BANDWIDTH AWARE LOAD BALANCING AND OPTIMAL BANDWIDTH ALLOCATION TECHNIQUES FOR VIDEO-ON-DEMAND SYSTEMS

Vinay. A[1], Bharath K[2], Prateek Saxena[3] , Shikhar Chandra[4], T N Anitha[5]

[1,2,3,4]Department of Information Science and Engineering, P E S Institute of Technology
Bangalore, Karnataka, INDIA
[1]a.vinay@pes.edu
[2]mailbharath@gmail.com
[3]talktoprateek@gmail.com
[4]chandra.shikhar@gmail.com
[5]Department of Computer Science and Engineering, S J C Institute of Technology
Chikkaballapura, Karnataka, INDIA
[5]nissureddy@rediffmail.com

ABSTRACT

The technological advancements in filed of computer hardware, software, communication and multimedia have paved a way for increase in number of multimedia services over internet. One such multimedia service that is booming over past decade is the Video-on-Demand (VoD) System. Today, thousands of users are using internet based VoD systems. Hence, video streaming over the Internet has been one of the most prolific research areas for researchers working in the domain of multimedia networks. Most related to this work are the research efforts for the designing of video distribution systems that can support a large number of users. This is because the performance of the VoD system is the maximum number of concurrent clients that can access media by streaming from the server without causing the server to malfunction or degrade streaming quality. However there are two important system parameters that affect the performance of the system namely, the load and the bandwidth. As number of concurrent users increase the load increases. Similarly to accommodate the incoming requests the bandwidth requirements too increase. The objective of this paper is to address the problem of load balancing. To achieve this objective the scope of the work has been focused on two crucial parameters, load and bandwidth. Thus, the paper aims to devise a bandwidth aware load balancing and optimal bandwidth allocation strategies for VoD systems.

Keywords : Video-on-Demand (VoD), Load Balancing, Optimal Bandwidth Allocation, Multimedia.

I. INTRODUCTION

A VoD system consists of several servers and clients over a network. The video files are stored at the server and streamed to the clients on their request. Streaming high quality videos consumes higher bandwidth. As large number of users tries to access the systems at the same instance, the server(s) get heavily loaded. Gradually, the performance of the server starts degrading. There is an utmost need for development of effective techniques for the distribution of the load based on the bandwidth availability. Load balancing is a strategy to direct the video requests from different clients to the specific server in the server farm to in order to minimize processing time, minimize communication latencies, maximize resource utilization and throughput.

Apart from balancing the load in bandwidth aware fashion, optimal bandwidth allocation is also an essential requirement to effectively utilize the available bandwidth. Although many algorithms and techniques are proposed in this direction, they are independent. This paper focuses on two crucial system parameters namely, bandwidth and load that affect the system performance.

The rest of the paper is organized as follows section II deals with related work; section III describes the system model and its components; section IV describes the proposed techniques for load balancing and bandwidth allocation; section V describes the analytical model; section VI illustrates the results obtained from simulating the model by applying the proposed techniques and finally section VII draws conclusion of the work carried out.

II. RELATED WORK

Researchers have tackled the problem of generating cost-efficient VoD network designs using different optimization techniques like placement of replica servers, video objects or allocation of available resources to minimize cost. Solving the replica placement or video placement problems independently of the resource allocation problem usually leads to suboptimal solutions because the location of the replicas has a direct impact on the amount of resources required.

Liang-Teh Lee et al proposed a load balanced PC-Cluster for the VOD server system has been proposed. Two-Tier model is used in the systematic architecture, and PC-Cluster are used as storage system of the VOD server. The load balancing mechanism is based on the Least-Connection-First algorithm. Furthermore, a video placement strategy is also proposed in this paper to share and balance the loads among video servers in the cluster. Accompany with the dynamically adjusted files in each video level, a dynamically cyclical video replacement mechanism has been proposed to replicate and allocate video files for improving load balance of the system [1].

Yitzhak Birk work focuses on load-balancing for the purpose of providing throughput that is independent of viewing choices. At the inter-disk level, data striping is the obvious solution, but may lead to a quadratic growth of RAM buffer requirements with system size. At the intra-disk level, multi-zone recording results in variable disk throughput. Deterministic schemes for solving each problem are discussed, as well as their joint operation. Finally, efficient staging of data from tertiary storage devices to disk is shown to be possible [2].

