A Model for Task Repartioning under Data Replication

Erkan Okuyan[1], Cevdet Aykanat[2] and B. Barla Cambazoglu[3]

[1] Bilkent University, Computer Science Department
Ankara, Turkey
eokuyan@cs.bilkent.edu.tr
[2] Bilkent University, Computer Science Department
Ankara, Turkey
aykanat@cs.bilkent.edu.tr
[3] Yahoo! Research
Barcelona, Spain
barla@yahoo-inc.com

Abstract. We propose a two-phase model for solving the problem of task repartitioning under data replication with memory constraints. The hypergraph-partitioning-based model proposed for the first phase aims to minimize the total message volume that will be incurred due to the replication/migration of input data while maintaining balance on computational and receive-volume loads of processors. The network-flow-based model proposed for the second phase aims to minimize the maximum message volume handled by processors via utilizing the flexibility in assigning send-communication tasks to processors, which is introduced by data replication. The validity of our proposed model is verified on parallelization of a direct volume rendering algorithm.

Keywords. Task repartitioning, data replication, hypergraph partitioning with fixed vertices, assignment flow network

1 Introduction

Many parallel scientific computing applications require repeating the same computation over a problem instance with different parameters. In general, the quality of the initial task-to-processor or data-to-processor mapping in such applications tends to deteriorate in terms of computational load balance and communication requirements, as the computational structure of the problem or application parameters change between successive computational instances. A promising solution to this problem is to rebalance the load distribution in the parallel system as needed by rearranging the assignment of tasks to processors via a process known as repartitioning (remapping).

Novel repartitioning models are essential for efficient parallelization. The success of a repartitioning model depends on its ability to rebalance the load distribution as well as to minimize the overheads introduced due to the repartitioning process itself. Although it is problem-dependent, most typical repartitioning

Dagstuhl Seminar Proceedings 09061
Combinatorial Scientific Computing
http://drops.dagstuhl.de/opus/volltexte/2009/2090

overheads are due to task migration, data replication, and repartitioning computations.

Recently, succesful combinatorial models [1], [2], [3], which are based on graph and hypergraph partitioning by fixed vertices, are proposed as solutions to the repartitioning problems arising in different types of applications. In all three models, the computational structure of an underlying application is represented by a graph/hypergraph, where vertices represent tasks and edges/hyperedges represent the interaction between/among the tasks. In [1] and [3], vertices are fixed to the parts according to the initial task-to-processor mapping, whereas in [2], the hyperedges, which represent the data primitives, are fixed to the parts according to the initial data-to-processor mapping. The fixed vertex formulations adopted in these models encapsulate the cost of task and/or data migration.

2 A task repartitioning model

The focus of this work is on parallel scientific computing applications in which similar type of computations are succesively repeated over the same dataset instance for many times with different parameters. There is no dependency between tasks and the only reason for inter-task interaction is the existence of data primitives that are inputs to several tasks. Both computational structure and expected task execution times may change during succesive computational instances. Change in computational structure means change in the data primitive requirements of tasks. Since the individual processors of the parallel system have a limited storage capacity, we can reserve a limited amount of storage for holding replicas at each processor. For the parallelization of a particular computational stage, the repartitioning model should utilize the replication pattern of the previous computational stage(s) for reducing the communication overhead due to the data replication requirement of the current stage.

We propose a two-phase model for solving this problem. The hypergraph-partitioning-based model proposed for the first phase is an enhanced version of our previous model in [2] and it aims to minimize the total message volume that will be incurred due to the replication/migration of input data while maintaining balance on computational and receive-volume loads of processors. The network-flow-based model proposed for the second phase is an adaptation of [4] and it aims to minimize the maximum message volume handled by processors via utilizing the flexibility in assigning send-communication tasks to processors, which is introduced by data replication. The validity of our proposed model is verified on image-space parallelization of a direct volume rendering algorithm.

References

1. C. Aykanat, B.B. Cambazoglu, F. Findik, and T. Kurc, "Adaptive Decomposition and Remapping Algorithms for Object-Space-Parallel Direct Volume Rendering of Unstructured Grids", *J. Parallel and Distributed Computing*, vol. 67, no. 1, pp. 77–99, Jan. 2007.

2. B.B. Cambazoglu and C. Aykanat, "Hypergraph-Partitioning-Based Remapping Models for Image-Space-Parallel Direct Volume Rendering of Unstructured Grids", *IEEE Trans. Parallel and Distributed Systems*, vol. 18, no. 1, pp. 1–14, Jan. 2007.
3. U.V. Catalyurek, E.G. Boman, K.D. Devine, D. Bozdag, R.T. Heaphy, and L.A. Riesen, "Hypergraph-based Dynamic Load Balancing for Adaptive Scientific Computations", in Proc. IPDPS'07, pp. 1–11, Mar. 2007.
4. A. Pinar and B. Hendrickson, "Improving Load Balance with Flexibly Assignable Tasks", *IEEE Trans. Parallel and Distributed Systems*, vol. 16, no. 10, pp. 956–965, Oct. 2005.