Analyitical Study of HPCC Performance Using HPL

Alpana Rajan[#1], B. K. Joshi[!], Anil Rawat[#], Sarthak Gupta[#]

[#]Computer Division
Raja Ramanna Centre for Advanced Technology
Indore, India
[1]alpana@rrcat.gov.in

[!]Professor
Military College of Telecommunication Engineering
Mhow, India
brijendrajoshi@yahoo.com

Abstract— **Performance study of a High Performance Computing Cluster (HPCC) is a complex and multi dimensional activity involving analysis and evaluation of several processes and parameters. Theoretical estimation of peak performance is generally done using simple calculations involving processor clock speed, FLOP rating of processor and number of processors & cores. This estimation typically provides a quantitative number, which is within acceptable limits of variation as against actual numbers. There are many factors which affect the actual performance of a HPCC.**

In this paper we are presenting results of certain computational experiments carried out on a medium size HPCC. The cluster used for carrying out the experiment is built using six identical server machines interconnected using 1 Gbps Ethernet network switch. To study the performance standard High-Performance Linpack (HPL) Benchmark has been used with certain variation in number of cores and complexity of task. Intel-optimized version of HPL has been used for carrying out the experiments.

Obtained results have been analyzed to understand the behavior of HPC for varying complexity of task. Variation in complexity of task and changing number of processing cores for a particular level of complexity have been studied and presented.

Keywords—HPCC, HPL, Peak Performance, Cluster

I. INTRODUCTION

HPC (High Performance Computing) Clusters are gaining popularity not only for scientific / engineering simulation and modeling applications, but also for commercial applications. Cost effective high throughput solutions are being designed and deployed using HPCCs. Estimation of deliverable performance by HPCC is worked on the basis of theoretical calculations based on numbers for cores, memory size, clock speed etc. For realistic assessment of actual deliverable computing power, various techniques are used. HPL Benchmark is one of the standard benchmark tools for this purpose.

Standard performance of various computing platforms is measured by LINPACK Benchmark. The LINPACK Benchmark was introduced by Jack Dongarra. A dense system of linear equations has been used by LINPACK Benchmark for benchmarking. LINPACK is a very standard yard stick for standalone server machines. With the advent of distributed memory computers, it became imperative to have a more realistic tool for evaluation of the performance of the computing clusters, which lead to introduction of HPL.

II. HIGH PERFORMANCE LINPACK BENCHMARK

HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers developed at Innovative Computing Laboratory, Computer Science Department, University of Tennessee by A. Petitet, R. C. Whaley, J. Dongarra, A. Cleary [1]. HPL is a software package and logical upgraded version of LINPACK for distributed-memory computers. It solves a (random) dense linear system in double precision (64 bit) arithmetic on distributed-memory computing machines [2]. It is portable and freely available implementation of High Performance Computing Linpack Benchmark and it latest version is 2.0.

For execution of HPL on a system, it is essential to have support for Message Passing Interface MPI (1.1 compliant). Availability of Basic Linear Algebra subprograms (BLAS) or the Vector Signal Image Processing Library (VSIPL) is also essential [4]. Performance of the systems depends on many factors, implementation of HPL ensures that parallel efficiency is maintained with respect to the per processor memory usage. The HPL software package solves a linear system of order N, for which the data is distributed over a two dimensional P x Q grid of processes. The resultant peak computing power delivered is measured in Gflops.

There are three major parameters which can be set for HPL execution [2]. The problem size N is defined as a number, which should be chosen so as to ensure largest problem size fitting in available memory. NB represents the block size used by HPL, which is affective for data distribution as well as for computational granularity. Typically a good block size is in the range of [32..256]. Process Grid P x Q is another parameter which can be chosen in HPL. It is suggested to keep P x Q in the ratio of 1:k with k in [1..3], this makes HPL use all the cores available in the cluster.

HPL can be used in variety of ways by varying these major parameters to get results on performance of HPC Clusters. Clusters are being built using various options for the major components, like processor, memory, interconnect etc. In this paper an attempt has been made to analyze performance of a medium size HPC having 24 cores in six processors, interconnected using 1 Gbps Ethernet network switch.