W. Jaiphakdee et al proposed a method that focuses on data placement issue in heterogeneous tiers of storage devices storing YouTube videos. In addition to storage capability and current workload, the data placement algorithm also took into account future workload. Four data placement algorithms were used to distribute 16, 314 videos across 3-tier storage system in the experiment. The proposed algorithm resulted in a more balance workload distribution compared to round robin and random algorithms [3].

Jonathan C Lu et al proposed a mathematical model to analyze the performance of the VoD system. Further they designed a replication and video placement strategies for their proposed architecture. Finally they devised a load balancing strategy to decide whether request is to be served locally or remotely [4].

H S Guruprasad et al proposes a load balancing algorithm for a distributed VoD architecture using agents. The proposed approach groups a set of local proxy servers into a Local Proxy Server Group [LPG] for load balancing among the proxy servers. However the proposed algorithm always uses maximum number of channels between the proxy servers in a LPG and also between the CMS and the proxy servers of a LPG by allocating more channels to the more popular videos [5].

Jun Guo et al proposed a conjecture by suggesting that such a system may only be load balanced when the traffic wishing to access movies of the same type is uniformly distributed among all combination groups of disks enumerated in the system for the associated movie type. At this state of CLB, the RBP of the LBF system is minimized. However in practice CLB may not be always achievable [6].

Anant Nimkar et al, first formulate the video placement as an INLP optimization problem and note that the video placement problem to distribute the number of copies of each video to disks or arrays of disks is NP-hard. Then they propose greedy video placement and disk load balancing algorithms to minimize the static and dynamic loads of disks respectively. The greedy video placement along with round robin DLB improves the performance of the VoD system by 10% as compared to random video placement along with round robin DLB. All I/O streams in the group of streams except one will experience buffer hits. But a stream of the client may leave the session resulting in the wastage of memory as the video-on-demand system provides VCR-like functions [7].

Deepthi K.Madathil et al, propose a novel load-balancing and performance oriented static data placement strategy, called perfect balancing (PB) which can be applied to distributed storage subsystems in clusters to noticeably improve system responsiveness. The basic idea of PB is to balance the load across local disks and to minimize the discrepancy of service times of data on each disk simultaneously. But the proposed methodology does not work in fully dynamic environment [8].

Pushpendra Kumar Chandra et al propose an algorithm for a wide variety of workload conditions including I/O intensive and memory intensive loads. However, the CPU requirements of the system is minimum as the tasks which arrive are mostly video fetch tasks which require negligible system interaction but a lot of I/O consumption. The proposed load balancing algorithm tries to achieve the effective usage of global disk resources in the VOD cluster [9].

III. SYSTEM MODEL

VoD system can be modelled as centralized architecture, distributed architecture and hybrid architecture. In a centralized architecture, the central server handles requests from all the clients. To serve all the incoming requests, the server should have high computational and transmission capabilities. It is simple, easy to implement and maintain. But it has a single point of failure and is not scalable. Also as number of users increase server gets overloaded and performance degrades [10, 5]. In distributed approach, a collection of videos are located at dispersed sites across a network. Distributed approaches have improved scalability, greater fault tolerance and optimal resource utilization ability when compared to that of the centralized approach [11, 12].

Each of the above said architectures have some pros and cons. In order to build efficient architecture, recently architectures based on combinations of the standard ones have been proposed. Such architecture is referred to as hybrid architecture. The system used here is one such hybrid architecture.

The VoD system model for which the bandwidth aware load balancing and optimal bandwidth allocation techniques are proposed is depicted in the figure 1. The components of the architecture are Main Multimedia Server (MMS), Global Server Farm (GSF), Local Server Farm (LSF), Local Tracker (LT), Global Tracker (GT) and a set of Clients. The MMS consists of high end data storage with all the video files. The GSF consists of a set of intermediate peer servers and is associated with clusters through a GT. The servers of GSF in turn contain a subset of the videos that are stored on MMS. The principle used in this architecture for choosing the files to be stored on the peer server is the moderate popularity (around 80% to 90%).

Similarly, each LSF consists of a set of servers and is associated with a set of clients through a LT. The servers in the GSF contain only highly popular videos (around 90% to 95%). The popularity of a video is the measured through the hit rate of the video. Some literatures refer to this as Hot Video also.

