Parallel Processing using Data Localization for MPEG2 Encoding on OSCAR Chip Multiprocessor

Takeshi Kodaka†, Hirohumi Nakano†, Keiji Kimura†† and Hironori Kasahara†
†Dept. of Computer Science, Waseda University
††Advanced Research Institute for Science and Engineering, Waseda University
Okubo, Shinjuku-ku, Tokyo, Japan, 169-8555, TEL: +81-3-5286-3371
URL: http://www.kasahara.elec.waseda.ac.jp/

1 Introduction

Need for efficient processing of multimedia applications on PCs, mobile phones, games and so on have been increasing. Especially, low cost, low power consumption and high performance processors for multimedia applications have been expected. To satisfy the demands, chip multiprocessor architectures which allow to give us scalability using multigrain parallelism are attracting much attention. However, to get performance of chip multiprocessor architectures, data locality optimization for target applications is also required. This paper describes a parallel processing scheme for MPEG2 encoding using data localization technique which improves execution efficiency by using global data locality optimization among different loops with coarse grain task parallel processing, and evaluates the performance of the proposal scheme on OSCAR chip multiprocessor architecture.

2 Coarse-grain Task Parallel Processing and Data Localization

This section describes coarse grain task parallel processing [1] and data localization [2].

In coarse grain task parallel processing [1], a sequential program is decomposed into three kinds of coarse grain tasks, or MacroTasks (MTs), such as Block of Pseudo Assignment statements (BPA) like a basic block and a fused basic block, Repetition Block (RB) like a loop, and Subroutine Block (SB) composed of a subroutine. After generation of MacroTasks, the compiler analyzes control flow and data dependence among MacroTasks and generates a directed acyclic graph called MacroFlow-Graph (MFG). Next, in order to find maximum parallelism among MacroTasks considering control dependencies and data dependencies, the compiler analyzes an earliest-executable-condition for each MacroTask. The result of this analysis is represented by a directed acyclic graph called MacroTask-Graph (MTG). After generation of MTG, the compiler assigns MacroTasks onto processor-groups (PGs), each of which consists of several processors logically.

To use processor local memory and/or cache memory efficiently, the data localization scheme has been proposed [2]. In the data localization, Loop Aligned Decomposition (LAD) is applied to loops that access the same shared data. LAD divides each loop into partial loops having smaller number of iterations so that data size accessed by the divided loops is smaller than processor local memory and/or cache memory. After LAD, each partial loop is treated as coarse grain task and exploit parallelism, and MTs using same range of data are grouped into "Data Localization Group (DLG)". Next, each partial loop inside a DLG is assigned onto same processor consecutively as much as possible statically or dynamically.

3 OSCAR Chip Multiprocessor Architecture

This section describes the OSCAR Chip Multiprocessor architecture (OSCAR CMP) [3].

The OSCAR Chip Multiprocessor architecture (OSCAR CMP) is shown in Figure 1. In this architecture, each processor-element (PE) has simple CPU core, local program memory (LPM) which stores program code exclusively generated for each PE by the compiler, local data memory (LDM) which stores PE local data, distributed shared memory (DSM) having two ports which provides low-latency data transfer and low-overhead synchronization and data transfer unit (DTU) which is used for overlapping of data transfer and task processing. These PEs are connected by interconnection network like multiple buses or crossbar network. Furthermore, this architecture has centralized shared memory (CSM).

The parameters of OSCAR CMP in this paper are following: The capacity of LDM on each PE is 256Kbytes respectively. The DSM is 16 Kbytes per PE. The access latency of LDM is one clock cycle. Local DSM access latency is one clock cycle and that of remote DSM is four clock cycles. The access latency of CSM is 20 clock cycles. A processor core inside a PE is simple single issue core based on pipeline configuration similar to UltraSPARC-II. Three buses connect PEs. DTU is not used in this evaluation.

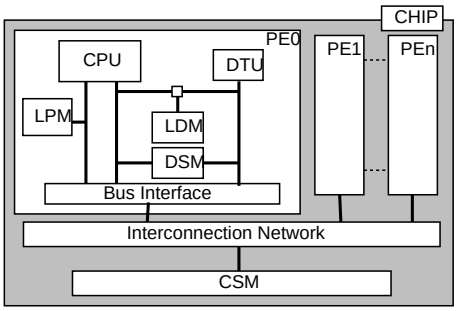

Figure 1: OSCAR CMP architecture

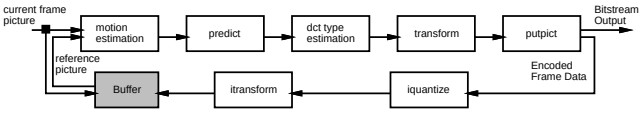

Figure 2: MPEG2 encode block diagram

4 Parallel Processing scheme for MPEG2 Encoding with the Data Localization

This section proposes parallel processing scheme for MPEG2 encoding with the data localization.

