INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN: 2278 – 0211 (Online)

High Performance Computing: A reality At Central-India

Ankush Rai
Department of Computer Science, Shri Shankrachariya Technical Campus, Junwani, (C.G), India

Abstract:

Supercomputers were once a resource limited to major universities and government agencies with huge funds at their disposal. Thus, the high costs of commercial parallel computing hardware and software place them beyond the reach of the researcher & students of many colleges and universities. Today, low-cost commodity hardware, free operating systems such as Linux, and faster, lower cost networking make it possible for institutions at all levels to provide supercomputing resources to their researchers. The article summarize design & performance information of the High Performance Computing (HPC) Laboratory that is builted up in Shankrachariya College of Engineering & Technology department of applied physics.

Key words: *Scientific computation, cluster, Computer Hardware Architecture, Parallel Computers.*

1.Introduction

1.1.Introduction Of Parallel Computing

All of today's fastest computers employ parallel processing technology in one form or another. What exactly does this mean, and would it work for your computing problem? Parallel computing is not always appropriate and for some applications there is little benefit to be made from switching to a parallel system. The real-time example & its implementation of parallel computing are discussed below.

The analogy of a supermarket is often used in describing parallel computation. Consider a large supermarket with a line of tills (drawer for money in bank & shop) at the front. Each till is a CPU and each customer a computer program. The items in the customer's trolley are the instructions which the CPU must process to run the program. Imagine the case where only one till is open, each customer has to queue and be served one at a time. This is a single-tasking operating system model, such as MSDOS.

Imagine now that the cashier scans one item each, in turn, of several customers. Everybody moves through the queue, though more slowly than if they were the only customer. This is a multi-tasking operating system example such as single CPU UNIX or Windows NT.

In the third scenario, several tills are opened. Each customer is processed by a separate cash register and the overall line moves much quicker. This is called SMP - Symmetric Multi Processing. Note in this case that although there are extra cash registers open you will not get through the line any quicker than the previous example if you are the only customer at the shop. The computer example here is UNIX or NT with multiple processors.

Better use of the system is developed through what is known as 'threading'. If you have a large amount of shopping you might be able to logically break up your shopping between multiple tills, say frozen goods at one till, fruit and vegetables at another and tins at the third. When the cashiers need to get sub-totals to work out the bill they can exchange information quickly by looking and talking to each other. In theory with three tills like this you should get served three times faster, but there are pitfalls. If you only have a small basket, sorting out the different items and adding up the totals at the end could take more time than would be gained by the extra tills. If you have far more frozen food than not then two of the tills will be sitting idle waiting for the third. This scenario

models a multi–threaded program running on a multi CPU motherboard with either NT or UNIX.

In order to serve customers more quickly the store adds more tills. There is no space at the front so the new tills are put in at the back of the shop. Now the cashiers have to phone in their subtotals to gather the total bill and it takes longer to take part of the order to the back of the shop. Even with this extra overhead though, if communication between the cashiers is minimized and you have a really big order then you can get a huge increase in speed by using all the cashiers at the front and the back of the shop. This is a classical parallel situation utilized in parallel machines, where the CPUs can be in different machines and communicate via messages.

For parallel computing to be successful, your application must have enough concurrency to make good use of multiple processors. This is not just a matter of identifying portions of the program that can execute independently. As we have seen, coordinating processes over different CPUs, especially in different machines, has an inherent time overhead, sometimes called network latency. If this is not managed carefully applications may run slower on a parallel system. For example, a section of code may take four seconds to run on a single CPU. It could run in one second on four parallel CPUs, but there would be no increase in speed if it took three seconds or more for these machines to co-ordinate their actions. Porting code to take advantage of a parallel system is not a trivial task but if handled correctly the benefits of increased speed can be enormous. In the taxonomy of parallel computers, Beowulf clusters fall somewhere between MPP (Massively Parallel Processors, like the nCube, CM5, Convex SPP, Cray T3D, Cray T3E, etc.) and NOWs (Networks of

Workstations).

1.2.Why Parallel Computing?

