International Journal of Computer Applications in Engineering Sciences
[VOL I, ISSUE II, JUNE 2011] [ISSN: 2231-4946]

An Efficient Video Request Scheduling Framework for Video-on-Demand Systems

Vinay. A[1], Abhinav Prakash[1], T N Anitha[2], M Bharathi[2]

[1] *Department of Information Science and Engineering, P E S Institute of Technology Bangalore, Karnataka, INDIA*

[2] *Department of Computer Science and Engineering, S J C Institute of TechnologChikkaballapura, Karnataka, INDIA*

[1]a.vinay@pes.edu

***Abstract*— Recent years have witnessed dramatic increase in computational capability and network bandwidth available to end-users in commercial, home, and mobile environments. Similarly, the field of entertainment has also seen a rapid increase in pay-per-view offerings and special-interest satellite channels. Even the educational environment has seen offerings which provide video and audio captures of classroom lectures over the Web. Such trends motivate investigations into video-on-demand (VoD) systems. The performance of such applications however is still far from satisfactory, especially during the peak hours. To reduce client-perceived access latencies as well as server/network loads, an effective strategy is to use efficient video streaming architecture with good scheduling algorithms being deployed. In this direction, the paper proposes a novel and efficient video request scheduling framework for hybrid VoD systems. The results obtained by implementing the proposed scheduling framework are significant from system Quality-of-Service (QoS) perspective. Experimental result illustrates that the servers are not congested for a very long duration, the request rejection rates and the starvation period are reduced yielding in good throughput from the system.**

***Keywords*— Video-on-Demand (VoD) systems, Scheduling, FCFS scheduling, Priority Scheduling, Gang scheduling, Quality-of-Service (QoS), Hybrid Architecture.**

I. INTRODUCTION

Streaming video contents to a large number of clients imposes a high load on the underlying network and the server. The voluminous nature of the video traffic along with its timing constraints make deploying a large-scale and cost-effective video streaming architecture a great challenge to researchers working in this domain. There are several approaches that can be used to stream video contents to the clients. These approaches can broadly categorized into two groups namely, unicast based and multicast-based approaches.

Unicast based approaches: In unicast based approach a single or unicast stream is established for every client that sends a request. There are three approaches that use unicast for on-demand video streaming. They are centralized, proxy, and content distribution networks (CDN) approaches (Figure. 1a, 1b and 1c).

Centralized Approach: Here powerful servers along with a high-bandwidth connection to the Internet are deployed. It is simple, easy to use, deploy and manage. However, it lacks w. r. t scalability and reliability. Apart from these major concerns, two additional concerns that may arise are high cost and heavy load on the centralized server.

Proxy Based Approach: Here proxy servers are deployed near the clusters / client domains. The proxy based approach helps in caching few videos in their entirety, thus acting as a cache. The proxy based approach saves bandwidth and is yields in short start-up delay and small jitter. On the other hand, this approach requires deploying and managing proxies at many locations. As number of proxies increase it not only increases the overall system capacity but also increases the cost involved.

Content Delivery Network (CDN) Approach: Here, third parties know as CDN are used for delivering contents to clients. Typically CDNs deploy servers at various edges of the network. The central concept behind this approach is to increase the availability and access to videos by placing them nearer to clients. This reduces load on network and servers and also results in shorter delays. However the cost incurred for deploying the servers is more and may not be always feasible.

Multicast Approaches: Here a multicast session is established to the clients. All clients receive the same portions of the stream at the same time. In simpler terms, multiple clients are served using the same stream thereby increasing the resource utilization of the system. Multicast distribution trees are either created at the network level or at the application level (Figure. 1d and 1e).

Network-level Multicast: The network-level multicast establishes a tree over the internal routers with the clients as the leaves of the tree. While network-level multicast is efficient, it is not widely deployed.

Application-level Multicast: Application-level multicast techniques, construct the distribution trees over the end systems. Building the tree over the end systems achieves deployability in the current Internet. However, it introduces another problem: it may overload some end systems beyond their capacities. An end system in the tree may become a parent of several other end systems.