III. TEST BED HPCC CONFIGURATION

For carrying the performance evaluation of a six processor, 24 core HPC Cluster has been used. This test bed cluster has been built using HP make servers, model no. HP PROLIANT DL 180 G6. Table I summarizes the main specifications of the test bed cluster used for carrying out the two experiments for performance evaluation of the test bed computing cluster.

TABLE I. SPECIFICATION OF TEST BED CLUSTER

Cluster Name	**Test Bed**
Number Of Nodes	6 (Processors)
Processor	Intel Xeon E5620 (Quad Core) 2.40 GHz
Processor Architecture	Nehalem
Processor Cache	12 Mbytes
Total Memory	96 GB (16 GB per node)
Memory Speed	DDR3 - 1066 MHz
Network	1 Gbps Ethernet
Operating System	RHEL 5.5
MPI used	Open MPI v1.4
GCC used	4.1.2

Two experiments have been performed using the test bed cluster having specifications as detailed in the above Table 1. These experiments have been designed to test performance by varying number of cores for the same complexity of a task and secondly by varying number of cores (increasing gradually) and simultaneously increasing the complexity of the task. Each run of the HPL is performed using a set of parameters and resultant delivered computing power has been recorded.

IV. PERFORMANCE EVALUATION

Performance of HPC Cluster is defined in terms of peak computing power delivered for a set of parameters [3]. Practically performance of cluster may vary from application to application, since it largely depends on total number of cores used, available memory and its usage by application, type of application like CPU bound / IO bound etc. For an assessment of performance two experiments have been performed and these experiments are described in the following two sub-sections:

A. Experiment No. 1

There are total six processors available in the test bed cluster. In this experiment, number of CPU has increased from one to six for each run of the HPL. The problem size, definable by the parameter N in the HPL is also increased for each run of HPL. This has been chosen to study the HPC performance under different parameter values. Peak computing power delivered has been observed and it is found that for problem complexity defined as N=83000, maximum Gflops have been recorded.

It is also observed that performance dips when N is set as 93000, but picks up again when N is kept at 100000 and all the twenty four cores available among the six processors are used. The process grid defined by P and Q is also varied to ensure that all the available cores are used and the problem is evenly distributed among the computing cores. Table II shown below summarizes the parameter values and results obtained in terms of peak computing power delivered by the HPC Cluster.

TABLE II. PARAMETER VALUES AND OUTPUT OF EXPERIMENT NO. 1

CPU, Cores	**P, Q**	**N**	**Gflops**
1	2, 2	42000	31.74
2	2, 4	58000	40.07
3	2, 6	72000	47.54
4	2, 8	83000	66.57
5	2, 10	93000	40.25
6	2, 12	100000	50.59

Value of NB, which defines the block size has been kept constant as 100 for all the six runs of HPL for the experiment. The results are also depicted in the form of a graph as shown in Figure 1 below:

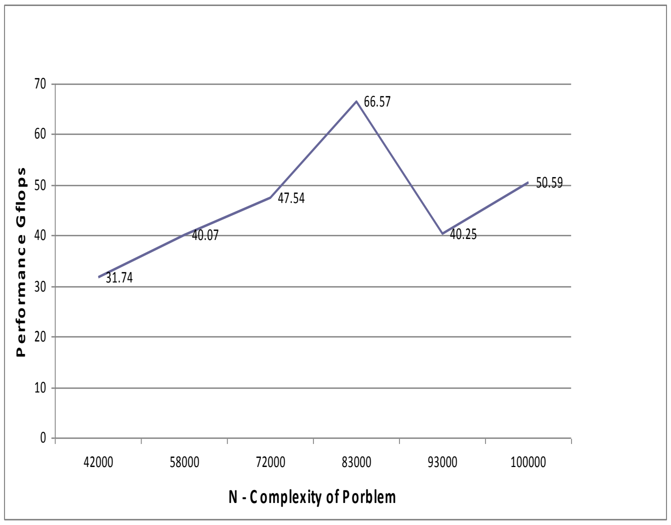

Figure 1. Performance in Gflops vs. Problem Complexity

For a small problem size the overheads are large and hence performance deteriorates. These overheads are due to message passing, local indexing etc. and could be significant. As is seen in this experiment, performance does not increase linearly with increasing problem size supported by increasing number of machines. Increasing number of machines translates into increasing number of cores available for the problem. The dip for problem size 93000 is a typical indication of performance deterioration due to message passing, synchronization etc.

