2012 International Conference on Information and Computer Networks (ICICN 2012)
IPCSIT vol. 27 (2012) © (2012) IACSIT Press, Singapore

Buffer Management Scheme for Video-on-Demand (VoD) System

Sudhir N. Dhage [1+] and B.B.Meshram [2]

[1] Sardar Patel Institute of Technology, Andheri(West), Mumbai-400 058, Mumbai University

[2] Veermata Jijabai Technological Institute, Matunga, Mumbai-400 019, Mumbai University

Abstract- There has been an increasing trend in designing the Video-on-Demand systems using parallel server-based architectures to meet the client demands for online streaming of videos. We use a distributed architecture for effective Buffer management which efficiently utilizes various server resources like the disk Bandwidth, memory utilization and provides clients with VCR like functionalities. In this paper we proposed a unique On-line dynamic buffer replacement policy that does not base its results solely on a single parameter such as video popularity but calculates a cumulative popularity based on several other factors and hence chooses the video with the least popularity thus calculated. It is necessary to batch requests to minimize the bandwidth requirement, to reduce I/O demand, improve throughput and increase the number of customers served by using less number of streams. We proposed a Modified Adaptive Batching Policy for batching requests arriving at a server. This is different from the existing adaptive batching techniques which batch a user request into a multicast stream by increasing the speed of transmission in the unicast stream and hence batching it. Here, instead we compare the user arrival with a threshold batch time and according to the difference, assign users to different streams classified as multicast queue, submulticast queue and unicast queue. Finally, We proposed to implement distributed streaming of videos to clients using the Real Time Streaming protocol (RTSP) which being a control protocol is designed to add more VCR like functionalities to the streaming process.

Keywords: Buffer Management, RTSP, Adaptive Batching Policy, Video-on-Demand.

1. Introduction

There has been an increasing trend in designing the Video-on-Demand (VoD) systems using parallel server-based architectures [2][4] to meet the client demands for online streaming of videos. By deploying a parallel server-based architecture the performance limit of a single server can be increased, while keeping the system cost low by leveraging on commodity hardware platforms. This research has various phases- Buffer management, Disk Load Balancing and Video placement. The buffer management [8] in a VoD system mainly focuses on the storage and efficient utilization of the available bandwidth to provide continuous streaming of the requested video without any staggering in it [1][7]. The smooth playback is made available if the required resources like I/O and network bandwidth are available, if not the number of users that can be served simultaneously reduces declining the very aim of a strong VoD system. The streaming is implemented using the more efficient Real Time Streaming Protocol (RTSP) rather than the conventional HTTP protocol [9][4]. The computational problems arising in large distributed Video-on-demand systems are very large and require appropriate mechanisms for distributing data among the processors. The goal is that load is be evenly divided among processing elements and that the communication between processors should be minimized. This leads to the formulation of effective load balancing algorithms.

The goal of video placement scheme is to decide the rank of the videos, the number of copies and place them on the proxy servers so that the number of hits is maximum. This method explicitly considers the impact of the available video server equipment unlike previous works which focused on theoretical caching

[+] Sudhir N. Dhage Tel.: +912226708520; fax: +912226701422.
E-mail address: sudhir_dhage@spit.ac.in.

concepts or the dynamics of cache filling and reclamation. The various services that can be offered by video-on-demand systems are: Conduct Forums/Conferences/Seminars from Other Institutions, Movies-on-demand, E-commerce, Interactive advertising. The main feature that a VoD server caters to is the VCR like features viz. to pause, rewind, forward and stop a live streaming video. The architecture is designed to facilitate services to geographically distributed regions. The two types of VoD systems are: first **Near-Video-on-demand*:*** This technology delivers the same content using multiple video streams with staggered start times. It broadcast its videos mainly at a set time, and often through multiple channels at multiple times, and the viewer can choose whether or not to view a video. And Second **Interactive/True-Video-on-demand*:*** This technology provides the users with the requested content immediately. The users can select any video from available list and can perform VCR-like functions with the selected video at any instance of time. The rest of the paper is organized as follows: In Section 2, we gives proposed distributed Architecture and Distributed Streaming. We review the existing replacement Algorithms in section 3. In section 4 we describe the Proposed Dynamic Buffer replacement Algorithm. In section 5 describe Implementation of the Real Time Streaming Protocol. In section 6 express the results and finally, in Section 6, we conclude our paper.

