Performance Comparison of Super Cheap Computing Cluster and Single Dual-Core Processor on Numerical Problem

Aye Aye Nyein
University of Computer Studeis, Hinthada
ayeayenyein2008@gmail.com

San San Aye
University of Computer Studeis, Kalay
sanaye.12.29@gmail.com

Tin Nwet Oo
University of Computer Studeis, Pakokku
Tinnweoo.mw@gmail.com

Abstract

Nowadays learning cluster computing and its applications are critical in all computing area. High computing can be gathered by using cluster and high speed processor.Almost all parallel programs that run on supercomputers are created using Message Passing Interface (MPI) library and also use on Raspberry Pi (RPi) cluster. .In this research, we constructed ten nodes forty cores RPi cluster and to analyze the performance of cluster, we have executed the numerical Pi value estimation problem on super-cheap computing cluster and as comparison, ran this computation on dual-core processor.The interesting results running on dual-core processors was as the number of random points increases, running time also increases and moreaccurate Pi valuewe got. According to the test result, the executing time ofRPI cluster was slightly slower than dual-core processor and dual core processor was approximately twice faster thanRPI cluster.

Keywords: Monte Carlo Method, Pi (π), Raspberry Pi, Super Cheap computing cluster, Dual-Core processor

1. Introduction

A cluster can be thought of as a set of interconnected computers . By connecting large number of computers in a cluster, one can gain potential increase in performance by doing operations in parallel and distributed environment.[3]. Message Passing Interface (MPI) allows us to choose how many processors in the super cheap computing cluster to use for each experiment.In this research, we analyze the computational power of ten nodes forty cores Raspberry Pi cluster, weexperiment the Pi (π) value estimation program on it. The program was tested with 10000 trails for one through forty cores and the time of computation was recorded. The test we ran on dual-core processor was similar to those we ran on the cluster. The main difference on dual-core processor was since this CPU has four cores, we tested with one through four processes as opposed to one through forty cores on cluster and we ran the program with various random points from 10^1 to 10^8.In this research,we present the test is run on forty-core clusters and dual- core i5 processor, the analysis also focuses on comparison of executing time of the RPi cluster and dual-core i5 processor

The remaining part of the paper is ordered in sections as: Section 2 will describe related research works, Section 3 describes theoreticalbackground. Section 4 describes design and configuration of the proposed system. Section 5 covers implementation and experimental results. Finally, section 6 we conclude the paper and followed by references.

2. Related Research Work

The researchers from the Department of Computer Science, Augsburg College presented a cluster based on eight Raspberry Pi 3 Model B modules [5]. The realized cluster was tested using the Monte Carlo method for calculating the value of Pi (π). Calculation was performed on eight nodes with one to four processes per node. This paper [4] presented two different types of clusters and both have almost same configuration. One cluster is constructing with two Dell computers and another cluster is constructed with two HP computers. The researchers test the two distinct parallel programs on the clusters for the Monte Carlo Pi (π) estimation and quick sort with different problem sizes and compared the

results that which cluster executes the algorithm in minimal time. In this paper [1], the authors reported the estimation of the value of Pi (π) and solved linear equations using Open MPI to improve performance by reducing execution time and also presented the execution time of both serial and parallel algorithm for computation of Pi (π) value and for the solution of system of linear equations. In this paper [2], the authors present the dual-core Window-based platform to study the effect of parallel processes number and also the number of cores on the performance of three MPI parallel implementations for some sorting algorithms. As a comparison the effect of parallel processes number and also the number of cores on the performance of parallel bubble sort, parallel merge sort and parallel quick sort algorithms and the execution time of the three studied parallel algorithm has been reported. In this research [3], the researchers describe the importance and uses of Cluster Computing in the IT industry and as a subjected research area, the Pi (π) value estimated tills the first 15 places after the decimal point In this research, we analyze the computational power of ten Raspberry Pi cluster and a Dual-Core i5 processor using Pi (π) value estimation experiments and as a comparison which execution time was the fastest..