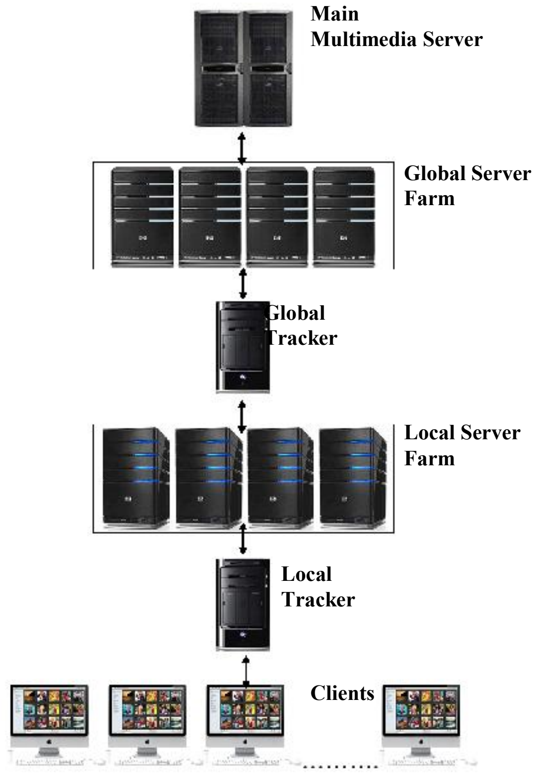

Figure 1. System model used for load balancing and bandwidth allocation

IV. PROPOSED METHOD

A. *Load Balancing*

Load balancing is the process by which incoming request can be distributed across multiple servers. Load balancing enhances the performance of the servers, leads to their optimal utilization and ensures that no single server is overwhelmed. Load balancing is particularly important for busy networks like Video-on-Demand (VoD), where it is difficult to predict the number of requests that will be issued to a server. Load balancing addresses some of the important requirements like scalability, high performance, high availability and disaster recovery. Load balancing technique can be either static or dynamic. If the load balancing method uses the current state information of the servers in its decisions, it is called dynamic load balancing technique. Otherwise, it is static load balancing static. The proposed architecture uses a dynamic load balancing strategy to balance the workload among the different servers at of CSF and GSF. The load balancing is done based on the current load and bandwidth status on the servers of different server farms. Always a minimum loaded and maximum bandwidth server with the requested video being present is selected. Thus, the probability of congestion is very low and the performance of the system enhances.

The load balancing strategy used is the dynamic load balancing. The load balancing and bandwidth allocation modules are located at tracker rather than at the server farms. This is because the tracker monitors and maintains the status information about the servers in the different server farms. Rather than just forwarding the request to the server farm, the tracker checks the bandwidth availability and balances the load.

Clients request Vreq arrives at the local tracker. Local tracker checks for the availability of the Vreq in its index table. If found, it marks the server containing the Vreq. Then, load status and bandwidth avail routines are called for each of the marked servers to return the current load on the server and remaining bandwidth available for each of the marked servers. Normally, the local tracker assigns the first server to be the min load server & max bandwidth server. Starting with the second server, it compares the load of the min load server with each server. If a server with load less than min load server is found, it assigns that server as min load sever. Similarly, the local tracker compares the bandwidth of max bandwidth server each server. If a server with bandwidth greater than max bandwidth server is found, assign that server as max bandwidth sever.

Finally, the local tracker checks the bandwidth requirement and the streaming time required for serving the Vreq and forwards the request to the appropriate server i.e. Min Load Server or Max Bandwidth Server.

It then calls the Optimal Bandwidth Allocation routine and starts streaming the video. Suppose, Vreq is not found at local tracker, it forwards the request to the global tracker. At, global tracker also the same sequence of operations is repeated. If the Vreq is not found at the global tracker also, it is forwarded to main multimedia server, where the probability of the Vreq is high. If not found at main multimedia server, the request is rejected.

