IJREAT International Journal of Research in Engineering & Advanced Technology, Volume 4, Issue 2, April - May, 2016
ISSN: 2320 – 8791 (Impact Factor: 2.317)
www.ijreat.org

Load Balancing Video Servers In Distributed System

Kunjal Surati, Vedashree Patil , Namoshi Roy, Ramakant Singh, Prof. Smita Patil

Department of Information Technology, Atharva College of Engineering, University of Mumbai Mumbai, Maharashtra , India

Abstract
Load Balancing effectively distributes the load among the available proxy servers. Load balancing is distributing the amount of work that a server has to do between two or more servers. Load balancing improves the performance of the system by balancing the loads among the servers. A Video-on-Demand (VoD) system is a system that allows geographically distributed users to select and watch videos when they choose to rather on a specific fixed broadcast time. Load balancing in distributed systems avoids single point failure and hence increases the availability of videos. In this paper, we have proposed load balancing policy for Video-on-demand systems using round robin algorithm.
Keywords: *load balancing, video-on-demand*, *Round Robin, distributed system* .

1. Introduction

The growth of the Internet and the increasing popularity of dynamically generated content on the World Wide Web, has created the need for more and faster web servers which are capable of serving over 100 million Internet users [8]. There is an increasing demand of high performance web servers, as dynamic contents change traditional web environment.

Load balancing becomes one among the most important problems while attaining high performance in distributed systems which may consist of many heterogeneous resources connected through one or more communication networks. Load balancing is the strategy by which load is redistributed among the computational elements (CE) of a heterogeneous network, where they work cooperatively so that large loads can be distributed among them in a fair and effective manner.

Video on Demand are Systems which allow users to select and watch video content as per their convenience. An interactive Video-on-Demand system (IVOD) is an extension of Video-on-Demand (VoD). Many VOD server have thrown away single servers and turn into VOD server clusters to improve the response speed, throughput and scalability of VOD server efficiently. VOD server cluster centrally accepts all the incoming requests and evenly dispatches them to the servers. Round Robin scheduling algorithm is used to distribute the incoming requests among video server. In distributed environment, all copies of videos are replicated on all nodes.

The remainder of the paper is organized as follows. In section 2, we have discussed about some related work. Section 3 includes Video-on-Demand system. Round robin scheduling policy is discussed in Section 4. In Section 5, proposed system is discussed.

2. Related Work

A clustered load balancing policy for a heterogeneous distributed computing system is proposed to balance load [1]. This algorithm estimates different system parameters like CPU Utilization, Memory Utilization, CPU Queue length, and Response time of the system to decide the workload of each node. A load balancing method is proposed that uses a two- level strategy to balance workload among the nodes thus reducing the complexity of the load balancing process and taking care of heterogeneity and scalability. A survey is done for existing IVOD systems and different system architectures [2]. Different scheduling policies are studied. Interactive Video-on-Demand is an extension of Video-on-Demand (VoD). Interactive Video-on-Demand provides functions such as pause/play, resume, forward, rewind, fast forward, fast rewind, slow forward and slow rewind. The user can select

any video, and watch it according to their schedule.A framework for server load-balancing over a single-operator OpenFlow network is proposed [3] to improve the quality of service levels of video streaming services. A new OpenFlow controller application is designed for dynamic server load balancing by continuous monitoring of load of each video server. Dynamic rerouting of clients to alternate servers is performed with lower loads when an overload condition is detected. A solution is developed to design interactive NVOD systems when stream merging is employed [4]. They have proposed an efficient dynamic I-Stream allocation policy which adjusts the number of I-Streams and B-Streams according to the system requirement at the moment. This policy allows the server to respond to workload variations. Algorithms for simple partitioning video-on-demand storage are studied to distribute content and workload among servers within a master-slave architecture [5]. They have studied several benefits of distributing video-on-demand storage across multiple servers. The first benefit of having a distributed video-on-demand service is scalability. A distributed architecture makes it easier to add capacity to the video-on-demand storage. The architecture and basic scheduling algorithm based on the Linux Virtual Server cluster is introduced [6]. They have improved the existing load balancing algorithm and put forward a new load balancing algorithm which combines the static and dynamic scheduling algorithm.

