Static Load-Balancing for Distributed Processing of Numerical Simulations

Graduate Advisor: Shuichi Ichikawa 963734 Shinji Yamashita

Keywords: Static load-balancing, recursive bisection, combinatorial optimization, branch-and-bound

1 Introduction

In the previous paper[1], the static load-balancing of parallel PDE (Partial Differential Equations) solver was modeled as an optimization problem, and solved by branch-and-bound method with off-the-shelf computers. This method takes both computation time and communication time into consideration.

The target computer system was assumed to be a parallel computer that consists of uniform processing elements in the previous paper[1]. On the other hand, this study deals with general parallel processing environment that consist of non-uniform processing elements. This situation is very popular in distributed processing environment.

2 Model

Let the number of blocks be m, and the number of processors be n. The relationship $m \ll n$ is assumed in this paper (as in the previous study [1]). Under this assumption, we can formulate the problem in two stages. First, we have to find the best allocation of n distinguishable processors to m distinguishable blocks to minimize the execution time. To determine the best processor allocation, we have to know the best partitioning of a block for a given set of processors. Here, we decided that each processor should deal with a single subblock, which is a rectangular fragment of block.

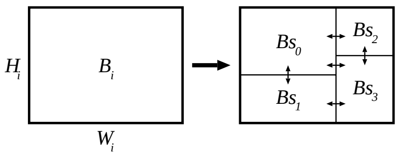

Figure 1: Partitioning of a Block

3 Partitioning

Partitioning of a block is a kind of combinatorial optimization with geometrical constraints. All heights and widths of subblocks must be integers, and the original block must be reconstructed from its rectangular subblocks like a jigsaw puzzle (Figure 1). Under these constraints, we have to find the best partitioning to minimize the execution time. This optimization problem is so difficult that five heuristic algorithms are presented and quantitively evaluated here against a theoretical lower bound. The heuristic partitioning algorithms are based on recursive bisection. See my master thesis for more details.

4 Processor Allocation

Next, we have to find the best allocation of n distinguishable processors to m distinguishable blocks so as to minimize the execution time. Basically, every combination of processors must be examined to solve this kind of combinatorial optimization problem, but this is a difficult computation problem. In this paper, a branch-and-bound method is adopted. A good approximation algorithm is also important for practical use of branch-and-bound method. The previous papers[2][3] showed the quantitive evaluation of three approximation algorithms (Approx1, Approx2, and Approx3) with local search (Local12). Though the error of

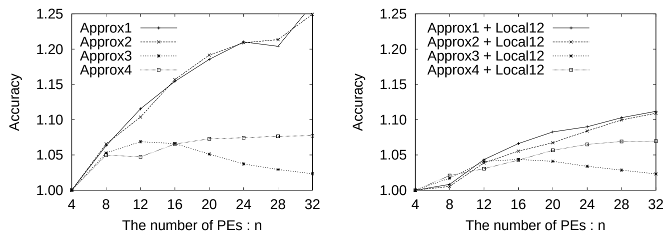

Figure 2: Accuracy of Processor Allocation Algorithms ($m = 4$)

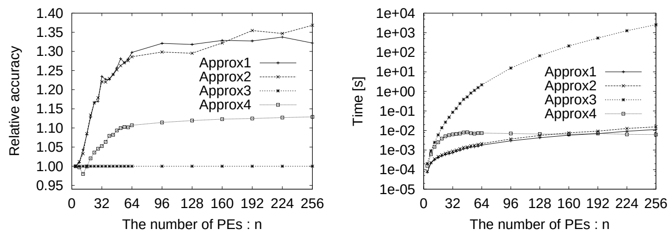

Figure 3: Comparisons of Processor Allocation Algorithms ($m = 4$)

Approx3[2][3] is less than 5% for ($m = 4, n = 32$), Approx3 is based on partial enumeration method and requires much time when $m \ll n$. Therefore, in this paper, we propose Approx4, which allocates processing elements with greedy method.

5 Results

The approximation algorithms are quantitively evaluated against the optimal grouping derived by branch-and-bound method. The results of numerical simulations are shown in Figure 2 and 3. In Figure 2, approximated solutions are compared to the optimal solutions. When $n > 32$, it is so hard to solve this combinatorial problem that approximated solutions are compared to Approx3's solutions (Figure 3).

Approx3 gives good approximation, but requires much time for solution. Approx4 gives the solution in almost constant time (about 10ms), and the accuracy is only 10% worse than Approx3. Consequently, Approx4 is regarded as the best approximation algorithm.

References

[1] Ichikawa, S., Kawai, T. and Shimada, T.: Static Load Balancing for Parallel Numerical Simulation by Combinatorial Optimization, *Trans. Information Processing Society of Japan*, Vol. 39, No. 6, pp. 1746 – 1756 (1998).

[2] Ichikawa, S. and Yamashita, S.: Static Load Balancing of Parallel PDE Solver for Distributed Computing Environment, Proc. ISCA 13th Int'l Conf. Parallel and Distributed Computing Systems (PDCS-2000), pp. 399 – 405 (2000).

[3] Ichikawa, S. and Yamashita, S.: Static Load-Balancing for Distributed Processing of Numerical Simulations, *Trans. Information Processing Society of Japan,* (submitted).