Data Allocation Algorithms for Distributed Video Servers*

James Z. Wang
School of Electrical Engineering & Computer Sci.
University of Central Florida
Orlando, FL 32816-2362, USA
zwang@cs.ucf.edu

Ratan K. Guha
School of Electrical Engineering & Computer Sci.
University of Central Florida
Orlando, FL 32816-2362, USA
guha@cs.ucf.edu

ABSTRACT

In this paper, We discuss the server level data allocation problems in the distributed Video-on-Demand systems. We proposed two data allocation algorithms, Bandwidth Weighted Partition (BWP) algorithm and Popularity Based (PB) algorithm, based on the bandwidth and storage capacity limits of the distributed multimedia servers. We compare those two algorithms with the traditional Round Robin (RR) algorithm. The analysis and simulation studies show that PB algorithm is a simple and practical video data allocation algorithm for the distributed video servers. It provides near optimal system performance in any system condition.

1. INTRODUCTION

There are many studies reported on data management for multimedia servers [1, 2, 3, 4, 5, 6]. However, only a few of them deal with the distributed multimedia environment [5, 6]. In this paper, we study the problems associated with the data allocation among the distributed multiple video servers located in different geographical locations across a WAN environment. Recent works discussing the data allocation issues in the distributed multimedia environment have focused on how to manage the multimedia data among the different storage devices [5, 6]. The objective of those data management schemes is to efficiently allocate and migrate the multimedia data between different storage devices in order to provide smaller user request latency and larger system throughput. We categorize those data management schemes as device level data management. In this study, we discuss the server level data management. At the server level, we only see the online multimedia data, i.e., data ready to be delivered to the clients when requests arrive. How to make the data online is the problem of the device level data management. We discuss three video data allocation algorithms and compare their performance using simulation studies.

*partial support by Army Research Contract No. DAAH04-95-1-0250.

The rest of the paper is organized as follows. In section 2, We propose the distributed multimedia data allocation algorithms. In section 3, we use simulation to model the distributed VOD servers and compare the proposed algorithms based on the system performance in terms of the average user request latency time. We have our concluding remarks in section 4.

2. DATA ALLOCATION ALGORITHMS FOR DISTRIBUTED VIDEO SERVERS

We discuss the video allocation algorithms based on two most important performance factors, server bandwidth and storage capacity, in the following paragraphs.

2.1 Round Robin Allocation Algorithm

Conventionally we assume each server has similar process power, i.e., they can handle equal number of concurrent streams at the same streaming rate. So the task of the video data allocation is to evenly distribute the hot video objects to the distributed servers so that no one server will have to handle most of the user requests. Let us assume the video object set is $V = \{v_1, v_2, \cdots v_m\}$ with the corresponding user access frequencies to the video objects is $F = \{f_1, f_2, \cdots f_m\}$ where $\sum_{i=1}^{m} f_i = 1$. Without losing generality, we also assume $f_1 \geq f_2 \geq \cdots \geq f_m$. We assume the server set is $\{S_1, S_2, \cdots S_n\}$ with the storage capacities $\{C_1, C_2, \cdots, C_n\}$, respectively.

Table 1: Round Robin video data allocation

S_1	S_2	.....	S_{n-1}	S_n
v_1	v_2	.....	v_{n-1}	v_n
v_{2n}	v_{2n-1}	.....	v_{n+2}	v_{n+1}
...	...	.....	...	...

The idea of the *Round Robin* (RR) algorithm is presented by Table 1. The actual algorithm varies slightly from the above basic idea because we need to consider the storage capacity of the servers.

2.2 Bandwidth Weighted Partition Algorithm

Instead of letting a single server handle the user requests for a specified object, we allow video servers to coordinate the delivery of the data for a single video object to the client's workstations. We partition each video object into the video servers in the way that the servers with higher server bandwidth will be allocated with a larger portion of the video data. Table 2 presents the BWP algorithm. We assume the server set is $\{S_1, S_2, \cdots S_n\}$ with the respective server bandwidths $\{B_1, B_2, \cdots, B_n\}$. For the object v_i, we

partition its data into n continuous chunks $v_{i_1}, v_{i_2}, \cdots, v_{i_n}$ where $size(v_{i_j}) = size(v_i) \cdot \frac{B_j}{\sum_{k=1}^{n} B_k}$.