3. Video-on-demand System

Video on Demand (VoD) is an interactive multimedia system that works like cable Television. The users can select a video from a large video database. Individual users in an area are able to watch different programmes when they wish to.

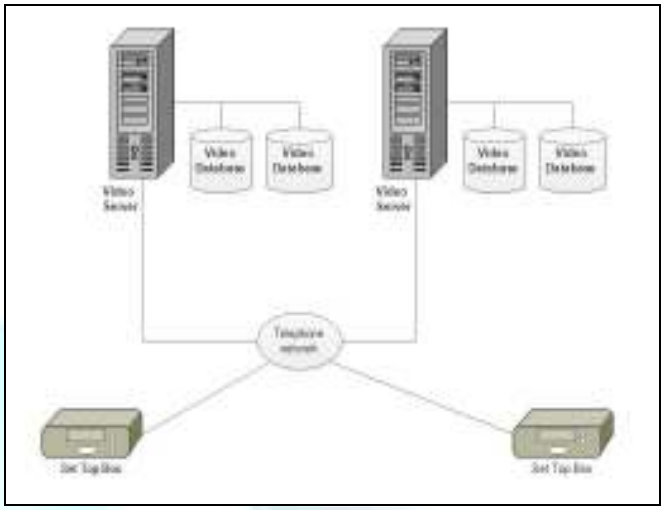

Fig 1. Video on Demand system

The main components of a video on demand service are shown in figure:

- **The video server**: It stores and provides access to video.
- **The data delivery network**: It interconnects the subscriber and the end user **set-top box** to interface home TV equipment with the VoD services.
- **Video Database:** This is the database of collection of various videos.

Prerecorded videos are digitally stored in a video server. These videos are then transmitted in a coded, compressed format. After the videos are ordered they are decoded and decompressed by set-top converters in individual homes.

The user can call on a range of services. User can perform operations, such as video selection, pause, rewinding etc. while watching videos as if it were a video player. These are processed by the *set-top-box* and sent to the local server.

Digital videos can be compressed and stored on hard disk and advertised for users on the network. Multiple archive servers can simultaneously be running over the network, depending on the bandwidth available.

4. Round-robin scheduling policy

Simple round robin comes under the category of static policy [8]. Static Load Balancing policies make their decision on statistical data from the system. Static policies do not consider any system state information. Static policies is performed when the system load and number of processes is fixed and known at compile time. Simple round robin implements a circular queue where pointer is pointed at the last elected server, that is

IF server i was the last chosen node for request dispatching,

THEN next request will be assigned to server i+1,

WHERE i=i+1 mod N here N is the number of servers.

As static algorithms do not rely on the current state of its system at the time of decision making, they are the fastest solution to prevent the web switch becoming the primary bottleneck of the web cluster as mentioned in. The advantage of the algorithm is its simplicity as it utilizes only information on past assignment decision. Hence the Static Load Balancing policy is simpler and takes less time to execute.

5. Architecture of proposed load balancing system

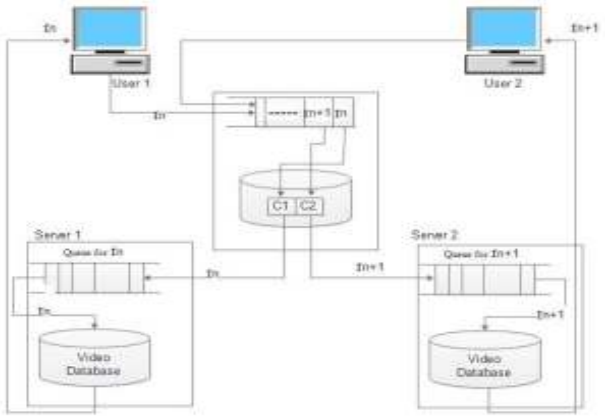

Fig 2. Architecture of proposed load balancing system

The architecture shows the proposed system in which when user1 placed a request say r_n ,for certain video V and the next request r_{n+1} was placed by another user say user2 for the same video V, then the procedure that is followed to fulfill these requests is as follows:

r_n and r_{n+1} will enter into rear end of scheduling queue of scheduler. From front end of scheduler queue r_n^{th} request will be dispatched to server1 and next request i.e. r_{n+1}^{th} request will be dispatched to server2. While this dispatching of requests occurs, for each r_n^{th} request, counter for server1 i.e. C1 will increment. Similarly for each r_{n+1}^{th} request counter C2 will increment.

