A Balanced Batching Scheme of User Requests in Near VOD Servers

Hong-Ki Jung*, Sung-Wook Choi**, Seung-Kyu Park*
Affiliation : Multimedia and Computer Architecture Lab, The professional graduated school of Information and Communication Technology in Ajou univ.*, Dept. of Computing, Incheon Junior College**
Address: Ajou univ, San 5, Wonchon-dong, Paldal-gu, Suwon, Republic of Korea.
Tel : +82-331-219-2532, Fax: +82-331-219-1614
E-mail : keeper, sparky@madang.ajou.ac.kr* , swchoi@falcon.icc.ac.kr**

Abstract : In a batch scheduling policy being different from real video system, the requests are not served immediately due to grouping user's requests upon every scheduling points. Such waiting delays by inefficient managements makes an unfair service to users and increases the possibility of higher reneging rates. This paper proposes an adaptive batch scheduling scheme which reduces the average waiting time of user's requests and reduces the starvation problem for requesters of less popular movies. The proposed scheme selects dynamically multiple videos in given intervals based on the service patterns which reflect the popularity distribution and resource utilizations. Experimental simulation shows that proposed scheme improves about 20-30 percent of average waiting time and reduces significantly the starving requesters comparing with those of conventional methods such as FCFS and MQL.

1. Introduction

One of typical applications of stored video services is video-on-demand(VOD). The recent studies in the issues regarding to the maximum throughput of user requests can be found in three areas: user requests scheduling, buffer managements, and disk store managements. Depending on the user service policy, the systems fall in the category of either real or near VOD systems. In near VOD systems, same video streams can be distributed to many users using multicast mechanism of ATM [1,3,4]. For grouping users to multicast, optimal batching intervals need to be determined so as to minimize the average waiting time of user requests and to minimize reneging probability.

Dan and Sitaram proposed a batching scheme called dynamic batching policy[3]. In the scheme, multiple reservation queues are assigned for videos and the number of requests in the queues are investigated for given intervals. The most frequently requested video is scheduled dynamically. The scheme, however, conservatively reserves the capacity of the server for the popular videos, which wastes the resources. Golubchik proposed different scheme called adaptive piggybacking based on the observation that 5 % of speed difference in playing video cannot be recognized by human being [2,5,7]. This scheme accepts users to have the real video service by piggybacking video streams afterwards so as to get merged as one video streams. Artificial skipping and duplications of frames degrade the QoS and waste of the resources utilizations[6].

An adaptive batch scheduling scheme is required for reducing the average waiting time of users' requests which reduces the starvation problem for the requesters of less popular movies. Based on the service patterns which reflect the popularity distribution and resource utilizations, this scheme can reduce the waiting delays which resulted in higher reneging rates.

We propose in this paper a balanced scheme of batch scheduling in near VOD systems which reduces the average waiting time and supports a fair scheduling for user requests.

2. Batch Scheduling

In a batch scheduled VOD system, a server provides the high capacity storages which supply the video streams to clients' requests. The requests from clients are listed in the request queue in the server during batch intervals and the necessary resources are assigned to requests on scheduling time. The receive queue in the client is a buffer for the streams received from the server and used for supplying the streams during playing back.

The performance of the system is mainly determined by the grouping interval of user requests and batch scheduling policies. Constructing VOD system based on batching scheme thus needs two major elements, the interval and scheduling policy.

Determining an optimal interval time comes from reflecting factors both from user requests, expressed by "grouping", and from a server which provides streams based on available resources. Determining batching interval can be made either by taking the queue length into consideration or by reflecting the distribution of user requests. The former counts the size of the queue. This leads to a long average waiting time when the arrival rates have a high deviation. The latter may use constant interval time regardless of the arrival rates or use the popularity distribution.

