Commodity Computing Clusters at Goddard Space Flight Center

John E. Dorband

Josephine Palencia Raytheon

Udaya Ranawake

Goddard Earth Sciences & Technology Center

Introduction

The purpose of commodity cluster computing is to utilize large numbers of readily available computing components for parallel computing to obtaining the greatest amount of useful computations for the least cost. The issue of the cost of a computational resource is key to computational science and data processing at GSFC as it is at most other places, the difference being that the need at GSFC far exceeds any expectation of meeting that need. Therefore, Goddard scientists need as much computing resources that are available for the provided funds. This is exemplified in the following brief history of low-cost high-performance computing at GSFC.

The GSFC group was formed in the mid-1970's in response to the need for large amounts of computation for processing LANDSAT images. It was recognized that mainframes and mini-computers could not provide sufficient processing power for processing this data. Even at that time funds were limited at NASA to build such a processing resource.

Initially it was decided to look into using optics to process the LANDSAT data. The idea was to process whole images at a time rather than one image pixel at a time. It soon became apparent that optical computing might not be ready soon enough for LANDSAT, yet, due to the advances in VLSI technology, electronics might be.

A shift in effort from optics to electronics resulted in procurement of the NASA/Goodyear Massively Parallel Processor (MPP). It was a 16,384 ALU SIMD (single-instructions stream/multiple-data stream) computer that could process a 128x128 section of a LANDSAT scene at a time, not a whole image, but a good-sized portion of one. The development cost of the MPP was only $7 million. It arrived at GSFC in 1983 and became an invaluable resource to study both computer architecture as well as NASA science applications running on a parallel architecture. Throughout the 1980's several similar architectures became commercially available: GAP, ASAP, ICL DAP, and Thinking Machine's Connection Machine to name a few. In 1990, GSFC obtained a MasPar MP-1, which was four times as fast as the MPP and cost $1 million. In 1992 GSFC obtained a MasPar MP-2, which was as fast as a Cray YMP on a PPM (piecewise parabolic method) fluid dynamics code. In 1993 in cooperation with MasPar Inc. GSFC clustered 4 MP-2's (cost ~$6 million), which perform faster than a 16-processor C90 on the above-mentioned PPM code.

During this period of time, the MP-2 was exhibiting uptimes of months if not a year or more, while users of MIMD (multiple -instructions stream/multiple-data stream) machines had to be satisfied with uptimes of a day or even hours. LINUX had just shown up on the scene and had the potential of being quite reliable due to the large number of developers and users. Later it became apparent that large numbers of developers could also be a disadvantage as well as an advantage.

In 1994, a team was put together at GSFC to build a cluster consisting only of commodity hardware (PC's) running LINUX, which resulted in the first Beowulf cluster (later renamed

Cluster	#proc	#node	type	CPUGhz	Net Ghz	RAM (GB)	Disk (GB)	Year	Cost $
theHIVE	128	64	P Pro	0.2	0.10	28	896	1997	250K
pivot-g	32	16	P3	0.5	1.28	8	140	1999	100K
pivot-d	40	10	P3	0.5	1.28	5	90	1999	100K
topaz	32	16	P3	0.7	1.28	16	200	2000	270K
bbblue	32	16	P3	0.65	1.28	16	300	2000	270K
orka	32	16	P3	0.7	1.28	16	100	2000	160K
medusa	128	64	Athlon	1.2	2.00	64	2560	2002	220K

TABLE 1
CT application proto-typing clusters.

(Wiglaf). It consisted of 16 100Mhz 486DX4-based PC's. The PC's were connected with 2 hub-based Ethernet networks tied together with channel bonding software so that the 2 networks acted like one network running at twice the speed. This demonstration cluster showed that one could utilize commodity hardware to build a very cost effective, moderately fast computing platform. The next year a 16 PC cluster, Beowulf II, Hrothgar, based on 100Mhz Pentiums was built and was about 3 times faster, but also demonstrated a reliability comparable to the previously mentioned MasPar MP-2. At one point it had not crashed in 9 months before someone accidentally shut off its power. At that time impressive performance had not yet been demonstrated. The next year (1996) a Pentium-Pro cluster at Caltech demonstrated a sustained giga-flop on an application. This was the first time a commodity cluster had shown high performance potential.