B. Bandwidth Aware Load Balancing Pseudo Code

```
Local_Tracker (Cid, Vreq)
{
   if (Vreq available in LSF server)
   {
       return (LFS_list  with Vreq);
    }
   MinLS_i = LServer_1;
   MaxBS_i= Lserver_1;
   MinLS = Load_Status (LServer_1);
   MaxBS = Bandwidth_Avail (LServer_1)
   for each Lserver_2 to Lserver_N in LFS_list
   {
     if (MinLS < Load_Status (Lserver_i))
     {  }
      Else
     {
          MinLS_i = Lserver_i;
          MinLS= Load_Status(MinLS_i);
      }
      if (MaxBS < Bandwidth_Avail (Lserver_i))
     {  }
      Else
     {
          MaxBS_i = Lserver_i;
          MaxBS= Bandwidth_Avail (MaxBS_i);
      }
    }
    if ( Vreq.MaxBandwidth == Banwidth_Avail
         (MaxBS_i) &&  Vreq.Bursttime == Load_Status
          (MinLS_i))
              Forward_Request (Cid, Vreq, MaxBS_i);
   Else
    if ( Vreq.MaxBandwidth == Banwidth_Avail
        (MaxBS_i)   && Vreq.Bursttime < Load_Status
        (MinLS_i))
              Forward_Request (Cid, Vreq, MaxBS_i);
   Else
    if ( Vreq.MaxBandwidth < Banwidth_Avail (MaxBS_i)
         && Vreq.Bursttime < Load_Status (MinLS_i))
              Forward_Request (Cid, Vreq, MinLS_i);
   Else
   if ( Vreq.MaxBandwidth > Banwidth_Avail (MaxBS_i)
         &&  Vreq.Bursttime <= Load_Status (MinLS_i))
              Forward_Request (Cid, Vreq, MinLS_i);
   Else
   if ( Vreq.MaxBandwidth <= Banwidth_Avail
      (MaxBS_i) &&  Vreq.Bursttime > Load_Status
      (MinLS_i))
              Forward_Request (Cid, Vreq, MaxBS_i);
   Else
    if ( Vreq.MaxBandwidth < Banwidth_Avail (MaxBS_i)
         && Vreq.Bursttime == Load_Status (MinLS_i))
              Forward_Request (Cid, Vreq, MinLS_i);
  Else
   if ( Vreq.MaxBandwidth > Banwidth_Avail (MaxBS_i)
         &&  Vreq.Bursttime > Load_Status (MinLS_i))
              Global_Tracker (Cid, Vreq);
}

Load_Status (Lserver)
{
     return (Burst time of all Vreq in Queue)
}
Bandwidth_Avail (Lserver)
{
     return (bandwidth);
}

Forward_Request (Cid, Vreq, MaxBS)
{
    MaxBS.Vreq_Cnt++;
    MaxBS.Bursttime+=Vreq.Bursttime;
    Bandwidth_allocation (Vreq, MaxBS);
    MaxBS.Stream(Cid,Vreq);
}

Forward_Request (Cid, Vreq, MinLS)
{
    MinLS.Vreq_Cnt++;
    MinLS.Bursttime+=Vreq.Bursttime;
    Bandwidth_allocation (Vreq, MinLS);
    MinLS.Stream(Cid.Vreq);
}
```

The routines for the global tracker are similar to that of the local tracker, except for the following condition and forward request mechanism

```
 if ( Vreq.MaxBandwidth > Banwidth_Avail (MaxBS_i)
         &&  Vreq.Bursttime > Load_Status (MinLS_i))
    Forward_Request (Cid, Vreq, Multimedia_Server);

Forward_Request (Cid, Vreq, Multimedia Server)
{
    Multimedia Server.Vreq_Cnt++;
    Multimedia Server.Bursttime+=Vreq.Bursttime;
    Bandwidth_allocation (Vreq, Multimedia Server);
    Multimedia Server.Stream(Cid.Vreq);
}
```

C. Optimal Bandwidth Allocation Pseudo Code

```
Bandwidth_allocation(Vreq, MaxBS)
{
  If (Vreq.MaxBandwidth == Banwidth_Avail (MaxBS))
          Allocate maximum available bandwidth
  Else
  If (Vreq.MaxBandwidth < Banwidth_Avail (MaxBS))
          Allocate required bandwidth
}

Bandwidth_allocation(Vreq, MinLS)
{
  If (Vreq.MaxBandwidth >= Banwidth_Avail (MaxBS))
          Allocate required bandwidth
  Else
  If (Vreq.MaxBandwidth < Banwidth_Avail (MaxBS))
          Allocate minimum bandwidth
 }

Bandwidth_allocation(Vreq, Multimedia Server)
{
          Allocate maximum required bandwidth
 }
```

V. ANALYTICAL ANALYSIS

Let C = {C_1, C_2, C_3,…, C_N} be the set of clients, V= {V_1, V_2, V_3,…., V_N} be the set of videos requested by the clients, B = {B_1, B_2, B_3, …, B_N} be the set of bandwidths for these videos and BT = {BT_1, BT_2, BT_3, ….., BT_N}be the set of burst time required for processing each of the videos in the set V.