A scheduling policy means that there is a certain way to select a requester which will be served among the requesters in the queue. The typical scheduling policies are First Come First Service (FCFS) and Multiple Queue List(MQL) [3,4]. The FCFS, as name implies, is a policy in which the serve is taken place for the requester who arrived first. This makes an even distribution of services regardless of popularity of the movies but leads to longer average waiting time. MQL on the other hand takes the frequency of requests into consideration. The most frequently requested movie in the queue is served for every given interval.

This work is supported in part by the Ministry of Information & Communication of Korea (Support Project of University foundation research <'99>) supervised by IITA

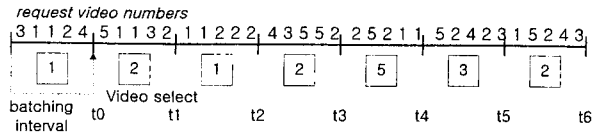

Fig. 1 Example of MQL scheduling

The figure 1 shows the requested video numbers and the selected movies by MQL policy on every scheduled interval. In the figure 1, the video 1 is selected at t0 since it has two requests (largest queue) in the first interval while the video 2 is selected at t1 since video 2 has same largest queue as that of video 1 but arrived earlier than that of video 1 in the second interval.

3. Adaptive Batch Policy

We propose an Adaptive Batch Scheme (ABS) in which the interval time is determined based not only on the users request rates but on the video popularity. The aim of the ABS is to provide a fair scheduling and small average waiting time. The adaptive scheduling and dynamic multiple selections are applied to the scheme.
Figure 2 shows the model of ABS. In the proposed scheme, each video has a queue of the users requesting a same movie. Only the videos with the queue lengths longer than k and the videos whose customer waited longer than the maximum waiting time $d_{\max}$, which is given as a parameter, are the candidates of scheduling.

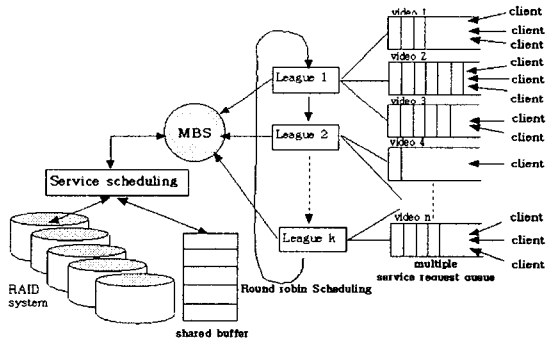

Figure 2. Model of ABS scheduling

3.1 Dynamic multiple selection

Simply applying a Zipf distribution [3] as a base for determining batch intervals introduces several problems. The intervals get longer when this scheme is used for unpopular video and it is not efficient when the popularity pattern changes often. To get over these problems, multiple videos are selected in the ABS scheme, whose queues are of the lengths more than k. Determining the length k contributes significantly to the performance. Two methods are suggested in the following section for deciding k, in which one is a heuristic method derived by observing the Zipf distribution and the other is derived using the utilization of the server resources. To avoid the starvation of the users whose requested videos are much less popular, the videos whose customers have waited a given maximum time are also selected for the candidates in the scheduling list in the ABS scheme.

Batch sequence	1					2				
Req. Video num.	1	2	*	4	*	1	4	1	*	*
FCFS					1					2
MQL					1					4
ABS										1

3					4					5				
4	1	3	1	3	*	4	1	4	2	*	3	4	2	1
				4					1					3
				1					4					3
				4					1 2					3 4

Fig 3. Example of multiple selection of ABS

The figure 3 shows an example of multiple selections by ABS scheme where $k_{\min} = 3$, where $k_{\min}$ means the length of a queue of the video for candidate scheduling, $d_{\max} = b_{inv} * 3$, in which b_{inv} is the batching interval. One or two videos are shown to be selected by the ABS in the example while FCFS and MQL has only one selected by their policy on every intervals.

3.2 Adaptive Scheduling

If a too big number is given to k, the service waiting time gets bigger. If we want a smaller k, the efficient management of the server resources are difficult. The number k must also be adaptively determined as time goes since the requests rate and popularity patterns may change over time.