Table 2: Bandwidth Weighted Partition Algorithm

S_1	S_2	.....	S_{n-1}	S_n
v_{11}	v_{12}	.....	$v_{1,n-1}$	v_{1n}
v_{21}	v_{22}	.....	$v_{2,n-1}$	v_{2n}
...	...	.....	...	...

This algorithm partition the video data based on the server bandwidth. It balances the server bandwidth utilization very well and hence derives excellent system performance.

2.3 Popularity Based Data Allocation Algorithm

The idea of the PB algorithm is to let the server with higher bandwidth handle larger portion of the user requests. Besides the same assumption as in the RR algorithm, We assume the respective server bandwidths are $\{B_1, B_2, \cdots, B_n\}$ for the servers $\{S_1, S_2, \cdots S_n\}$. Without losing generality, we further assume the access to each object requires equal server bandwidth. To allocate the video objects, we first calculate what percentage of the total user requests a certain video server may handle based on its server bandwidth. We transform the server bandwidths $\{B_1, B_2, \cdots, B_n\}$ into their normalized server bandwidths $\{\bar{B}_1, \bar{B}_2, \cdots, \bar{B}_n\}$ where $\bar{B}_i = \frac{B_i}{\sum_{j=1}^{n} B_j}$. Obviously we have $\sum_{i=1}^{n} \bar{B}_i = 1$. Since we have $n \ll m$, it is reasonable to assume $f_i < \bar{B}_j$ for any $i \in [1, m]$ and $j \in [1, n]$. It is also reasonable to assume the total storage capacity of the servers is not less than the entire video database size, i.e., $\sum_{i=1}^{n} C_i \geq \sum_{j=1}^{m} Size(v_j)$. Thus the ideal video data allocation algorithm divides the video set V into n subsets $\{V_1, V_2, \cdots, V_n\}$ and allocates subset V_i to server S_i where $i \in [1, n]$, so that we have $C_i \geq \sum_{j=1}^{k} Size(v_{i_j})$ and $\bar{B}_i = \sum_{j=1}^{k} f_{i_j}$ for video subset $V_i = \{v_{i_1}, v_{i_2}, \ldots, v_{i_k}\}$. However, implementing such an ideal algorithm requires $C_m^n = \frac{n \cdot (n-1) \cdots (n-m+1)}{m \cdot (m-1) \cdots 2 \cdot 1}$ computation complexity. Furthermore, there is no guarantee the optimal solution exists. So we design the approximate video allocation algorithm as depicted in Figure 1.

```
Algorithm: Popularity_Based_Allocation
 for(i=1; i≤n; i++) {  RB(i) = B̄_i; RC(i) = C_i; }
 ServerNum = n;
 for(i=1; i≤m; i++) {
   Find the server j with largest RB.
   if(RC(j) ≥ Size(i)) {
     RC(j) -= s_i;   RB(j) -= f_i;
     Allocate Object i to Server j.
     Consider all servers for the next object.
     ServerNum = n;
   }
   else { /* This server is full */
     Delete server j from consideration.
     ServerNum = ServerNum - 1; i = i - 1;
     if(ServerNum = 0)
       Fail due to lack of storage capacity.
   }
 }
```

Figure 1: Popularity Based Data Allocation Algorithm

3. PERFORMANCE STUDY

We have developed a simulator to compare the performance of the proposed data allocation algorithms. Our simulator consists of *Request Generator*, *Dispatcher*, and *Servers*. The request generator is responsible for creating requests. The dispatcher routes the requests to the server where the requested object is located. There are four servers in our simulator. User requests are modeled using a Poisson process. The access frequencies of objects in the database follow a Zipf-like distribution. The access frequency for each video object, v_i, is determined as $f_i = \frac{1}{i^z \cdot \sum_{j=1}^{m} 1/j^z}$, where m is the number of objects in the system, and $0 \leq z \leq 1$ is the z-factor [1]. We summarize the simulation parameters in Table 3. We use those parameters to perform our simulation unless otherwise stated. Because users are most concerned about the latencies of their requests, we use average request latency as our performance metric.