Let $BLSFS_1$, $BLSFS_2$, $BLSFS_3$, $BLSFS_4$ be the bandwidth allocated to servers $LSFS_1$, $LSFS_2$, $LSFS_3$, …. , $LSFS_N$ in the LSF. Then the total bandwidth allocated to LSF is

$$BA_{LSF} = \sum_{S=1}^{N} BLSFSi \qquad (1)$$

If LSFS1V = {$LSFS_1V_1$, $LSFS_1V_2$, $LSFS_1V_3$, …, $LSFS_1V_N$} are the subset of the videos of V that are waiting in or being served by the server $LSFS_1$ of LSF. Then total bandwidth utilized by the $LSFS_1$ is

$$B_{LSFS1} = \sum_{V=1}^{N} BLSFS1Vi \qquad (2)$$

Similarly, the load on LSFS1 is given by,

$$L_{LSFS1} = \sum_{V=1}^{N} BTLSFS1Vi \qquad (3)$$

Where $L_{LSFS1} \leq MAXL_{LSFS1}$

The, equation (2) holds good for all the servers in the LSF. At any instance of time, the total bandwidth consumed by all the servers in the LSF is

$$BU_{LSF} = \sum_{S=1,V=1}^{N} BLSFSiVi \qquad (4)$$

Where $BU_{LSF} \leq BA_{LSF}$

Similarly, the total load on all servers in LSF is

$$L_{LSF} = \sum_{S=1,V=1}^{N} BTLSFSiVi \qquad (5)$$

Where $L_{LSF} \leq MAXL_{LSF}$

Let $BGSFS_1$, $BGSFS_2$, $BGSFS_3$, …, $BGSFS_4$ be the bandwidth allocated to servers $GSFS_1$, $GSFS_2$, $GSFS_3$, …. , $GSFS_N$ in the GSF. Then the total bandwidth allocated to GSF is

$$BA_{GSF} = \sum_{S=1}^{N} BGSFSi \qquad (6)$$

GSFS1V = {$GSFS_1V_1$, $GSFS_1V_2$, $GSFS_1V_3$, …, $GSFS_1V_N$} be a subset of the videos of V that have been forwarded to the GSF and are waiting in or being served by the server $GSFS_1$ of GSF, then total bandwidth utilized by the $LSFS_1$ is

$$BG_{SFS1} = \sum_{V=1}^{N} BGSFS1Vi \qquad (7)$$

Similarly, the load on $GSFS_1$ is given by,

$$L_{GSFS1} = \sum_{V=1}^{N} BTGSFS1Vi \qquad (8)$$

Where $L_{GSFS1} \leq MAXL_{GSFS1}$

The equation (7) holds good for all the servers in the GSF. At any point in time, the total bandwidth consumed by all the servers in the GSF is

$$BU_{GSF} = \sum_{S=1,V=1}^{N} BGSFSiVi \qquad (9)$$

Where $BU_{GSF} \leq BA_{GSF}$

At any instance of time, the total load on all the servers in the GSF is

$$L_{GSF} = \sum_{S=1,V=1}^{N} BTGSFSiVi \qquad (10)$$

Where $L_{GSF} \leq MAXL_{GSF}$

Let BA_{MS} be the bandwidth allocated to multimedia server. If MSV = {MSV_1, MSV_2, MSV_3, …, MSV_N} are the subset of the videos of V that have been forwarded to MS and are waiting in or being served by MS. Then total bandwidth utilized by the MS is

$$BU_{MS} = \sum_{V=1}^{N} BMSVi \qquad (11)$$

Where $BU_{MS} \leq BA_{MS}$

Similarly, the total load on MS is given by

$$L_{MS} = \sum_{V=1}^{N} BTMSVi \qquad (12)$$

Where $L_{MS} \leq MAXL_{MS}$

The total load on the architecture is

$$L_{Total} = L_{LSF} + L_{GSF} + L_{MS} \qquad (13)$$

Then, the total bandwidth allocated to the architecture is

$$BA_{Total} = BA_{LSF} + BA_{GSF} + BA_{MS} \qquad (14)$$

Similarly, the total bandwidth required by the architecture to serve the request is

$$BU_{Total} = BU_{LSF} + BU_{GSF} + BU_{MS} \qquad (15)$$

Finally, the current available bandwidth for the architecture is

$$BAvail_{Total} = BA_{Total} - BU_{Total} \qquad (16)$$

Let $ICost_{LSF}$ be the total infrastructure cost of the servers in LSF, $ICost_{GSF}$ be the total infrastructure cost of the servers in GSF and $ICost_{MS}$. Then total infrastructure cost for the architecture is denoted by

$$ICost_{TOT} = \sum_{i=1}^{N} ICost_{LSFSi} + ICost_{GSFSi} + ICost_{MS} \qquad (17)$$

