DYNAMIC SCHEDULING OF MULTIPLE VIDEO OBJECTS FOR MPEG-4 ENCODING WITH USER INTERACTIONS

Yong He[†], Ishfaq Ahmad[‡], Ming L. Liou[†]

[†]Department of EEE, [‡]Department of Computer Science,
The Hong Kong University of Science and Technology,
Clear Water Bay, Kowloon, Hong Kong
Email: [†]*eehey@ee.ust.hk*, [‡]*iahmad@cs.ust.hk*, [†]*eeliou@ee.ust.hk*

Abstract

MPEG-4 video consists of various video objects, rather than frames, allowing a true interactivity and manipulation of separate arbitrary shape object. Software-based encoding of MPEG-4 video objects can be carried out by using parallel processing with efficient scheduling scheme to speedup the computation. In this paper, we propose two dynamic scheduling algorithms which have different scheduling costs and performance levels. The algorithms assign the multiple video objects encoding tasks to the cluster of workstations with proper load balancing. The algorithms allow user on-line interactions and perform the concurrent encoding on the video objects to achieve real-time speed. The experimental results, while showing real-time encoding rates, exhibit trade-offs between load balancing, overhead scheduling cost and global performance.

1 Introduction

MPEG-4 is a new standard providing the best solution for storage, presentation, transmission, and management of various types of media objects, both natural and synthetic, and is equipped with advanced functionalities, such as content-based interactivity, high compression, and random access [1].

For most MPEG-4 based multimedia applications, an MPEG-4 video encoder should be highly efficient and capable of operating multiple video objects in real-time. Because software-based approach usually employs general-purpose hardware and programming primitives which allow the flexibility, portability and scalability, it can be run on a vast variety of platforms and the performance can be optimized to different applications. In addition, software-based approach permits the users to configure the special program with various downloadable tools and audio-visual objects, therefore, software-based approach is a natural and viable option for MPEG-4 implementations. Furthermore, with the developments of the parallel and distributed computing technologies, a higher degree of performance at an affordable cost (such as a network of workstations or PCs) can be achieved through properly scheduling of multiple tasks.

Because MPEG-4 treats a scene to be coded as multiple individual video objects with different spatio-temporal characteristic, and each video object could be manipulated dynamically by the user interactions which result in time varying computational cost, implementation of an MPEG-4 encoder using parallel processing is a challenging task and cannot be accomplished using a straightforward multi-tasking or data partitioning strategy.

Generally, in an MPEG-4 system, with a single video encoder server and multiple clients configuration, clients may send sporadic requests to interact with certain video objects. The request which must be handled by the encoder server demands fast response time and a certain quality of service. This requires the encoder to determine the encoding schedule of each video object quickly while ensuring necessary synchronizations. The trade-off between the overhead time cost, responding time and overall encoding performance should be addressed carefully.

In this paper, we present two dynamic scheduling algorithms which have different scheduling costs and performance levels. The performance of the encoder can be scaled according to the number of workstations used and the interactive responding latency is low.

2 Dynamic Scheduling of Video Objects

The objective of scheduling in a parallel environment is to minimize the overall execution time of a concurrent program by properly allocating its tasks (in this case, video objects) to the processors [2]. Various scheduling algorithms have been reported in literature [3]. One of the drawbacks of most existing

0-7803-5471-0/99/$10.00©1999 IEEE

algorithms is that they assume the knowledge of tasks are known before processing and the system state remains stable during the entire processing. In an MPEG-4 application, the states of video objects may change at any time. Moreover, multiple objects scheduling problems are known to be NP-hard problems, and, therefore, heuristic methods are widely used as the feasible solutions. Earliest deadline first (EDF) is an optimal and dynamic algorithm which is widely employed in the parallel and distributed environment computing [4]. The principle of EDF is to assign the tasks with earlier deadlines higher priorities. We will present two dynamic scheduling algorithms based on EDF.

First, we introduce a low cost approach called *round robin scheduling* (RRS) algorithm which schedules the video objects to workstations in a sequential EDF order and with a round-robin fashion. It adapts to the size variation of the video objects which results in a minimum overhead scheduling cost. Hence, RRS algorithm is suitable for the applications with heavy client/server interactions due to its quick responding and scheduling time. The second algorithm, called *GOV (Group of VOPs)-based scheduling* (GOVS) algorithm, divides the processors into a number of groups such that each group performs the encoding on a single video object concurrently using an EDF order. In order to deal with video objects whose sizes change greatly with varying computation power requirements, rescheduling is performed periodically on the basis of GOV for adjusting the processing configuration and related parameters. With enough workstations and low frequent user requests, the GOVS algorithm can achieve a higher speedup.

2.1 The Round Robin Scheduling Algorithm

The round robin scheduling (RRS) algorithm uses the EDF rule by sorting all VOPs in a non-decreasing order of their playout deadlines. If two VOPs have the same playout deadline, the smallest processing time (SPT) is applied, that is, a VOP with a smaller size precedes the one with a larger size. A data partitioning method [5] partitions each VOP into a number of pieces equal to the number of processors. The RRS algorithm allocates the pieces to the processors in a round-robin fashion. Whenever a user request is received, the scheduler suspends the encoding procedure and updates the state of the video objects. Then, the processors perform the encoding on the video objects according to the new sorting order.