2. Proposed Distributed Architecture and Distributive Streaming

The advances in the field of Computer Science, Multimedia and Communication technology have paved a way for increase in the number of multimedia services and applications. One such exiting multimedia service's that is currently gaining popularity across the world is Video-on-Demand (VoD) System. However, VoD is becoming costly due to the load placed on the servers. With number of users increasing, the load on the servers may proportionally increase, network may get congested and the performance of the system gradually degrades. A Video-on-Demand system can be designed using any three of the major network configurations viz. Centralized, Networked and Distributed. Proxy servers are widely used in multimedia networks to reduce the load on the central server and to serve the client requests faster [10]. We proposed a new VOD architecture for distributed interactive VOD system shown in the figure 1. This architecture consists of a Central Multimedia Server and a set of Proxy servers which are connected to form a partial mesh topology. All the proxy servers are connected to the Central Media Server. Buffer management in a VoD system mainly focuses on the storage and efficient utilization of the available bandwidth to provide continuous streaming of the requested video without any staggering in it. The smooth playback is made available if the required resources like I/O and network bandwidth are available, if not the number of users that can be served simultaneously reduces declining the very aim of a strong VoD system.

This research mainly focuses on architectural solution for improving the performance of the VoD systems. This architectural solution is Distributed Architecture consisting of a Central Media server and several Distributed Proxy Servers. This system aims at combining the features of proxy based and content distribution architecture and is distributed in nature. The system will provide high availability while supporting a high number of video streams. It will eliminate the drawbacks of the existing system.

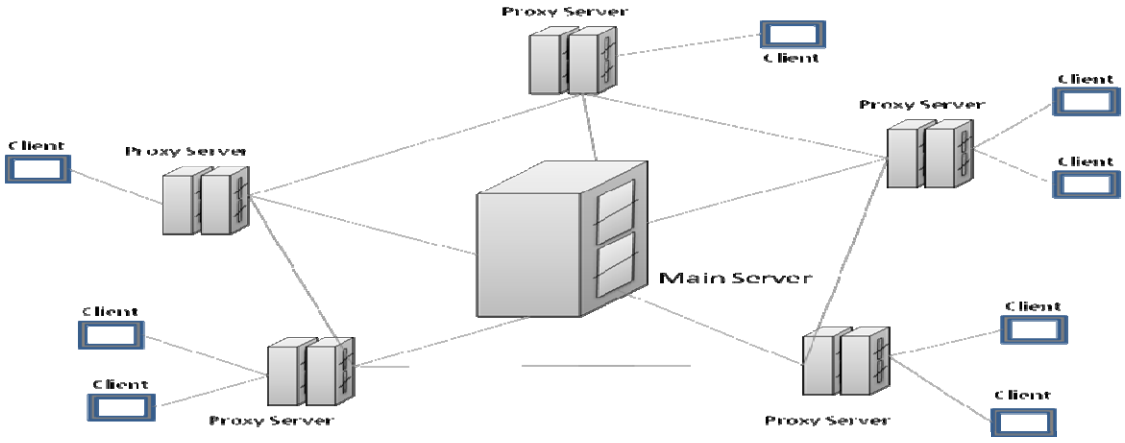

Fig. 1: System Design Architecture

The main components of this architecture are: **Main Server: -** The main or central server is the main multimedia server where all multimedia contents are available. It is the main video library from where videos are placed on the proxy server. **Proxy Server: -** Proxy servers are a subset of the central server. A set of clients are connected to each proxy server, i.e. client requests are served by the proxy server. **Client: -** The clients are the users of the VOD system. They request to proxy server for any multimedia content they wish to view. The above steps clearly represent the modular break down of the entire setup required for the system.

