Enhanced Optimization Scheme for Parallel PDE Solver of NSL

Shuichi Ichikawa †, Takamitsu Kawai ‡, and Toshio Shimada ‡

ichikawa@tutkie.tut.ac.jp, {kawai,shimada}@nuee.nagoya-u.ac.jp

† Toyohashi University of Technology, Toyohashi 440-8580, Japan

‡ School of Engineering, Nagoya University, Nagoya 464-8603, Japan

Abstract

Authors have been developing a numerical simulation environment NSL [1], which automatically generates parallel PDE (partial differential equations) solver from high-level description of problem. Two of the notable features of NSL are boundary-fitted coordinate system and multi-block method. Physical domain is mapped onto a group of rectangular computational blocks, each of which is partitioned into one or more congruent sub-blocks. Each processor takes charge of a single sub-block. Static load balancing of such system can be modeled as a combinatorial optimization, which can be solved by branch-and-bound method [2][3]. However, in this model, the number of processors (n) is required to be greater than or equal to the number of blocks (m). This restriction can be a major obstacle to handle many blocks on a modest-scale parallel computer.

This paper presents an enhanced scheme that works regardless of the relationship between m and n, with additional performance improvement. Basic idea is that each processor handles a few sub-blocks instead of one. Assume that each processor is equivalent and in charge of the same number of sub-blocks. Let this number be k. The enhanced scheme is formulated as distributing kn virtual processors among m blocks and then allocating k sub-blocks for each physical processor. Applying the earlier algorithm [2][3] to kn virtual processors, the former part is easily solved. Let this result be $T_{m,kn}$. The latter allocation problem is beyond the scope of this short paper, so let us just take the worst case estimation here. The overall execution time T is estimated to be $kT_{m,kn}$.

Figure 1 shows the simulation results of T for various k, while both m and n are fixed to 16. No load balancing is achieved at $k = 1$, because n is equal to m. When block size (b) is small, T monotonically increases as k increases. In this case, calculation time is not enough to conceal communication latency and communication time increases in proportion to k. If b is big enough, T first decreases as k increases by the effect of load balancing when calculation time is superior to communication. At some point, communication is getting dominant and T turns to increase as k increases. Therefore, the execution time can be improved by choosing the optimal k. Figure 2 shows T for various m for n = 16 and b = 200. The new scheme applies to m that is bigger than n, while improving T with adequate k.

It is simple to implement this new scheme by iterating the earlier scheme for kn virtual processors until finding the optimal k. This new algorithm needs longer time than earlier because each search incurs $O((kn)^m)$ time in iteration. However, our simulation demonstrates that better solution can be derived in practical time.

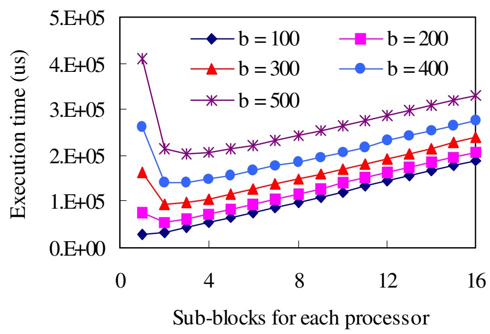

Figure 1. T vs. k

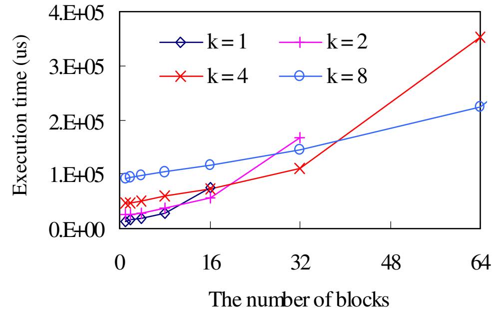

Figure 2. T vs. m

References

[1] T. Kawai, S. Ichikawa, and T. Shimada, "NSL: High level language for parallel numerical simulation," *Transactions of Information Processing Society of Japan*, Vol. 38, No. 5, pp. 1058-1067 (1997).

[2] S. Ichikawa, T. Kawai, and T. Shimada, "Mathematical programming approach for static load balancing of parallel PDE solver," in *Proceedings of the 16th IASTED International Conference on Applied Informatics*, Acta Press (1998).

[3] S. Ichikawa, T. Kawai, and T. Shimada, "Static load balancing for parallel numerical simulation by combinatorial optimization," *Transactions of Information Processing Society of Japan*, (to appear).

Acknowledgment: This research is partially supported by the Telecommunications Advancement Foundation.

In the proceedings of Joint Symposium on Parallel Processing 1998 (JSPP '98), Information Processing Society of Japan, Tokyo, 1998.