The RRS algorithm enforces the processors to encode each VOP concurrently. By storing all reference data in the local memory, each processor is able to process the partitioned data locally, and no data exchange is required. The scheduling time for sorting the VOPs can also be neglected because the calculation and comparison of playout deadline is very fast. Moreover, such RRS algorithm can adapt to the variations of video object size automatically. Since all the processors process the same task with different partition region concurrently, if the object size becomes larger, the partitioned data area will also be enlarged, and vice verse. Each processor may spend more time on larger VOPs and less time on smaller VOPs simultaneously.

Because the RRS algorithm can respond to the user request and perform the rescheduling very quickly, it is suitable for the applications supporting interactivity from multiple users in real-time. However, since each processor has to deal with all the available video objects, a data structure should be specified to identify the related reference data storage address for each video object and local memory should be large enough for the storage. In real-time applications, the input data rate for each processor is also high.

2.2 The GOV-based Scheduling Algorithm

According to the video encoder structure specified in MPEG-4, each video object is encoded independently. We can divide the processors into a number of groups to gain the extra speedup if large quantities of processors are available.

Therefore, we propose a periodical scheduling algorithm, called GOV-based scheduling algorithm (GOVS), where GOV stands for the Group of VOP (Video Object Plane). In order to minimize the inter-processor communication cost for collecting the load information and exchanging the reference data, the scheduling period is based on GOV which is an optional syntax level specified by the standard for random access and error recovery purpose. The GOV length (a collection of VOPs) can be defined by the encoder and its header is followed by the I-VOP performing Intra-coding which is independent of the previous VOPs. Therefore, the change in processor assignment by the scheduler due to the user interactions will not introduce any additional inter-processor communication.

GOVS algorithm divides the existing processors into a number of groups and each group handles single video object concurrently. Such allocation is performed periodically on the basis of the GOV. Within each group, a balanced data partitioning method is employed for further encoding speedup.

The criterion of processor allocation in GOVS algorithm is tied to the shape size, namely, the larger object size, the more processors assigned. While this may cause load imbalance between the groups since the distribution of the processors may not be proportional to the size of the video objects due to the excessive difference between the object size. One feasible solution is to merge the smallest object tasks together recursively.

Such task merging is usually performed whenever the number of tasks is greater than the number of physical processors by merging a pair of tasks into a single co-task [6]. There are various merging approaches for different purpose. In our approach, such task merging step is to guarantee the load balancing among the processors. We define the GOV of each video object as a task, and find a pair of tasks which have minimum workload among all the clusters; then the pair of tasks are merged as one task and such operation is recursively executed until the load balancing can be met.

2.3 Experimental Results

We have implemented the encoder using the above proposed scheduling schemes on the test sequences. Our software-based encoder uses the MPEG-4 video verification model (VM8.0) [7]. We conducted the experiments on a cluster of 20 Sparc Ultra 1 workstations connected by an ATM switch which provides fast communication among the workstations. Furthermore, we have used various additional software optimization, such as fast motion estimation algorithm, *Visual Instruction Set* (VIS) and compiler optimization, for performance improvement in the encoding speed [8].

We designed a test sequence with two foreground video objects retrieved from the standard test sequences called '*Akiyo*' and '*Weather*'. Figure 1 shows the encoding frame rate achieved by the encoder using the proposed scheduling algorithms with various number of workstations. Generally, the performance of GOVS is better than that of RRS due to its periodical adjustments. The encoder can achieve frame rate higher than the real-time performance (30 frames/second) on the QCIF sequence. For the two CIF object sequences, a frame rate close to 14 frames/second has been achieved by using 20 workstations.

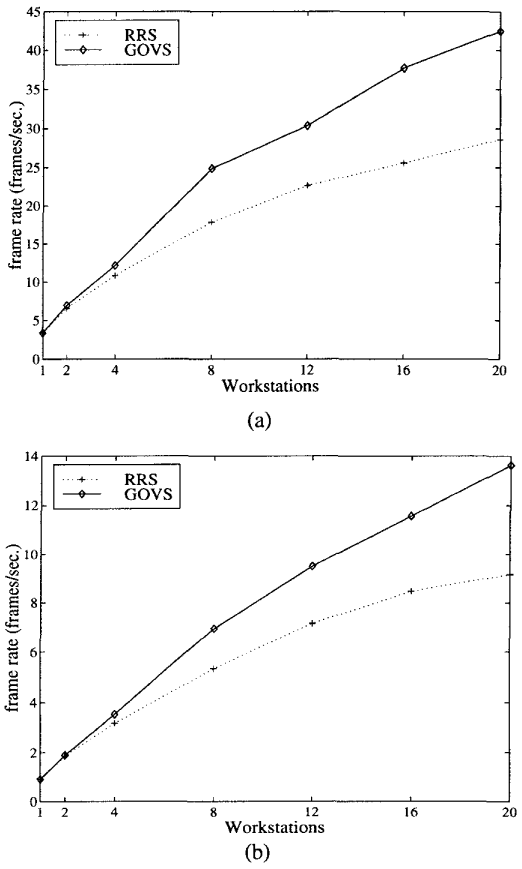

Figure 1: Encoding rate of video session with two video objects (VO_0 '*Weather*' and VO_1 '*Akiyo*'). (a) QCIF format (b) CIF format

We also tested the GOVS algorithm on several composed sequences. All of the video objects, such as '*Akiyo*', '*News1*' and '*Weather*', as labelled in Figure 2, are obtained from the MPEG-4 standard test library with QCIF format and represent various characteristics in terms of spatial detail and movement.

