EFFICIENT CHANNEL ALLOCATION BASED ON THE WEIGHTS OF VIDEOS USING AGENTS

[1]H S GURUPRASAD, [2]DR. H D MAHESHAPPA

[1]Research Scholar, Dr. MGR University, Assistant Professor & HOD, Department of ISE, BMSCE, Bangalore, India.

[2]Professor & HOD, Department of E & C, RevaITM, Bangalore, India

E-mail: hs_gurup@yahoo.com hdmappa@gmail.com

ABSTRACT

This paper proposes a load sharing algorithm with higher priority given to the videos with higher weights using agent technology. A mobile agent periodically updates the popularity and the weight of the videos which is used for efficiently allocating the channels. The proposed approach reduces the load on the central multimedia server, allocates more channels for higher weight videos and maximizes the channel utilization between the neighboring proxy servers and the central multimedia server. The simulation results prove the load sharing among the neighboring proxy servers and hence reducing the load on central multimedia server, maximum channel utilization and more channel allocation for weight videos.

Keywords: *Channel Allocation, Weights, Mobile Agents, Distributed VoD, Popularity, Videos, Load sharing, Algorithm.*

1. INTRODUCTION

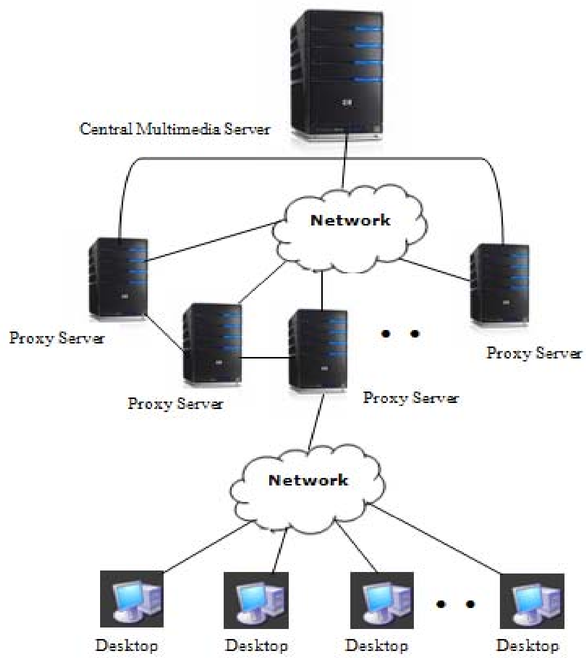
Agents are autonomous programs which can understand an environment, take actions depending upon the current status of the environment using its knowledge base and also learn so as to act in the future. Autonomy, reactive, proactive and temporally continuous are mandatory properties of an agent. The other important properties are commutative, mobile, learning and dependable. These properties make an agent different from other programs. The agents can move around in a heterogeneous network to accomplish their assigned tasks. The mobile code should be independent of the platform so that it can execute at any remote host in a heterogeneous network [1, 2, 8, 10].

A video-on-demand system can be designed using any of the 3 major network configurations – centralized, networked and distributed. In a centralized system configuration, all the clients are connected to one central server which stores all the videos. All the client requests are satisfied by this central server. In a network system configuration, many video servers exist within the network. Each video server is connected to a small set of clients and this video server manages a subset of the videos. In a distributed system configuration, there is a central server which stores all the videos and smaller servers are located near the network edges. When a client requests a particular video, the video server responsible for the requests ensures continuous playback for the video [3].

In [5], Tay and Pang have proposed an algorithm called GWQ [Global Waiting Queue] which shares the load in a distributed VoD system and hence reduces the waiting time for the client requests. This load sharing algorithm balances the load between heavily loaded proxy servers and lightly loaded proxy servers in a distributed VoD. They assumed that videos are replicated in all the servers and videos are evenly required, which requires very large storage capacity in the individual servers. In [6], Sonia Gonzalez, Navarro, Zapata proposed a more realistic algorithm for load sharing in a distributed VoD system. Their algorithm maintains small waiting times using less storage capacity servers by allowing partial replication of videos. The percentage of replication is determined by the popularity of the videos. Proxy servers are widely used in multimedia networks to reduce the load on the central server and to serve the client requests faster.