Up until 1997, the commodity clusters at GSFC were in essence engineering prototypes, that is, they were built by those who were going to use them. In spring of 1997 a project was started to build a commodity cluster that was intended to be used by those who had not built it, the HIVE (highly parallel virtual environment) project. The idea was to have workstations distributed among many offices and a large number of compute nodes (the compute core) concentrated in one area. The workstations would share the compute core as though it was apart of each. Though the original HIVE only had one workstation, many users were able to access it from their own workstations over the Internet. Many

non-builders were able to develop parallel scientific applications on the HIVE and one scientist, Dr. Michael Gross, produced the first published scientific results facilitated by the HIVE. The HIVE was also the first commodity cluster to exceed a sustained 10 Gflop on an algorithm. Since 1997 GSFC has had clusters built from components from DELL, Gateway, SGI, IBM, and VALinux, which have become the mainstay of scientific parallel program prototyping for commodity clusters here at GSFC.

Recently the cluster, medusa, was built from 64 dual 1.2 Ghz AMD Athlons connected with 2 Ghz Myrinet. This cluster finally embodied the intended concept of the HIVE. Along with the 128-processor compute core, medusa has more than a dozen workstations throughout a building at GSFC attached to the core with 2 Ghz optical fibre Myrinet. Several clusters are being planned and built throughout GSFC to support missions and scientific research. Table 1 lists the CT application prototyping clusters.

Applications

Many applications and algorithms have been developed on CT clusters. The following are just a few of these applications.

Hierarchical Image Segmentation (HSEG) in Remotely Sensed Multi-spectral or Hyperspectral Imagery

Dr. James C. Tilton
Code 935/Applied Information Sciences Branch
NASA/Goddard Space Flight Center
Email: *James.C.Tilton.1@gsfc.nasa.gov*

The process of image segmentation is highly useful in the field of Earth remote sensing. Computing speeds can be enhanced when an image is partitioned into sections or regions among the nodes of a Beowulf Cluster. The regions may consist of groupings of multi-spectral or hyper-spectral image pixels with similar data feature values. These related regions are then tagged with informational labels to extract information regarding the ground cover or land use.

In his previous work with the old HIVE Cluster, Dr. Tilton programmed a version of his HSEG algorithm using C and PVM for parallelization. He did encounter some processing window artifact problems that occur when processing large images.

In the new version of HSEG running on the new HIVE2 Cluster, he is able to overcome this problem. He plans to use the new C++/MPI version of HSEG in his NRA-funded research project: "Knowledge Discovery and Data Mining Based on Hierarchical Segmentation of Image Data". He will be reporting the preliminary results from this project in a paper he presented at the International Geoscience and Remote Sensing Symposium 2002, Toronto, CA, June 24-28, 2002. The paper, co-authored with G.

Marchisio, K. Koperski and M. Datcu, is titled "Image Information Mining Utilizing Hierarchical Segmentation."

URL: http://www.nasamedicalimaging.com/hseg
http://code935.gsfc.nasa.gov/code935/tilton

Parallel Matlab on Beowulf Machines

Dr. J. Anthony Gualtieri
Code 935/Global Science and Technology
NASA/Goddard Space Flight Center

Interactive programming environments with powerful graphical and image display capabilities, such as Matlab, are available on single processor workstations. To extend the capability of this tool to tackle more advanced problems, we must learn to integrate these environments in a Beowulf cluster.

