ISSN 2278-3091
Volume 10, No.2, March - April 2021
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse961022021.pdf
https://doi.org/10.30534/ijatcse/2021/961022021

Effects of Combining Multi-Core Processors in Creating Super-Computers Using Raspberry Pis

Kenneth Oliver S. Lopez[1]
[1]Computer Engineering Department, College of Engineering and Architecture, Pangasinan State University, Urdaneta City, Pangasinan 2428 Philippines, klopez@psu.edu.ph

ABSTRACT

The vast resources of old working computers are getting bigger and bigger. Companies that consider these old computers as obsolete but still working put these computers to the storage, or worse, junk. Different entities are more and more relying on supercomputers to replace their ordinary and old computers. Supercomputers have become a basic necessity in this generation. It has a wide variety of functions that make our daily work easier. Having a well-built supercomputer with excellent specifications is a must; however, it comes with a cost. The computers can be clustered to create a supercomputer. The computers used in these clustered supercomputers have the exact specifications, such as uniform models with the same number of cores per processor and the same memory capacity. This study focuses on creating a supercomputer using different processors with a different number of cores per processor, processor speed, and different memory capacity. This study can be a valuable system for entities that have old computers in their workplace. They can reuse their old computers and combine them with other computers to create a supercomputer, thus reducing waste and recycling old resources. This study will be using different types of Raspberry Pi (RPi) computers combining old and new models to simulate the scenario of combining multi-core processors in creating supercomputers.

Key words: combining multi-core processors, Raspberry pi cluster, supercomputer.

1. INTRODUCTION

Computers have become an integral part of our society. Anyone could go anywhere they want, and computers will always be present. It is shown in virtually every aspect of our modern practice. It is something we use in our houses, schools, and offices. As a result, owning a computer has now become the standard. The advancement in computer technologies has been proliferating over the years. Anyone can buy the latest model today, and it will be considered an old model a month later. Therefore companies have constantly been upgrading their computer from time to time, in order to enjoy the added features and to be able to say that their company is using the best equipment possible.
In some government-owned agencies, that is not the case. The computers were working on different and computing-intensive tasks. Some are very slow and inefficient because they are still using technologies that were built a decade ago. The proposed prototype will help solve the problems of slow computers that have been encountered in different government agencies. This study combined different Raspberry Pis to work as a single unit. Furthermore, the output will be faster in computing speed and can handle the intensive task given by its node cluster controller. It will also open up an avenue that combining computers that have different speeds and memory is possible.

2. RELATED LITERATURE AND STUDIES

2.1 Raspberry Pi

Raspberry pi was launched the year 2012, and it popularised the use of SBCs. The ability to create a compute cluster for approximately the cost of workstation has meant that using and evaluating these micro-data centers is now within reach of student projects. Edge can be used to reduce the amount of bandwidth needed for data transfer. When deploying edge compute facilities in remote locations, the only available power supplies are batteries and renewable sources such as solar energy. Power efficiency in terms of GFLOPs/W is a critical consideration. The two most recent Raspberry Pi releases have not been evaluated previously to determine their power efficiency. The location of an edge compute infrastructure may mean that maintenance or repairs are not practical. In such cases, the ability to over-provision the compute resources means spare S BCs can be installed and only powered when needed. The energy efficiency of an SBC cluster is essential when investigating the use of SBCs in next-generation data centers. The small size and ruggedness of these clusters enable the creation of portable clusters. Iridis-Pi, Pi Cloud, and the Mythic Beast cluster are designed for the education use case [1].
Raspberry Pi's has been used in a wide array of applications, from simple single-board computer system to interactive

home automation system through email. The algorithm for the same has been developed in a python environment. Results show the efficient implementation of the proposed algorithm for home automation [2]. The wide application of the Raspberry pi is so robust that it has been shown effective and efficient in creating a scale model for cloud computing infrastructures. The majority of Cloud computing research is conducted in small machine emulation or testbed environments. Construction of a miniature Cloud DC has become more affordable thanks to the Raspberry Pi, a low-cost, low-power single-board device. From resource virtualization to network behavior, the PiCloud simulates any layer of a Cloud stack. [3].

2.2 Raspberry Pi Clustered Supercomputers and Parallel Programming

Cluster computing research and technologies are important for the future of computer science. Modern supercomputers are based around the concept of connecting a group of computers to provide more computing power than a single computer would provide alone. Using 5 Raspberry Pis, researchers from Texas Woman's University's Quality Enhancement project built a low-cost cluster computer. The data gathered from the cluster computer's tests were equivalent to those obtained from a single Raspberry Pi. In the paper, the findings of such studies are discussed [4].