Many applications require more computing power than a traditional sequential computer can offer. Parallel computing provides a cost effective solution to this problem by increasing the number of CPUs in a computer & by adding an efficient communication system between them. The workload now can be shared between different processors. This, result is in much higher power, performance that could be achieved with traditional single processor system. The development of parallel processing is being influence by many factors. The prominent among them include the following:

- Solve bigger problems faster.
- Sequential architecture reaching physical limitation as they are constrained by the speed of light & the thermodynamic laws. The speed with which sequential CPU's can operate is reaching saturation point & hence an alternative way to get high computational speed is to connect multiple CPU's.
- Significant development in network technology is paving a way for heterogeneous computing.
- The technology of parallel processing is mature & can be exploited commercially; this is significantly excels in R&D work.
- Hardware improvements in pipelining, superscalar, etc, are non scalable & requires sophisticated compiler technology. Developing such compiler technology is difficult task.

In context of nanotechnology research High performance Computing will solve problems that we encounter during our research:

- Opportunities in Molecular Simulations.
- Application of Beowulf cluster Fidelity of Molecular Models.
- Bond energies critical for describing many chemical phenomena.
- Accuracy of calculated bond energies increased significantly.
- Long term simulations for phenomena to occur.
- Need potentials to model nano scale processes.
- Large data from experiment need accurate calculations.
- Computation of Bulky nanotechnology data.
- Mass modeling & data analysis.

2.Hardware System Structure

2.1.Architecture Detail

The most visible and discussed aspect of cluster computing systems are their physical components and organization. The two principal subsystems of a Beowulf cluster are its constituent compute nodes and its interconnection network that integrates the nodes into a single system. Design schematic of our hardware system is given in Figure 1.

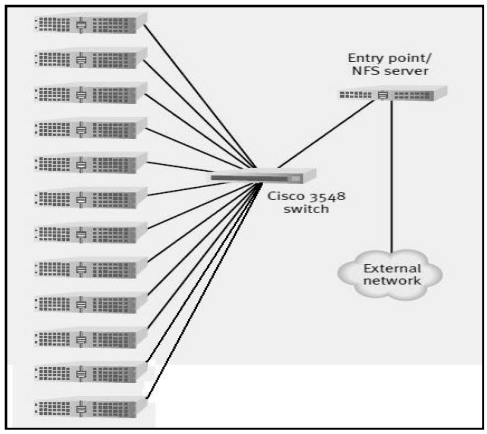

Figure 1: Design schematic of hardware system of Beowulf Cluster of Shri Shankrachariya College of Engineering & Technology, High performance Nanoscience computing Laboratory.

The compute or processing nodes incorporate all hardware devices and mechanisms responsible for program execution, including performing the basic operations, holding the working data, providing persistent storage, and enabling external communications of intermediate results and use command interface. Five key components make up the compute node of a Beowulf cluster: the microprocessor, main memory, the motherboard, secondary storage, and packaging.

- The Microprocessor provides the computing power of the node with its peak performance measured in Mips(millions of instructions per second) and Mflops(millions of floating-point operations per second). While choosing the microprocessor the Mflops is important for the scientists because they use it for complex arithmetic operations. In our project, Pentium(R) Dual-Core CPU E5500 @ 2.80GHz (2048Kb L2 cache) is used as the microprocessor of our Beowulf system, because it has 4 floating-point units(FPU) [10,11]. Although Itaniums have 4 FPU and also they have 64 bits architecture, but their clock frequency is slow. Intel's P4 Pentium(R) Dual-Core microprocessors' peak performance will be 2.80x4=11.2 Gflops.
- Main Memory stores the working data set and programs used by the microprocessors during job execution. We will going to use at least 1GB Double Data Rate (DDR) 266(133x2) Mhz SDRAM, in each node to provide enough space for the microprocessors to work comfortably.

- The Motherboard is the medium of integration that combines all the components of a node into a single operational system. Far more than just a large printed circuit board, the motherboard incorporates a sophisticated chipset almost as complicated as the microprocessor itself. The chipset manages all the interfaces between components and controls the bus protocols. Dual CPU system is going to is used for performance and cost. One of the reasons to buy this board is it will give us chance to change the clock ratio and CPU multiplier if the CPU's are unlocked. It will be very important in our research because we will be testing the system to find out the overheads of the system in different clock rate speeds.
- Secondary storage provides high capacity persistent storage. While memory loses all its contents when the system is powered off, Secondary stage fully retains its data in the power down state. EIDE hard disks are used for this purpose. In design of the system, each node is going to have a raid card to supply enough bandwidth as SCSI and this will be a cheaper solution. Each node have dual 8 MB cached 7200 RPM 80x2=160 MB storage. This is very important when working with large data sets. As an example a 3D matrix with 100000 elements in each dimension will require a huge storage space.
- Packaging for PC's originally was in the form of tower cases. These cases must be cooled well for proper operation of the system. Another important issue is about the electricity system. Line interactive uninterrupted power supply (UPS) is proposed to be used for each node to provide protection and fail free system.