VoD services usually consume a large amount of resources, primarily the storage space and the network bandwidth. Moreover, the maximum number of concurrent streams that a server can support is limited by server with limited resources like storage capacity, I/O bandwidth and the interconnection bandwidth. Apart from these issues, a significant area that affects the quality provided by the network is the scheduling process. Since several applications should be concurrently serviced, the role of the scheduling process is to select each time the application that should be served next. It is therefore important to design efficient scheduling algorithms for video streaming to improve the bandwidth utilization and overall system performance. This in turn increases the Quality-of-Service being offered by the systems [1,2].

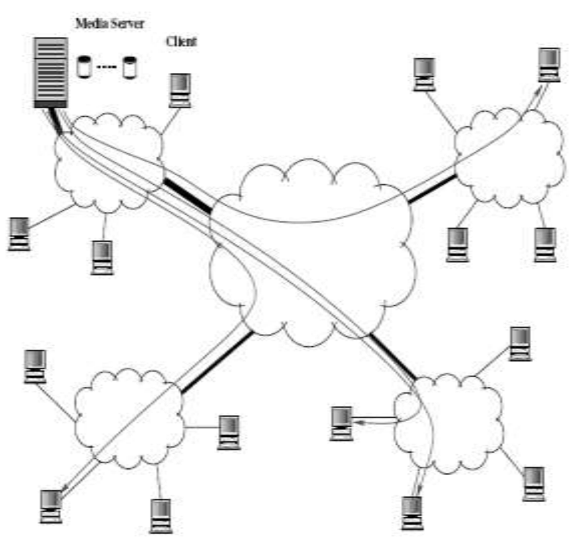

Figure 1(a) Centralized Approach

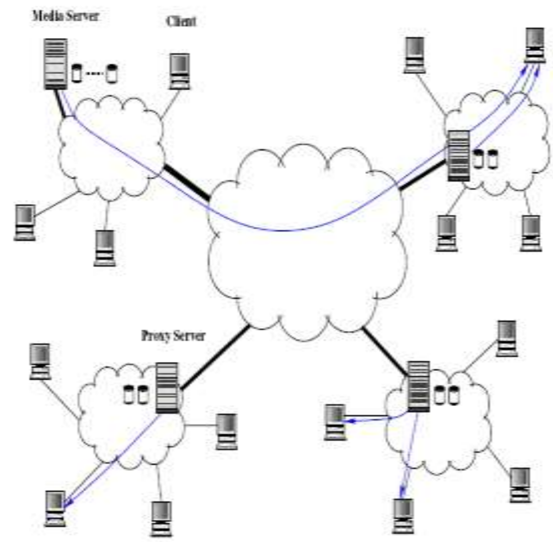

Figure 1(b) Proxy Based Approach

The rest of the paper is organized as follows; section II deals with the problem formulation for video request scheduling in VoD systems; section III focuses on the existing scheduling techniques that are either video scheduling techniques that make use of broadcast protocols, channel optimization or disk scheduling that make use of techniques like SCAN, CSCAN, SCAN-EDF, etc.; section IV deals with proposed scheduling framework which is a tri-combo framework; section V illustrates simulation results and section VI concludes the work carried out.