Parallel programming is a computer paradigm in which computations are performed concurrently on several processors. A low-cost cluster was created to demonstrate the technical capability of parallel programming. This cluster is made up of computer using 20 Raspberry Pi 3s [5]. This study however will only focus on eight (8) Raspberry Pis but of different types and models.

Vishnu Govindaraj states that parallel applications may be one of the two forms. The first is asymmetric multiprocessing, which assigns separate tasks to each processor. Through sending and allocating tasks, one master processor can manage all worker processors. The second type of multiprocessing is symmetric multiprocessing, in which physical memory is shared among the processors. Worker model with no supervisor, both processors behave in a peer-to-peer manner [6].

Node architectures based on System-On-a-Chip technologies are used in a novel massively parallel supercomputer of hundreds of teraOPS. The architecture has outstanding numerical efficiency and could be used to calculate new classes of parallel algorithms. Specific node packaging strategies that use midplane and other hardware devices allow the supercomputer to be partitioned across several networks in order to optimize supercomputing capabilities [7].

Raspberry Pi 4 is the latest model that the Raspberry Pi has. It is a Single Board Computer (SBC) with a 2,4 or 8GB RAM variation. It has two (2) USB 3 ports and two(2) USB 2 ports, four USB ports. It is equipped with a Gigabit Ethernet port, two (2) micro HDMI ports that support 2x4K displays. Powered by a Broadcom BCM2711, Quad-Core Cortex -A72 @ 1.5Ghz, this computer needs a 5V 3A Type C USB power supply unit. Raspberry Pi 3B and 3B+, on the other hand, have slightly lower specifications. [8].

In 2015, researchers measured the importance of results, FLOPS, Processor time, and score using Raspberry Pi devices that run with the model cluster. The received FLOPS value was then converted to the load held by the cluster computing Raspberry Pi. The i5 and i7 CPU architectures are both being studied in the same way. To analyze the processor and memory allocation, the researchers used himeno98 and himeno16Large. The evaluation is conducted on a 1000x1000 matrix before being benchmarked using OpenMP. The focus of the study is on CPU Time in FLOPS and each design ranking. The result reveals that the Raspberry Cluster Architecture has a Processor Time of 2576.07 seconds, an MLPOS of 86.96, and a score of 2.69. On the Core i5 architecture, the result is 55.57 seconds of CPU time, 76.30 MLOPS, and a performance of 0.92. In the Core i7 architecture, the result is 59.56 seconds of CPU time, 1427.61 MLOPS, and 17.23. The influence of architecture models on the computation process can be seen in the cluster and multicore architecture performance. The contrast revealed that the processor power supply architecture has a significant effect on processing performance, with i5 and i7 performance improving. According to the study, all cluster and core i5 and i7 versions will process data to completion. [9]

Kathiravan Srinivasan and colleagues used Apache Hadoop and a Raspberry Pi cluster to compute big data created by feature point extraction on an image in 2018. As seen in the analysis, the MapReduce procedure in Hadoop and the main features of the Raspberry Pi 3 help the experiment more effectively than a single high-processing device. When dealing with a smaller dataset, though, the same cannot be said. Hadoop allows researchers to scale their setup to handle much larger datasets quickly. Hadoop's HDFS storage framework provides the researchers with improved protection and fault tolerance when opposed to single-host servers. HDFS divides data into several blocks distributed through many nodes, making it difficult to read the contents of data from a single node and thereby improve data protection. [10]

2.3 Message Passing Interface (MPI)

The message passing interface (MPI) is a structured way of sharing messages between several computers running a parallel program in distributed memory. Different processors, or sometimes multiple processor cores within a single computer, are referred to as parallel computation nodes. In a parallel arrangement, each node usually operates on a subset of the overall computation problem. The task then becomes synchronizing each parallel node's behavior, exchanging data between nodes, and commanding and controlling the whole parallel cluster. The message passing interface describes a basic set of functions for these tasks. [11]

Before the word "translational research" was coined, the MPICH project was an example of computer science translational research. The project started in 1992, intending to create a lightweight, high-performance implementation of the Message-Passing Interface (MPI) Standard. It has paved the way for MPI to be widely adopted as a means of writing scalable parallel programs on systems of all sizes, including the upcoming exascale supercomputers. [12]

Mpi4py-fft is an open-source Python program for computing (in parallel) FFTs with multidimensional arrays that can be very large and distributed. A multidimensional FFT is calculated in order, one axis at a time, overall axes. ultidimensional arrays must be distributed around some, but not all, of their axes to fit in the memory of several processors. As a result, parallel FFTs are computed using sequential (linear) transformations over undivided axes and global array redistributions (using interprocess communication) that realign the arrays for further serial transforms. [13][14]