2.1.1.Node Details

Architecture	i686
CPU op-mode(s)	32-bit, 64-bit
CPU MHz	1203.000
Virtualization	VT-x
L1d cache	32K
L1i cache	32K
L2 cache	2048K
Model name	Pentium(R) Dual-Core CPU E5500 @ 2.80GHz
CPU MHz	1203.000
Cache size	2048 KB

CPU cores	2
Clflush size	64
Cache_alignment	64
Address sizes	36 bits physical, 48 bits virtual
Operating System	Fedora 13
Kernel version	2.6.33

Table 1: Architectural details of each node of the cluster

2.1.2.Network Details

Without the ability of moderate cost short hand network technology, Beowulf cluster computing would never have happened. Ethernet was developed as a local area network for interconnecting distributed single user and community computing resources with shared peripherals and file servers. A network is a combination of physical transport and control mechanisms associated with a layered hierarchy of massage encapsulation. The network is characterized primarily in terms of its bandwidth and its latency. Bandwidth is the rate at which the massage bits are transferred usually cited in terms of peak throughput as bits per second. [19] Latency is the length of time required to send the message. Both bandwidth and latency is important [4]. The network switch dedicates running of parallel programs which share data by MPI (Message Passing Interface) or PVM (Parallel Virtual Machines) or any other method. The system will be connected to internet via a firewall on the fast Ethernet network. So the users from all over the world will connect to our supercomputer using SSH telnet connection and line up to run their programs easily with this project.

Type	Fast ethernet (100 Mbps)
Topology	Switched ethernet, single switch
Interconnect	Twisted-pair (RJ45)
Network switch	HP ProCurve 2424M

Table 2: Architectural details of interconnected network for the cluster

3.Performance Overview Of HPC

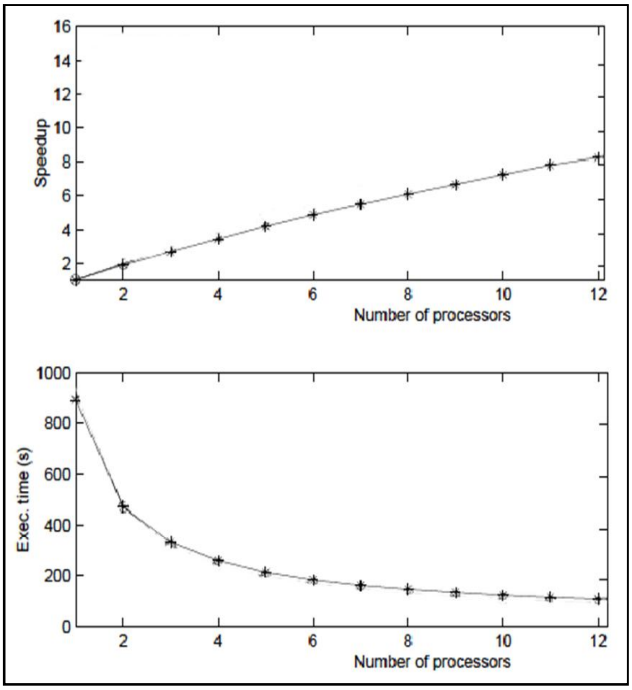

Figure 2 &3: Shows the system performance graph of maximum 12 processors in parallel

We evaluated the performance and scalability of HPC system using Linpack (Linear Algebra Pack) benchmark[11]. The improvement of speed of instruction execution through Linpack and matrix multiplication kernel on single node to 12 nodes is shown in figure 2. It shows our optimization strategies improve performance by about 8 times more powerful than an ordinary core 2 duo processor. The optimized HPC system can achieve 70% efficiency of the peak performance. Figs. 2 and 3 present Linpack performance on a single computing node to over 12 nodes in parallel. When the problem size is large enough to consume 80% of memory space, Linpack achieves 68.17% efficiency of peak performance. Finally, the scalability of Linpack on HPC system is summarized in Fig.2. The first point to break 100 GFLOPS barrier is 7^{th} nodes, where Linpack achieves 128 GFLOPS for the whole system with 12 nodes.