- A software system called Matlab*P (written by Parry Husbands and Charles Isbel) combines these two approaches to the scientific computing and offers a resolution of the implementation challenge. Currently, Matlab*P is installed on several clusters at GSFC for the development of remote sensing classification codes.
- The user avails herself of Matlab syntax with polymorphism to write, for example, a two-dimensional square random array as a = rand (4000*p); where the *p is parsed to indicate that a parallel variable that will distribute itself across multiple cluster processors. Then operator overloading transparently allows the use to invert this matrix by writing ainv = inv (a). A client node of the cluster runs a single copy of standard Matlab which has been extended with Matlab functions that are coupled to dynamically loaded packages that link Matlab on the client to the server running on the compute nodes of the cluster. The server nodes are running SCALAPACK on top of MPI to provide the actual computation. To the user the full interactive capability of Matlab is always available, including all the single processor capability of ordinary Matlab, but now all the standard linear algebra functions from SCALAPACK are available to perform the computations. The user never has to deal with any message passing additions to the standard Matlab code.

References

Proceedings of the Third International Conference on Vector and Parallel Processing, "The Parallel Problems Server: A Client-Server Model for Large Scale Scientific Computation (1998)", Husband and Isbel

Advances in Neural Information Processing Systems 12, "The Parallel Problem Server: An Interactive Tool for Large Scale Machine Learning (1999)," Husband and Isbel

Hyper-spectral Imagery Dimension Reduction Using Principal Component Analysis on the HIVE

Sinthop Kaewpijit
School of Computational Sciences
George Mason University
Sinthop@science.gmu.edu

Tarek El-Ghazawi
Department of Electrical and Computer Engineering
George Washington University
tarek@seas.gwu.edu

Jacqueline Le Moigne
Applied Information Science Branch
Code 935, NASA/ Goddard Space Flight Center
Lemoigne@backserv.gsfc.nasa.gov

Hyper-spectral data, with observations collected at hundreds of bands, are produced by some of the operational NASA instruments. These remote sensing technology developments will facilitate many new applications of Earth and Space Science. On the other hand, the increased data volumes prompt the need for much faster processing and methods for data reduction. Dimension Reduction is a spectral transformation, aimed at concentrating the vital information and discarding redundant data. One such transformation, used widely in remote sensing, is Principal Component Analysis (PCA). Parallel algorithms have been developed for PCA, along with implementations and performance measurements on the HIVE Cluster. An innovative technique via wavelet decomposition is introduced as a new choice for reducing dimensionality of hyper-spectral data.

Reference:

Science Data Processing Workshop 2002
URL: http://that.gsfc.nasa.gov/gss/workshop2002

Parallel Adaptive Mesh Refinement (PARAMESH)

Dr. Peter MacNeice
Code 935, HPCC
NASA/ Goddard Space Flight Center
E-Mail: macneice@alfven.gsfc.nasa.gov

Dr. Kevin Olson
University of Chicago

NASA/Goddard Space Flight Center
Email: olson@bohr.gsfc.nasa.gov

Paramesh is a software package used primarily for the automatic parallel adaptive mesh refinement (AMR). It consists of Fortran 90 subroutines with MPI calls designed to extend an existing serial code, which uses a logically Cartesian-structured mesh into a parallel version with block-structured adaptive mesh refinement. The package builds a hierarchy of sub-grids to cover the computational domain, with varying spatial resolution to satisfy the demands of the application. These sub-grid blocks form the nodes of a tree data-structure (quad-tree in 2D or oct-tree in 3D); each grid block has a logically Cartesian mesh.

Paramesh maintains the mesh data structure, distributes them to processors for load balancing, and handles all communications. The users can then construct their code to call a subset of the high-level Paramesh subroutines. The package supports 1, 2, and 3D models.

URL: http://sdcd.gsfc.nasa.gov/RIB/repositories/inhouse_gsfc/Users_manual/amr.html

Applications developed with Paramesh:

1) FLASH, A Parallel, Adaptive Code for Astrophysics

Kevin Olson, Bruce Fryxell, Frank Timmes, Paul Ricker, Mike Zingale, Jonathan Dursi, Alan Calder, Henry Tufo, Robert Rosner, and Peter MacNeice

FLASH is a general-purpose Astrophysics code that uses PARAMESH. It currently includes modules for compressible fluid dynamics using PPM (Prometheus, Fryxell and Muller), gamma law equation of state, stellar equation of state, reactive flow with a multi-species nuclear burning network, nuclear energy generation, constant gravitational field and other forcing terms and AMR and parallelism using Paramesh.