MPI for Python was developed on top of the MPI-1 specification, which specified an object-oriented interface that closely matched the MPI-2 bindings and facilitated communications of general Python objects in its first version. This kit has been updated to support nearly all MPI-2 functionality as well as direct blocking/non-blocking communication with numeric arrays. A Beowulf class cluster was used to assess improvements in communication efficiency. In addition to compiled C code, the results revealed a negligible overhead. [15]

Rather than raising the number of transistors in a computer, we are now heading toward parallelism. Clusters operate in the same way as today's supercomputers. The other nodes are slave nodes, except for the master node, which communicates with the user. The load is distributed evenly among all nodes, who then send their results to the master. The results are combined by the master, who then displays the final output to the customer. A network over Ethernet is used to communicate between nodes. MPI4py is a Python-based Message Passing Interface (MPI) and MPICH implementation. [16]

Figure 1 describes the framework of the study. The process starts with the collection of Input parameters: identifying what raspberry pi computers will be integrated as a cluster, programming language to create software for the cluster, and interconnecting the raspberry pi to create a single cluster. A Process is a direct approach to integrating all the inputs and observing the cluster's performance concerning the different number of slaves connected in the cluster. The number of cluster members is gradually increased to see what the immediate effects of every element being added are. Until such time that all members of the cluster have been integrated, only then we could have the result of the effects of combining multi-core processors in creating super-computers using raspberry pis.

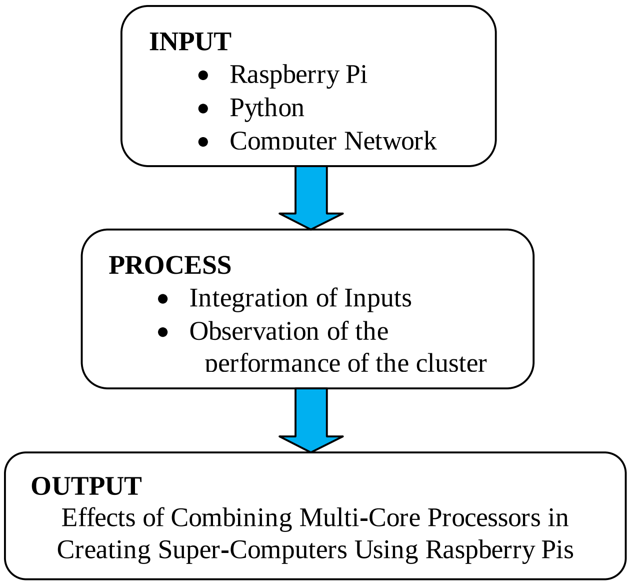

INPUT
Raspberry Pi
Python
Computer Network
PROCESS
Integration of Inputs
Observation of the performance of the cluster
OUTPUT
Effects of Combining Multi-Core Processors in Creating Super-Computers Using Raspberry Pis

Figure 1. Conceptual Framework

3. MATERIALS AND METHODS

3.1 Research Design

The research is a purely quantitative approach combining developmental and experimental methods. The combination of a different number of processor cores and different processing speed of computers and their effect on their performance is the main objective of this research. In the form of raspberry pi, the number of computers is gradually increased as the researcher observe the effect of having multi-core processors in creating supercomputers using raspberry pis. Furthermore, the raspberry pis are connected through a local area network using a simple switch.

3.2 Equipment Used

Eight units were used in order to form the supercomputer. The researcher used different models of Raspberry Pis a 3B [17] model, three units of 3B+ [18] model, and a model 4 with 4GB [19] and three units of 2 GB [19] of memory of the same model 4, a total of eight units. The Raspberry Pis are also eight units of 16GB [20] micro sd cards that will hold its operating system. Raspberry pi 4 needs around 3A, while older models use 2.5A [21]; with this, the project will need at least 24A with a 5Vdc power supply. No power supply supports the specifications; hence the researcher converted an Advanced Technology eXtended (ATX) power supply from a desktop computer to a bench supply that will suffice the cluster's needs [22]. A 24 port TP-Link, TLSF1024 switch [23] is used to interconnect all Raspberry pis using a simple Cat 5 UTP cable crimped using TIA-568B [24] [25] straight-through configuration. Furthermore, four generic 80mm 5vdc case fan was used as a cooling system. Lastly, a generic 2U rack hard case with 19inch x 17inch x 3.5inch dimension was used to house all the components and equipment.

3.3 Preparation of the Raspberry Pi Supercomputer

The first step to prepare the Raspberry pi is to format all the micro-sd cards appropriately. Raspbian Jessie [26] [27] operating system is installed in all of the cards using a third party software called Balena [28] [29]. There exist a technique that formatting and installing the operating system in a micro-sd card and then clone it to other cards will save time [30][31][32], however, in this case, the cards were formatted one by one because the raspberry pi being used are different models.