The number k can be obtained either by Zipf distribution or by resource utilization. The Zipf distribution says that more that 50 percent of users are requesting only 10 ~ 15 percent of most popular videos. The figure 4 shows an algorithm how to get the number k based on the popularity patterns.

Algorithm for popularity

```
repeat
while (time-interval = null)  do skip;
k:= previous_minimum_service_frequency;
n:=(current_service_video_count
service_count_for_maximum_waiting_time)/
(all_request_video_count);
if n > given_popularity_ratio
then k:= k + 1
else
if n < given_popularity_ratio
then k := k - 1;
until false;
```

Fig. 4. Determining k with video popularity.

The ratio in the algorithm is obtain by calculating the percentage: the number of videos with queue length more than k minus the number of videos with waiting time more than $d_{\max}$ over the number of total requests during the intervals of observation. This method works good for large servers but it suffers the performance for small systems which have relatively small resources.

The figure 5 shows an algorithm how to get the number k based on the resources available. For a given situation the resources, the buffers and disk bandwidth, are over utilized, the number k is increased. The k is de-

```
Algorithm for Resource
repeat
while (time-interval = null)  do skip;
    k:= previous_minimum_service_frequency;
    if current_available_resource_count <
            lower_limit_resource_count
        then k:= k + 1
        else if currant_available_resource_count >
                upper_limit_resource_count
        then k := k - 1;
until false;
```

Fig. 5. Determining k with resource utilization

creased on the opposite situation.

4. Simulation Results

The conventional scheduling methods FCFS and MQL are evaluated comparing with the proposed scheme ABS in the simulation. In the simulation, the average waiting service time and maximum waiting time are obtained for a given server capacity. The Table 1 lists the parameters used in the simulations. The demand pattern of selecting videos is assumed to have a Zipf distribution and the demanding rate has a Poisson Distribution with λ=1, one per second. The maximum waiting time in ABS is set to 600 seconds and initial k is 6, which is changing over time in the adaptive method. The given population ratio is set to be 10 % considering 10 to 15 percent is the most popular movies in the Zipf distribution.

Table 1. Parameters for the simulation.

Parameters	Contents	Value
Request pattern for videos	Zipf distribution	
λ (poisson)	Arrival rate	1/sec
V_no	Number of video	100
V_play_time	Video length	3600 sec
Max_wait_time	Max. waitng time	600 sec
Time_interval	Batching interval	60 sec
Data_count	Data count	2000

The figure 6 shows the pattern of requested frequencies for 100 videos, in which the video index is shown horizontally.

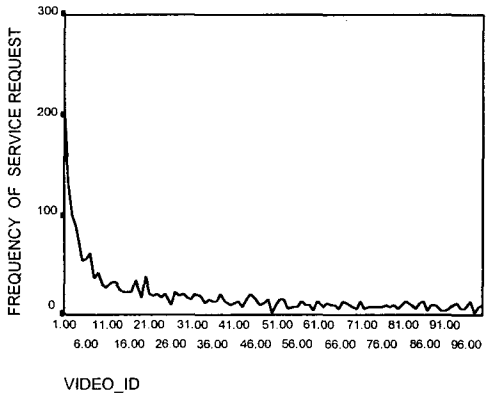

Fig. 6. Service requests with respect to video numbers.

The figure 7,8,9, and 10 shows the results of the analysis on the waiting time for service. The scheduling methods under the simulations are FCFS, MQL, ABS1, which means the ABS scheme with the interval obtained by considering the video popularity, and ABS2, where the resource utilization is used to get the interval.