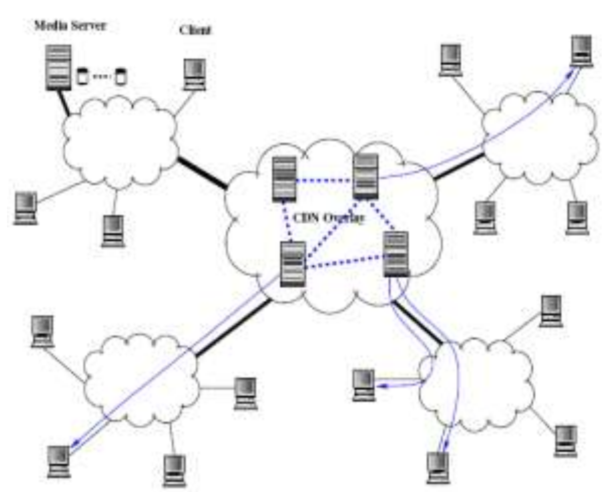

Figure 1(c) CDN Approach

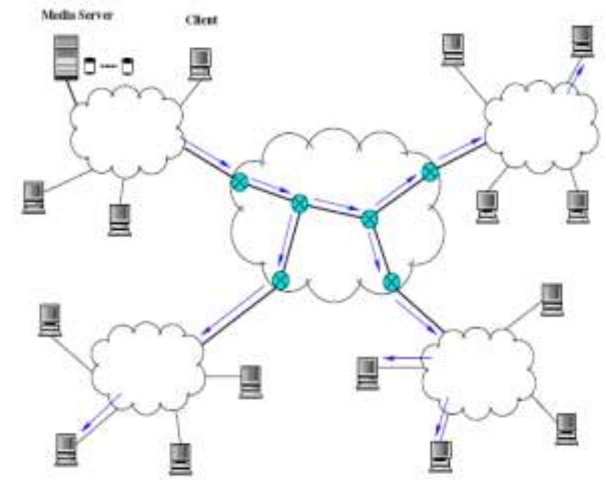

Figure 1(d) Network Level Multicast Approach

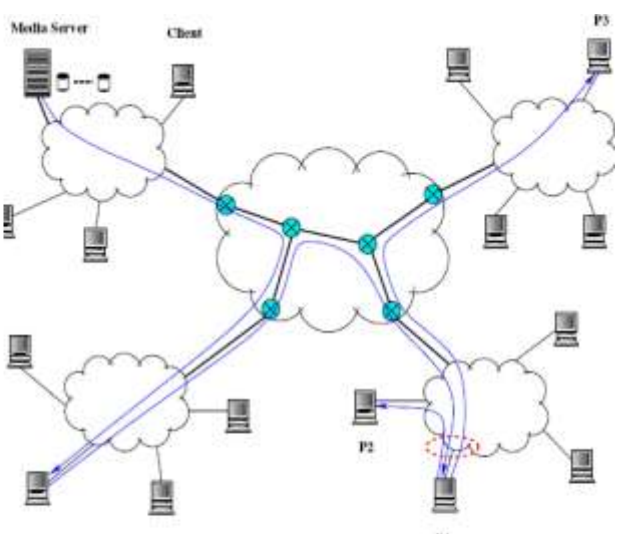

Figure 1(d) Application Level Multicast Approach

Figure 1: Typical Video-on-Demand (VoD) System architectures

II. Video Request Scheduling Problem Formulation

Let the number of video files by M denoted by $V_1, \ldots, V_M$. The length of each video be of same size L. The request arrive with mean arrival rates of $\lambda_1, \ldots, \lambda_M$ respectively. The video requests are served using N server streams. Without any loss of generality, let $\lambda_i > \lambda_j$ for $1 \leq i < j \leq M$, i.e., popularity of these videos decreases gradually with the index so that V_1 and V_M be the most- and the least-popular video respectively. The total arrival rate for all videos is given by:

$$\lambda = \sum_{i=1}^{M} \lambda_i$$

The probability of receiving a user request for a video V_i is given by p_i, where $p_i = \lambda_i/\lambda$ and $i = 1, \ldots, M$. Let the number of server streams allocated to V_i be n_i. The average waiting time for a user to get a server stream n_i is $d_i(n_i)$ where $di()$ is a non-linear function and $i = 1, \ldots, M$.

The video request scheduling problem here is to schedule the requests such that the average user waiting time

$$\sum_{i=1}^{M} d_i (n_i) \bar{p}_i$$

is minimized.

III. Related Work

The paper [6] surveys broadcasting protocols and describes the development of these protocols, starting with Staggered broadcasting protocols, in which the movies are simply transmitted repeatedly on the channels (e.g., [7]), through Pyramid-based broadcasting protocols, in which movies are partitioned into segments and different segments are broadcast on different channels [20], and finally Harmonic broadcasting protocols in which segment i is allocated bandwidth proportional to $1/i$ (e.g., [11]).

The papers [10, 12] present a simple schedule of one movie on h channels by partitioning the movie into $2^h–1$ segment. Their schedule implies a maximal start-up delay of $1/(2^h–1)$ for a movie of length 1. The Pagoda scheme [17] is based on a schedule for 3 channels with maximal start-up delay 1=9. It is then generalized to a schedule that asymptotically guarantees a start-up delay of at most $1/O(2.236^h)$. The new Pagoda scheme ([13]) deals with small values of h. Their maximal start-up delays for h = 3, 4, 5, 6, 7 are 1/9, 1/26, 1/66, 1/172, 1/442 respectively.