The buffer management technique used in this system implementation caches certain portions of frequently accessed videos in order to reduce the demand on I/O bandwidth. The architecture designed is distributive in nature, i.e. a request is attended by a proxy server that is balanced in terms of the load currently on it and the disk space and bandwidth available with it. The buffer management scheme will try to minimize the bottleneck related to handover of requests from the proxies and the disk bandwidth. The inter-proxy communication in the architecture is fruitful to cater to the request of the client immediately by streaming video from requested proxy or buffering the video from the neighbor proxy. The videos in the VoD server will be stored in terms of frames or blocks and will be streamed/buffered accordingly considering the disk load and the feasibility of buffering the video at the proxy server. It results in efficient usage of buffer and bandwidth for the videos and also results in low client waiting time by balancing the load among the neighboring proxy servers.

3. Existing Replacement Algorithms

In most of the previous systems on buffer management, it is observed that the main drawback in the system lies in their replacement algorithm. The previously proposed scheme ranks the videos in the external media server as per the popularity of the respective video locally. This popularity is calculated taking into consideration only the total no. of views of each video respectively.

3.1. Least Recently Used (LRU) Replacement Algorithm[3][5]

This algorithm replaces the block in the cache that has not been used for the longest period of time. From the basics of temporal locality, the blocks that have been referenced in recent past will likely be referenced in the near future. This policy works well when there is a high temporal locality of references in the workload but not otherwise. An Early Eviction LRU (EELRU) evicts the blocks when it notes that too many pages are being touched in a roughly cyclic pattern that is larger than the main memory.

3.2. Least Frequently Used (LFU) Replacement Algorithm[3][9]

This policy keeps a reference count for each document in cache and when a document needs to be replaced, the document with the least reference count is removed. In a situation where there are two documents with the same reference count, it uses LRU to break the tie (i.e. it removes the least recently used among them).

4. Proposed Dynamic Buffer Replacement Algorithm

To choose the best caching technique for VOD, one should note that traditional caching algorithms, such as LRU, do not work well for video servers due to the large size of video objects [8][9]. When the entire video objects need to be cached, a server cache may not be able to hold more than a few video objects; this results in a low cache hit ratio. One alternative to traditional caching algorithms is interval caching. This is an innovative caching algorithm developed for video servers. It employs temporal locality of client requests to the movies and caches intervals between successive requests to the same movie, as opposed to the entire video object. We intend to implement the Client Assisted Interval Caching scheme which tends to use the cache memory available in clients to serve the first few blocks of streams so as to dramatically reduce the demand on the I/O bandwidth of the server. In most of the previous research work on buffer management, it is observed that the main drawback in the system lies in their replacement algorithm. The previously proposed scheme ranks the videos in the external media server as per the popularity of the respective video locally. This popularity is calculated taking into consideration only the total no. of views of each video respectively.

The Proposed system ranks the video not only taking into consideration of the total no. of views but also the no. of views in the last 'x' days. This x is known as the threshold constant. It is important to include this parameter into the algorithm since it is not only the total no. of views that decides which video should be replaced but the combination of the above mentioned parameters that does the required job of replacement in a more efficient way. The parameters used in the replacement algorithm are mentioned below:-

Table 1: Parameters for replacement Algorithm

T	Total no. of views of each video	t	Number of views of each video in the past X number of days
X	Threshold number of days	D	Deciding factor

The algorithm for replacement is stated as follows: - first calculate the deciding factor using the formula

D1(Deciding factor for video 1) = t1/T1

Now deciding factor for each video is calculated in the same way. Now if during replacement if Di<Dj for all j!=i then replace 'i'th video i.e. During the time of replacement the video with lowest deciding factor will be removed from buffer or memory and a new video will replace it. Thus the video with the most views in past 'X' no. of days will be retained and with the least views will be replaced.

Significance of taking parameter X days in algorithm:- If we had not taken any threshold and gone about replacing videos in the traditional way that is in the order of most number of views then the number of replacements i.e. collision would occur more frequently wasting many CPU cycles as well as precious energy behind unwanted transfers. To reduce the no. of collisions and increase the HIT ratio we proposed the algorithm. The above four parameters rate each video locally and on the basis of deciding factor, the video to be replaced is obtained. The videos in the VoD server will be stored in terms of frames or blocks and will be streamed/buffered accordingly considering the disk load and the feasibility of buffering the video at the proxy server. It results in efficient usage of buffer and bandwidth for the videos and also results in low client waiting time by balancing the load among the neighboring proxy servers.