Let $BCost_{LSF}$ be the total infrastructure cost of the servers in LSF, $BCost_{GSF}$ be the total infrastructure cost of the servers in GSF and $BCost_{MS}$. Then total cost of the bandwidth consumed by the architecture is denoted by

$$BCost_{TOT} = \sum_{i=1}^{N} BCost_{LSFSi} + BCost_{GSFSi} + BCost_{MS} \qquad (18)$$

Thus, the total cost incurred in implementing the proposed technique is the sum of total infrastructure cost and total bandwidth cost. This total cost is represented as

$$Cost_{TOT} = ICost_{TOT} + BCost_{TOT} \qquad (19)$$

VI. SIMULATION RESULTS

A. *Simulation Model*

There are no standard simulators available for modelling VoD systems. Hence a simulator based on the proposed system model was designed and implemented in Java. The simulation model consisted of a main multimedia server, four servers in Local Server Farm, three servers in Global Server Farm, two trackers. Some of the assumptions made were:

i) The user requests obey Zipf distribution.
ii) The average interarrival time between consecutive requests is 1/λ.
iii) The sizes of the video files are uniform in the range of 300 MB to 500 MB.

B. *Simulation Result*

The following results were obtained from the simulation. Figures 2 to 5 illustrates the total number of requests made and load on each server of the LSF. As illustrated in these figures, if a video replica is found in two or more servers the load is almost evenly balanced. However if the replica is in only one of the servers, then load on that server is one request more than that of other servers.

Similarly figures 6 to 8 illustrates the total number of requests made and load on each server of the GSF. The concept used to balance load among GSF's servers is same as that of the technique used for load balancing at CSF's servers. Finally figure 9 illustrates the load on the MS. The load of MS is almost the same of that of a single server in GSF. Thus, the load on the MS is less.

