

# **A High Performance Computing Primer**

**Academic Services and Technologies**

**Leslie P. Michelson, Ph.D.**

**John E. Kerrigan, Ph.D.**

**July 2006**

High Performance Computing might be described as the application of a level of resource that is well matched to the demands of the problem to be solved. This is particularly true when the problem is well above the ordinary challenge. ENIAC, one of the first electronic digital computers, developed at the Moore Laboratory at the University of Pennsylvania could be classified as High Performance in its day. After all, it was able to compute the complete trajectory of a ballistic missile in less time than the actual flight—certainly high performance at the time and many, many times faster than the scores of “human” computers” previously assigned the task. Today, the most challenging problems find solution on a class of machinery that is reaching speeds of one petaflop or one thousand trillion ( $10^{15}$ ) floating point operations/second. ENIAC operated at speeds of several operations to five thousand operations/second depending on the actual function or functions.

The life sciences contribute more than their share to some of the most vexing computational problems. As a perspective, it would not be unusual for the simulation of a nanosecond of “life” in a real system to require weeks of computer time on some very capable machinery. The class of Grand Challenge Problems whose solution is amenable in whole or part to computer simulation will also require the most powerful resources available at any point in time. It is not likely that any one organization, other than large government funded labs, certain private sector interests, and to a far lesser extent, Universities will own this level of resource. However, through judicious planning, modest investment and active collaboration, Universities can make some inroads into computationally intensive biological/biophysical problems. Many problems of real importance to the life sciences are tractable on “relatively” modest resources. In fact some problems (e.g. structural dynamics of membrane-bound proteins) can only be addressed via the computer. A number of our competitor institutions have made significant investments in high performance computing assets based on cheaper, simpler cluster technologies.



#### Homology Model of Hamster Tumor Necrosis Factor alpha Converting Enzyme.

Researchers can learn how structural changes caused by genetic mutations can lead to disease using high performance computing tools like molecular dynamics, docking and comparative modeling.

HPC environments exist in multiple flavors including high performance multiple processor servers with very fast processor and I/O interconnects, clusters of identical computers (typically Linux) with specialized software designed to leverage aggregate capability, computing grids comprising heterogeneous computing types under the management of software capable of dispatching and controlling jobs based upon a set of parameters describing program requirements, priority and other factors. New technologies, e.g., cell computers, may offer vastly more computing density than is currently available, perhaps at the expense of scope of applicability.

No single environment is suitable for all types of applications. Programs requiring ultra high connectivity between processors are generally more suitable to a single multiprocessor environment or cluster with very fast interconnects (e.g., Infiniband). Others, which can be easily "chunked," may be efficiently dispatched to a grid environment. Available programming resources are yet another consideration. AST can assist researchers with these decisions and suggest whether UMDNJ or external resources are better suited.

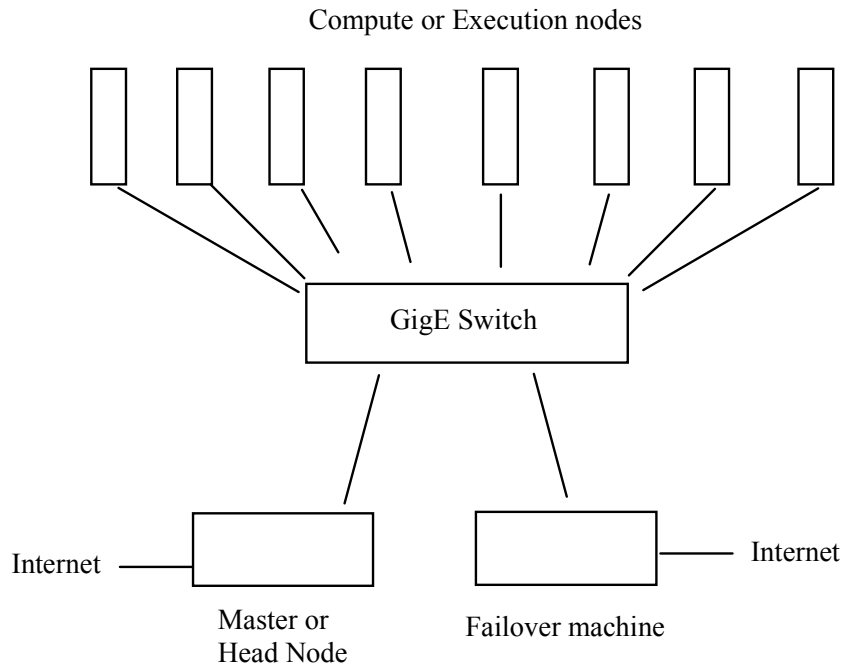
## HPC Architectures

Today, High Performance Computing (HPC) can be defined as computing devices, attendant storage and network services/connections which permit computational problems that would be intractable on common desktop devices for reasons that include but are not limited to processor speed, number of processors, memory capacity, memory cache size, external storage, bandwidth to external storage and other resources needed to successfully tackle the problem at hand. A standard approach to achieving very high performance and one that has been used successfully for quite some time is to employ multiple processors in a conventional chassis. Computers with dozens to hundreds of processors connected by a very fast common backplane are widely available. Increasingly, these machines are populated with multiple core processors with each core supporting multiple threads. Architectures of this type work equally well for broad mixes of unrelated tasks or a single task able to take advantage of parallelization. These architectures benefit from their legacy heritage and are often specified for applications that require a significant amount of computing muscle coupled with stable, high availability performance. The EPIC, electronic health record environment implemented at University Hospital runs on such equipment. The Academic Systems and Technologies division of IST maintains a SunFire 6900 with 24 dual-core processors, 48 Gbytes of main memory and fiber interconnects to a high performance SAN (storage area network). The 6900 is capable of tackling fairly demanding computing tasks and supports a repertoire of scientific codes in widespread use in computational biology. These programs are described elsewhere in the current portal topic.

