ISSUE 19 JUNE 2022

GAṆANAM (गणनम्)

HIGH PERFORMANCE *COMPUTING* NEWSLETTER
INSTITUTE FOR PLASMA RESEARCH, INDIA

INSIDE THIS ISSUE
(3 Pages)

From Testbed to Petascale: A Case Study of Parallel Applications Development

Deepak Aggarwal (SO-E, Computer Division, IPR)
Email: deepakagg@ipr.res.in

Scientific application development using parallel programming which can run simultaneously on multiple cores of a High Performance Computing (HPC) cluster is a difficult task. The nature of complexity due to a large number of hardware and software components in a cluster-based HPC system makes it difficult to write applications/codes which can interact with the system as a single unit and utilize the parallel capabilities to get better performance. There is a lack of expertise in writing efficient code, particularly using parallel programming on such massively parallel systems. Also, there are challenges in moving the codes running on desktop/PC including the legacy codes [1] directly to the HPC system which requires building the code from scratch to utilize the capabilities of the system. The HPC systems are ubiquitous yet intangible to researchers for the development and testing of a scientific application. With the advent of small and highly affordable single-board computers, it is possible to make an HPC like cluster environment [2,3] by connecting several such boards. In an earlier work [4], we have made a 4-node working cluster named Pradyut (meaning "Light") using Raspberry Pi boards to demonstrate HPC like environment. Pradyut with all its components housed in a portable acrylic box is shown in Figure 1. This dedicated system can serve as a low-cost and green testbed (<50W) [5] alternative to an expensive large-scale HPC system like ANTYA which is very expensive and uses high power (>150KW) for operation.

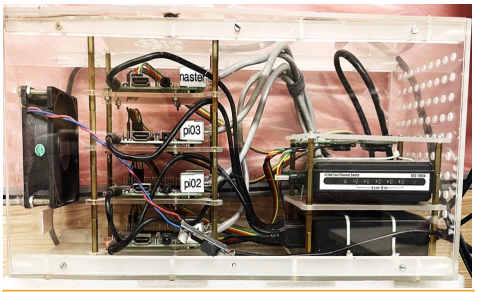


Figure 1: Pradyut housed in a portable acrylic box (30cm x 15cm x 15cm) has been used as a testbed with theoretical performance capacity of less than 15 GFLOPS.

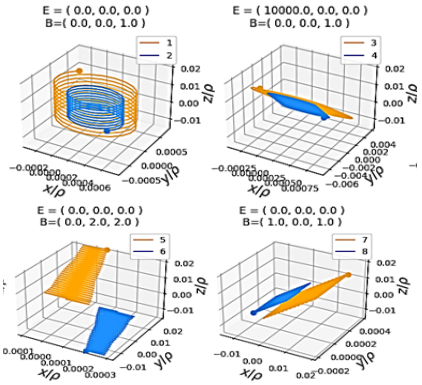


Figure 3: Lorentz_Force application showing the motion of two charged particles in each subplot.

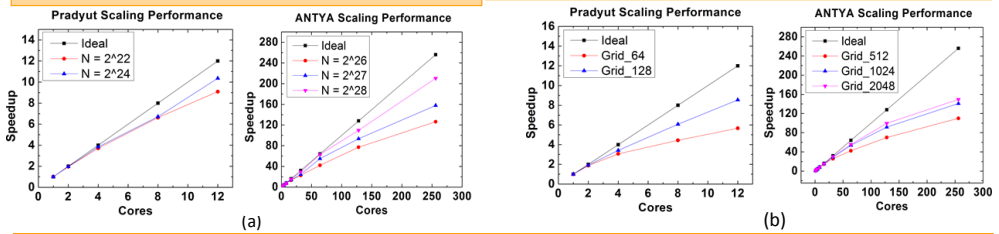


Figure 2: Scaling Performance of (a) Pi_Estimation and (b) Heat_Diffusion applications on Pradyut and ANTYA.

> ***"A low cost testbed cluster can serve as an efficient platform for parallel application development before running on big production petascale clusters"***

In the case study presented here, we have used Pradyut as a testbed for developing and testing applications which demonstrate the well-known behavior of parallel and distributed computing (PDC) systems [2]. With Pradyut being easy to maintain, it allowed us to do all experiments with no restriction while developing the applications. The following 3 applications were developed in Python for PDC demonstration:

- *Pi_Estimation (PE):* This code estimates the value of Pi (π) using Monte Carlo method.
- *Heat_Diffusion (HD)*: This code calculates the transfer of heat from high temperature to low temperature areas in a 2D plane using a 2D heat equation.
- *Lorentz_Force (LF)*: The code demonstrate the motion of independent charged particles in electric and magnetic fields.