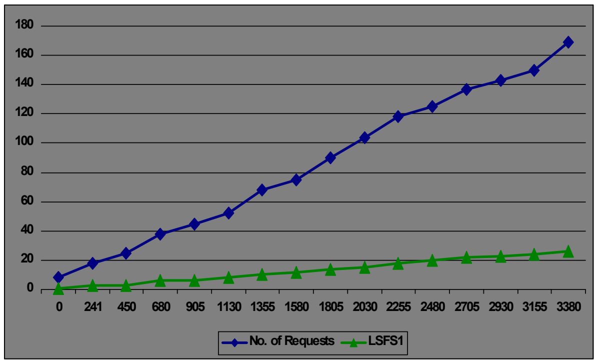

Figure 2. Load on local server farm's server 1

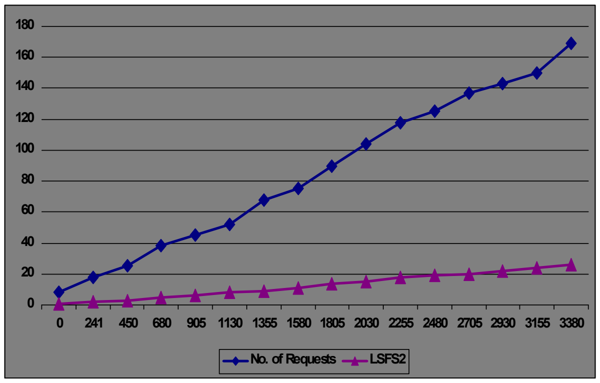

Figure 3. Load on local server farm's server 2

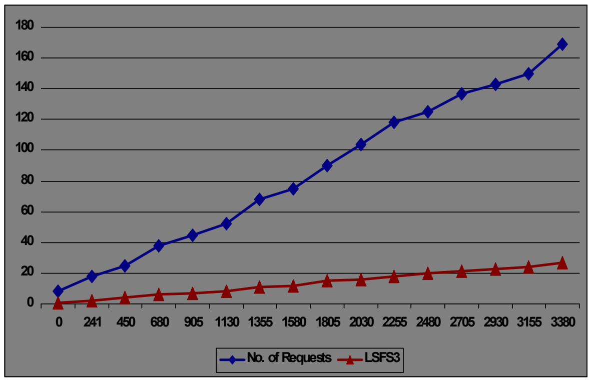

Figure 4 Load on local server farm's server 3

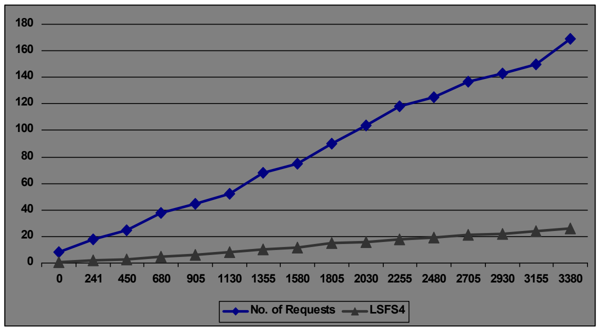

Figure 5 Load on local server farm's server 4

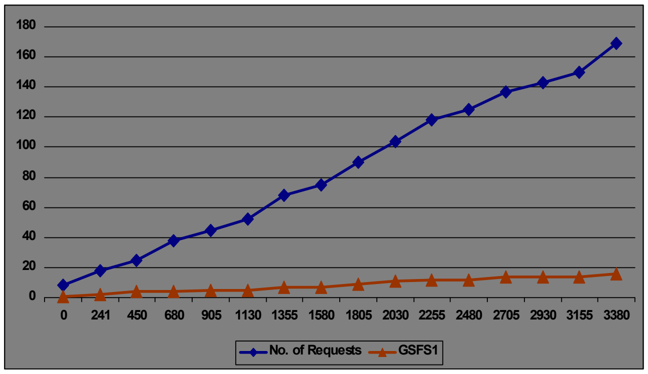

Figure 6 Load on global server farm's server 1

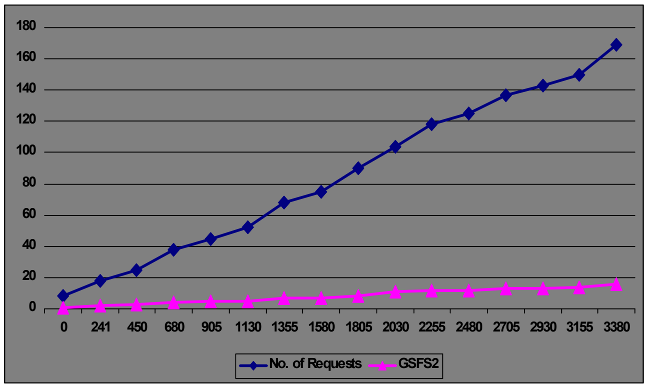

Figure 7 Load on global server farm's server 2

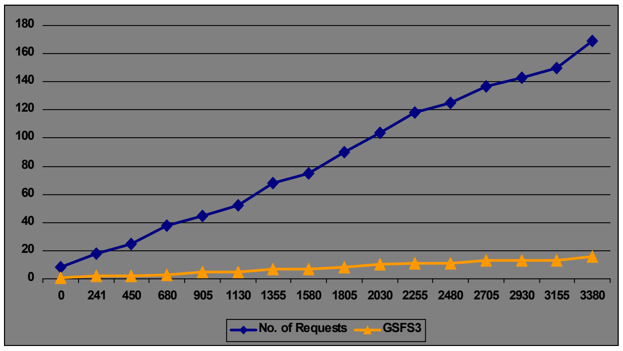

Figure 8 Load on global server farm's server 3

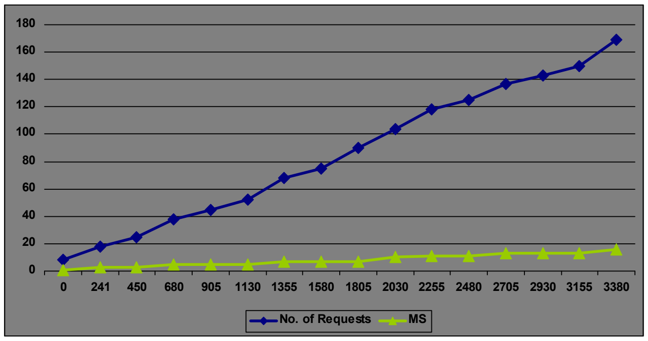

Figure 9 Load on main server

VII. CONCLUSIONS

This paper proposed a unique bandwidth aware load balancing and optimal bandwidth allocation techniques for VoD system. The trackers implement the load balancing, bandwidth allocation routines and request forward routines. Here load balancing is achieved by distributing the load on different servers in the server farms. Optimal bandwidth allocation is achieved through allocation of bandwidth based on current bandwidth availability and load on servers. The performance evaluations show that around 92% of the load balancing is achieved at LSF and GSF and bandwidth utilization is to the core.

