IMPLEMENTATION OF ASYNCHRONOUS PARALLEL PROCESSING WITH RASPBERRY PICLUSTER

Pradana Atmadiputra[1]

Computer Science Engineering, International University Liaison Indonesia, BSD, Indonesia
email: [1]pradana.atmadipura@iuli.ac.id

Abstract. Modern parallel computing has ever been so chunky and expensive. With the use of a miniature sized Raspberry Pi and Cluster HAT, it becomes more affordable and leaves a small physical footprint for easy storage. Parallelism can be used to spread the workloadof computation-intensive applications across the multiple cores of a Raspberry Pi. It is the principle upon which modern supercomputers are built. This paper will describe the implementation of asynchronous parallel processing with Raspberry Pi Cluster. This research involves a Raspberry Pi 3, four Raspberry Pi Zeroes, and a ClusterHAT to act as the controller of the cluster. The cluster is then tested against a Python-based module specifically used to exploit clusters and a 'nginx' web server. The experiment proves thatclustered Raspberries perform better than single ones. It also proves that the cluster seamlessly functions as a single unit regardless of inactive nodes.

Keywords: Parallel Computing, Raspberry Pi, Supercomputers, Cluster

1. Introduction

Parallel computing and related concepts have since been used by capital intensive industries in the late 1960s. With the costs of hardware dropping overtime and the birth of open-source applications and operating systems; enthusiasts, students, young professionals, and small companies would have the ability to leverage these technologies for their own. Traditionally, the term parallel computing was found within High Performance Computing (HPC) architectures whereby systems were categorized by high speed and dense calculations which is also familiar with super-computing.

The beginning of supercomputing can be found in the 1960s with an organization called Control Data Corporation (CDC). Seymour Cray was an electrical specialist working for CDC who got known as the dad of supercomputing because of his work on the CDC 6600, generally viewed as the principal of supercomputers. The CDC 6600 was the quickest operating PC somewhere in the range of 1964 and 1969. In 1972 Cray left CDC and founded his own organization, Cray Research. In 1975 Cray Research publicly announced the Cray-1 supercomputer.

The Cray-1 would proceed to be one of the best supercomputers in history and was still being used among a few establishments until the late 1980s. The 1980s likewise observed various players enter the market including Intel through the Caltech Concurrent Computation venture, which contained 64 Intel 8086/8087 CPU's and Thinking Machines Corporation's CM-1 Connection Machine.

The boom occurred in the 1990s concerning the number of processors being integrated for super-computing machines. It was in this decade that IBM notoriously beat world chess ace Garry Kasparov with the Deep Blue supercomputer.

The Deep Blue machine contained around 30 nodes each including IBM RS6000/SP parallel processors and various "chess chips". By the 2000s the number of processors had bloomed to many thousands working in parallel. As of June 2013, the quickest supercomputer title was held by the Tianhe-2, which contains 3.120.000 cores and is equipped for running at 33,86 petaflops every second (Parallel Computing Backgrounder, 2021).

Parallel processing isn't simply restricted to the domain of supercomputing. Today we see these ideas present in multi-core and multiprocessor machines. To an extent, single gadgets we likewise have, frequently containing a single core, that can be associated to cooperate over a network.

Constructing a cluster for computing enthusiasts has often required the purchase of expensive and massive hardware like a full desktop PC's or implementing complex virtual machine setups. Essentially what a cluster means is a group of separate devices paralleled together.

With the low cost and small compact physical space, the Raspberry Pi allows computing enthusiasts to further explore parallel computing and the premise of building a cluster has become far cheaper and simple to follow for users at home. It is an excellent method of teaching both the hardware and software at the same time. While the Raspberry Pi wouldn't be classified as a full-fledged PC, it still provides a learning kit that professional clusters were built upon. For example, working with industry standards along with MPI and asynchronous parallel processing on clusters. The Raspberry Pi 3B comes with a built-in Wi-Fi, which allows other Raspberry Pi's to connect on the same wireless network. Unlike

a PC which may contain more than one CPU, the Raspberry Pi 3B contains a single ARM processor operating at 4 cores with an internal clock of 4x1.200MHz, multiple of Raspberry Pi's combined gives more CPU cores to work with.

