Resource Management for Digital Signal Processing via Distributed Parallel Computing

Albert Reuther and Joel Goodman
(reuther, jgoodman)@ll.mit.edu
MIT Lincoln Laboratory, Lexington, MA 02420

May 31, 2002

Abstract

The resource management of real-time, pipeline streamed digital signal processing applications is not considered in the common Grid Computing paradigms. For this class of applications, scheduling for task- and data- parallelism is needed in order to meet extremely high throughput and communication requirements. This research focuses on the development of a resource manager for dynamically launching such applications on distributed parallel, embedded computers.

Introduction

Fast computational and internetworking technologies are the enabling technologies for Grid Computing[1], in which applications are executed on clusters of computational resources without regard to geographic locality. The Globus Toolkit [2] along with such resource managers as Legion [3] and Condor [4] provide many of the fundamental Network Resource Management (NRM) tools needed to realize this form of location independent computing, but these tools do not adequately addresses the requirements of real-time, pipeline streamed digital signal processing applications because such applications have strict throughput and latency requirements while using a minimal number of computational resources. Therefore, a different approach must be taken to satisfy these requirements. Further, in military tactical situations, real-time quality-of-service is essential so that mission-critical results can be delivered in a fixed time interval. To expedite such delivery, applications are launched on marshaled resources so data delay or loss will be mitigated.

The Research

The scheduling of real-time, pipeline streamed digital signal processing applications is done in two stages: system model graph generation and graph search for determining a feasible scheduling.

To facilitate the generation of a representative processing graph to be used for scheduling,, the applications must be partitioned into a series of pipelined tasks. (For example, in a simple radar processing application, it may be partitioned into three tasks: beamforming, pulse compression, and detection.) Each task is mapped onto a set of one or more processing resources; these mappings are modeled as vertices of the graph, while the communication links between these mappings are modeled as graph edges as was done for the S3P algorithm [5]. Once these graphs are generated, the vertices and edges for the mappings must be populated with weights such task latency, resource utilization, and throughput.

The focus of this research is the development of the graph search to determine a feasible path through the graph; that is, choosing a feasible mappings for each of the pipelined, real-time tasks in the application given processors per mapping, latency, utilization, and throughput weight measures. The feasible solution should mitigate resource contention while being computationally efficient because of the potentially vast number of permutations of resource configurations.

This research addresses these challenges with a novel approach to an efficient search for a solution to the resource-scheduling problem of optimizing a discrete objective function while meeting a set of inequality constraints. The objective function and the constraints are operational parameters that are chosen to expedite the delivery of time-sensitive information while consuming as few computational and communication resources as possible.

We have found that for high throughput, streaming applications, our algorithm which we call Decision Directed Learning (DDL) outperforms other combinatorial optimization techniques such as search-optimized genetic algorithms (GA), as is used in the IOS resource manager [6], in the quality of the solution and time to solution.

Figures 1 and 2 show the GA and DDL graph-search scores and times where the objective is to find a minimum score given a set of constraints for a set of fully-connected graphs that are 1000-vertices (mappings) by ten tasks in size. The simulation results were obtained using 1000-weighted graphs automatically generated using weights randomly selected from a uniform distribution with a fixed set of constraints, upon which both DDL and GA operate. In Figure 1, DDL has a mean score centered near 40 with the predominate number of search-scores ranging from 30-60, while GA scores are centered near 60 with the predominate number of search-scores ranging from 40-80, or roughly a 35% decline in performance with respect to DDL. Similarly comparing the time to a best solution from Figure 2, DDL outperformed GA on average by roughly 50%. Note that GA was unable to find a solution (path through the graph) that met constraints in roughly .6% of the graphs searched, while DDL always was able to find a solution.

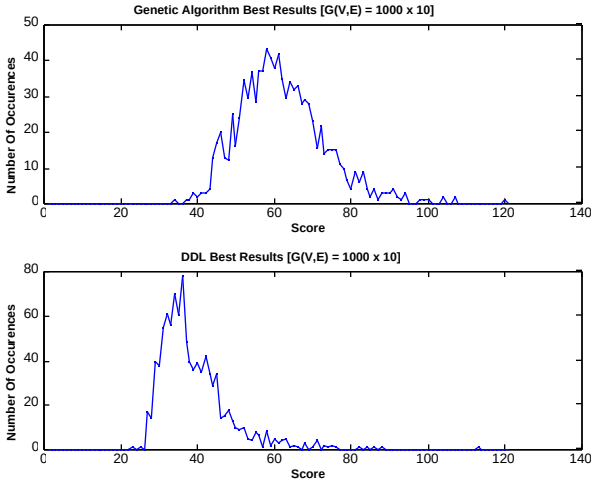

Figure 1: Histograms of solution score of DDL vs. GA.

This technology can be applied to a variety of parallel computing systems that are typically used for real-time pipelined parallel digital signal processing applications. For instance, ground-based supercomputing centers could receive real-time sensor data and process the data stream. As another example, large clusters of embedded multiple processor cards could be installed in several racks in a command and control aircraft. For both of these systems, the DDL algorithm would choose the appropriate resources upon which the sensor data processing would be executed.

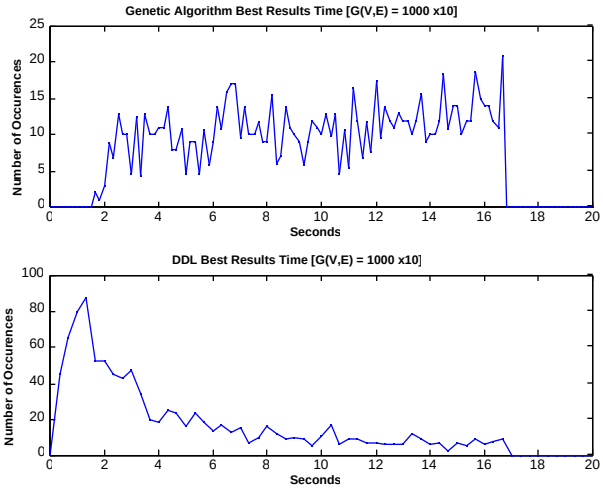

Figure 2: Histograms of time to produce best result for graph searching for GA and DDL.

References

[1] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure*, Morgan-Kaufman, 1999.

[2] I. Foster and C. Kesselman, Globus: "A Metacomputing Infrastructure Toolkit." I. Foster, C. Kesselman. *Intetnational Journal of Supercomputer Applications*, 11(2):115-128, 1997.

[3] A. Grimshaw, A. Ferrari, F. Knabe, and M. Humphrey, "Wide-Area Computing: Resource Sharing on a Large Scale," *IEEE Computer*, May 1999, pp. 29-37.

[4] D. Epema, M. Livny, R. van Dantzig, X. Evers, and J. Pruyne, "A Worldwide Flock of Condors: Load Sharing Among Workstation Clusters." *Future Generation Computer Systems*, 12:53-65, 1996.

[5] H. Hoffmann, J. Kepner, and R. Bond, S3P: Automatic, Optimized Mapping of Signal Processing Applications to Parallel Architectures," HPEC 2001.

[6] J.R. Budenske, R.S Ramanujan, and H.J Siegel, "On-Line Use of Off-Line Derived Mappings for Iterative Automatic Target Recognition Tasks and a Particular class of hardware," In *Proceedings of the Heterogeneous Computing Workshop, 1997 (HCW '97)*, pp. 96- 110.