The Recursive Frequency-Splitting scheme ([19]) improves some of the results of the new Pagoda scheme. In particular, this scheme guarantees maximal start-up delays for h = 3, 4, 5, 6, 7 are 1/9, 1/26, 1/73, 1/201, 1/565 respectively. This scheme is almost equivalent to the greedy scheme presented in [1]. The latter paper presented better results for small values of h. Their best maximal start-up delays for h = 3, 4, 5, 6, 7 are 1/9, 1/28, 1/77, 1/211, 1/570 respectively.

The poly-harmonic protocol [16] always forces the receiver to delay the same amount of time before beginning playback. Using channels of differing bandwidth enables a worst case delay to asymptotically approach $1/(e^b- 1)$ for total bandwidth b. Several papers [5, 7, 9] have shown this bound on delay to be optimal. The lower bound result has been recently extended and asymptotically optimal protocols have been presented for the case where the receiving bandwidth is less than the sending bandwidth [6].

Harmonic broadcasting is implemented in [3] by a reduction from the *window-scheduling* problem. Specifically, the movie is partitioned into s equal-sized segments that are scheduled on the channels such that the gap between any two consecutive appearances of segment i is at most i. For a given number of channels, the goal is to maximize s, and as a result, minimize the start-up delay (which is at most $1/s$). A schedule based on this principle is shown to approach the lower bound as $h \rightarrow \infty$

The papers [14, 15] also apply the observation that clients may start buffering segments before they start viewing the movie to achieve better results. However, they demonstrate the usefulness of this observation only for small examples. The first paper ([14]) presents superior results in which schedules achieve a shorter start-up delay for the same amount of bandwidth. On the other hand, the second paper ([15]) presents simpler schedules that can be applied to other variants such as when the receiving bandwidth is less than the sending bandwidth.

The paper [21] proposed a Pyramid Scheme that for a given amount of bandwidth guarantees exponentially smaller start-up delay to clients that can buffer parts of the movie and can receive data from several channels concurrently. Many variants and generalizations of the Pyramid Scheme followed and all of them showed this dramatic improvement by employing the following schedule design principle: *Early segments should be broadcast more frequently than later segments.*

The paper [22] describes a cost-driven disk scheduling algorithm for environments consisting of multi-priority requests. The proposed algorithm minimizes costs by maintaining one-queue and managing requests intelligently in order to meet the deadline of as many priority 1 requests as possible while maximizing the number of priority 2 requests that meet their deadline.

The paper [23] proposes a rate-based real-time disk-scheduling algorithm called WRR-SCAN (Weighted-Round-Robin-SCAN). WRR-SCAN guarantees to meet the deadline of a real-time task by reserving disk bandwidth according to its real-time constraints. WRR-SCAN services scheduled tasks in scan order to minimize the disk-seek time. The experimental results show that WRR-SCAN reduces non-transmission overhead significantly and produces a guaranteed minimum data rate for aperiodic tasks while keeping the deadlines of real-time tasks.

The paper [24] proposed a SCAN-EDF, that combines the features of SCAN type of seek optimizing algorithm with an Earliest Deadline First (EDF). It investigates the impact of buffer space on the maximum number of video streams that can be supported. It illustrated that by making the deadlines larger than the request periods, a larger number of streams can be supported. It also described how the current work can be extended in the PRISM multimedia architecture.

IV. Proposed Method

Different issues involved in the design of VoD systems have been studied by researchers during last few decades. Architectural issues, physical storage requirements, load balancing and bandwidth allocation issues have been studied. But there are hardly a few papers that focus on scheduling in a VoD system. Some of the existing work concentrates on disk scheduling or movie scheduling rather than video request scheduling. Thus, this paper takes into account the fact that most requests tend to concentrate on popular videos and presents an efficient video request scheduling framework that effectively distributes the video requests across the architecture using a tri-combo scheduling techniques namely First Come First Served (FCFS) scheduling, priority scheduling and gang scheduling. The main advantages of this framework are a balanced load across the architecture, a good QoS being provide to users and drop in the rejection rate when compared to other existing solutions.