We started with legacy Fortran codes which ran only on a single CPU core (serial). These codes were first converted into Python. Even with friendly syntax along with an enormous pool of free high-quality libraries, Python is considered to be inefficient for multi-core architectures and does not perform as fast as lower-level languages like Fortran or C/C++. This case study demonstrates how we have used Python to overcome these challenges to run the converted codes in Python efficiently on multiple cores in both shared and distributed memory architectures. Many Python libraries exist for doing parallel processing with different approaches [6], like Just In Time (JIT) Compilation which is based on numba, cython etc., symmetric multiprocessing which uses multiprocessing, joblib, etc., and distributed computing using mpi4py, dask, etc for cluster like environment. Since a majority of the HPC systems including ANTYA use Message Passing Interface (MPI) standard library for developing codes that can use thousands of cores simultaneously, we have written the Python codes using MPI for Python (mpi4py) library. This means we have the same environment for the execution of codes on both Pradyut and ANTYA. The only difference in the execution of the codes was the use of the open-source batch scheduler SLURM on Pradyut which understandably will not impact the results.

Figure 2 shows the scaling performance of PE and HD applications obtained on Pradyut and ANTYA. Both the codes suggest with increasing workloads the scaling performance improves. The PE problem does not rely on communication between the processes whereas in the HD problem the neighboring values need to be communicated between the processes (calculating derivatives). LF code starts with initializing charged particle parameters and position-velocity vectors and updating them as they are iterated over time along with evolving electric and magnetic fields. Using mpi4py, simulations of multiple particles with different values and in different environment settings (electric and magnetic fields) can be done simultaneously by dividing particles among the multiple processes as shown in Figure 3.

This work was done in collaboration with Arka Bokshi as part of an academic project with Students, Deep Lad and Raj Patel from DDU.

References:

1. *Deepak Aggarwal and Ankita Shingala, 2020 7th International Conference on Computing for Sustainable Global Development (INDIACom), 2020, pp. 102-107, doi: 10.23919/INDIACom49435.2020.9083729.*
2. *Christian Baun, Performance and Energy-Efficiency Aspects of Clusters of Single Board Computers, IJDPS Vol.7, No.2/3/4, July 2016.*
3. *Simon J. Cox et al., Iridis-pi: a low-cost, compact demo. cluster, Cluster Com.,Vol.17, No.2, pp 349-358.*
4. *Deepak Aggarwal et al., "EduPar Posters," 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW), 2019, pp. 347-349, doi: 10.1109/IPDPSW.2019.00065.*
5. *Zhonghong Ou et al., Energy and Cost-Efficiency Analysis of ARM-Based Clusters, 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid), pp 115–123 (2012).*
6. *https://wiki.python.org/moin/ParallelProcessing.*

Running MATLAB Programs on ANTYA

Part-6: Checking MATLAB Licenses Availability Including Toolboxes and Accelerating MATLAB Code with GPU

The last article of the MATLAB series covers how a user can check the availability of MATLAB licenses and run the MATLAB jobs accordingly in ANTYA. In the earlier issue, it was demonstrated that a MATLAB code can be executed on multiple cores faster. In this issue, it will be demonstrated that the performance can be improved by running the compute intensive codes on the powerful GPU cards (P100) available in ANTYA.

"MATLAB licenses including toolboxes along with the users holding the licenses can be checked from the following url: http://licensewatch.ipr.res.in/lwm/"

"The url also provides details of the available toolboxes, total n. of licenses in each toolboxes, user details and historical usage pattern."

"MATLAB module available on ANTYA is GPU compatible and GPU MATLAB codes can be executed on ANTYA gpu nodes."

"GPU Acceleration is optimum if the MATLAB code is vectorized i.e. avoiding the 'for loops' in the code."

How to Implement in ANTYA?

An example MATLAB problem that performs fast convolution on the columns of a matrix has been taken for demonstration. The source code tested on ANTYA has been taken from MATLAB website. The CPU and GPU versions have been executed on a CPU node and GPU node (1 P100 GPU card) respectively.

