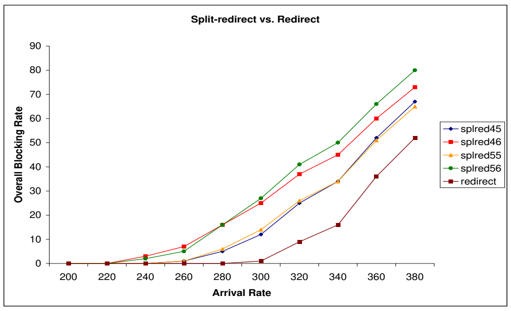

Figure 2. Blocking Rate for Policies :Redirect and Split-Redirect

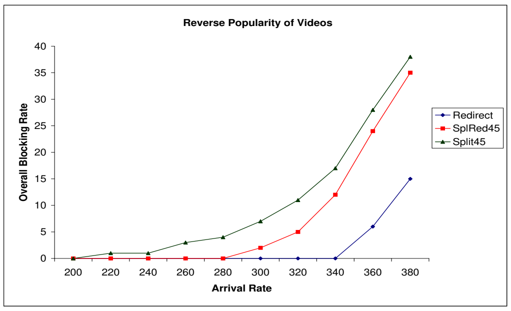

Figure 4. Blocking Rate for Reversed Popularity

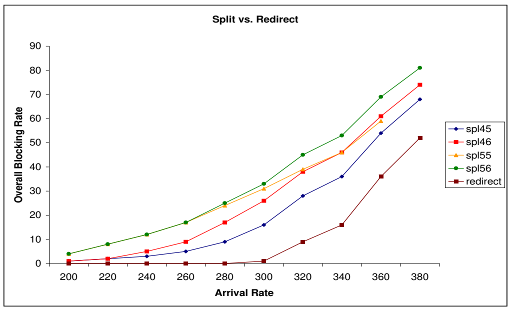

Figure 3. Blocking Rate for Policies: Redirect and Split

We also show that the hierarchical organization is appropriate for storing less popular videos and archival videos at remote sites. In addition, the remote sites act as sources of supplemental request handling capacity minimizing overall blocking. Though the three basic policies, redirect, split-redirect, and split, differ in path establishment complexity, redirect having the highest overhead, we show that the performance is strikingly better for redirect. The difficulty in finding an optimal split that minimizes overall blocking rate makes the split based policies less attractive, despite their implementation simplicity. We also show that the performance of these policies is insensitive to the holding times and the redirect policy is still a good option.

Figure 2 compares performance of redirect policy with various combinations of split-redirect policies. In Figure 3, redirect policy is compared to split policies. Redirect policy has the minimum overall blocking rate in both cases. The redirect policy gives maximum utilization at each resource. Remote service is delivered only when the local cluster cannot handle the load. Between the two remote clusters, remote cluster2 is twice as many hops away as the remote cluster1. Minimizing traffic toward remote cluster2 is valid since we will be occupying twice as many resources if we serve out of remote cluster2. Among the split policies, the 45 split shows the minimum overall blocking rate for both split-redirect and split policies. The rest of the analysis with respect to split policies is based on the 45 split.

In Figure 4, we show that redirect policy still does well when the popularity of the videos is reversed. This result also indicates that the performance exhibited by the three basic policies is insensitive to service times. The relative order of the performance of the policies is the same for any video collection and the service times of the videos.

In Figures 5 to 7, we show that the results of our end-to-end analysis are scalable. The significance of these graphs is that as the VoD system size is scaled up from two servers to 20 servers with appropriate network bandwidth increases, the overall blocking stabilizes after 12 servers and remains about the same as the capacity and the load are increased. The results show that we are able to predict system performance for each request handling policy when the demand increases and the system size is scaled up. The high blocking observed at two server capacity is because of the large number of videos not assigned to the local cluster as a result of limited storage. Only the top 6 of the 15 videos are stored on the local cluster. The demand for the rest of the videos is met at the remote clusters, overloading the network and increasing the overall blocking probability. For a local cluster of more than four server capacity, all 15 videos are stored on the local cluster.