Algorithm Dynamic Buffer Replacement Policy {
Begin: ***If memory is empty then*** *V1 gets data from disk;*
If next block request Vi arrives then
{ Increment Ti; Calculate deciding factor Di = ti/Ti;
If space available in memory then *Store the request Vi;*
Else if memory is full then
{ ***for each(block j in memory)***
{ ***If Dj<Dj+1 then*** *r=j; } Replace Vr with request Vi }*
End }

Algorithm Modified adaptive Batching Policy{
//t,t1,t2,t3: time at which user requests arrive
Begin: ***while (****queue requests are in the multicast queue****)***
{ if (*arriving time t <= tl****) then*** *Wait in the previous multicast queue;*
else if (*arriving time t <= t2****) then*** *Cache the data of the previous multicast stream;*
Wait in the sub-multicast queue;
else if (*arriving time t <= t 3 ****) then*** *Cache the data of the previous multicast stream;*
Wait in the unicast queue; ***else*** *Initialize a new multicast queue;* ***}***
End }

5. Implementation of the Real Time Streaming Protocol

The traditional approach that is generally used consists of the compressed audio/video file to be downloaded as a text file. The client (browser) can use the services of HTTP and send a GET message to download the file. The Web server can send the compressed file to the browser. The browser can then use an application, normally called a media player, to play the file. This approach is very simple and does not involve streaming. However, it has a drawback. An audio/video file is usually large even after compression. An audio file may contain tens of megabits, and a video file may contain hundreds of megabits. In this

approach the file needs to be downloaded completely before it can be played. Using the conventional data rates, the user needs some seconds or tens of seconds before the file can be played.

The RTSP is a control protocol designed to add more functionalities to the streaming process. Using RTSP we can control the playing of audio/video. RTSP is an out-of-band control protocol that is similar to the second/ the control connection in FTP. The media player can send other types of messages. For example, a PAUSE message temporarily stops the downloading; downloading can be resumed with a PLAY message. An RTSP server needs to maintain state by default in almost all cases, as opposed to the stateless nature of HTTP. Both an RTSP server and client can issue requests which are not possible while using HTTP. RTSP is defined to use ISO 10646 (UTF-8) rather than ISO 8859-1, consistent with current HTML internationalization efforts. The Request-URI always contains the absolute URI. This makes "virtual hosting" easier, where a single host with one IP address hosts several document trees. To enable the implementation of the RTSP protocol, we intend to implement a customized portal which supports the RTSP protocol. Considering the benefits of the RTSP protocol, we intend to perform the streaming using it in this research.

6. Results

The results of our actual distributed system for a run time which involved the specifications mention in the table 2. The figure 2 indicates the Buffer Hit Ratio for the entire Video range at all proxy servers. Videos are ranked from 1 to the max number, here 50. Buffer Hit Ratio can be defined as for all requests, (Request for which video found in buffer/Total video requests) from figure 2 it is visible that the lowest 20% ranked videos are almost each time found in the buffer, i.e. either they are currently being streamed or are present in buffer in offline mode (not currently being streamed). After that as rank decreases the buffer hit ratio also decreases. But the highly ranked videos have less chances of already being in buffer. Thus a considerable amount of bandwidth is utilized to bring from disk to buffer or from buffer of other proxy server.

Table 2: run time specification

S.N.	Description	Input	S.N.	Description	Input
1	Number of Proxyservers	3	4	Number of disks on Proxyserver 3	6
2	Number of disks on Proxyserver 1	8	5	Total number of Videos	50
3	Number of disks on Proxyserver 2	7	6	Buffer Size	16 GB
7	Total client requests received	Approx. 1000			