Massively parallel machines, comprising thousands of processors (typically Intel architecture), have also been constructed. Such custom-built machines are characterized very high price tags and the software complexity associated with exploiting the potential of a specialized architecture. Often, these machines are built to tackle a particular set of problems.

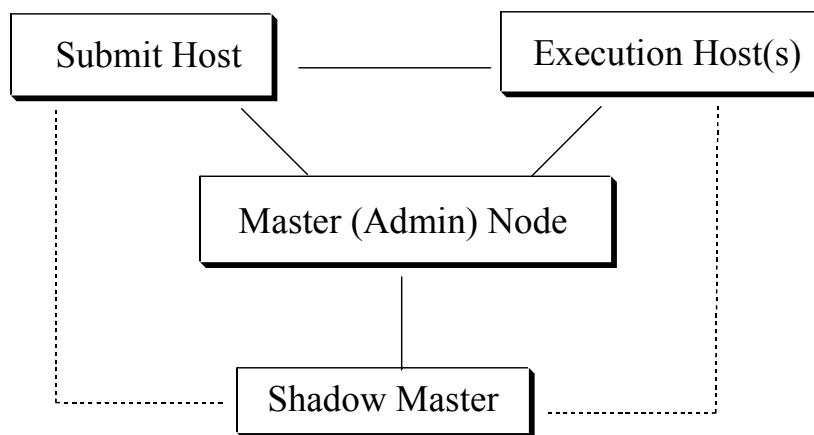
Another approach, and currently the most popular, is compute clusters comprised of relatively low-cost PC-class processors, managed by a “head” node. Clusters comprising thousands of processors can and have been developed. The two major open source software ensemble packages for cluster building are ROCKS (<http://www.rocksclusters.org/wordpress/>) and Clustermatic (<http://www.clustermatic.org/>). Other open source packages are available. The IBM Blue Gene ([http://domino.research.ibm.com/comm/research\\_projects.nsf/pages/bluegene.index.html](http://domino.research.ibm.com/comm/research_projects.nsf/pages/bluegene.index.html)) project (uses 8000 PowerPC chips) and was developed to tackle Grand Challenge protein folding problems.

---



Typical Arrangement for a Cluster System

Linux is the most popular operating environment for such clusters. A number of open source and proprietary packages are available to manage cluster resources, dispatching and matching tasks to resources and maintaining local utilization policy. These “managers” are often capable of supporting ensembles of heterogeneous computing architectures including clusters and multiprocessors such as the Sunfire 6900. AST has implemented Grid Engine, to manage its current assets. New compute environments can be added to the Grid Engine management environment. Although Grid Engine (<http://gridengine.sunsource.net/>), developed by Sun Microsystems and available in the public domain is certainly a candidate for evolving HPC assets at UMDNJ, AST will continue to investigate alternatives.



Grid Computing Setup

Several Linux clusters are in operation at the University. Please contact AST for more information.

Many considerations must be taken into account when planning a cluster for a particular performance target. A design such as the Sunfire 6900 employs a very fast (but expensive) backplane to connect all of the computing elements. Clusters use very fast network interconnects such as GigE, Myrinet and Infiniband for such connections. Network connectivity such as Infiniband can add a considerable cost to the cluster and must be carefully considered. Not all applications run over an Infiniband or Myrinet interconnect.

A relatively recent trend in Linux compute clusters uses processor assemblies called “blades.” Blades simplify the implementation and expansion of compute clusters by virtue of a dense plug-in design coupled with common hardware management for all blades in an enclosure. The downside is somewhat higher cost for a custom, proprietary design.

In theory, clusters can be arbitrarily large and, as noted, large clusters are not uncommon. Eventually, though, size and power become a limiting factor. One solution to this problem is “cellular architecture” developed by IBM (<http://www-03.ibm.com/press/us/en/pressrelease/19229.wss>) in which many processors, local memory and the switching necessary to connect the processor cores exist on a single chip. Although the idea is not new, product has not been available until quite recently. Cellular computers are likely to be less adaptable to a wide range of problems than more conventional designs. The Sony Play Station 3 is based on a cellular architecture designed for graphical computation. As such, this particular architecture may have application in biomedical visualization.

A significant challenge in HPC is adapting program code for a particular problem. Some of the most widely used computational codes in the life sciences have been parallelized for popular cluster and massively parallel platforms and work in this area continues with advancements in the popular MPI (message passing interface) and MPICH libraries. Compilers capable of generating parallel code are available for all HPC environments. Such compilers are, in general, only moderately successful in producing parallel code, but do provide a starting point. Oftentimes, a program will spend a significant percentage of time in a small portion of code, leading to the possibility significantly increasing performance by a handcoding a relatively small sections of the program. AST’s Sunfire 6900 supports a complement of parallelizing compilers for the C and Fortran programming languages capable of exploiting the Sunfire’s SMP (symmetrical multiprocessor environment).

---

---

## **Moving Forward**

AST is working with its neighboring research institutions to share knowledge and develop collaboration that may lead to more sharing of physical resources and human expertise. We expect to develop a better picture of HPC requirements at UMDNJ by meeting with researchers already using our HPC resources and others with an expressed interest in order to identify the best architectures for our needs.

---

---