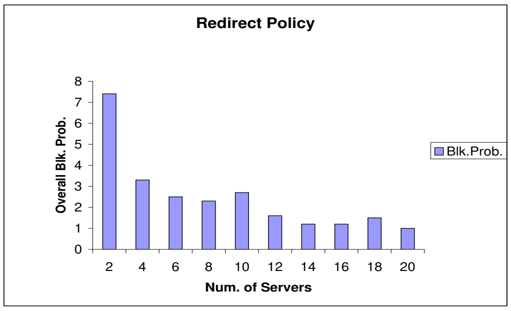

Figure 5. Overall blocking as VoD system size scales up with increase in arrival rate

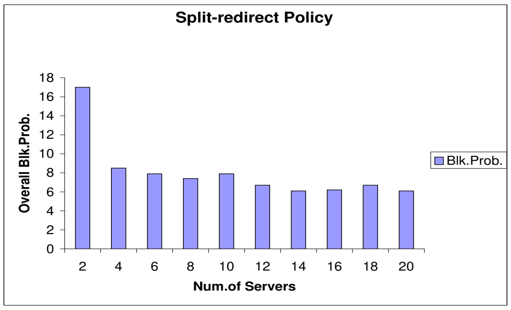

Figure 6. Overall blocking as VoD system size scales up with increase in arrival rate

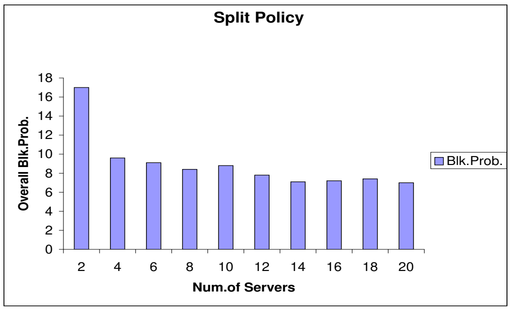

Figure 7. Overall blocking as VoD system size scales up with increase in arrival rate

6 CONCLUSION

In this paper, we have shown that the results of our end-to-end analysis are scalable. The results show that we are able to predict the system performance as demand for videos increases and the system size is scaled up. We show that relative performance order among different request handling policies remains the same independent of the video collection and the system size.

References

[1] C. Bisdikian and B. Patel. Issues on movie allocation in distributed Video-on-Demand systems. In *Proceedings of the IEEE International Conference on Communications*, pages 250–255. IEEE Press, 1995.

[2] D. W. Brubeck and L. A. Rowe. Hierarchical storage management in a distributed VOD system. *IEEE Multimedia*, pages 37–47, Fall 1996.

[3] A. Chervenak, D. Patterson, and R. Katz. Storage systems for movies-on-demand video servers. In *Proceeding of the fourteenth IEEE symposium on mass storage systems*, pages 246–256, September 1995.

[4] L. Georgiadis, R. Guerin, V. Peris, and R. Rajan. Efficient support of delay and rate guarantees in an internet. In *ACM SIGCOMM*, pages 106–116, August 1996.

[5] P. Mundur, R. Simon, and A. Sood. Issues in distributed Video-on-Demand systems: Replication, placement, and network transfers. Technical report, George Mason University, Fairfax, VA, USA, 1998.

[6] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control - the multiple node case. *IEEE/ACM Transactions on Networking*, 2(2):137–150, 1994.

[7] P. V. Rangan and H. M. Vin. Designing file systems for digital video and audio. *Operating Systems Review*, 25(5):69–79, 1991.

[8] H. Schwetman. *CSIM18 User's Manual*. Mesquite Software, Inc.

[9] Y. Wang, J. Liu, D. Du, and J. Hsieh. Efficient video file allocation schemes for Video-on-Demand services. *Multimedia Systems*, 5(5):283–296, September 1997.

[10] J. Wolf, P. Yu, and H. Shachnai. Disk load balancing for Video-on-Demand systems. *Multimedia Systems*, 5(5):358–370, September 1997.