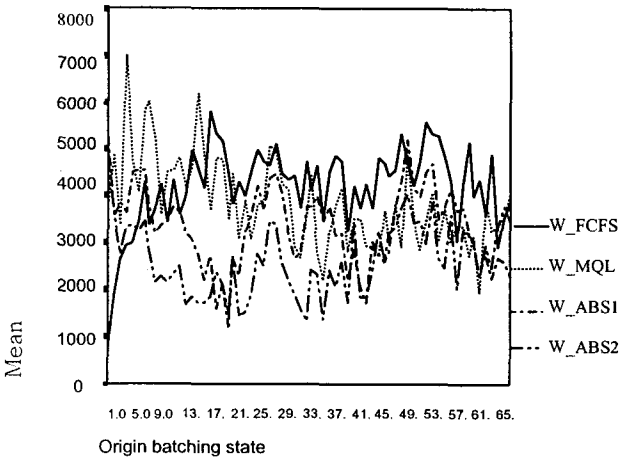

Fig. 7. Average waiting time over batch scheduling points.

The figure 7 and 8 shows the average waiting time over elapsed batch intervals for each scheduling methods.

The unit in the figures is all 10 msec (0.1sec). The ABS1 and ABS2 have smaller waiting time by 20 to 30 percent.

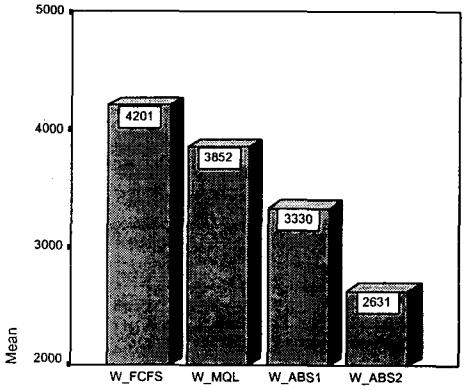

Fig. 8. Average waiting with respect to scheduling policies.

The figure 9 shows the number of requests which exceed the maximum waiting time, 600 seconds, where the 600 seconds is guaranteed as a maximum waiting time. The ABS1 and ABS2 have the half of exceeded ones than those of FCFS and MQL but they do not keep the promise of guaranteed waiting time. Increasing the maximum waiting time by 10 percent, which becomes 650 second, makes the ABS1 and ABS2 satisfy all customers with zero exceeding waiting time. The figure 10 shows the results in which the FCFS and MQL still have high waiting requesters

The Figure 11 and Figure 12 show the resource utilization over batch scheduling points and average utilization for scheduling policies, respectively.

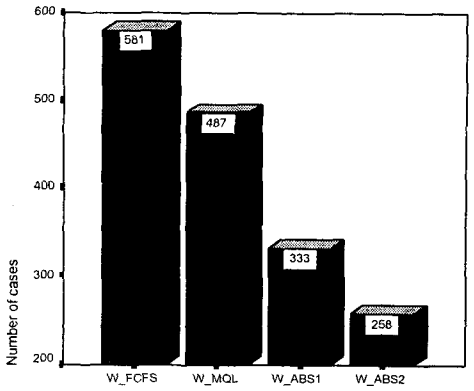

Fig. 9. Number of requests exceeding the maximum waiting time (600 seconds).

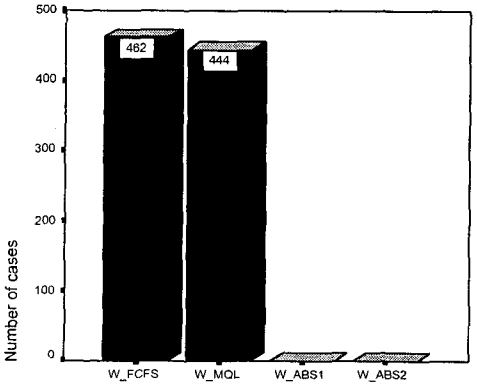

Fig. 10. Number of requests exceeding the 650 seconds.

The ABS2 which was based on the resource utilization shows a highest utilization where the initial level is set to 70 percent. The higher utilization of resource in ABS2 gives the lowest waiting time in the figure 8.