In this paper, we propose a load sharing algorithm and VoD architecture for distributed VoD system which gives higher preference to the videos which have higher weights. This architecture consists of a central multimedia server which is connected to a group of proxy servers and these proxy servers are assumed to be connected by fiber optic cables in a ring fashion. The rest of the paper is organized as follows: Section 2 presents the proposed architecture, section 3 presents the proposed algorithm, Section 4 presents the simulation model, Section 5 presents the simulation results and discussion, Section 6 finally concludes the paper and further work.

2. PROPOSED ARCHITECTURE



In the proposed architecture, a Central Multimedia Server [CMS] is connected to a group of proxy servers. All these proxy servers are connected through fiber optic cables in the form of a ring. Each proxy server is connected to a set of clients (users). The video content that is currently requested by its clients is stored in each proxy server.

Consider n videos $v_1, v_2, \cdots, v_n$. The mean arrival rates for the videos are $\lambda_1, \lambda_2, \cdots, \lambda_n$ respectively. There are m server channels. The total arrival rate of all the videos is $\lambda = \sum_{i=1}^{n} \lambda_i$. The probability of receiving a user request for a video v_i is given by $P_i = \lambda_i / \lambda$ for $i = 1,2,\cdots,n$.

There are 3 classes of customer's c_1, c_2 and c_3 and the profit associated with each class is p_1, p_2 and p_3 respectively. Let $k_1, k_2, \cdots, k_n$ be the number of requests for the n videos $v_1, v_2, \cdots, v_n$. Also, $k_i = k_{i1} + k_{i2} + k_{i3}$, where k_{i1} is the number of requests of class 1, k_{i2} is the number of requests of class 2 and k_{i3} is the number of requests of class 3. Now, the weight associated with each video is $w_i = k_{i1} * p_1 + k_{i2} * p_2 + k_{i3} * p_3$.

The CMS contains all the N number of videos. The distribution of the videos is done as follows: The popularity of a video depends on the number of users requesting for this video. i.e. higher the number of user requests for a video, higher the popularity of the video. The number of requests to a video is done according to Zipf's law. Initially, all the N videos are arranged according to the decreasing order of their popularity. The first k videos are selected from this popularity based sorted list and stored in each proxy server. The remaining videos are stored depending on the local popularity in the proxy servers.

The CMS periodically invokes a mobile agent which travels across the Proxy Servers and updates the video popularity and weight profile at the Proxy servers and the CMS.

When a request for a video arrives at the proxy server [PS], one of the following 4 cases happens:

- The requested video may be present in the PS

- The requested video is not present in the PS, but is present in either left neighbor Proxy Server[LNPS] or Right neighbor Proxy Server[RNPS]
- The requested video is present in both LNPS and RNPS
- The requested video is not present in LNPS and RNPS

If the requested video is present in the PS, then the real time transmission of the video starts immediately from the PS to the client. If the requested video is not present in the PS, then the weight of the requested video is calculated as explained above.

If the requested video is present only in LNPS and not in RNPS, then if more numbers of channels are allocated for higher weight videos b/w LNPS & PS, we select the path LNPS-PS, otherwise we select the path CMS-PS. If the requested video is the first request, then all the channels are allocated for this video. Otherwise, the weight of the requested video is checked. If the requested video has more weight than the videos being streamed in the channels, then more number of channels is allocated for the requested video by deallocating channels from the lesser weight videos being streamed in the channels. Otherwise, appropriate numbers of channels are allocated depending on its weight and the weight of the videos streamed in the channels and finally the channel allocation of the other videos are dynamically adjusted.

If the requested video is present only in RNPS and not in LNPS, then if more numbers of channels are allocated for higher weight videos b/w RNPS & PS, we select the path RNPS-PS, otherwise we select the path CMS-PS. If the requested video is the first request, then all the channels are allocated for this video. Otherwise, the channel allocation is done as the same way as when the video is found in LNPS only.

If the requested video is present in both LNPS and RNPS, then we check the number of channels allocated for higher weight videos b/w LNPS & PS, RNPS & PS and CMS & PS. We select one of these three paths, in which more number of channels is allocated for higher weight videos. If the requested video is the first request, then all the channels are allocated for this video. Otherwise, the channel allocation is done as the same way given above.

If the requested video is not present in LNPS and RNPS, then we select the path b/w CMS-PS. If the requested video is the first request, then all the channels are allocated for this video. Otherwise, the channel allocation is done as the same way given above.