A. *System Model*

The VoD system model considered for video request scheduling is hybrid system (figure 2) that consists of a tracker, local video servers, intermediate video servers and main video server. All the servers act as a database of movies and support a fixed number of video streams. The clients connect to the server via dedicated links and make video requests. However the video replicas that are stored in these servers vary in number based on the popularity measure. The communication network used is equipped with a multicast facility. Thus, the same movie-stream can be sent to more than one user without causing any extra overhead to the server, and therefore, multiple users can participate in a single session. Let M be the set of videos stored in the server, U be the set of users making requests, and S be the set of streams / channels in the system, and let m, u, and s be their cardinalities, respectively. The important system parameters are:

T	:	The duration of video
L	:	Length of the video
τ	:	Maximum delay between request and start of the requested video
v	:	Maximum delay between a request and its response.

Note: Here $v \leq \tau \leq T$ and $\tau = v$.

The scheduling framework determines the systems response for an incoming video request at any moment. It streams videos over the channels / streams based on the requests to the system up to the moment the scheduling is taking place. If v=0 then the request is served at its arrival, otherwise (if v>0) the request is served at v times units after the request arrival.

B. *SCHEDULING*

Scheduling is the method by which threads, processes or data flows are given access to system resources (e.g. processor time, communications bandwidth). Video-on-Demand (VoD) applications are usually soft real-time applications. When a video request arrives, certain time constraints need to be met. One significant area that affects the quality provided by the network is the scheduling process. Since several video requests should be concurrently serviced, the role of the scheduling process is to select each time the video request that should be served next. The proposed framework makes use of a tri-combo scheduling technique i.e. it uses three different scheduling techniques namely First Come First Served (FCFS) scheduling, priority scheduling and gang scheduling.

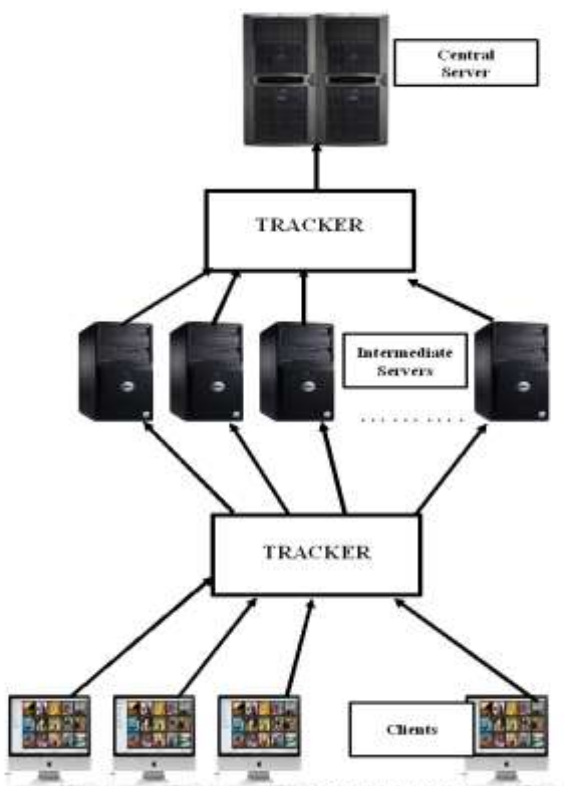

Figure 2: Video-on-Demand (VoD) System Model

FCFS is also the shorthand name for the FIFO scheduling algorithm, which gives every client the server time in the order they request videos. Fixed priority pre-emptive scheduling is a scheduling system commonly used in real-time systems. With fixed priority pre-emptive scheduling, the scheduler ensures that at any given time, the server serves the video requests with highest priority of all those request that are currently ready to execute.

Gang scheduling is a scheduling algorithm that schedules related threads or processes to run simultaneously on different systems. Usually these will be threads all belonging to the same process (video). Gang scheduling is used so that if two or more threads or processes communicate with each other, they will all be ready to communicate at the same time. If they were not gang-scheduled, then one could wait to send or receive a message to another while it is sleeping, and vice-versa.

The usage of these scheduling techniques in the proposed framework is as follows: When a user request arrives at the tracker, they arrive in FCFS order. Then the tracker groups them into two categories namely Class I and Class II. The basis for this classification is popularity of the video. Here the tracker makes use of priority and gang scheduling. Then tracker checks the status of the servers and forwards requests based on individual class or a combination of both based on the number of streams / channels available in the server. Once the server receives a gang of requests it starts streaming the video to the clients.