The next step is to boot every raspberry pi to configure its Ethernet port assigning a static IP address so it could then be accessed through a secured shell (SSH) by enabling its SSH capability [33][34]. The static IP address [35] has been used instead of a DHCP IP address because to make different raspberry pi work together, every computer's IP address should be listed and pre-defined as they will be registered in the program to be used. Else, the IP address of RPi number 1 can be different the next time it reboots. The variation of IP address every reboot will cause a problem since the IP addresses to be used in the program should be constant.
The micro-sd cards are inserted on every Raspberry Pi and boot the computer with its peripherals. Due to the keyboard, mouse, the monitor is limited, the researcher configured one raspberry pi at a time by plugging in the micro-sd card and connecting the mouse (A), keyboard (B), Ethernet cable (C), monitor (D), and power supply (E) as shown in Figures 2a and 2b.

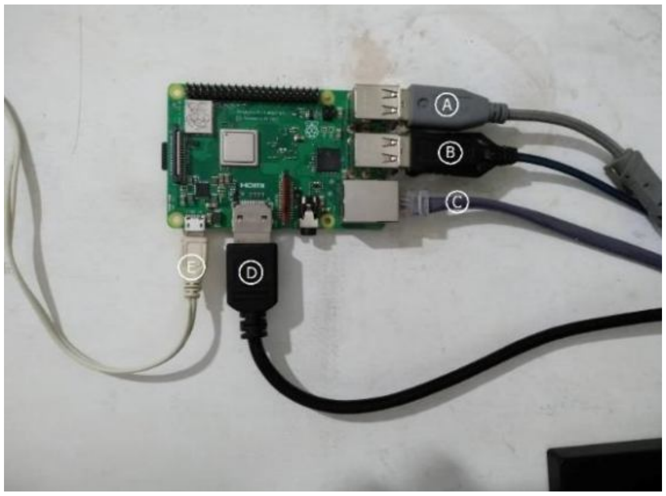


Figure 2a. Connecting the Peripherals of RPi

Figure 2b. Birds eye view of Raspberry pi with peripherals

After all the raspberry pi computers' Ethernet ports have been assigned their respective IP address and enabled the SSH access, the computers are then connected in a local area network using the said TLSF-1024 Switch. Figures 3a and 3b show the network topology used and the actual image connection. One of the Raspberry pis, the Pi-01, acted as the node controller. A separate laptop computer is used for accessing the cluster. Table 1 shows the tabulated information of all the models of raspberry pis that have been used in this research with their respective static IP addresses. The network used a standard class C network following a /24 subnet. The IP address of the Node Controller (Pi-01) is 192.168.100.151 with a 255.255.255.0 subnet mask. Pi-02 to 08 uses 192.168.100.152-158 with the same subnet mask.

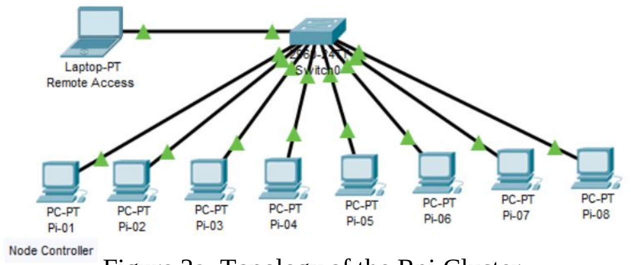


Figure 3a. Topology of the Rpi Cluster

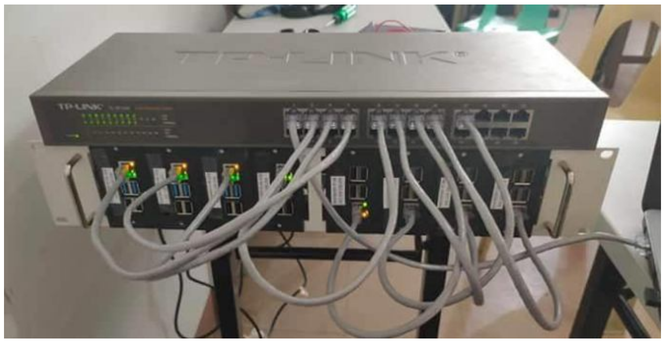


Figure 3b. Actual Connection of eight (8) Rpis

Table 1. Raspberry Pi models used with their IP Addresses

Rpi	Host name	Model Number	Ip Address/ Subnetmask
A	Pi-01	Raspberry Pi 3 Model B+	192.168.100.151/24
B	Pi-02	Raspberry Pi 3 Model B+	192.168.100.152/24
C	Pi-03	Raspberry Pi 3 Model B+	192.168.100.153/24
D	Pi-04	Raspberry Pi 3 Model B	192.168.100.154/24
E	Pi-05	Raspberry Pi 4 4GB	192.168.100.155/24
F	Pi-06	Raspberry Pi 4 2GB	192.168.100.156/24
G	Pi-07	Raspberry Pi 4 2GB	192.168.100.157/24
H	Pi-08	Raspberry Pi 4 2GB	192.168.100.158/24