The current scientific work being done with FLASH includes XRAY bursts in 2-3D (img013.gif), novae and supernovae studies, detonation front instabilities (img014.gif), crushed turbulence, and burning front-vortex interaction. Future plans for the FLASH code include development of modules for self-gravity using the multi-grid algorithm, implicit thermal diffusion, radiation, different fluid solvers (e.g., incompressible), MHD, AMR using other packages (e.g., SAMRAI) and time adaptivity.

URL:http://www.csar.uiuc.edu/~hoefling/workshop-may00/Slides/KevinOlson/chicago/sld001.htm

2) *AMRMHD3D: A 3-D Flux-Corrected Transport Code with Adaptive Mesh Refinement for Ideal, Compressible Magnetohydrodynamics*

AMRMHD3D is a new, three-dimensional, magnetohydrodynamics model developed on the Cray T3E computer system. It extends the FCTMHD3D model for fixed grids to adaptively refined grids constructed by the parallel meshing package PARAMESH. The equations are solved conservatively in a finite-volume representation with explicit two-step Runge-Kutta to advance the variables. The multiple subroutine package of AMRMHD3D is written in Fortran 90 and consists of 79 Fortran source files and 27 header files in two groups: the FCTMHD3D application group (22 sources, 19 headers) and the PARAMESH AMR group (57 sources, 8 headers).

URL:
http://www.lcp.nrl.navy.mil/hpcc-ess/amrmhd3d.10.html
AMRMHD3D source code:
ftp://lcp.nrl.navy.mil:/pub/hpcc-ess/amrmhd3d.t3e.tar.Z.

3) ATHENA: A 3-D MHD Code with Adaptive Mesh Refinement for Modeling Global Magnetosphere and Accreting Magnetized Stars

Dr. Daniel Spicer
Code 930, Senior Scientist
NASA/ Goddard Space Flight Center

ATHENA is a new, three-dimensional 3D magnetohydrodynamics (MHD) code that uses dynamic adaptive mesh refinement. The equations are used in a finite volume representation using Colella's High Order Godunov Corner Transport Upwind scheme, together with a staggered mesh scheme to insure the magnetic fields remain divergence free. The AMR package used is the PARAMESH package developed by Peter MacNeice and Kevin Olson. Presently it is used to model the global magnetosphere and accreting magnetized stars.

4) ATHENA/AMR

Michael L. Rilee
Maharaj K. Bhat
High Performance Computing
NASA Goddard Space Flight Center
mbhat@hannibal.gsfc.nasa.gov
mrilee@hannibal.gsfc.nasa.gov

In this work, the interaction of solar flux with earth's magnetic field is studied, employing Athena/Paramesh codes. While Athena provides the numerical algorithms, Paramesh

software divides the computational domain into large numbers of blocks, distributes these blocks onto processors and facilitates runtime refinement/derefinement. Run time flow visualization reveals evolving physics and monitors progress of simulation. Athena/Paramesh computational environment is ideal for distributed computing that is provided by a cluster like HIVE II.

5) Numerical Relativistic Astrophysics Group

Dr. Joan Centrella
Relativistic Astrophysics Group
Code 661, LHEA
NASA/Goddard Space Flight Center
jcentrel@milkyway.gsfc.nasa.gov

A core objective of the numerical relativistic astrophysics group is to develop and apply codes for solving Einstein's gravitational field equations using finite difference simulations with adaptive mesh refinement (AMR). These tools will be used to model astrophysical sources of gravitational waves as required for the analysis of data produced by the planned Laser Interferometer Space Antenna (LISA).

The group is also involved in the Lazarus project, an effort producing approximate models for gravitational waves binary black hole systems using a combined approach involving numerical simulation together with perturbation theory techniques in the regimes where they are applicable.