If channel allocation was not possible between LNPS-PS, RNPS-PS and CMS-PS, then the requested video is rejected.

3. PROPOSED ALGORITHM

[Nomenclature: PS: Proxy Server
CMS: Central Multimedia Server
LPS: Left Neighbor Proxy Server
RPS: Right Neighbor Proxy Server
NOVS(x, y): Number of videos being streamed between x and y]

When a request for a video arrives at a particular time t, do the following:

```
If the requested video is present in PS
        Start streaming the video from PS
else
        Dynamic channel allocation is done
        according to the algorithm DCA
        If the channels are allocated
        then
                the video is downloaded and
                stored at PS and streamed to the
                requested client
        else
                the request is rejected
```

Algorithm DCA

```
Begin
If the requested video is present in LPS only
  then
    call Channel_allocation (LPS, PS)
  else
    if the requested video is present in RPS only
    then
      call Channel_allocation (RPS, PS)
    else
      if the requested video is present in both LPS
      and RPS
      then
           if (NOVS (LPS, PS)>NOVS (RPS, PS))
           then
                call Channel_allocation (LPS, PS)
           else
                call Channel_allocation (RPS, PS)
       else
           call Channel_allocation (CMS, PS)
End
```

Algorithm Channel_allocation(X, PS)

Begin
When a request R_n arrives at time t,

If no other requests are currently being served
then
assign all the channels to the request Rn
else
identify the only request R_i with more than one channel assigned
if found
then
if weight of the video of R_i > weight of the video of Rn
then
free one channel from the request R_n
assign this channel to the request R_n
else
no channels are assigned to R_n
End
Begin
When a request R_c is completed at time t,
If there are other requests being served
then
Find the request R_i with the highest weight
Assign all the channels of R_c which have become free to R_i
Else
Mark all the channels as free channels
End

4. SIMULATION MODEL

The simulation model consists of a single Central Multimedia Server [CMS], and a few proxy servers. The following are the assumptions made in the model:

The user requests for the video follows Zipf's law of distribution. The sizes of the videos are uniformly distributed over a range 300MB to 800MB. The number of channels b/w PS & LNPS, b/w PS & RNPS and b/w PS & CMS are assumed to be 60.

The performance parameters are load sharing among the proxy servers, more channel allocation for the videos in more demand by the users and maximum channel utilization b/w PS & LNPS, PS & RNPS and PS & CMS.