3.4 Enabling Parallel Computing.

This research used a python-based script called Message Passing Interface to connect all of the Raspberry Pis and make them function as a single cluster (MPI). It is essentially a structured way of sharing messages between several computers running a parallel program through distributed memory, similar to how a supercomputer works. Multiple processors are referred to as nodes of parallel computation. Each node in a parallel arrangement usually operates on a subset of the overall computational challenge, reducing each CPU's workload and speeding up the operation. [36]
Furthermore, MPICH, a lightweight MPI (Message Passing Interface) implementation, was added. It is a high-performance and uniformly compact MPI implementation. It operates on multicore nodes, clusters, and massive supercomputers, among other parallel structures.
It also acts as a forum for MPI application testing and the development of new and enhanced parallel programming environments [37]. MPICH is at the core of the supercomputer's activities, connecting the Raspberry Pi and allowing them to interact. It also distributes the functions that must be completed and considers the data that the supercomputer has stored.

The supercomputer needs to be able to process parallel tasks with an MPI that has python bindings. MPI4PY package provides python programs used in MPI standard. Both MPI and MPI4PY are installed in all members of the cluster to communicate with each other [38]. Additionally, SSH keys for each Raspberry Pi are a must. MPI would be able to connect with each device without thinking about passwords as a result of this. If one raspberry pi is programmed improperly, it will fail to communicate with other raspberry pis. Furthermore, configuring SSH keys allows data to be transferred from one Raspberry Pi to another without the need to verify hostnames or passwords. Each Raspberry Pi must have all of the keys.

Now that the connectivity walls have been built up, the next move is to establish a host file containing the contact information for all cluster members. MPICH examines this file and the verified contact details of all computers in the supercomputer's nodes. The host file has been labeled as "machinefile". This file serves as the contact information of MPICH to pass the message from one raspberry pi to another.

4. RESULTS AND DISCUSSION

4.1 Computing Prime Numbers

The speed of the Raspberry Pis was estimated in millions of instructions per second (MIPS) using the Dhrystone [39] benchmark to understand their fundamental capabilities better. It is a measure that determines how fast a CPU's computing capacity is [40]. The Raspberry Pi 3B received 3475 MIPS in all. The Raspberry Pi 3B+ scored 4028 MIPS, which is higher than the Raspberry Pi 3B. Finally, the Raspberry Pi 4 received a total of 8176 votes, which is more than twice as many as the Raspberry Pi 3B+. Since the Raspberry Pi 4 has the most recent chipset, the benchmark result is as expected. It is the best of the three Raspberry Pis. The python software was written to use MPI to split the job among the Raspberry Pis. The python software detects all prime numbers in the range of 0 to 1,400,000 and displays the number of prime numbers found. The role was split among the supercomputers, allowing them to return results more rapidly as shown on the Table 2 and Figure 4.

Table 2. Seconds to Finish Computing Prime Numebrs

Trial\Rpi	A	AB	ABC	ABCD	ABCDE	ABCDEF	ABCDEFG	ABCDEFGH
1st	29	23	18	14	15	9.8	12	12
2nd	29.4	24.3	18.9	15.2	15.1	10.5	12.2	10.8
3rd	29.5	24.7	18.5	14.9	16.1	10.3	11.3	11.3

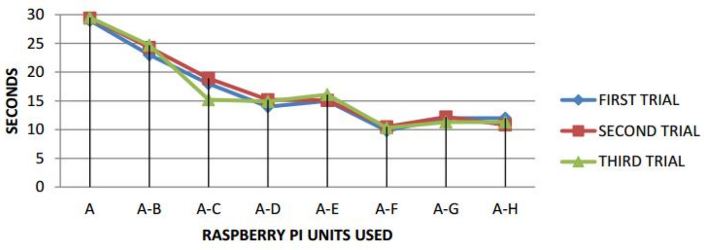


Figure 4. Line Graph of Time vs Number of Rpis

Table 2 shows the different numbers of raspberry pi in running the code to see if there will be any difference in the time it will take to complete the code given. As the number of raspberry pi increases, there is a significant decrease in the number of seconds the task is completed. From 29 seconds, they were using only Raspberry pi A at the first trial, down to 14 seconds when raspberry pis A, B, C, and D are already involved—considering that all raspberries involved in this scenario are of all the same model series, Rpi 3 (as shown in Table 1). However, when the 5th raspberry pi, a model 4, was added (ABCDE), there was a drastic change in the pattern. Adding a more powerful computer into the cluster increased the number of seconds from 14 to 15 seconds. It should have lowered the time needed for the raspberry pis to find the prime numbers, but instead, it made it higher. As the number of Model 4 increases in the clusters, it is seen that the pattern of decreasing time is already significant. The results are depicted in Figure 4 as the line graph states that the more significant number of raspberry pi's in the cluster, the lesser time it will take to perform a task. All three trials depict similar results.