Fast Convolution on the Columns of a Matrix

CPU Code

```
# CPU code: cpu.m
a = complex(randn(40960,1000),randn
(40960,1000));              % Data input
b = randn(16,1);            % Filter input
c = fastConvolution(a,b);   % Cal. output

% Measure CPU time
ctime = timeit(@()fastConvolution(a,b));
disp(['Execution time on CPU = ',num2str
(ctime)]);
```

```
fastConvolution.m
function y = fastConvolution
(data,filter)
m = size(data,1);
% Zero-pad filter to the length of data, and
transform
filter_f = fft(filter,m);
% Transform each column of the input
af = fft(data);
% Multiply each column by filter and compute
inverse transform
y = ifft(bsxfun(@times,af,filter_f));
end
```

GPU Code

```
# GPU Code: gpu.m
a = complex(randn(40960,1000),randn
(40960,1000));          % Data input
b = randn(16,1);        % Filter input
ga = gpuArray(a);       % Move data to GPU
gb = gpuArray(b);       % Move filter to GPU
gc = fastConvolution(ga, gb);%Cal. On GPU

% Measure GPU time
gtime = gputimeit(@()fastConvolution
(ga,gb));
disp(['Execution time on GPU =
',num2str(gtime)]);
```

```
fastConvolution.m
function y = fastConvolution
(data,filter)
m = size(data,1);
% Zero-pad filter to the length of data, and
transform
filter_f = fft(filter,m);
% Transform each column of the input
af = fft(data);
% Multiply each column by filter and compute
inverse transform
y = ifft(bsxfun(@times,af,filter_f));
end
```

Vectorized code used upto 20 cores

On 1 GPU card Vectorized code used 5GB RAM

ANTYA UPDATES AND NEWS

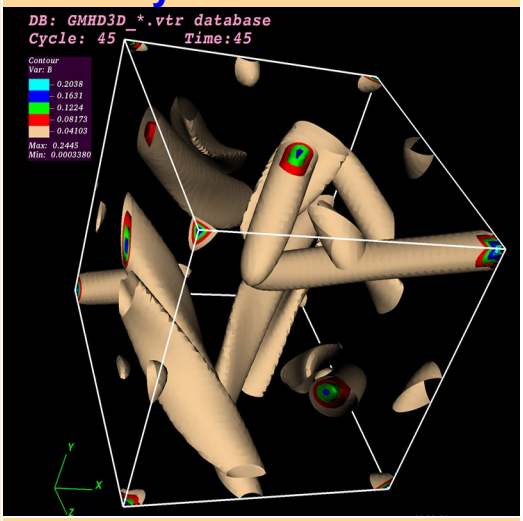
1. New Packages/Applications Installed

⇒ **The no. of Applications stays the same for this issue.**

⇒ **ANTYA Users /home data monthly backed up cycle completed.**

HPC PICTURE OF THE MONTH

Generation of Magnetic Flux Cigar for an Induction Dynamo Model



Pic Credit: **Shishir Biswas**

The figure above shows the special kind of structure which is known as cigar like magnetic field iso-surface structure for an induction dynamo model, in a three dimensional magnetohydrodynamic plasma. It is generated using In-house developed multi-node multi-card weakly compressible magnetohydrodynamic GPU based solver GMHD3D.

For this simulation 256^3 grid resolution was taken. The full simulation took 63.526 hours in 4xP100 GPU cards (2 Nodes) on ANTYA. For generating this 3D Iso-surface visualization, open source visualization package VisIt 3.1.2 installed in Antya Visualization node have been used.