5. RESULTS AND DISCUSSION

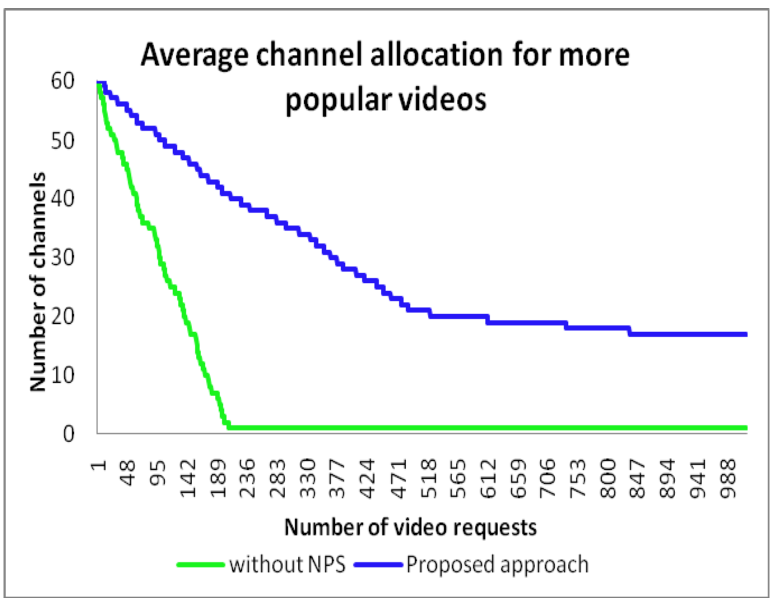


Fig. 1. Average channel allocation for more popular videos

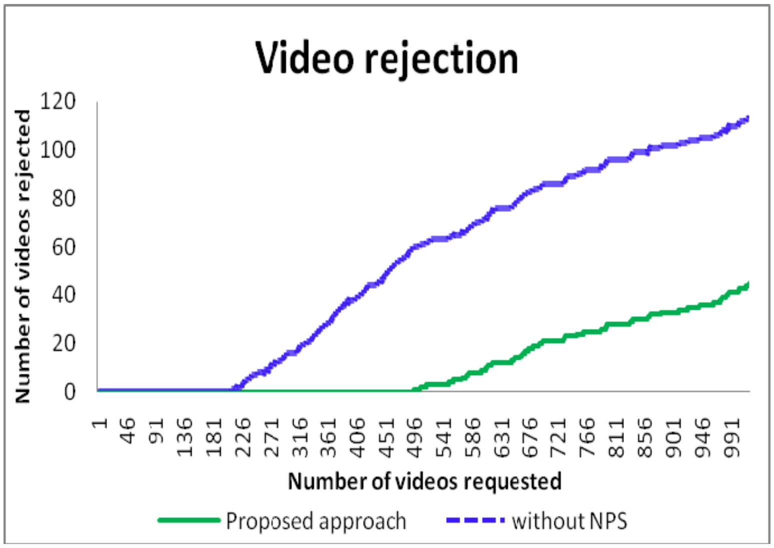


Fig. 2. Video Rejection

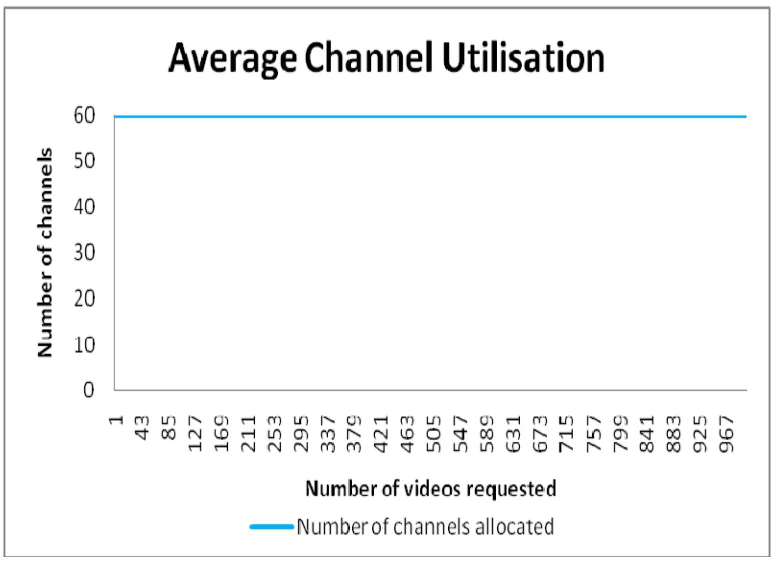


Fig 3. Average Channel Utilisation

The results presented are an average of several simulations conducted on the model. The sizes of the videos are taken in the range 300MB to 800MB. The number of proxy servers considered is 5 and each simulation is carried out for 10000 seconds.

Fig 1 shows the average channel allocation for more popular videos in our approach and in the approach without neighboring proxy servers given in [1]. Since, there are neighboring proxy servers in our approach, the videos need not be downloaded from the CMS always. If a video is present in a neighboring proxy server, it can be downloaded from that proxy server. In the approach given in [1] without NPS [Neighboring proxy servers], when the video is not found in the PS, it has to be downloaded from the CMS always. This puts a lot of load on the channels between CMS and PS and they get exhausted and then the video requests are rejected. Thus more channels are allocated to the popular videos in our approach than in the approach given in [1] as shown in Fig1.

Fig 2 shows the number of videos rejected. In the proposed approach, the videos are downloaded from the neighboring proxy servers if present, and downloaded from CMS if they are not present in the neighboring proxy servers. This reduces the demand for the channels between CMS and PS and hence the number of videos rejected is less and rejection starts after the number of video requests become higher. In the approach given in [1], there is a lot of demand for the channels between CMS and PS. Thus the number of videos rejected is more and the video rejection starts after the number of video requests become quite moderate as shown in Fig 2.

The average channel utilization is always maximum because all the channels are allocated among the videos being streamed in both cases as shown in Fig 3.

6. CONCLUSION

In this paper, we have concentrated on the load sharing among the proxy servers by giving higher priority for higher weight videos using agents. The simulation shows promising results. The algorithm always uses maximum number the channels between the neighboring proxy servers and the central multimedia server by allocating more channels to the higher weight videos so that they are streamed faster. Further work is being carried out to investigate load balancing by grouping a set of local proxy servers.