B. *Experiment No. 2*

In the second experiment, problem size was kept constant at 100000 and number of cores was increased from 12 to 24 in steps of 4 for each run of the HPL. The processor grid ratio defined by parameters P & Q were also varied, thus ensuring that for each run the problem spreads over all the cores available in the HPC Cluster.

Table III shown below summarizes the parameter values taken for the experiment and also includes the results obtained for each run in terms of Gflops. Value of NB block size was kept constant at 100 and the problem was distributed evenly over all the cores and available memory of 512 GB.

TABLE III. PARAMETER VALUES AND PERFORMANCE OUTPUT OF EXPERIMENT NO. 2

Nodes	Cores	P & Q	Gflops
6	12	2,6	25.29
6	16	2,8	26.38
6	20	2,10	23.00
6	24	2,12	50.59

It is observed that performance of the test bed HPC Cluster remains in close range when the numbers of cores were taken as 12, 16 and 20. Peak performance delivered for 24 cores almost doubled as compared to that for 20 cores.

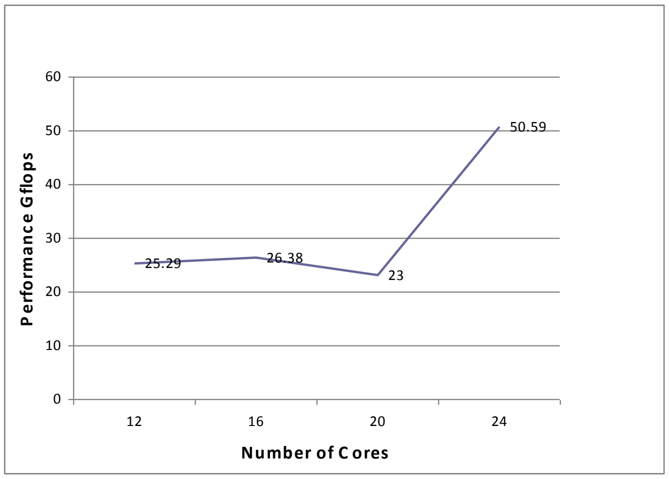

Figure 2. Performance in Gflops vs. Number of Cores

In the Figure 2 results are plotted in the form of a graph and as can be seen in the graph that the performance of the HPC Cluster is within a range of 23 – 25 Gflops when number of cores used are varied from 12 to 20, while other parameters are kept constant. The performance almost doubles when all the 24 cores available in six processors are used.

The behavior of the HPC Cluster in delivering performance clearly indicates that optimization of cores for a particular complexity of task is important while deciding HPC Cluster configuration for any application. Peak performance of HPC does depend on the way the cluster configuration is matched with the problem size and problem complexity.

CONCLUSION

Performance of HPC can be optimized to deliver maximum output for an application depending on complexity of the application and chosen parameters of HPC to engage in executing the application. If an application is to be executed for a large number of times, it is worth trying out certain combinations of parameter so as to optimize the output performance for the application. HPL is only an indicative technique to assess the computing power of an HPCC.

It is also observed that spreading the task evenly among the available cores can deliver maximized output for a set of values. Investigations presented in this paper are a good indication for the relation complexity of task has with number of cores available in the HPC Cluster.

REFERENCES

[1] Dongarra, J., Luszczek, P., Petitet, A. "The LINPACK Benchmark: Past, Present, and Future," *Concurrency: Practice and Experience*, 15, pp. 803-820, 2003.

[2] Langou, J., Dongarra, J. "The Problem with the Linpack Benchmark Matrix Generator," *International Journal of High Performance Computing Applications*, Volume 23, No. 1, pp. 5-14, Spring, 2009.

[3] Cyril Banino, “SIF8064: Miniproject Measuring the Performance of ClustIS with the High Performance Linpack Benchmark”, Department of Computer and Information Science, Norwegian University of Science and Technology, June 2003.

[4] Thander Prnabesh Kanti, Rajan Alpana, Rawat Anil, “Open Source Software Tools for HPC Linux Clusters”, National Conference on Emerging Trends in Computing & Communication (ETCC 2010), Medi-Caps Institute of Technology and Management, Indore, November 19-20, 2010.