4.Conclusion

We have created a High performance Computing power from scrap pieces of computer parts and a few open source software packages. This will enhance research opportunities

at our educational institutions that have combined their energies to make this cluster an invaluable tool for our research community. The speed of computation & simulation empowered by this cluster will help researchers to perform extensive tasks that were either tedious or even impossible to accomplish on ordinary system. The best example could be illustrated with it is that

the researchers of our institute have successfully crafted a DNA based nano-composite bio-switch [12]. This requires higher computational power & speed which is unsatisfactory from ordinary systems. The HPC has facilitated us to perform simulations up to great detail as required in this case. Otherwise, it would have taken months of constant computation with less precession.

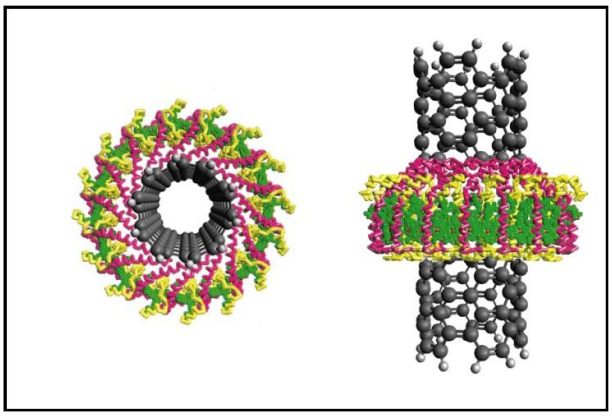

Figure 4: Shows top & side view of DNA based nano-composite bio-switch

This type of communication represents a significant conceptual advance relative to today's fastest processors, which execute only one instruction at a time. The HPC project has been successful & further we wish to start a new project to build up a supercomputer that shall be in the top 500 supercomputer list of the world.

5.Acknowledgment

We thanks Mr. Abhishek Mishra for supporting our project & providing all financial aids. In addition we thank Mr. Mangal, System Administartor, SSCET for helping us in all the field work.

6.Reference

1. Adams, J., The Ohm Project, Calvin College, 2001. Online. Internet. [Aug. 15, 2001]. Available WWW:http://cs.calvin.edu/research/ohm/.
2. Adams, J., Laverell, D., Ryken, M. MBH'99: A Beowulf Cluster Capstone Project, Proceedings of the 14th Annual Midwest Computer Conference, Whitewater, WI, March 2000.
3. Becker, D., Sterling, T., et al. Beowulf: A Parallel Workstation for Scientific Computation, Proceedings, International Conference on Parallel Processing, Oconomowoc, Wisconsin, Aug. 1995, p. 11-14.
4. Becker, D., et al. Scyld Beowulf Cluster Operating System, Scyld Computing Corporation, 2001. Online. Internet. [Aug. 15, 2001]. Available WWW: http://www.scyld.com/.
5. Geist, A. PVM: Parallel Virtual Machine, Oak Ridge National Laboratories, 2001. Internet. [Aug. 15, 2001]. Available WWW:http://www.csm.ornl.gov/pvm/.
6. Grop, W., Lusk, E. The Message Passing Interface (MPI) Standard, Argonne National Laboratories, 2001. Online. Internet. [Aug. 15, 2001]. Available WWW: http://www-unix.mcs.anl.gov/mpi/.
7. Hoffman, F., Hargrove, W. The Stone Soupercomputer, Oak Ridge National Laboratories, 2001. Online. Internet. [Aug. 15, 2001]. Available WWW: http://stonesoup.esd.ornl.gov/.
8. Merkey, P. The Beowulf Project, Scyld Computing Corporation, 2001. Online. Internet. [Aug. 15, 2001]. Available WWW:http://www.beowulf.org/.
9. The Netlib Repository at UTK and ORNL, Oak Ridge National Labs, 2001. Online. Internet. [Aug. 15, 2001]. Available WWW:http://www.netlib.org/linpack/.
10. [10] Warren, M. Loki — Commodity Parallel Processing, Los Alamos National Laboratories, 2001. Online. Internet. [Aug. 15, 2001]. Available WWW: http://loki-www.lanl.gov/.
11. Linpack Source-Code Available WWW: http://www.netlib.org/linpack/
12. Ankush Rai., Bringing DNA Computing With Bio Nano-Compostie Photosensitive material, 2012. Journal of Fundamental & Applied Science, Dec. 10, 2012, p.10-14.