REFERENCES

[1] R Ashok kumar, H S Guru Prasad, H D Maheshappa, Ganesan, "Mobile Agent Based Efficient Channel Allocation For VoD", *IASTED International Conference on Communication Systems & Applications [CSA 2006*] 3rd -5th July, 2006, Banff, CANADA

[2] H S Guruprasad, H D Maheshappa, "Dynamic Load Sharing Policy in Distributed VoD using agents", International Journal of Computer Science and Network Security, Vol 8, No 10, 2008 pp: 270-275.

[3] Santosh Kulkarni "Bandwidth Efficient Video on Demand Algorithm (BEVA) " *10th International conference on Telecommunications*, Vol 2 pp 1335-1342, 2003

[4] Hongliang Yu, Dongdong Zheng, Ben Y. Zhao, Weimin Zheng, "Understanding User Behavior in Large-Scale Video-on-Demand Systems", *Proceedings of the 2006 EuroSys conference*, Volume 40 , Issue 4 (October 2006), PP: 333 - 344

[5] Y.C Tay and HweeHwa Pang, "Load Sharing in Distributed Multimedia-On-Demand Systems", *IEEE Transactions on Knowledge and data Engineering*, Vol.12, No.3, May/June 2000.

[6] S. Gonzalez, A. Navarro, J. Lopez and E.L. Zapata, "Load Sharing in Distributed VoD (Video on Demand) Systems". *Int'l Conf. on Advances in Infrastructure for e-Business, e-Education, e-Science, and e-Medicine on the Internet (SSGRR 2002w*), L'Aquila, Italy, January 21-27, 2002.

[7] Meng Guo and Mostafa H. Ammar and Ellen W. Zegura, "Selecting among Replicated Batching Video-on-Demand Servers", *Proceedings of the 12th International Workshop on Network and Operating System Support for Digital Audio and Video*, pp 155—163, 2002

[8] Mohammed A. M. Ibrahim, "Distributed Network Management with Secured Mobile Agent Support", *International Conference on Hybrid Information Technology (ICHIT'06)*, 2006

[9] Frederic Thouin, Mark Coates, Dominic Goodwill, "Video-on-Demand Equipment Allocation," *Fifth IEEE International Symposium on Network Computing and Applications (NCA'06*), pp 103-110, 2006

[10] S S Manvi and P Venkataram, "Mobile Agent based online Bandwidth allocation Scheme in Multimedia Communications", *IEEE GLOBECOM 2001 Conference* USA

[11] A Dan, D Sitaram and P Shabuddin, "Dynamic batching policies for an on demand video server ", *Multimedia Systems*, pp 51-58, 1996

[12] H S Guruprasad, H D Maheshappa, "Dynamic Load Balancing Architecture for Distributed VoD using Agent Technology", International Journal of Computer Science and Security, Vol(1),Issue(5) Nov 10, 2008 pp: 13-22.

H S Guruprasad is an Assistant Professor and Head of the Department of Information Science & Engineering, BMS College of Engineering, Bangalore, India. He is an Engineering Graduate from Mysore University and did his Post Graduation at BITS, Pilani, India in 1995. He has vast experience in teaching and has guided many post graduate students. His research interests include multimedia Communications, distributed systems, computer networks and agent technology.

Dr. H.D. Maheshappa Graduated in Electronics & Communication Engineering from University of Mysore, India in 1983. He has a Post Graduation in Industrial Electronics from University of Mysore, India in 1987. He holds a Doctoral Degree in Engineering from Indian Institute of Science, Bangalore, India, since 2001. He is specialized in Electrical contacts, Micro contacts, Signal integrity interconnects etc. His special interests in research are Bandwidth Utilization in Computer Networks. He has been teaching engineering for UG & P G for last 25 years. He served various engineering colleges as a teacher and at present he is a Professor & Head of the Department of Electronics & Communication in Reva Institute of Technology & Management, Bangalore, India. He has more than 35 research papers in various National and International Journals & Conferences. He is a member of IEEE, ISTE, CSI & ISOI. He is a member of Doctoral Committee of Coventry University UK. He has been a Reviewer of many Text Books for the publishers McGraw-Hill Education (India) Pvt., Ltd, Chaired Technical Sessions, National Conferences and also has served on the advisory and technical national conferences.