V. RESULTS AND DISCUSSIONS

The simulation of the proposed scheduling framework for VoD system was carried out for 4500 seconds with 380 requests. The results obtained from simulation are illustrated in figure 3 to 8. Figure 3 illustrates the total number of requests made and the number of requests being served. According to the results 94% of the total requests were accepted and served. Similarly figure 4 & 5 illustrates the ratio between the request made / served and the requests rejected which is 6%. The reason behind rejection includes factors like non-availability of the requested videos, server's threshold limits being reached at that time interval, etc.

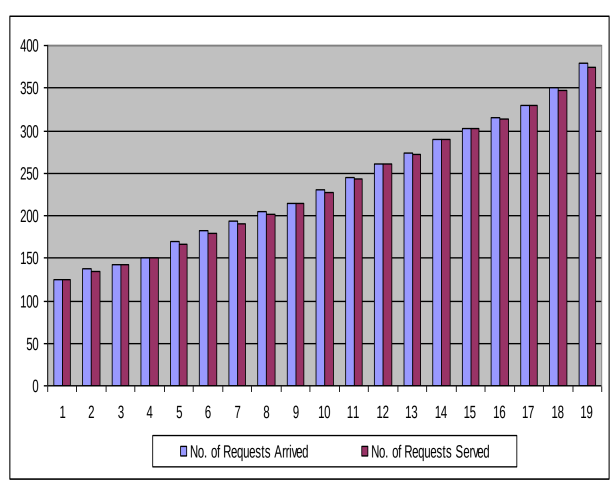

Figure 3: Requests Arrived vs. Served

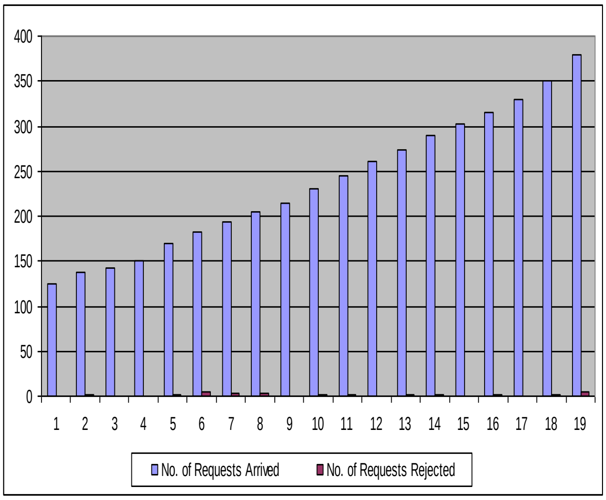

Figure 4: Requests Arrived vs. Rejected

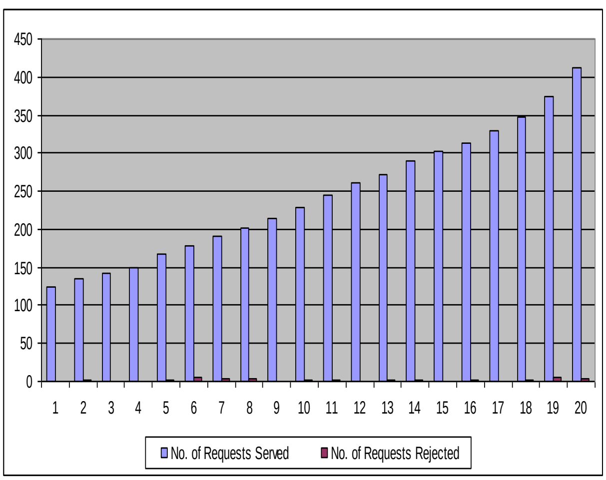

Figure 5: Requests Served vs. Rejected

Figure 6 illustrates the request arrival time and response start time. As depicted in the graph, initially there is a start-up delay. But in the later cases the delays are considerably shorter and acceptable. Figure 7 illustrates the request arrival time and the time that the user needs to wait for the services from the system. Again the results show that the wait times are very short and even some times negligible.

Finally, figure 8 depicts the load status of the servers used in the architecture. Almost 88% to 90% of implicit load balancing is achieved by using the proposed scheduling framework.