4.2 Rendering Animation Using Blender Application

The cluster is put to another test utilizing rendering animation in the form of the Blender application. The method of gradually increasing the number of computers in the cluster for a single task is still the same as what we have done in computing prime numbers. The animation is a cube spinning using the following settings in Table 3 using 100% CPU usage on all Raspberry Pis. By doing so, the cluster was tested for stability under full load while observing, on the other hand, the temperature of every processor for possible overheat.

Table 3. Sample Image of the Rendered Video and its Settings

Sample Image	Settings used
	Resolution: 1280 px x 1080px Frames: 50 Frames Render Samples: 35 Preview Samples: 10 Other Settings: Default

As shown in Figure 5, the result illustrates a drastic decrease in the processing time as the number of Raspberry pis increases. It did not show any glitch when the 5th computer was added compared to the Computing Prime Numbers test. The decrease of time from 46.27 minutes for a single computer down to 3.22 minutes when all eight computers work on the same job is highly significant. However, the gap between the A-G and A-H was just 44 seconds. It also made the supercomputer quicker, but the difference was not as noticeable from A to A-E, as if adding more Raspberry Pis would not make a massive difference in terms of processing pace. It could be due to a hardware constraint or a network issue, where the only issue is data transmission from one device to another, or it could be due to the cluster's use of multiple computer types.

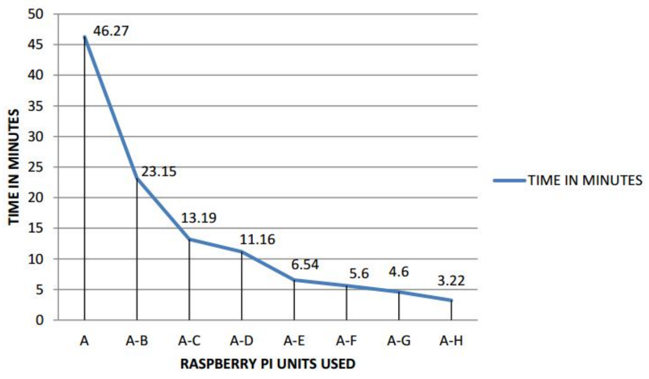


Figure 5. Time Graph for Animation Rendering

4.3 Clustering the Same Models and Comparing the Results.

The series of results show significance on the model numbers. To further solidify this theory, a test was conducted by creating a test using a cluster of all similar models and a cluster with combination models. As shown in Table 4, a cluster of four models 3 raspberry pi took 11.16minutes in rendering the same animation as shown in Figure 6. 4.4 minutes is the time it took for four models 4 clusters. The combination of two models 3 and two models 4 took 6.6 minutes.

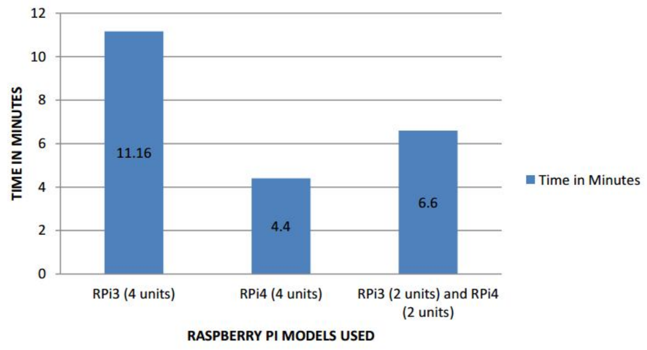
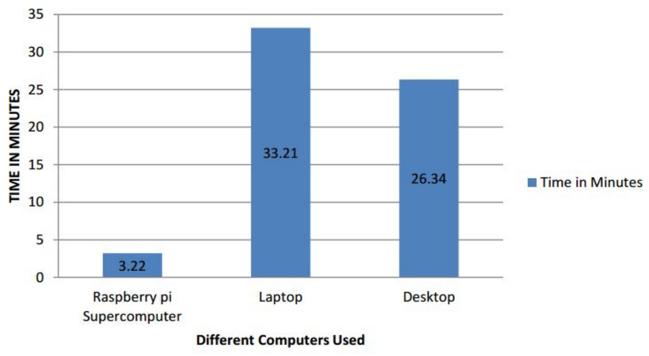