2. Review

Previous implementation of Raspberry clustering uses a Raspberry Pi 3, a ClusterHAT, and four Pi Zeroes. The implementation uses HPC Challenge Linpack for benchmarking on single Pi Zero, four Pi Zeroes , and single Pi 3. However, the benchmarking was not done using the cluster yet (Smith, 2016).

a. Brief Overview of Asynchronous Parallel

Asynchronous parallel generally means executing multiple tasks at the same time in parallel. There would be no blocking process, like waiting to complete, but rather continue with delivering other tasks. Sometimes many people find it difficult to articulate the difference with parallel programming. Simple processing of independent data is a great place for asynchronous parallel and provides a better UX and, most of the time, better performance for the application (Stringfellow, 2017). A framework that allows asynchronous parallelism would be Message Passing Interface (MPI). It is a language- independent message-passing communication method developed in the early 1990s to aid parallel computing application development. The MPI standard defines a core set of routines that can be used by a programmer for distributing their application and handles passing back the results of the executed code seamlessly. In MPI's earlier days, C and Fortran were the languages associated with parallel computing; however, Java and Python among others have also offered to support MPI. There are several parallel asynchronous implementations besides MPI, which are SHMEM and OpenMP. It is Jacobi's method for solving systems of linear equations.

The precise implementation details of asynchronous algorithms can strongly affect the resulting performance and convergence behavior in unexpected ways (Barney, 2021).

b. Raspberry Pi Cluster Design

Cluster computing is a setup where a set of computers works together as a single system. Computers—or nodes—within a cluster must have identical hardware and software configurations. Resources in a cluster is managed by centralized r esource manager. Cluster computing is typically used to host databases or web applications.

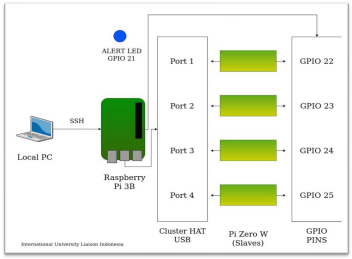

Figure 1 Cluster Connections Diagram

Figure 1 illustrates the connections between the individual components that make up a cluster. The implementation focuses on the connections between these components and to harness the processors from the slaves to combine as a cluster.

The Cluster HAT comprises five micro-USB 2.0 hubs: four for the Pi Zero W and one for the power delivery to the hardware, and GPIO controlled for each Pi Zero W including the Alert LED (ClusterCTRL - Setup Control, 2021). Secure Shell (SSH) is a necessary for accessing each individual Pi as

there will be initial setup and program installations. Also, in the diagram, one SSH connection has been clearly shown as the initial connectionto the master node and from there SSH can be widely used to connect to other Pi Zero W ports after it has been configured with its wireless setup.

Another factor to consider is that if one slave node dies or is inactive due to failure in port connections or requires backup storage, the cluster will still work fine as there are no dependencies between each slave node. It is treated as an individual node and with this in mind, users can respectively attach or detach nodes from clusters. In this cluster, adding more nodes will have a significant amount of processing cores and is beneficial for this research on computing heavy prime numbers – though the speed in run time execution differs with a few minor tolerances.

3. Methodology

a. Network Address Translation (NAT)

For the networking of the cluster, it can be done in different types of system networks such as Network Address Translation (NAT). NAT works in a cluster for translation of public IP addresses to private IP addresses or vice versa so then every computer on the Local Area Network (LAN) can easily access the internet. It is a system that merges more than one computer to the internet by only using an IP address. There are various benefitsto using NAT. One of them is to save the legal IP that is provided by ISP (Internet Service Provider) (Riyadi, 2019).

The architecture of the network using NAT can be seen in Figure 2 where each of the nodes is connected to the controller so public addresses of 192.168.0.160 are translated to the private IP address and these addresses can access the internet as it is connected to the controller as it has the public address. The benefit of using NAT itself is creating a subnet for the nodes to the controller which makes it more secure, as the nodes can only be visible to the controller and not the external devices who want to access it through SSH(L., 2019).

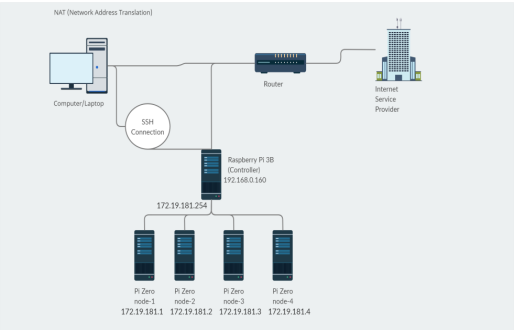

Figure 2 Displaying the network architecture of NAT (Network Address Translation)

Unlike NAT, bridge networking is another alternative way of the system network that can be used to handle the cluster. This bridge network works to connect each network with other networks to exchange data packets and regulate the data circulation. It connects separate LANs. If the bridge has received a data packet, it will try to find the destination and source of the data sent. If the delivery destination is not recognized, the bridge will reject it. But if the destination and source are matched, the data packet will be forwardedto the destination address (Hidayat, 2019). CBRIDGE image allows bridging the USB Gadget Ethernet from the Pi Zeros to eth0 on the controller which allows them to receivean IP address from a DHCP (Dynamic Host Configuration Protocol) server on local network (ClusterCTRL - Setup Software, 2021).