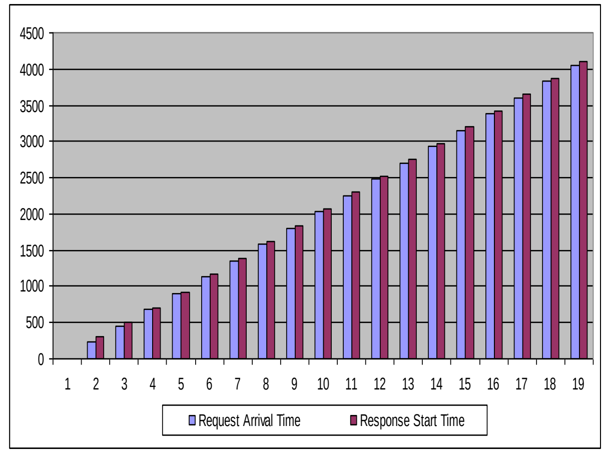

Figure 6: Requests Arrival Time vs. Response Start Time

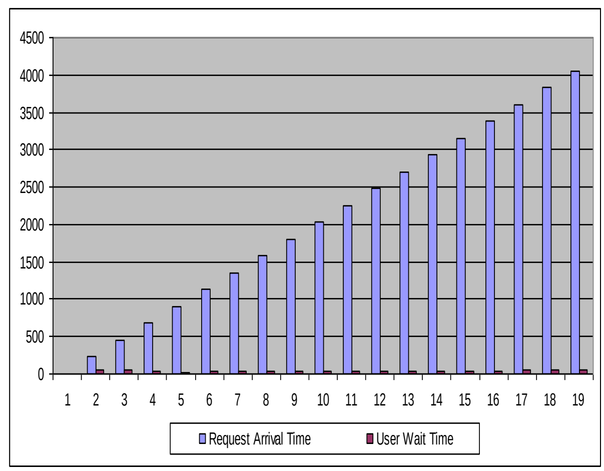

Figure 7: Requests Arrival Time vs. User Wait Time

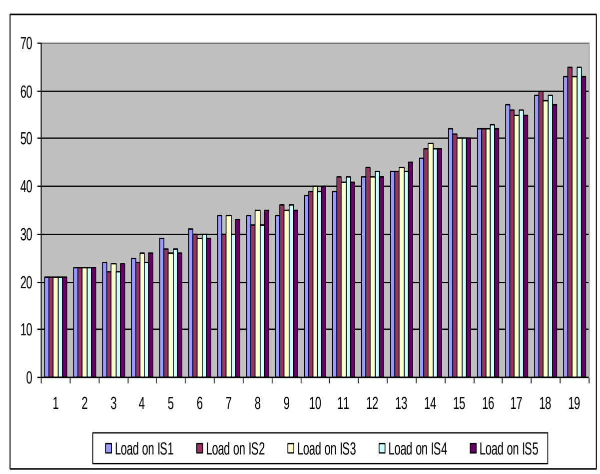

Figure 8: Requests vs. Load

VI. CONCLUSIONS & FUTURE WORK

This paper presented an efficient video request scheduling framework for Video-on-Demand systems. The proposed framework is robust, unique and efficient in that, it is a request scheduling framework rather than video or disk scheduling techniques being designed or deployed till day. The simulation results illustrate the fact that the said advantages of framework namely balanced load across the architecture, good QoS being provide to users and drop in the rejection rate when compared to other existing solutions are achieved. As per the experimental results 94% of the total requests are served, 6% are rejected and 90% of implicit load balancing is achieved.

Future work includes deploying explicit load balancing algorithms which in turn may minimize the rejection rate, enhancing the architecture to be more scalable, reliable and fault tolerant.

ACKNOWLEDGMENT

The authors would like to express heartfelt sincere gratitude to P E S Institute of Technology and S J C Institute of Technology for all the support provided for carrying out this research work.

REFERENCES

[1] D. Ghose and H.J. Kim, “Scheduling video streams in video-on-demand systems: A survey”, In the proceedings of Multimedia Tools and Applications, Chapter 11, pp. 167–195, 2000.

[2] Notre-Dame de la Paix, Namur, “Video-on-Demand over Internet : A Survey of Existing Systems and Solutions”, 2008

[3] A. Bar-Noy and R. E. Ladner., “Windows Scheduling Problems for Broadcast Systems”, In the proceedings of SIAM Journal on Computing (SICOMP), pp.1091 - 1113, 2003.

[4] A. Bar-Noy, A. Nisgav, and B. Patt-Shamir, “Nearly Optimal Perfectly-Periodic Schedules”, In the proceedings of International Conference on Distributed Computing, pp. 207-220, 2002.