3. Theoretical Background

3.1. Monte Carlo Method

Monte Carlo methods are vital set of algorithms in computer science. They involve estimating results by statistically sampling a parameter space with thousands to millions of experiments. The algorithm requires a small set of parameters as input, with which it generates a large amount of computation and outputs a concise set of aggregated results [6]. Monte Carlo Pi(π) value estimation program are make use of statistical simulation to mathematically model. It estimates the probability of obtaining a successful outcome. [6]. Pi (π) is an irrational mathematical constant describing the ratio of a circle's circumference to its diameter. We proceed to enclose the circle in the tightest square possible, where one side of the square is equal to the diameter of the circle. Using the formula for the area of a circle of radius=1, Ac = πr2 = π and the area of the square AS = 4r2 =4. We approximate Pi (π) by calculating the ratio of the generated random points in the circle and the total number of generated random points in the square is equal to the ratio of the two areas, π/4. Multiply this ratio by 4 to estimate Pi (π) value. The steps for estimation of the value of Pi (π) are as follows [7]:

1. The numbers of random points generated inside the unit square (N).

2. Whether the generated random point falls inside or outside the circle enclosed the square and documents the result.

3. Repeat steps 1 and 2 Q times.

4. Number ofrandom points fall in the circle enclosed in the unit square (n).

5. The value of Pi (π) can be calculated as:

$$\pi/4 = n/N \quad (1)$$

$$\pi = 4* \ n/N \quad (2)$$

Where in this experiment N is the number of generated random points and n is the number of points fall inside the circle. The more generated random points the more accurate estimated Pi (π) value we get.

3.2. Parallel and Distributed Computing

Distributed computing system is combination of hardware and software components that are located on different networked computers and communicate and coordinate actions among each other through passing messages. Each node in a distributed system consists of a memory and a processor component. The distribution and managing of data/tasks across the system are performed by master and slave nodes execute the tasks assigned to them. To solve the computational problem efficiently in a fault-tolerant manner by utilizing distributed system on all available memory and the processing power of various nodes [7]. Raspberry Pi

brought multicore computing to the eager hands of many financially challenged institutions and the implementation offree Message Passing Interface (MPI) libraries package software require for the Raspbian operating system allowed for the proliferation of computing research on a smaller budget. MPICH2 is a portable implementation of MPI (Message Passing Interface) and permit processes to correspond with each other by transmitting and getting messages between them. It is usually utilized for creating parallel programs and executing on cluster computers and supercomputers. Process can communicate with each other by sending and receiving messages and support the shared-memory architectures. This implies that several processes can read or write to the similar memory location. MPI is specifically designed to run on multiple machines and passes messages between machines in an efficient manner to parallelize the task to 2, 3, and 4 subtasks, with each subtask being executed on a separate processor in a single node. Total ten RPi boards are used to build super-cheap cluster for this research. One single board processor consists of 4 cores. So, our cluster has 40 cores to compute any parallel execution In this analyze, the whole clusters (forty cores) were used for Raspberry Pi cluster and four cores are used for Dual-Core processor for to compare the performance of these.

4. Design and Configuration of the Proposed System

. We constructed a super cheap computing cluster using ten Raspberry Pi 3 Model B and one RPi is the master node, responsible for distributing jobs and resources including itself the rest nine RPi's are simply the slave nodes obeying in the master node instructions. One single node processor consists of 4 cores. So, this cluster has 40 cores to compute any parallel execution.In the following figure (Figure 1) illustrates the Pi value estimation program design.

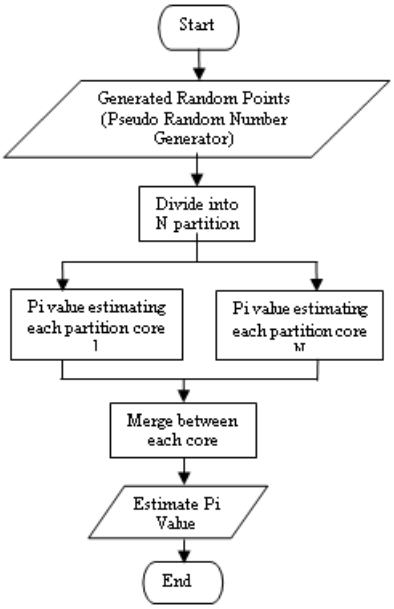

Figure 1.System Flow Diagram for Pi Value estimation Program

The appearance of our proposed system architecture for UCS Monywa is as shown in Figure 2.