Figure 6. Time Needed to Render Using Different Cluster Types

The results between the Model 3 cluster to Model 4 cluster are expected since the latter's base clock/CPU is 1.5GHz and the former is only 1.4GHz. In addition, Model 4 has a newer chipset of Cortex-A72, compared to the Model 3 chip, which is Cortex-A53. Furthermore, the Model 3 Cluster result to the combined cluster of two-Model 3 and two-Model 4 shows that the latter is faster by 4.56 minutes. Lastly, the Model 4 Cluster result to the combined cluster of two-Model 3 and two-Model 4 shows that the latter is slower by 2.2 minutes.

4.4 Comparing the Supercomputer to other Computers

The final test was done by comparing the supercomputer to a standard desktop and a typical laptop. The laptop has an Intel Core i3-5005U processor with 2.0Ghz speed, and the desktop has an Intel Core i5-2500 processor with 3.30 GHz speed. All computers rendered the same image and settings as mentioned in Table 3. Figure 7 shows the results.



5. CONCLUSION

Combining different processors' models affects the rate of speed performance of the cluster compared to a cluster with similar models/processor speed. Also, the significant reduction of time may come into a plateau as the number of computer increases in the cluster.

ACKNOWLEDGEMENT

The author thanks Mr. Arturo B. Malapit III and Ms. Dorothy A. De Guzman for their assistance.

REFERENCES

1. Philip J. Basfor et.al., “Performance Analysis of Single Board Computer Clusters,” January 2020, Future Generation Computer Systems Vol 102, pp 278-291, https://www.sciencedirect.com/science/article/pii/S0167739X1833142X#bib1
2. S. Jain, A. Vaibhav and L. Goyal, "Raspberry Pi based interactive home automation system through E-mail," 2014 International Conference on Reliability Optimization and Information Technology (ICROIT), Faridabad, India, 2014, pp. 277-280, doi: 10.1109/ICROIT.2014.6798330. https://www.sciencedirect.com/science/article/pii/S0167739X1833142X#bib1
3. F. P. Tso, D. R. White, S. Jouet, J. Singer and D. P. Pezaros, "The Glasgow Raspberry Pi Cloud: A Scale Model for Cloud Computing Infrastructures," 2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops, Philadelphia, PA, USA, 2013, pp. 108-112, doi: 10.1109/ICDCSW.2013.25. https://www.sciencedirect.com/science/article/pii/S0167739X1833142X#bib1
4. Kevin Doucet, et.al., “Learning Cluster Computing by Creating Raspberry Pi Cluster,” April 2017, ACM SE ’17: Proceedings of the SouthEast Conference, pp 191-194 https://dl.acm.org/doi/abs/10.1145/3077286.3077324
5. Kevin Doucet, et.al., “The Creation of a Low-cost Raspberry Pi Cluster for Teaching,” May 2019, WCCE ’19: Proceedings of the Western Canadian Conference on Computing Education, Article No.7 pp 1-5 https://dl.acm.org/doi/10.1145/3314994.3325088
6. Vishnu Govindaraj. “Parallel Programming in Raspberry Pi Cluster,” A Design Project Report, School of Electrical and Computer Engineering, Cornell University, New York, 2016. https://courses.ece.cornell.edu/ece6930/ECE6930_Spring16_Final_MEng_Reports/RPi_Cluster/RPi_Cluster.pdf
7. Matthias A. Blumrich. “Massively Parallel Supercomputer,” US Patent # US7555566B2, https://patents.google.com/patent/US7555566B2/en
8. https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/
9. Ahmad Ashari, et.al., “High Performance Computing on Cluster and Multicore Architecture”, December 2015, TELKOMNIKA, vol 13 no. 4 pp. 1408-1413 http://journal.uad.ac.id/index.php/TELKOMNIKA/article/view/2156/1899
10. Kathiravan Srinivasan et. al. “An Efficient Implementation of Mobile Raspberry Pi Hadoop Clusters for Robust and Augmented Computing Perfromance”, August 2018, Journal of Information Processing Systems, vol 14, No. 4 pp.989-1009 http://jips-k.org/q.jips?cp=pp&pn=588
11. Stephen J. Bigelow, “Message Passing Inteface”, July 2013, TechTarget Network, https://searchenterprisedesktop.techtarget.com/definition/message-passing-interface-MPI
12. Gropp, W., Thakur, R., & Balaji, P. (Accepted/In press). Translational research in the MPICH project. Journal of Computational Science, [101203]. https://doi.org/10.1016/j.jocs.2020.101203
13. Mortensen, M., Dalcin, L., & Keyes, D. E. (2019). mpi4py-fft: Parallel Fast Fourier Transforms with MPI for Python. Journal of Open Source