Table 3: Simulation parameters

Server Storage Capacity	700,000 MB ($\approx$ 685 GB)
Playback rate B_p	6 Mbits/sec
Zipf factor	0.7
Requests inter-arrival time	5 seconds
Number of objects	1000
Object size	2,800 MB
Number of requests	20,000

In addition to the parameters listed in Table 3, we also need to determine a reasonable server bandwidth for us to compare the difference of the algorithms. Through simulation study, We choose the average server bandwidth to be 580 MB/sec in the rest of performance study.

3.1 Performance under various data skew condition

In this simulation, We varied the skew factor between 0.0 and 1.0. The results are plotted in Figure 2. It shows the BWP algorithm does not affect by data skew condition and its performance is the best in this situation. When the data skew condition was mild (less than 0.5), the RR algorithm performed well with the average latency less than 40 seconds. However, when the data skew condition became severe, the average latency for this algorithm increased drastically. The PB algorithm performed very well in any access skew condition. All of its latencies were less than 40 seconds which is practical for a 1-hour video.

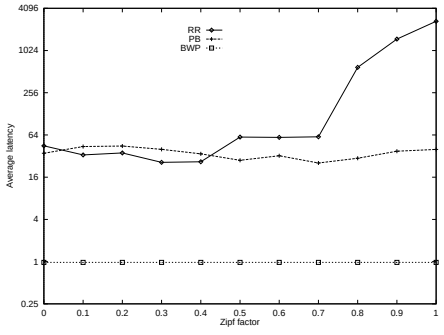

Figure 2: Performance under different access skew

3.2 Servers have different bandwidths

In the real distributed Video-On-Demand systems, the server bandwidths are different in different video servers. We studied system performance under this condition. We let the servers have an average bandwidth of 580 MB/sec. Their individual server bandwidth were different as $B_1 =$ 505 MB/sec, $B_2 =$ 555 MB/sec, $B_3 =$ 605 MB/sec and B_4

= 655 MB/sec. We also assume each server has enough storage capacity to complete all three algorithms. We varied the skew factor between 0.0 and 1.0 as in the previous simulation. The results are plotted in Figure 3. It shows the RR algorithm is practically useless in this situation. On the other hand, BWP algorithm and PB algorithm performed very well.

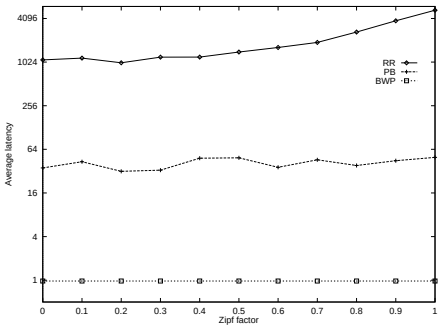

Figure 3: Servers have different bandwidths

3.3 The impact of the server storage capacity

In the previous simulations, we assumed all the servers have enough storage capacity to distribute the objects by the algorithms. However, in a real VOD system, it is not always the case. In this situation, BWP algorithm and PB algorithm will have to comply with storage limitation on the servers. We studied the similar situation using our simulator. We let all servers have same storage capacity of 685 GB. The server bandwidth in the servers were $B_1 = 505$ MB/sec, $B_2 = 555$ MB/sec, $B_3 = 605$ MB/sec and $B_4 = 655$ MB/sec. Because the servers have equal storage capacity, the data partition in the BWP algorithm should not be based on the server bandwidth. Each server has to store equal portion of an object. We also varied the zipf factor in the simulation, the results are presented by Figure 4.