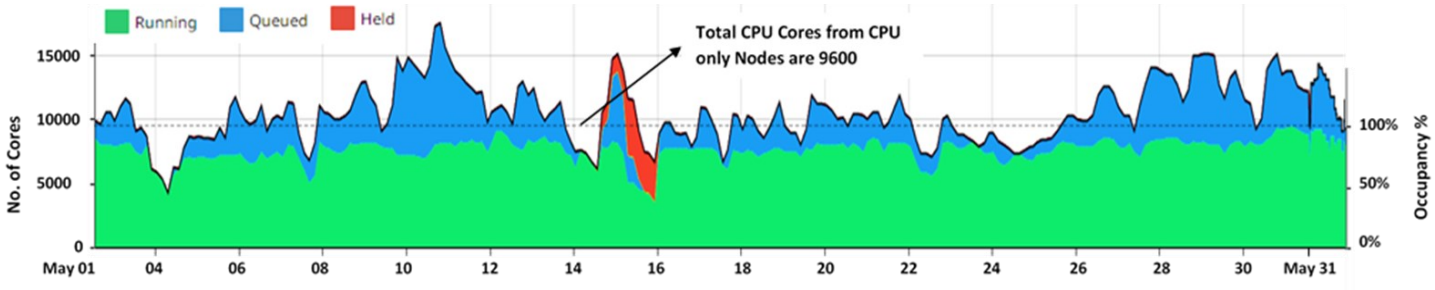
TIP OF THE MONTH

ANTYA has several versions of OpenFOAM, open source code for CFD simulation with parallel capabilities. The available versions can be checked with the command:

```
[user@login1 ~]$ module avail OpenF
OpenFoam  OpenFOAM-2.1.1  OpenFOAM-plus  Open-
FOAM-v2106  OpenFoam7.0  OpenFoam8.0
```

ANTYA Utilization and User Job Performance Ratio: May 2022

ANTYA Daily Observed Workload

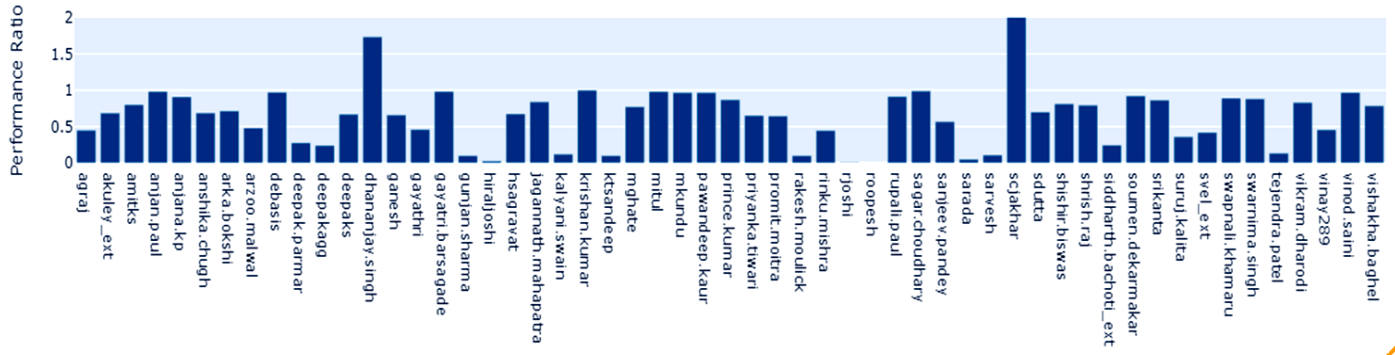


Users' Jobs Performance Ratio

The performance ratio (PR) is a quantitative measure of how well the allocated resources are being utilized. Ideally, PR should be 1. For example, if a user has requested 100 cores for a job and all the 100 cores are being utilized throughout the execution, then the performance ratio for this job/user would be 100/100 which is 1. Here the performance ratio is averaged over jobs submitted in a month by a user.

PR < 1: Job Underperforming (using less cores than requested/allocated). **Users with 0.1<PR<0.8, should check their jobs regularly and improve the resource allocation as per the PR values of their jobs which can be self calculated as explained above.**

PR > 1: Job doing implicit multithreading



Other Recent Work on HPC (Available in IPR Library)

प्लाज़्मा प्रयोगों के लिए अतिचालक चुम्बकों का विकास: अद्यतन और योजना	SWATI ROY
Re-entrant phase separation in a collection of self -propelled non-reciprocally aligning disks	SOUMEN DE KARMAKAR
Design of New Target Handling System, Extension Chamber and Modifications to the Existing Target Handling System of High Heat Flux Test Facility	RAJAMANNAR SWAMY KIDAMBI

Acknowledgement

The HPC Team, Computer Division IPR, would like to thank all Contributors for the current issue of *GANANAM*.

ANTYA HPC USERS' STATISTICS—

MAY

- **Total Successful Jobs — 4367**
- **Top Users (Cumulative Resources):**

• CPU Cores	Amit Singh
• GPU Cards	Suruj Kalita
• Walltime	Gayathri
• Jobs	Arzoo Malwal

On Demand Online Tutorial Session on HPC Environment for New Users Available
Please send your request to ***hpcteam@ipr.res.in.***

Join the HPC Users Community
hpcusers@ipr.res.in
If you wish to contribute an article in GANANAM, please write to us.

Contact us
HPC Team
Computer Division, IPR
Email: ***hpcteam@ipr.res.in***