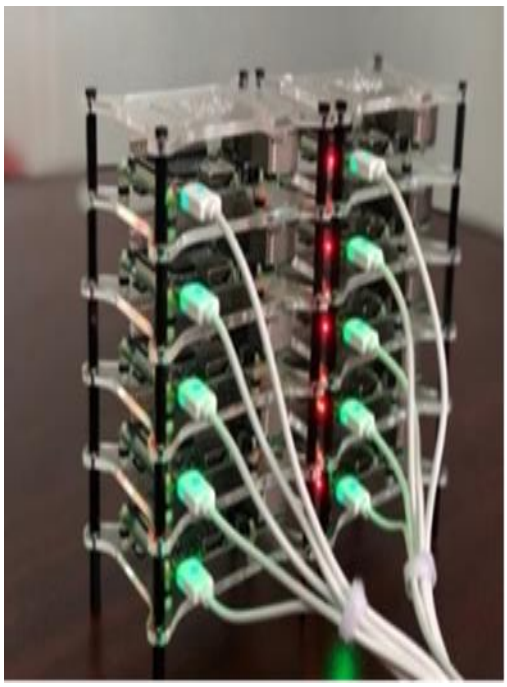

Figure 2. Appearance of Super-Cheap Computing Raspberry Pi Cluster

The requirements and the total costs for building the cluster as seen in Table 1.

Table 1. Requirement and Total Cost for Construction of Raspberry Pi Cluster

	Item	Price	Qty	Cost
1	Raspberry Pi 3, Model B, RAM (1GB)	$39.6	10	$396
2	D-link 24 port switch	$128	1	$128
3	60W USB 16-port fast charger	$35.5	1	$ 71
4	8G SD card	$7.00	10	$ 70
5	A Male to micro-B 1.5m network cable(5Pack)	$7.99	2	$ 15.98
6	RPI Cluster Frame	$7.5	2	$ 15
Total Cost (in US$)				$695.98

5. Implementation and Experimental Results

To obtain the results as well as to test the performance of super cheap computing cluster,we implemented the Pi (π) value estimation programson the cluster. To test the efficiency of the cluster, we ran the MPI scripts with one through four processes per node. Our tests used 100000 random points to estimated Pi (π) value and to obtain more accurate results we did 10000 trials used [3]. The running time to execute the Pi (π) takes 10.86 seconds and the estimated Pi (π) value is 3.141511016000000 on forty cores cluster as seen in Figure 3.

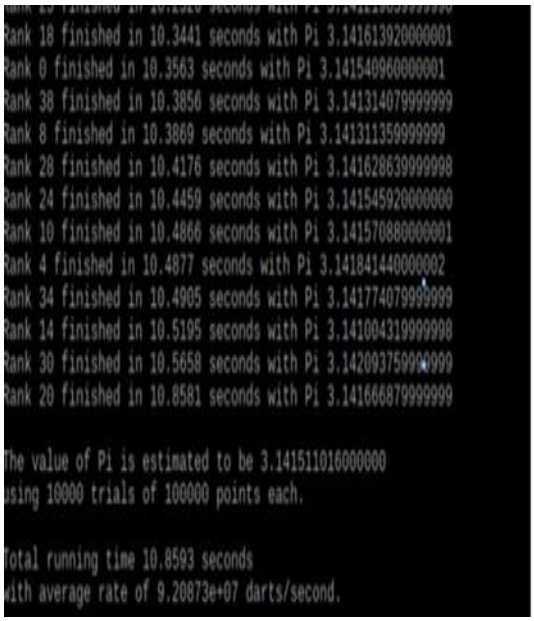

Figure 3. Evaluation of Running Time and Pi *(π)* value for Forty CoresRPiCluster

The test we ran on the dual-core processor was similar to those we ran on the cluster. The computer we used this research was Intel (R) Core(TM) i5-5200U @ 2.20GHz processor with 4GB of RAM, the running time was 5.76 seconds faster than thecluster and the estimated Pi value3.143243000000000 using on same data sets . Then we also experiment using various data sets on dual core processor with generated random points from 10^1 to 10^8 and record the running time. As can be seen in graphical representation (Figure.4), we conclude that the number of generated random point increases, the running time also increases and the more accurate Pi (π) value we get.Figure 5 shows the comparison of execution time of super cheap computing cluster and Dell dual-core i5 processor after compiling the Pi (π) using 100000 generating random points as input [4].