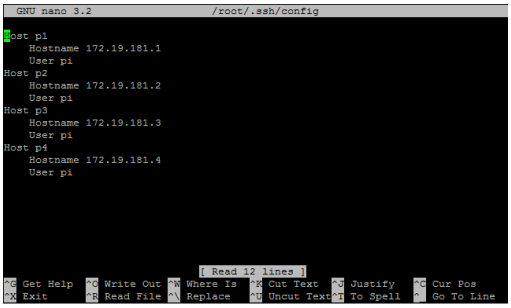

Figure 3 NAT (Network Address Translation) architecture of the cluster

NAT implementation in the cluster can be done by burning a CNAT image designed for the NAT implementation to the controller SD card. After successfully connecting the cluster to network, each of the node needs to be configured so they can be accessed through SSH. By using NAT, each node only has local address relative to the controller. Individual node can only be accessed by accessing the controller first. Figure 3 shows thehost name configuration of each node.

Bridge networking is implemented by burning the image of CBRIDGE to the controller SD card. Bridge connection allows each of the nodes to connect to the Wi-Fi so then it can be accessed by external devices without access to the controller first. This is achieved by assigning public address to each of the node. Bridge networking will be used in this implementation.

4. Result

a. Docker distribution implementation on Raspberry Pi Cluster

One way the cluster can be tested is by using Docker software, which is an open platform for developing, shipping, and running applications (Docker Overview, 2021). Docker is installed in each of the node, while Docker swarm is installed in the controller as the orchestration tool.

The activities of the cluster are controlled by a swarm manager. Machines that have joinedthe cluster will be referred to as nodes. These nodes can be either swarm manager, workeror perform both roles, depending on the user's need. Swarm managers manage the membership and delegation while workers are responsible to run swarm services (What is a Docker Swarm?, 2021). Figure 4 shows one node named 'cnat' assigned to be leader/manager.

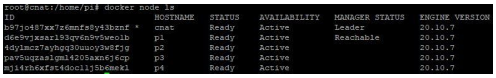

Figure 4 List of the nodes joined into Docker

In this implementation, a web application named 'visualizer-arm' is used as the tool to monitor activities in each node. Figure 5 shows the interface of the application. The figure shows that 'viz' (the web application itself) is running in 'cnat' node, which acts as the manager.

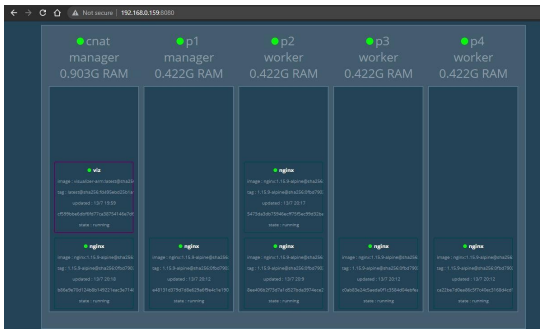

Figure 5 Docker Swarm visualizer

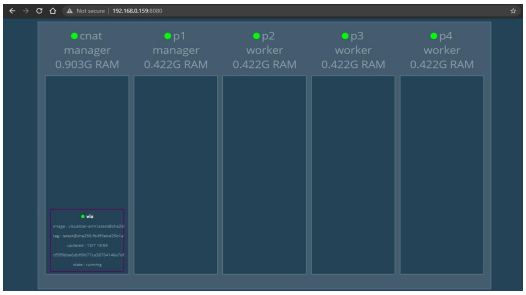

Figure 6 Adding nginx application running in Docker

Figure 6 shows that the 'nginx' web server is installed and run in the Docker hub. In this case, the manager node assigned p2 worker node to run the nginx web server.