Work can be further extended to minimize the number of servers deployed in the Local Server Farm as well as Global Server Farms with out affecting the performance of the system. The proposed load balancing strategy tries to balance load based on the availability of video i.e. if requested video is present in the servers of LSF, it tries to balance load among servers of LSF only even when the servers of LSF have many requests to serve. Thus, the work can be modified to provide system wide load balancing rather than just at the server groups. Optimal replication strategies need to be devised and deployed to handle huge concurrent requests.

ACKNOWLEDGMENT

The authors would like to express heartfelt sincere gratitude to Prof. Shylaja S S, Head, Department of Information Science and Engineering, PESIT, for providing constant support, valuable suggestions and guidance for this work. The authors extend sincere gratitude to Dr S Natarajan, Professor, Department of Information Science and Engineering, PESIT, Dr. K N Balasubramanya Murthy, Principal & Director, PESIT, for their motivation and PES Management for all the support extended in carrying out this research activity.

REFERENCES

[1] Liang-Teh Lee, Hung-Yuan Chang, Der-Fu Tao, and Siang-Lin Yang, " A Load Balanced PC-Cluster for Video-On-Demand Server Systems", In the proccedings of International Journal of Grid and Distributed Computing, pp. 63 – 70, 2005.

[2] Yitzhak Birk, "Deterministic Load-Balancing Schemes for Disk-Based Video-On-Demand Storage Servers", In the proceedings of IEEE Mass Stroage Symposium, pp. 1- 10, 1995.

[3] W. Jaiphakdee, C. Srinilta, ""Dynamic Load Balancing of Short Videos in Heterogeneous Storage Environment", In the proceedings of the International MultiConference of Engineers and Computer Scientist, Vol I, pp. 1- 5, 2010

[4] Jonathan C Lu, Chung Han Chen, "Performance Study on Video Placement and Load Balancing of distributed Video-on-Demand Systems", In the proceedings of International Conference on Communications, pp. 12- 18 , 2004.

[5] H S Guruprasad, H D Maheshappa, "Dynamic Load Balancing Architecture for Distributed VoD using Agent Technology", In the proceedings of International Journal of Computer Science & Security (IJCSS), Volume (2) : Issue 5, pp. 13-22, 2008.

[6] Jun Guo, Eric W. M. Wong, Sammy Chan, Peter Taylor, Moshe Zukerman, and Kit-Sang Tang, "Combination Load Balancing for Video-on-demand Systems", In the proceedings of International

[7] Anant Nimkar, Chittaranjan Mandal, Chris Reade, "Video Placement and Disk Load Balancing Algorithm for VoD Proxy Server", In the proceedings of 3rd IEEE international conference on Internet Multimedia Services Architecture and Applications, pp. 141-146, 2009.

[8] Deepthi K.Madathil, Rajani B. Thota, Paulina Paul, Tao Xie, "A Static Data Placement Strategy towards Perfect Load-Balancing for Distributed Storage Cluster", In the proceedings of IEEE International Advance Computing Conference, pp. 408-412, 2009

[9] Pushpendra Kumar Chandra, Bibhudatta Sahoo, "Performance Analysis of Load Balancing Algorithms for cluster of Video on Demand Servers", In the proceedings of IEEE International Advance Computing Conference, pp. 408-412, 2009

[10] Giovanni Branca, Thomas Schierl, and Luigi Atzori, "Theoretical Models for Video on Demand Services on Peer-to-Peer Networks", In the International Journal of Digital Multimedia Broadcasting, Volume 2009, Article ID 263936, 8 pages, 2009

[11] Carlos Eduardo Coelho Freire Batista, Tiago Lima Salmito, Luiz Eduardo Cunha Leite, Guido Lemos de Souza Filho, Glêdson Elias da Silveira, "Big Videos on Small Networks, A Hierarchical and Distributed Architecture for a Video on Demand Distribution Service",In the proceedings of Multimedia Services Access Networks, pp. 15 -19, 2005.

[12] K. Tsang, Sai Kwok, "Video Management in Commercial Distributed Video on Demand (VoD) Systems", In the proceedings of Pacific Asia Conference on Information Systems, 2000.