Dispatcher in proposed system is using the simple Round Robin algorithm for dispatching of incoming requests. Once the dispatched request is received by associated proxy server, proxy server attempts to fetch the desired video from media library. When requests arrive at intended servers, they are locally queued for some time and then the desired video from media library is fetched. This video will then starts responding on browser of respective user.

6. Load Balancing Algorithm

- **Input Parameters:**

V: Video

Id: video identifier

C: count

- **Output Parameters:**

Server1: previous location of video

Server2: new location of video

- **Algorithm:**

1. Begin
2. Select v from available video list.
3. Call function videoclick()
4. Videoclick(id)
   - 4.1. If (null==id)
     - 4.1.1. Return
   - 4.2. Else
     - 4.2.1. Call function incrementView()
   - 4.3. End
5. End

incrementView()

1. Begin
2. Initialize c=0
3. Function getparameter(video)
   3.1. Update database
4. Initialize resultset to null
   4.1. Select count
   4.2. Increment C
   4.3. Update database
   4.4. End
5. If (C %2 != 0)
   5.1. Server= Server2
6. Else
   6.1. Server = Server1
   6.2. End
7. End

7. Results

7.1 Homepage

Fig 3. 1st attempt for playing a video

As shown in the above fig. 5.1, this is a home page of the implemented system. Here we will get listing of uploaded videos.

7.2 Playing a video

Fig 4. 1st attempt for playing a video

When user selects a video, it will redirect to the respective page. If user is selecting first video in the list then it will be redirected on 1.jsp. If we will see at the bottom of the page, it is shown that video is playing from the local machine.

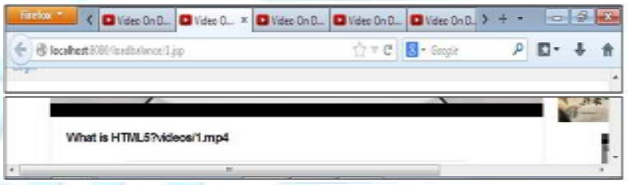

Fig 5. 2nd attempt for playing a video

As shown in the above fig 5.3 this is second attempt to view the video. Here path is shown as first selection that is HTML5?videos/1.mp4

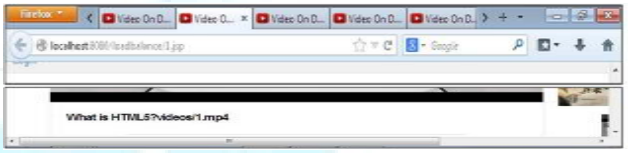

Fig 6. 3rd attempt for playing a video

As shown in the above fig. 5.4 this is third attempt to view the video. Here path is shown as first selection that is is HTML5?videos/1.mp4.

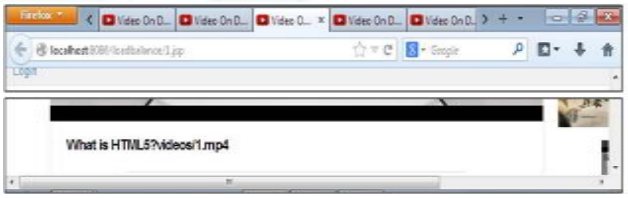

Fig 7. 4thattempt for playing a video

As shown in the above fig.5.5 this is fourth attempt to view the video. Here path is shown as first selection that is is HTML5?videos/1.mp4.

