Dynamic Resource Management for a Sensor-Fusion Application via Distributed Parallel Grid Computing

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

May 31, 2003

Abstract

Sensor data fusion applications generally use networked computational resources to process information. For high sensor data rates, distributed, parallel computing is necessary to keep up with the processing of the incoming signals, and the dynamic management of these geographically dispersed resources is necessary for fault tolerance and efficient use of resources. This research reports on a proof of concept project and demonstration in which a high data-rate sensor fusion application was deployed on a distributed, parallel, dynamic "grid" network of computers.

Introduction

Fast computational and internetworking technologies enable grid computing [1], in which applications are dynamically 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 tools needed to realize this form of location independent computing; however, these tools do not adequately address the requirements of real-time, pipeline-streamed digital signal processing applications that have strict throughput and latency requirements. The Precision Targeting via Collaborative Networking (PTCN) project addressed these issues; in the project a novel Network Resource Manager (NRM) software system [5] was developed to manage how pipelined signal processing applications are launched and executed on a set of networked computational resources. The NRM is particularly useful for managing sensor fusion applications with real-time processing performance requirements.

The System Testbed and the Application

A block diagram of the experimental testbed is shown in Figure 1. The testbed consisted of two SGI Origin servers (named Or-J and Or-S), an SGI Octane2 workstation (named O2), a Pentium4 Linux workstation (named P4-NRM), and eight Pentium3 Linux servers (named P3-1, …, P3-8).

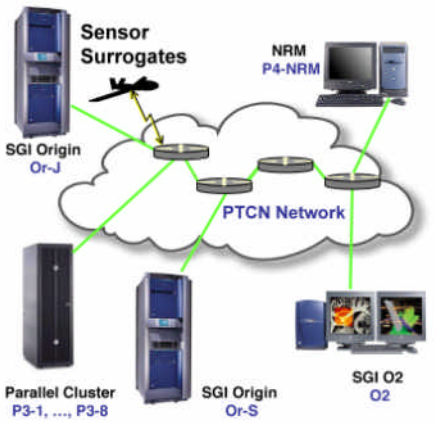

Figure 1: Topology of experimental network of computers.

The sensor data fusion application used for the proof of concept demonstration was OASIS (Operator Assisted Search Integrated System), which is an automatic target recognition and visualization suite. As depicted in Figure 2, OASIS processes both real-time synthetic aperture radar (SAR) data and archived data generated by sensors operating in different modalities (e.g., EO, IR, etc.) to generate terrain maps over which target locations and identities are overlaid.

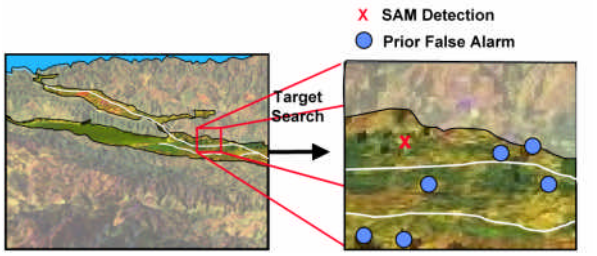

Figure 2: OASIS target detections on terrain-delimited imagery.

The Proof of Concept Experiment

As shown in the four-stage task mapping graph in Figure 3, an SGI Origin (Or-J) was used as sensor surrogates in the first stage to transmit unprocessed complex SAR imagery generated with range and cross-

This work is sponsored by the Defense Advanced Research Projects Agency (DARPA) under Air Force contract F19628-00-C-002. Opinions, interpretations, conclusions and recommendations are those of the author and are not necessarily endorsed by the United States Government.

range resolutions of 1m and ⅓m, depending on application parameters. The second stage, front-end processing, consisted of detection, geo-location, discrimination, and recognition work, which was all conducted by Or-S. The third stage was classifier processing which consisted of High-Definition Vector Imaging (HDVI) and matched filter processing [5]. This highly compute-intensive stage was conducted on each target detected in the SAR images. Finally, the back-end processing and user displays were handled by O2.

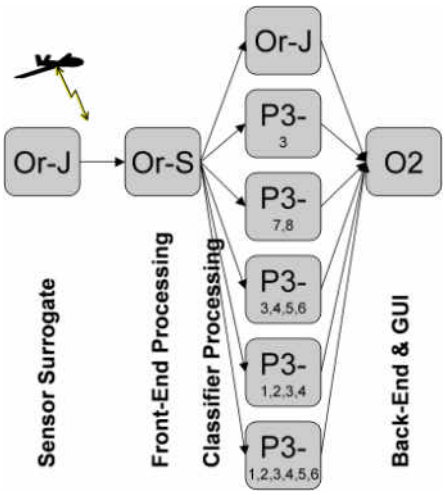

Figure 3: Task assignment map for OASIS applications.

Since the classifier (HDVI) and matched filter stage required the most processing power, this stage was chosen for the dynamic task-resource allocation testing. Since the network was processor resource limited, it required judicial resource allocation to be able to run multiple OASIS application instantiations simultaneously. Therefore, the NRM had to take into account processors' utilization, processors' bandwidth, task latency, system latency, stage throughput, network utilization, and network link bandwidth, in order to make an optimal resource allocation for each OASIS application instantiation that was launched. Furthermore, a degree of fault tolerance was required, and the NRM could deliver task restart or full processing chain failover depending on application parameters.

The proof of concept experiment involved launching multiple instantiations of the OASIS application. For 1m SAR data streams, the classifier (HDVI) stage required at least two P3 cluster processors, while for the ⅓m, it required at least four P3-cluster processors. As can be seen in Figure 3, this required the NRM to be extremely judicial in loading the processors with tasks.

In the demonstration, we were able to execute up to four unique instantiations of the OASIS application simultaneously. The NRM was able to optimally load each of the processors for this to occur. Also the NRM successfully executed both task restart and full processing chain failovers.

Future Work

The OASIS application data stream was limited by the speed at which the sensor surrogate computer was able to read sensor data off of disks and the performance of a target database. As part of our future work, we intend to tailor the NRM for use with more typical stream data processing applications such as those written in PVL [7]. Also, the task-resource map generation and stage timings were derived by hand and using system models. We intend to develop an automated task-resource map generator, which will facilitate the generation of much larger task-resource maps, and a task-stage execution timing mechanism so that actual system measures can be used in the graph that the NRM analyses. Finally, we would like to upgrade this testbed to become a platform for evaluating technologies for real-time Linux processing including Myrinet, RT-Linux, etc.

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." *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] J.I. Goodman, A.I. Reuther, et al., "Discrete Optimization using Decision-Directed Learning for Distributed Networked Computing," In *Proceedings of the IEEE Asilomar Conference on Signal, Systems, and Computers*, 2002.

[6] G.R. Benitz, "High-Definition Vector Imaging," *MIT Lincoln Laboratory Journal, Special Issue on Super-Resolution*, 10(2), pp. 147-170, 1997.

[7] E. Rutledge and J. Kepner, "PVL: An Object Oriented Software Library for Parallel Signal Processing," In *Proceedings of the 2001 International Conference on Cluster Computing*, 2001.