[5] S. W. Carter, D. D. E. Long, and J. Paris, “Video-on-Demand Broadcasting Protocols” In the proceedings of Multimedia Communications: Directions and Innovations, Academic Press, pp.179-189, 2000.

[6] A. Dan, D. Sitaram, and P. Shahabuddin, “Dynamic Batching Policies for an On-Demand Video Server”, In the ACM Multimedia Systems Journal, pp.112-121, 1996.

[7] L. Engebretsen and M. Sudan, “Harmonic Broadcasting is Optimal”, In the proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 431-432, 2002.

[8] W. Evans and D. G .Kirkpatrick, “Optimally Scheduling Video-on-Demand to Minimize Delay” , In the proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1041-1049, 2004.

[9] L. Gao, J. Kurose, and D. Towsley, “Efficient Schemes for broadcasting Popular Videos”, In the proceedings of Multimedia System, pp. 284-294, 2002.

[10] A. Hu. “Video-on-Demand Broadcasting Protocols: A Comprehensive Study”. In the proceedings of IEEE INFOCOM, pp. 508-517, 2001.

[11] K. A. Hua, Y. Cai, and S. Sheu, “Exploiting Client Bandwidth for More Efficient Video Broadcast”, In Proceedings of the 7th International Conference on Computer Communication and Networks (ICCCN), 848{856, 1998.

[12] L. Juhn and L. Tseng., “Harmonic Broadcasting for Video-on-Demand Service”, In the IEEE Transactions on Broadcasting, pp.268-271, 1997.

[13] L. Juhn and L. Tseng.,” Fast Data Broadcasting and Receiving Scheme for Popular Video Service”, In the IEEE Transactions on Broadcasting, pp.100-105, 1998.

[14] J. Paris. “A Simple Low-Bandwidth Broadcasting Protocol for Video-on-Demand”. In the Proceedings of the 8th International Conference on Computer Communications and Networks (IC3N), pp.118-123, 1999.

[15] J. Paris. “A Fixed-Delay Broadcasting Protocol for Video-on-Demand”. In Proceedings of the 10th International Conference on Computer Communications and Networks (IC3N), pp. 418-423, 2001.

[16] J. Paris. “A Simple but Efficient Broadcasting Protocol for Video-on-Demand. In Proceedings of the 24th International Performance of Computers and Communication Conference (IPCCC 2005), pp. 167-174, 2005.

[17] J. Paris, S. W. Carter, and D. D. E. Long. “A Low Bandwidth Broadcasting Protocol for Video on Demand”, In Proceedings of the 7th International Conference on Computer Communications and Networks (IC3N), pp 690-697, 1998.

[18] J. Paris, S. W. Carter, and D. D. E. Long. “A Hybrid Broadcasting Protocol for Video on Demand”. In Proceedings of the IS&T/SPIE Conference on Multimedia Computing and Networking (MMCN), pp.317-326, 1999.

[19] J. Paris and D. D. E. Long. “Limiting the Receiving Bandwidth of Broadcasting Protocols for Video-on-Demand”, In Proceedings of the Euromedia Conference, pp.107-111, 2000.

[20] Y. C. Tseng, M. H. Yang, and C. H. Chang. “A Recursive Frequency-Splitting Scheme for Broadcasting Hot Video in VOD Service”, In the IEEE Transactions on Communications, pp.1348-1355, 2002.

[21] S. Viswanathan and T. Imielinski. “Metropolitan Area Video-on-Demand Service Using Pyramid Broadcasting”. In the proceedings of ACM Multimedia Systems, 197-208, 1996.

[22] Shahram Ghandeharizadeh, LiGuo Huang, Ibrahim Kamelm, A Cost Driven Disk Scheduling Algorithm for Multimedia Object

[23] Cheng-Han Tsai, Edward T.-H. Chu, Tai-Yi Huang, WRR-SCAN: A Rate-Based Real-Time Disk-Scheduling Algorithm”, In the proceedings of EMSOFT’04, pp. 1- 8, 2004

[24] A. L. N. Reddy, Jim Wyllie, K. B. R. Wijayaratne, “Disk scheduling in a multimedia I/O system”, In the ACM Transactions on Multimedia Computing, Communications, and Applications - TOMCCAP , vol. 1, no. 1, pp. 37-59, 2005.