7.3 Video views

Fig 8. Total count for all videos

These are total counts i.e. how many times videos have been seen by different users.

8. Comparison of Round Robin Algorithm

Table 1: Comparison of Round Robin Algorithm

Parameters	Round robin	Adaptive batching	Dynamic buffer allocation	Preemption based buffer allocation (PBBA)
Network bandwidth	40 % reduction in the bandwidth requirement	Less because update rate decide the bandwidth requirement	-	-
Storage space at servers	The central server has all the videos so storage increases	As per requirement	Larger space required to store all videos	Larger space required to store all videos
Storage space at client	-	-	Relatively small since buffer is allocated dynamically with smaller sizes	Larger
Cost of setting up servers	-	High	Average	High
Access time delay	Reduced	-	Reduced due to less buffer size and RAM requirements	Reduced due to less buffer size
Video popularity	Independent of the popularity of the video	Main focus on popularity as popular video's access is taken from arrival rate of request	Since smaller buffers are allocated, the saved memory can be used for the popular videos	Independent
Network topology	Central Server	-	-	-
Staertup latency	Initial latency very much reduced faster than the real time	Initially multicast stream=-2* batching time	Initial latency is very much reduced	Initial latency very much reduced faster than the real time
Number of Concurrent User Requests	Independent of the simultaneous request arriving at the peak rate	Any	Services more number of concurrent user requests regardless of disk distribution load	Any

6. Conclusion

The presented design works well for distributed systems and ensures that no process suffers starvation and no processor is overwhelmed. Benefit of having a distributed video-on-demand service is scalability. Round robin algorithm is used for distributing incoming requests among associated n number of proxy servers. Simple round robin algorithm is used because of its simplicity. We have used n proxy servers which will contain n replicas of media library. Future plan is to offer security algorithm with load balancing. Optimization of this policy can be further enhancement.

References

[1] Moumita Chatterjee and S K Setua, "A New Clustered Load Balancing Approach for Distributed Systems", University of Calcutta Kolkata, India, Computer, Communication, Control and Information Technology (C3IT), 2015 Third International Conference , pp. 1-7, 7-8 Feb. 2015.

[2] Sudhir N. Dhage, Smita K. Patil and B. B. Meshram,"Survey on: Interactive Video-on-Demand (VoD) Systems", Mumbai, Circuits, Systems, Communication and Information Technology Applications (CSCITA), 2014 International Conference, pp .435-440, 4-5 April 2014.

[3] Selin Yilmaz, A. Murat Tekalp, Bige D. Unluturk, 'Video Streaming Over Software Defined Networks With Server Load Balancing', 2015 International Conference on Computing, Networking and Communications, Internet Services and Applications Symposium, pp.722-726, 16-19 Feb 2015,Garden Grove, CA

[4] Kamal K. Nayfeh and Nabil 1. Sarhan "Design and Analysis of Scalable and Interactive near Video-on-Demand Systems", San Jose, CA,Multimedia and Expo (ICME), IEEE International Conference, pp. 1-6, 15-19 July , 2013.

[5] Bairong Lei, Ivan Surya, Shahin Kamali, Khuzaima Daudjee, "Data Partitioning for Video-on-Demand Services",2013 IEEE 12th International Symposium on Network Computing and Applications, Cambridge, MA.

[6] Yang Hu and Shanan Zhu, "Load-balancing Cluster Based on Linux Virtual Server for Internet-based Laboratory" , Hangzhou, Industrial Electronics and Applications (ICIEA), 2014 IEEE 9th Conference, pp. 2181-2185, 9-11 June 2014.

[7] M.Thejovathi, ‘‘Dynamic Load Balancing Algorithms for Distributed Networks”, IJCSNS International Journal of Computer Science and Network Security, VOL.14 No.2, February 2014.

[8] Kunjal Surati, Vedashree Patil, Namoshi Roy, Ramakant singh, Smita Patil, “A Study on Load Balancing Video Servers in Distributed System”, International Journal of Innovative Research in Computer and Communication Engineering Vol. 3, Issue 10, ISSN:2320-9801, October 2015