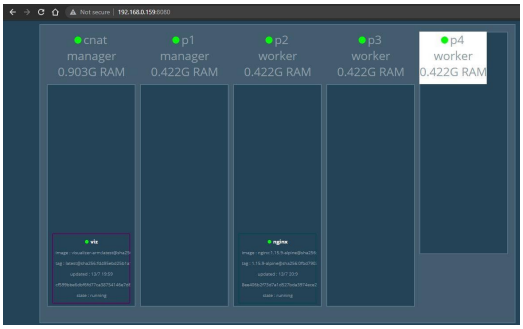

Figure 7 Scaling up nginx service

In order to test the task distribution in the cluster, the 'nginx' web server is scaled up to run six tasks in the Docker Swarm, as shown in Figure 7.

```
root@cnat:/home/pi# docker service scale nginx=6
nginx scaled to 6
overall progress: 6 out of 6 tasks
1/6: running   [==================================================>]
2/6: running   [==================================================>]
3/6: running   [==================================================>]
4/6: running   [==================================================>]
5/6: running   [==================================================>]
6/6: running   [==================================================>]
verify: Service converged
root@cnat:/home/pi#
```

Figure 8 Task distribution across the cluster nodes

Figure 8 shows the task distribution of the 'nginx' task after it's scaled up. Docker automatically distributes the tasks across the cluster nodes (including the managers) as even as possible, so no single node is overloaded with numerous tasks.

When nodes were detached from the master node which is 'cnat', the 'viz' application would indicate that some nodes are offline. Figure 9 shows that nodes p2 and p4 are inactive.

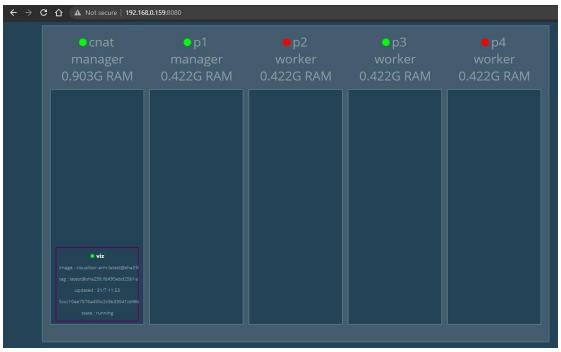

Figure 9 viz application displays inactive nodes p2 and p4

Even though two nodes has been deactivated, the cluster can still distribute the tasks correctly. Figure 10 shows that the 'nginx' web server has been scaled up to run five tasks, and the tasks can be distributed evenly across the active nodes and manager, namely 'cnat', p1, and p3.

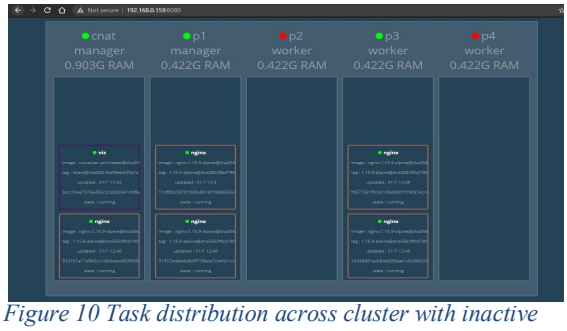

Figure 10 Task distribution across cluster with inactive nodes

This experiment shows that Docker swarm is capable to manage to run multiple tasks across the cluster available nodes. Regardless of the number of nodes available in the cluster, the 'nginx' web server is still accessible in the client's browser, as shown in Figure 11. In the client's view, there is no

indication whether any node is active or not.

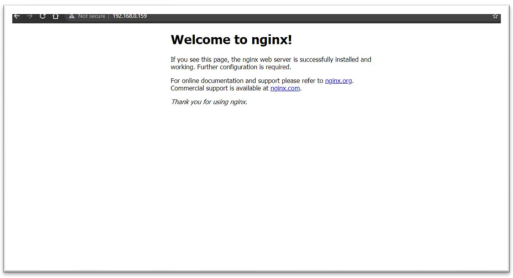

Figure 11 Web application sample running on nginx server, displayed in client's browser

b. Implementation using MPI4PY

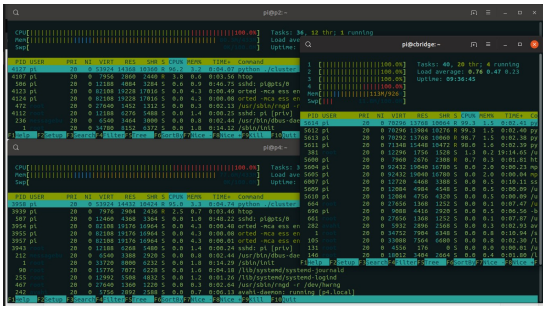

Figure 12 Aynchronous parallel process using 3 Raspberries.