MPEG2 encoding algorithm used here is that of "mpeg2encode" from MediaBench [4]. MPEG2 encoding consists of the following seven stages as shown in Figure 2:

1. Calculation of a motion vector of the luminance frame in each macroblock (motion estimation)
2. Predicting motion compensation (predict)
3. Selection DCT mode from frame or field DCT (dct type estimation)
4. Discrete cosine transformation (DCT) using DCT mode (transform)
5. Quantization DCTed blocks and output of bit stream code (putpict)
6. Inverse quantization to quantized blocks (iquantize)
7. Inverse DCT to dequantized blocks for reconstruction (itransform)

In "mpeg2encode", each stage is performed on one MacroBlock at a time.

To exploit parallelism in MPEG2 encoding, parallelism among multiple macroblocks with no data dependence among macroblocks except putpict stage is used. Therefore, motion estimation, predict, dct type estimation, transform, iquantize and itransform stages can be executed in parallel using macroblock level parallelism. However, putpict stage has loop carried dependence caused by the bit rate control and bitstream output order. Therefore, currently putpict stage is executed sequentially.

Considering the memory capacity of OSCAR CMP, LDM is not enough for the encoding process on an entire frame for popular data size like QCIF and QVGA. To solve this problem, the data localization scheme is applied. At first, each stage is decomposed into partial loops considering data dependency among different loops. In MPEG2 encoding, since each stage is performed in macroblock level, each stage is decomposed into macroblock level tasks. After the decomposition, these macroblock level tasks are defined as coarse grain tasks or MTs, and parallelism among MTs is exploited. Figure 3 shows the MacroTaskGraph in which each stage is decomposed into four partial loops or MTs. In Figure 3, each node represents MTs, each edge represents data dependency among MTs. Figure 3 shows that there is no data dependence among macroblock level tasks which are originally defined as the same stage except putpict stage. As to the data dependence edges of putpict stage, the size of shared data between putpict and iquantize stage (ex. between MT5_1 and MT6_1) is much larger than inside between putpict stage (ex. between MT5_1 and MT5_2). Putpict and iquantize stage (ex. MT5_1 and MT6_1) should be assigned onto the same processor for data locality. Based on the above, macrotasks that treat same macroblock are assigned onto the same processor to exploit data locality as much as possible. Figure 4 shows the scheduling result by the proposed scheme in which each stage is divided into eight partial loops. Therefore, shared data can be passed through among different stages via LDM, so that improvement of efficient execution can be achieved.

5 Performance Evaluation

This section describes performance evaluation results of MPEG2 encoding by the proposed data localization scheme on OSCAR CMP, and loop parallel processing is also evaluated for comparison with the proposed scheme.

In this evaluation, clock level detailed simulator of OSCAR CMP architecture is used. The evaluation program in this paper is rewritten MPEG2 encoding program derived from "mpeg2encode" in MediaBench [4] by Fortran to use the OSCAR multigrain parallelizing compiler. To apply the proposed scheme , exploitation of coarse grain task parallelism and machine code generation are performed by the OSCAR multigrain parallelizing compiler. Input data is four frames of QCIF (176×144) data which is reduced size used in MediaBench because of our architecture simulator requires very long time to evaluate full size of data.

Figure 5 shows evaluation result of MPEG2 encoding on OSCAR CMP. In this figure, horizontal axis shows the number of processors or PEs, and each bar shows speedup against sequential execution time. The left bar for 2PEs, 4PEs and 8PEs shows the speedup