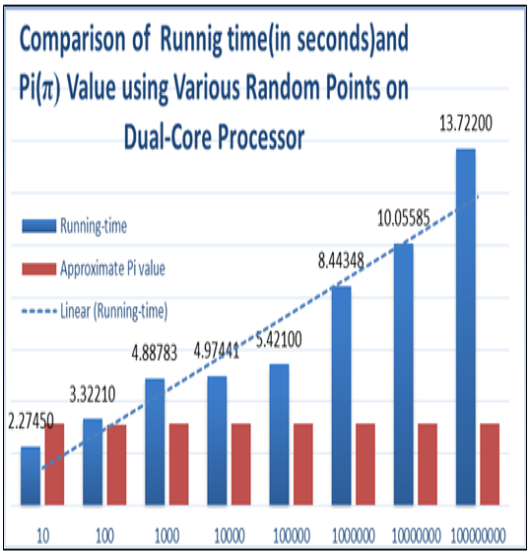

Figure 4.Estimated Pi *(π)* Value for Different Data Sets on Single Dual-Core Processor

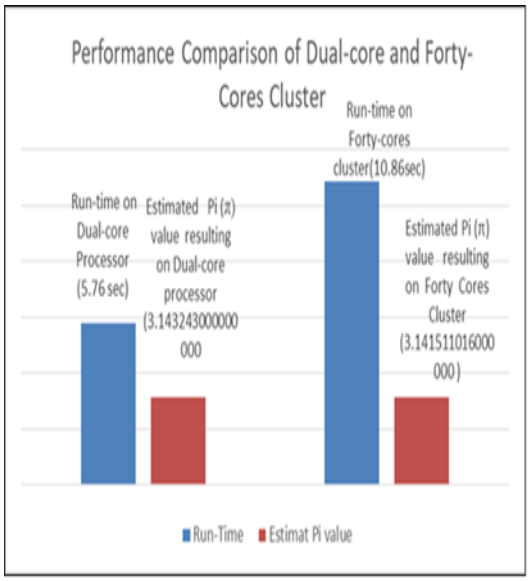

Figure 5. Performance Comparison of Dual-Core Processor and Forty-Core Cluster

6. Conclusion

In this research, we study cluster computing and its applications. We successfully constructed and experiment a ten node, forty cores super cheap computing cluster using Raspberry Pi 3 model B. We implemented the forms of parallel processing using MPI with one through four processes per nodes. To analyze the performance of the cluster we have executed numerical parallel program on the clusters for the estimation of value of Pi (π) using Monte Carlo method. As a performance comparison, we ran this program on Dell dual-core i5 processor. The execution timeon the cluster was slower than dual core processor. The running time on forty cores Raspberry Pi cluster was only 53% slower than the dual core i5 processor. Based on our study, the dual core i5 processor is approximately twice faster than the forty core Raspberry Pi cluster. The cluster we built in this research was only had ten nodes, if we were to further research we built the cluster adding more nodes it would be interesting to see how adding more nodes would affect the running time of cluster and we could create more computationally heavy algorithms to test on the cluster and compare the performance of the cluster and dual-core processor.

References

[1] S. K. Sharma, "Performance Analysis of Parallel Algorithms on Multi-core System using Open MP", International Journal of Computer Science, Engineering and Information Technology (IJCSEIT), Vol.2, No.5, October 2012 ,pp.55-64

[2] A. I. El-Nashar "Parallel Performance of MPI Sorting Algorithms on Dual–Core Processor Windows-Based Systems ", International Journal of Distributed and Parallel System (IJDPS) VOL.2, NO.3, MAY 2011, PP.1-14

[3] S.S.Aye, M.L.Win, M.H.Zaw,, "Design and Implementation of Super Cheap Computing Cluster using Raspberry Pi for High Performance Computing ", SJIR Vol1,Issue1 ,August 2019,pp 171-175.

[4] M.Nazir, D.Kumar, L.A.Thebo" Critical Analysis of High Performance Computing (HPC)" International Journal of Computer Science and Information Security (IJCSIS), Vol. 16, No. 9, September 2018 , pp(153-163).

[5]G.Dorr,D.Hagen.B.Laskowski,Dr.E.Steinmetz,D.Vo, "Introduction to Parallel Processing with Eight Node Raspberry Pi Cluster", The Department of Computer Science Augsburg College

[6] https://www.researchgate.net/publication/ 228865283J. Chong, E.Gona, K. Keutzer, "Monte Carlo methods: A computational pattern for our pattern language", January 2010.

[7] https://digitalcommons.mtech.edu/grad_rsch