Software, 4(36), 1340. https://doi.org/10.21105/joss.01340
14. https://bitbucket.org/mpi4py/mpi4py-fft
15. Lisandro Dalcín, Rodrigo Paz, Mario Storti, Jorge D'Elía, "MPI for Python: Performance improvements and MPI-2 extensions," 2008, Journal of Parallel and Distributed Computing, Volume 68, Issue 5, Pages 655-662, ISSN 0743-7315, https://doi.org/10.1016/j.jpdc.2007.09.005.
16. S. Wazir, "Performance comparison of MPICH and MPI4py on raspberry Pi-3B Beowulf cluster", 2019, Journal of Advanced Research in Dynamical and Control Systems 11(03-Special Issue):1766-1774, https://arxiv.org/abs/1911.03709
17. https://www.raspberrypi.org/products/raspberry-pi-3-model-b/
18. https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/
19. https://www.raspberrypi.org/products/raspberry-pi-4-model-b/
20. https://kb.sandisk.com/app/answers/detail/a_id/22697/
21. Rpi 3 vs 4 power supply, https://www.raspberrypi.org/forums/viewtopic.php?t=276434
22. Electronics Tutorials, "Convert ATX PSU to Bench Supply" https://www.electronics-tutorials.ws/blog/convert-atx-psu-to-bench-supply.html
23. https://www.tp-link.com/ph/business-networking/unmanaged-switch/tl-sf1024/
24. EIA/TIA 568A and 568B Standard, http://www.utm.edu/staff/leeb/568/568.htm
25. The Internet Centre Inc., "Ethernet Cable Color Coding Diagram", 2018, https://incentre.net/ethernet-cable-color-coding-diagram/
26. Simon Long, "Jessie Is Here", 2015, Raspberry Pi Blog, https://www.raspberrypi.org/blog/raspbian-jessie-is-here/
27. Tyler, " How to Install Raspbian Jessie on the Raspberry Pi", 2020, Howchoo, https://howchoo.com/pi/how-to-install-raspbian-jessie-on-the-raspberry-pi
28. https://www.balena.io/what-is-balena/
29. https://github.com/balena-io/etcher
30. Ask Ubuntu, "SD Card cloning using the DD Command", 2013, https://askubuntu.com/questions/227924/sd-card-cloning-using-the-dd-command
31. Shivam Raj, "How to Clone Raspberry Pi SD Card on Windows, Linux and macOS", 2016, Beebom, https://beebom.com/how-clone-raspberry-pi-sd-card-windows-linux-macos/
32. Superuser, "Cloning an SD card onto a larger SD Card", 2013, https://superuser.com/questions/460657/cloning-an-sd-card-onto-a-larger-sd-card
33. Raspberry pi Foundation, "SSH (Secure Shell)", https://www.raspberrypi.org/documentation/remote-access/ssh/
34. Goran Jevtic, "How to enable ssh on raspberry pi {Linux, MacOS, Windows}, 2020, Phoenix NAP, https://phoenixnap.com/kb/enable-ssh-raspberry-pi
35. 1&1, "How to set a Raspberry Pi with a static IP address?" 2019, Digital Guide IONOS, https://www.ionos.com/digitalguide/server/configuration/provide-raspberry-pi-with-a-static-ip-address/
36. George K. Thiruvathukal, Sarah Kaylor, " Message Passing Interface ", 2019, Distributed Systems, https://ds.cs.luc.edu/mpi/mpi.html
37. Rajeev Thakur, Mark Snir, Robert J. Latham, Robert B. Ross, "MPICH: A High-Performance, Portable Implementation of MPI", Mathematics and Computer Science Division, Argonne National Laboratory, https://www.anl.gov/mcs/mpich-a-highperformance-portable-implementation-of-mpi
38. Lisandro Dalcin, "MPI for Python", 2020, https://mpi4py.readthedocs.io/en/stable/
39. John C McCallum, "Benchmark results for microcomputers and large computers: Performance tests on IBM and DEC Vax" , 1986, Data Processing, Volume 28, Issue 8, Pages 426-433, ISSN 0011-684X, https://doi.org/10.1016/0011-684X(86)90426-0.
40. Park, G. H., Sung, W. C., Kim, H. J., Im, J. B., Park, J. W., Kim, S. D., & Park, S. B. (2006). "Practice and experience of an embedded processor core modeling." In High Performance Computing and Communications - Second International Conference, HPCC 2006, Proceedings (pp. 621-630). (Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics); Vol. 4208 LNCS). Springer Verlag. https://doi.org/10.1007/11847366_64