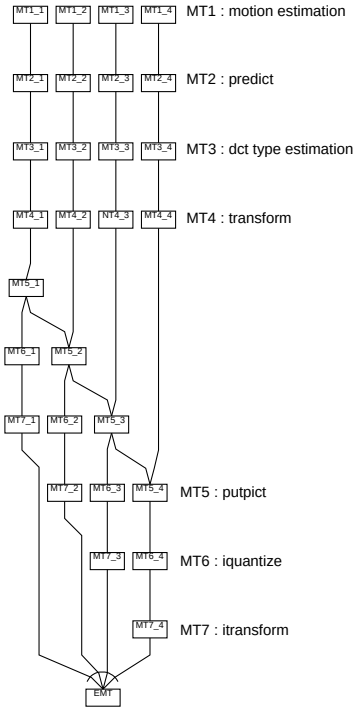

Figure 3: MacroTask Graph of MPEG2 encoding

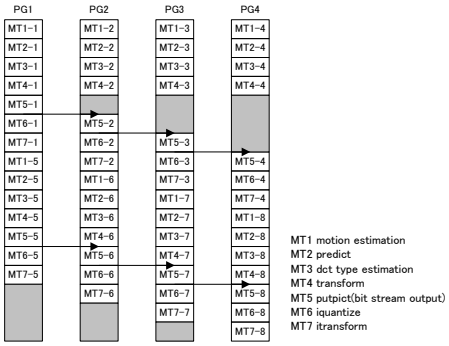

Figure 4: Scheduling result

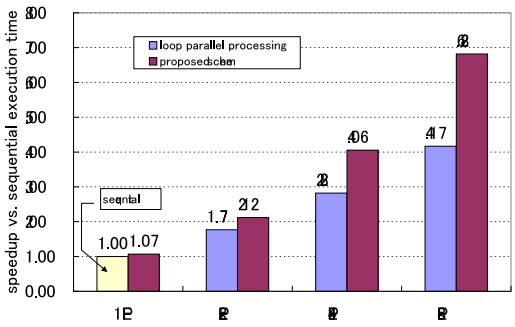

Figure 5: Evaluation result of MPEG2 encoding

ratio of MPEG2 encoding applied loop parallel processing (LOOP) and the right bar for 1PE, 2PEs, 4PEs and 8PEs shows that of proposed scheme (LOCAL).

When proposed scheme (LOCAL) is applied, 1PE gives us 1.07 times speedup, 2PEs gives us 2.12 times speedup, 4PEs gives us 4.06 times speedup and 8PEs gives us 6.82 times speedup against sequential execution time respectively. When loop parallel processing (LOOP) is applied, 2PEs gives us 1.77 times speedup, 4PEs gives us 2.82 times speedup and 8PEs gives us 4.17 times speedup against sequential execution time respectively. As to compare LOCAL with sequential execution, LOCAL gives us 7% speedup against sequential execution. This result shows that efficiency of memory access is improved by the data localization scheme. Improvement of speedup ratio of LOCAL is better than that of LOOP with the increase of the number of processors. The reason of these differences is that when LOCAL is applied, while one PE is executing putpict stage, the other PE(s) can execute motion estimation to transform stage of other macroblocks using the data on the LDM.

6 Conclusions

This paper has proposed parallel processing for MPEG2 encoding using the data localization. The performance evaluation on OSCAR CMP showed that proposed scheme gives us 4.06 times speedup for 4 processors and 6.82 times speedup for 8 processors against sequential execution time, and 1.43 times speedup for 4 processors and 1.63 times speedup for 8 processors against the loop parallel processing.

Acknowledgments

This research has been supported by "Automatic Parallelizing Compiler cooperative Chip Multiprocessor" in STARC and Advanced Research Institute for Science and Engineering, Waseda University, and JSPS Grants-in-Aid for Young Scientists (B) (#15700074) and for JSPS Fellows(#1501202). The authors thank to Mr. Miyamoto (STARC), Mr. Takahashi (Fujitsu), Mr. Takayama(Panasonic), Mr. Yasukawa (Toshiba) and Mr. Kurata (Sony).

References

[1] H. Kasahara, M. Obata, and K. Ishizaka. Automatic coarse grain task parallel processing on smp using openmp. In *Proc. 12th Workshop on Languages and Compilers for Parallel Computing*, Aug. 2000.

[2] A. Yoshida, K. Koshizuka, M. Okamoto, and H. Kasahara. A data-localization scheme among loops for each layer in hierarchical coarse grain parallel processing. *Trans. of IPSJ*, 40(5), May. 1999.

[3] K. Kimura, T. Kato, and H. Kasahara. Evaluation of processor core architecture for single chip multiprocessor with near fine grain parallel processing. *Trans. of IPSJ*, 42(4), Apr. 2001.

[4] C. Lee, M. Potkonjak, and W. H. Mangione-Smith. Mediabench: A tool for evaluating and synthesizing multimedia and communications systems. In *30th International Symposium on Microarchitecture (MICRO-30)*, Nov. 1997.