Space Interferometry Mission on Dynamics of Galaxies (SIMDOG)

Dr. Edward Shaya
Institute of Science and Technology at Raytheon
Code 630.1, NASA/Goddard Space Flight Center
Email: shaya@mail630.gsfc.nasa.gov

As part of the SIM Key Project on Dynamics of Galaxies (SIMDOG), we will be calculating the trajectories of nearby galaxies given the present positions and velocities. This is somewhat like running an N-body gravitational code, however, because we are also constrained by the fact the peculiar motions were zero at t=0, it is more similar to a boundary valued differential equation problem. The actual method is called numerical action, and it relies on classical mechanics. In action theorems, one solves for paths for which the integrals of the actions over all time are extreme. This typically requires solving for the minima of about 300 parameters per galaxy, and we usually calculate paths for several thousand galaxies. Since the potential field changes as the paths are varied, it is a very challenging numerical problem. We typically run grids of nodes with each node solving for orbits in universes of different cosmological parameters. The problem thus parallelizes very efficiently and easily.

URL: http://sim.jpl.nasa.gov/ao_support/ao_abstracts.html

Using a Beowulf Cluster in Land Information System (LIS)

Dr. Paul Houser
Dr. Christa Peters-Lidard
Hydrological Department
NASA Goddard Space Flight Center
Paul.R.Houser.1@gsfc.nasa.gov
Christa.D.Peters-Lidard.1@gsfc.nasa.gov

The Land Information System (LIS) will be a hardware transparent integrated software and database system, focused on a high-resolution (1km) global land data assimilation with several independent community land surface models, land surface data assimilation technologies, and integrated database operations for data management. The eventual throughput and storage requirements of the LIS (approximately 1TB per day simulated) exceed any available or planned NASA computing platform, and hence a custom 194-node Beowulf cluster is being constructed to support the work. The relatively weak horizontal physical coupling of global land surface processes works well with a large-scale distributed memory parallel processing software design.

The custom Linux Beowulf cluster at NASA/GSFC will consist of one "queen" or control node and at least 192 "compute" nodes connected by at least fast Ethernet. The queen node will control the data staging and model execution, and the compute node segments will be partitioned into input data preprocessing/interpolation, model execution, and output data post-processing and gathering. The queen node will consist of dual 1.5Ghz processors, 4 GB RAM, and 2 TB storage or better while the compute nodes will consist of single 1.2Ghz processors, 512 MB RAM, and 80 GB storage or better. We anticipate gigabit connections between the compute node switches and the queen nodes as well as between the queen nodes and the GSFC network.

URL: http://lis.gsfc.nasa.gov/

Scientific Visualization Studio

Randall Jones
Scientific Visualization Studio
NASA/Goddard Space Flight Center

The SVS (Scientific Visualization Studio) utilizes various scientific visualization tools to provide products and services to scientific communities. We have a wealth of experience working with high-performance, professional graphics computers, which utilize advanced, hardware-based graphics rendering. The SVS seems to be an ideal place to introduce a new generation of interactive hardware graphics tools.

In the last few years, commodity, PC-based graphics accelerators have made great advances in performance while maintaining their commodity pricing. It has been shown and proven that a significant increase in graphics performance can be achieved by breaking the rendering work into smaller components and distributing these to many PC computers (nodes), each running a hardware graphics accelerator. The graphics accelerators on the individual nodes compute a portion of the scene, which is then assembled to produce an output image. There are two common models of parallel hardware rendering we intend to explore. The first method utilizes multiple displays, each attached to its own graphics accelerator and outputting a portion or "tile" of the total display image. The final result is high (higher than a single display) resolution image composed of tiled displays. Second is a parallel rendering approach where a portion of the scene to be rendered is sent to each node, computed using the graphics accelerator and then read back into a "master" node to be displayed in a single graphics window. This parallel rendering approach has the advantage of graphics performance increases while maintaining the familiar use and interaction of a local window.

A small cluster was specified to provide an adequate test bed for learning and experimenting with current technologies in the area of hardware-assisted, cluster-based parallel rendering. There are a few software packages that one can start with. A package called WireGL is a parallel OpenGL library and framework for running existing OpenGL graphics applications in parallel without modification to the application. There is also a follow-on project to WireGL, based on the WireGL code base called Chromium provides a more general and more customizable framework. OpenGL-based image viewer can be used to view high-resolution imagery continuously on a large tiled display.