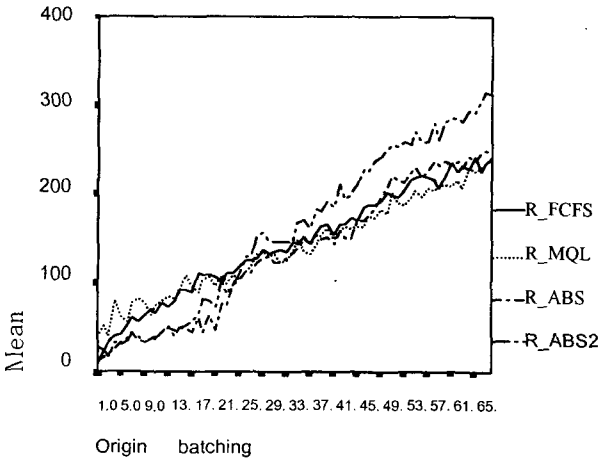

Fig. 11. Resource utilization over batch scheduling points

5. Conclusion

An adaptive scheduling scheme is proposed in this paper to give the lowest average waiting time of requesters while the scheme guarantees a maximum waiting time for requesters for low popular videos. The queue length k is obtained reflecting the popularity and resource status

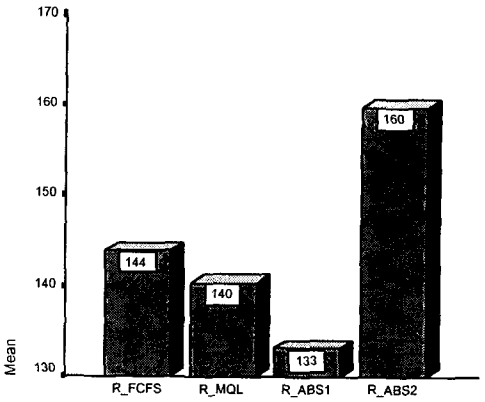

Fig. 12. Average utilization for scheduling policies

adaptively.

The simulation results show that the proposed schemes, ABS1 and ABS2, reduces the average waiting time by 20 - 30 percent than those of FCFS and MQL scheduling policies. The exceeded waiting time of ABS scheme can be eliminated all by increasing 10 percent of guaranteed waiting time. The utilization factors of resources for each scheduling policy did not make much difference. The proposed adapted scheme is expected to fit well specially in the heavily changing environment of services.

Reference

[1] Heek-Young Woo, Chong-Kwon Kim, "Multicast Scheduling for VOD Services". Multimedia Tools And Applications, pp 157-171, 1996

[2] Leana Golubchik, John C.S. Lui, "Adaptive piggybacking: a novel technique for data sharing in video-on-demand storage servers". Multimedia Systems, pp 140-15, 1996

[3] Asit Dan, Dinker Sitaram, "Dynamic batching policies for an on-demand video server". Multimedia Systems, pp 112-121, 1996

[4] Asit Dan, Dinkar Sitaram and Perwez Shahabuddin, "Scheduling Policies for an On-Demand Video Server with Batching". ACM Multimedia, pp 15-23, 1994

[5] L.Goubchik, J.C.S. Lui. and R. Muntz, "Reducing I/O demand in video-on-demand stoage servers", In proceeding of Intl. Conference on Measurement and Modeling of Computer System st. (SIGMETRICS '95), pp 25-36, 1995

[6] Kurt Rothermel, Tobias Helbig, "An adaptive protocol for synchronizing media streams", Multimedia Systems, pp 324-336, 1997

[7] Hong-ki Jung, Seung-Kyu Park, "Grouping and Buffer Management Methods in a VOD Server" ITC-CSCC'99, Sado, Niigata, Japan, pp899-902, July 13-15, 1999

[8] Wen-Jiin Tsai and Suh-Yin Lee, "Dynamic Buffer Management for Near Video-On-Demand Systems". Multimedia Tools and Applications, pp 61-83, 1998