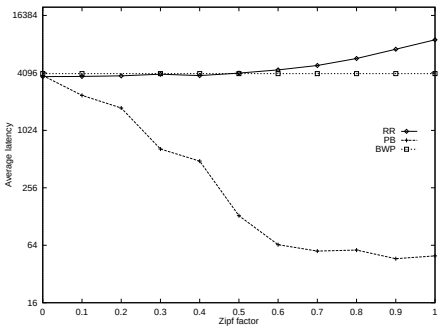

Figure 4: Impact of the server storage capacity

The simulation shows that the RR algorithm and the BWP algorithm are practically useless in this situation. When the zipf factor was zero, i.e., all objects had equal opportunity being accessed, the average latency for the PB algorithm was almost the same as that for the other two algorithms. This result is not surprising at all because there is no way for the PB algorithm to distribute the user requests to the servers based on their server bandwidth because of the server storage constraints. However, when the data skew factor is increased, PB algorithm takes advantage of the data access skew condition and put the hotter objects in the higher bandwidth servers. Hence, the average latency dropped. When the zipf factor was greater than 0.5, the average latency time dropped below 1 minute. It is reasonable for 1-hour video objects. Based on the results of simulation we expect the PB algorithm will perform very well in the real distributed VOD systems.

4. CONCLUDING REMARKS

In this paper, we thoroughly discussed the data allocation problems in the distributed Video-On-Demand environment. We proposed two different algorithms, *Bandwidth Weighted Partition* (BWP) algorithm and *Popularity Based* (PB) algorithm, to address the problems that affect the system performance. Through our analysis and simulation, we found the BWP algorithm balanced the server bandwidth utilization very well if the server storage capacity is not limited. Its performance is not affected by the user request patterns. In the ideal situation, the BWP algorithm provides the optimal system performance. This algorithm requires larger storage capacities for the higher bandwidth servers. The ideal situation is the server storage capacity proportional to the server bandwidth. But those conditions are not guaranteed in the real VOD systems. On the other hand, partitioning the data across the servers causes problems in video presentation. We need extra processing effort and extra bandwidth to ensure there is no jitter or delay when switching from one data server to another data server. The PB algorithm does not partition the video objects. It allocates the video objects based on their access frequencies. This algorithm not only factors in all resource constraints in video servers but also takes advantage of the access skew to the video objects. The simulation results show it provides near optimal results in any condition. Thus in the VOD systems where the data access skew condition is usually severe, the PB algorithm is an excellent video data allocation algorithm.

5. REFERENCES

[1] James Z. Wang and Kien A. Hua. A bandwidth management technique for hierarchical storage in large-scale multimedia servers. In *Proc. of IEEE Int'l Conf. on Multimedia Computing and Systems*, pages 261–268, Ottawa, Ontario, Canada, June 1997.

[2] Y. Wang and D. Du. Weighted striping in multimedia servers. In *Proc. of IEEE Int'l Conf. on Multimedia Computing and Systems*, pages 102–109, Ottawa, Ontario, Canada, June 1997.

[3] T.-S. Chau, J. Li, B. Ooi, and K-L. Tan. Disk striping strategies for large video-on-demand servers. In *Proc. of ACM Multimedia*, pages 297–306, Boston, MA, November 1996.

[4] S. Berson, S. Ghandeharizadeh, R. Muntz, and X. Ju. Staggered striping in multimedia information systems. In *Proc. of ACM SIGMOD Conf.*, pages 79–90, May 1994.

[5] D. W. Brubeck and L. A. Bowe. Hierarchical storage management in a distributed VOD system. *IEEE Multimedia*, pages 37–47, Fall 1996.

[6] Jamel Gafsi and Ernst W. Biersack. Performance and reliability study for distributed video servers: Mirroring or parity? In *Proc. of IEEE Int'l Conf. on Multimedia Computing and Systems*, volume 2, pages 628–634, Florence, Italy, June 1999.