References

S. Molnar et al., "A Sorting Classification of Parallel Rendering," IEEE Computer Graphics and Applications, vol. 14, no. 4, July 1994, pp. 23-32.

H. Igehy, G. Stoll, P. Hanrahan, "The Design of a Parallel Graphics Interface," ACM Computer Graphics Proceedings, 1998, pp. 141-150.

G. Humphreys, M. Eldridge, I. Buck, G. Stoll, M. Everett, P. Hanrahan, "WireGL: A Scalable Graphics System for Clusters," ACM Computer Graphics Proceedings, 2001, pp. 129-140.

G. Humphreys, R. Frank, S. Ahern, "Specification for Stanford/DOE Cluster-Rendering Infrastructure," WireGL/Chromium Project documentation.

G. Humphreys, P. Hanrahan, "A distributed graphics system for large tiled displays," IEEE Visualization Proceedings, 1999, pp. 215-227.

B Wylie, C. Pavlakos, V. Lewis, K. Moreland, "Scalable Rendering on PC Clusters," IEEE Computer Graphics and Applications, vol. 21, no. 4, July/August 2001, pp. 62-70.

Benchmarks

We evaluated the performance of the communication network of the PC cluster and also its performance on the class A NAS parallel benchmarks. The communication performance of the network was measured using the MPI communication libraries for the Ethernet and the Myrinet. These include the latency, the bandwidth, and the MPI group communication functions such as reduce, broadcast, and barrier.

The round-trip time between two nodes was measured using the MPI_Send and MPI_Recv functions for various message lengths. For small messages, the fixed overhead and latency dominate the roundtrip time. For large messages, the roundtrip time increases linearly with message size. Half the roundtrip time for short messages is a measure of the latency of the system. For MPICH under Ethernet, the latency was about 0.1 ms. For Myrinet, it was about 0.01 ms. The bandwidth between two nodes is obtained by dividing the message length by half the roundtrip time. For the Ethernet and the Myrinet the bandwidths were 94 Mbits/sec and 1,266 MBits/sec respectively. Therefore, the bandwidth under Myrinet is about 10 times the bandwidth under Ethernet. It was noticed that the MPI group communication functions also showed the same performance ratio for various message sizes under the two networks.

The NAS parallel benchmark consists of six parallel kernels and three simulated application benchmarks. These are:

Elegantly Parallel	EP
Fast Fourier Transform	FT
LU Decomposition	LU
Multi-grid	MG
Conjugate Gradient	CG
Integer Sort	IS

Here, we consider the performance of the six classes A kernel benchmarks. The programs were compiled using the g77 compiler and linked with the MPICH libraries for the Ethernet and Myrinet. Table 2 lists the execution times for the 64-processor case. We notice that on 64 processors, the PC cluster under Myrinet performs better than the CRAY T3e 900 on all the computational kernels of the NAS benchmarks.

	PC(ether)	PC(myri)	T3e
EP	2.78	2.82	3.2
FT	1.1	5.59	2.9
LU	11.42	20.89	27.6
MG	0.42	0.97	0.8
CG	0.75	2.93	1.3
IS	0.31	--	1.1

TABLE 2. NAS class A benchmark times in seconds.

References:

William Gropp, Ewing Lusk, Nathan Doss and Anthony Skjellum, “A High-Performance, Portable Implementation of the MPI Message Passing Interface Standard", available at: http://www.mcs.anl.gov/mpi/mpich/

"NAS Parallel Benchmarks" available on the WWW at: http://www.nas.nasa.gov/NAS/NPB/

"NAS Parallel Benchmarks 2 Detailed Results" available on the WWW at: http://www.nas.nasa.gov/Software/NPB/NPB2Results/

Conclusion

By necessity, the overwhelming computational needs of earth and space scientists have driven GSFC to be one of the leaders in the application of low cost high-performance computing.