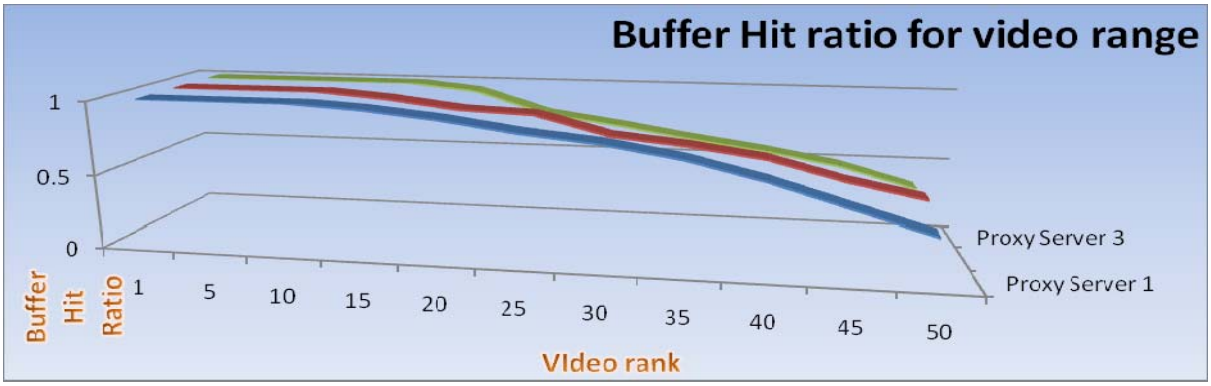

Fig. 2: Buffer Hit Ratio for video range

7. Conclusion

After a brief study of various research papers and algorithms related to buffer management in distributed and interactive VoD systems, we devised new architecture for VoD system and then proposed novel replacement algorithm and modified adaptive batching policy for buffer management in VoD systems. The Novel dynamic buffer replacement policy that does not base its results solely on a single parameter such as video popularity but calculates a cumulative popularity based on several other factors and hence chooses the video with the least popularity thus calculated. It is necessary to batch requests to minimize the bandwidth requirement, to reduce I/O demand, improve throughput and increase the number of customers served by using less number of streams. We proposed to implement distributed streaming of videos to clients using the

Real Time Streaming protocol (RTSP) which being a control protocol is designed to add more VCR like functionalities to the streaming process.

8. References

[1] T R Gopalakrishnan Nair, P Jayarekha, “A Rank Based Replacement Policy for Multimedia Server Cache Using Zipf-Like Law” , Journal of Computing, Vol. 2, Issue 3, March 2010.

[2] Wujuan L., Yong L. and Leong Y.,“A Novel Interval Caching Strategy for Video-on-Demand Systems”, Pg. 1-5, ICON’06, Networks, 2006.

[3] F.Moser, A.Kraiss, and W.Klas, "L/MRP: A Buffer Management Strategy for Interactive Continous Data Flows in a Multimedia DBMS", Proceedings of the 21th IEEE VLDB conference, pp.275-286, 1995.

[4] A.Dan and D.Sitaram, "Buffer Management Policy for an On-Demand Video Server", IBM Research Report, RC 19347, 1993.

[5] S.Sheu, K.A.Hua, and W.Tavanapong, "Dynamic Grouping: An Efficient Buffer Management Scheme for Video-on-Demand Servers", Proc. of the 2nd International Conference on Multimedia Information Systems, 1997.

[6] M.Andrews and K.Munagala, "Online Algorithms for Caching Multimedia Streams", Proceedings of European Symposium on Algorithms, pp.64- 75, 2000.

[7] A.Dan and D.Sitaram, "A Generalized Interval Caching Policy for Mixed Interactive and Long Video Workloads", Proceedings of Multimedia Computing and Networking, pp.344-351, January 1996.

[8] B. Wang, S. Sen, M. Adler, and D. Towsley, “Optimal Proxy Cache Allocation For Efficient Streaming Media Distribution,” In IEEE INFOCOM, June 2002.

[9] Bing Wang, Subhabrata Sen, “Optimal Proxy Cache Allocation for Efficient Streaming Media Distribution”, VOL. 6, NO. 2, April 2004.

[10] Sang-Ho Lee, Kyu-Young Whang, Yang-Sae Moon, “Dynamic Buffer Allocation in Video-on-Demand Systems”, VOL. 15, NO. 6, NOVEMBER/DECEMBER 2003