To evaluate the performance of the scheduling algorithm with respect to the user interactions, we simulated the user interactions on the sequence and varied the request inter-arrival rate. The request inter-arrival time represents the average interval between two users request. The experiments were performed to the sequence '*Weather* + *Akiyo*'. Figure 3 depicts the response and schedule time of proposed algorithms

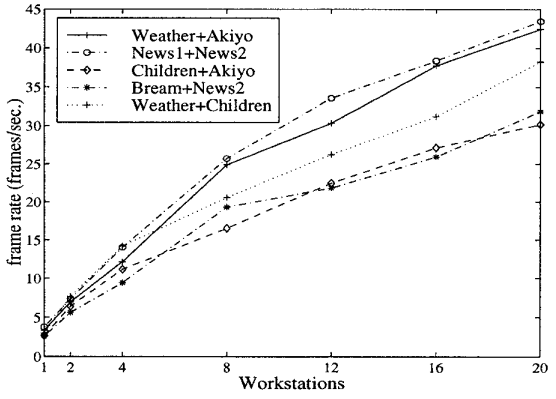

Figure 2: Encoder rate of composed sequence in QCIF format using GOVS scheduling scheme.

using 8 workstations. As indicated by these results, the RRS algorithm outperforms GOVS in terms of responding and scheduling time implying that it is more suitable for the environment with high frequent client/server interactions. Because the GOVS algorithm performs the rescheduling periodically, the user request has to wait until the encoding of current GOV is finished. With more frequent interactions, new requests suffer from the queuing delay and the response time becomes larger.

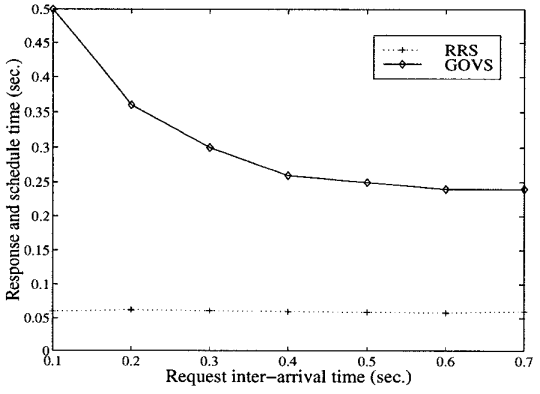

Figure 3: Response and schedule time with user interactions

3 Conclusions

In this paper, we present two scheduling schemes which assign the video encoding tasks to a cluster of workstations with proper load balancing. Each algorithm has its own trade-off between performance and complexity and therefore is suitable to different application environments. With 20 workstations, the encoder can achieve a high encoding rate on the composed sequences which demonstrate its potential to be used in a real-time system. In our future work, we will explore MPEG-4 decoders and interactive rendering methodology for supporting multimedia communication between server and clients of our multimedia system.

4 Acknowledgments

This work was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China and a grant from the Hongkong Telecom Institute of Information Technology.

References

[1] L. Chiariglione, "MPEG and Multimedia Communications," *IEEE Transactions on CSVT*, vol. 7, no. 1, pp. 5-18, Feb. 1997.

[2] Y. K. Kwok and I. Ahmad, "Dynamic Critical-Path Scheduling: An Effective Technique for Allocating Task Graphs to Multiprocessors," *IEEE Trans. on Parallel and Distributed Systems*, vol. 7, no. 5, pp.506-521, May 1996.

[3] G. C. Buttazzo, Hard Real-Time Computing Systems, *Kluwer Academic Publishers*, 1997

[4] S. Cheng et al., "Scheduling Algorithms for Hard-Real Time Systems - a Brief Survey," Hard Real-time Systems, IEEE Computer Society Press, 1988

[5] Y. He, I. Ahmad and M. L. Liou, "A Shape-adaptive Partitioning Method for MPEG-4 Video Encoding," *Proceedings of the 5th International Conference on Electronics, Circuits and Systems*, pp. 239-242, 1998

[6] J. C. Liou and M. A. Palis, "A Comparison of General Approaches to Multiprocessor Scheduling," *Proceedings of 11th International Parallel Processing Symposium*, pp. 152-156, 1997

[7] ISO/IEC, "MPEG-4 Video Verification Model 8.0," JTC1/SC29/WG11 N1796, July 1997

[8] Y. He, I. Ahmad and M. L. Liou, "A Software-based Video Encoder using Parallel Processing," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 8, no. 7, pp.909-920, November 1998