Figure 12 illustrates the implementation of three Raspberry Pi Zeroes running together as a cluster: two Pi Zeroes, and the master node Raspberry Pi 3B. In total, the overall core summed up to six single cores where four cores came from the Raspberry Pi 3B and the single cores from the two Pi Zero W's. The htop command is used to visually indicate the number of processes distributed and also see the overall workloads.

The testing ran on Python 2.7 with the external MPI4PY module as a workload-intensive program, which basically continues to compute large prime numbers and display the run time of execution. The controller measures the status on all active Pi Zeroes as well as the Raspberry Pi 3B over SSH. Initially, a single-core Pi Zero W took a mere 2.46 minutes, a lengthy amount of waiting time. Whereas a four-core Raspberry Pi 3B took 1.42 minutes, much faster response and execution speed time. The results that were done however were not consistent for many reasons. The distribution between master and slaves was random, meaning there would be a slight performance issue if a process has more workload than another. Another reason that may cause a disruption comes from the SSH connections of nodes, in particular the controller as it uses the hostname to carry out the distribution of processes via SSH.

Table 1 Individual node average run time results

Test Unit	Run Time
Raspberry Pi Zero W (1 Core)	00:02:46 minutes
Raspberry Pi 3B (4 Cores)	00:01:42 minutes

Table 1 shows the time required to run the module by each node individually. With all four Pi Zeroes attached to the controller, the average results were determined to be roughly ± 55 seconds. This experiment shows that clustering Raspberries would increasethe computing performance compared to running a unit individually.

5. Conclusion

In conclusion, the small-sized and affordable Raspberry Pi allows computing enthusiasts to conduct research on parallel computing anywhere while also benefiting from the programming tools used by today's industry standards. The ClusterHAT allows the bridging connections between the Pi Zeroes and the Raspberry Pi 3B. Clustering Raspberries is proven to improve computing

performance and ensure availability when one or more nodes are offline.

References

Barney, B. (2021, Decemeber 2). Message Passing Interface (MPI). Retrieved fromLawrence Livermore National Laboratory: https://hpc-tutorials.llnl.gov/mpi/

ClusterCTRL - Setup Control. (2021, November 12). Retrieved from Cluster CTRL: https://clusterctrl.com/setup-control

ClusterCTRL - Setup Software. (2021, December 2). Retrieved from Cluster CTRL: https://clusterctrl.com/setup-software

Docker Overview. (2021, December 2). Retrieved from Docker Documentation: https://docs.docker.com/get-started/overview/

Hidayat, T. (2019, August 19). Fungsi Bridge dalam Jaringan Komputer, Jenis dan CaraKerjanya. Retrieved from Universitas Djuanda Bogor: https://www.unida.ac.id/teknologi/artikel/fungsi-bridge-dalam-jaringan- komputer-jenis-dan-cara-kerjanya.html

L., D. (2019, August 6). The Missing ClusterHat Tutorial. Retrieved from Medium: https://medium.com/@dhuck/the-missing-clusterhat-tutorial-45ad2241d738

Lithmee. (2018, September 8). Difference Between Cluster and Grid Computing. Retrieved from Pediaa: https://pediaa.com/difference-between-cluster-and-grid-computing/#Cluster%20Computing

Parallel Computing Backgrounder. (2021, December 2). Retrieved from Intel.com: https://www.intel.com/pressroom/kits/upcrc/ParallelComputing_backgro under.p df

Riyadi, H. (2019, September 6). Pengertian NAT Beserta Fungsi dan Cara Kerja NAT dalam Jaringan Komputer. Retrieved from NesabaMedia: https://www.nesabamedia.com/pengertian-fungsi-dan-cara-kerja-nat/

Smith, N. (2016, August). ClusterHAT Review for the Raspberry Pi Zero. Retrieved from Climbers.net: https://climbers.net/sbc/clusterhat-review-raspberry-pi-zero/

Stringfellow, A. (2017, September 25). When to Use (and Not to Use) Asynchronous Programming: 20 Pros Reveal the Best Use Cases. Retrieved from Stackify.

What is a Docker Swarm? (2021, December 2). Retrieved from Sumo Logic: https://www.sumologic.com/glossary/docker-swarm/