Cluster Computing Analysis Based on Beowulf Architecture

Purnawansyah[a,1], Ramdan Satra[a,2]

[a] *Universitas Muslim Indonesia, Street Urip Sumoharjo KM 05, Makassar 90231, Indonesia*
[1] *purnawansyah@gmail.com;* [2] *ramdan@umi.ac.id*

ARTICLE INFO

Article history:
January 18, 2016
January 25, 2016
February 2, 2016

Keywords:
Cluster Beowulf
MPI
Parallel Computing

ABSTRACT

The need for computing speed in data processing is in high demand indicated by the increasing data processing in many companies. Aviation companies, for instance, show the growing number of passengers daily. The data processing for a big number of customers requires supercomputer technology at the cost of exponential fund. Therefore, the technology able to process big data with low cost is necessary. This research has improved a cluster computer which can process data with a big capacity processing that is generally cannot be processed using one computer. Cluster computer is built using Beowulf cluster architecture using Linux Debian operating system and SSH, NFS, MPI library and GANGLIA services. The result of this research is a cluster computer prototype which is able to process 38.000x38.000 matrix calculation data. In the first testing, the computer was able to work for a maximum of 30.000x30.000 matrix calculation whereas cluster computer can process up to 38.000x38.000 matrix size. This result indicates that the designed cluster computer is successful to calculate big data with a low computer specification. The further improvement can be applied in a more complicated computer calculation process and bigger data.

I. Introduction

A computer with hundreds of processors (massively parallel processor) has a high ability in the computing processing but it costs much money to purchase it. One of the solutions recommended to a fast and inexpensive computing system is a cluster computer, a standalone group of computers but interconnected one another in a parallel computer network to work on computing process [1,2].

In regard to the technology advancement, the improvement of a big and faster data processing is necessary. The design of cluster computer using low specification computers and the unused computers is the solution recommended to design a cluster computer.

This research would design a parallel computing system based upon Beowulf architecture. It is expected that the design can contribute to solve big data processing problem.

II. Literature review

A. Beowulf Cluster

Cluster in computing is a set of computers where each computer works independently but they are connected by computing network. The computers can share data communication and support software which shares the workload of each computer at the same time in order to enable the computer to work on big data faster [2]. Cluster is also defined as a group of computers working together which can be viewed as a single system [3].

Beowulf Cluster is a group of computers using cheap hardware of consumer grade, for example used and unused computer's components. One of the options to design cluster computer is by using free software,

FOSS (free open source software), to design the entire cluster system and to accommodate parallel computing needs [4]. The typical Beowulf Cluster show in Figure 1.

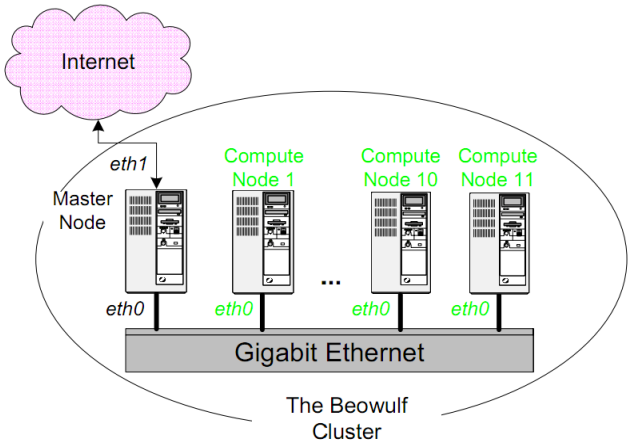

Fig. 1. The Typical Beowulf Cluster [5]

B. *Parallel Computing*

Parallel computing is one way to process computing concurrently by using several computers at the same time. Parallel computing employed at cluster computer is called distributed memory [6]. Example Parallel Computing show in Figure 2.

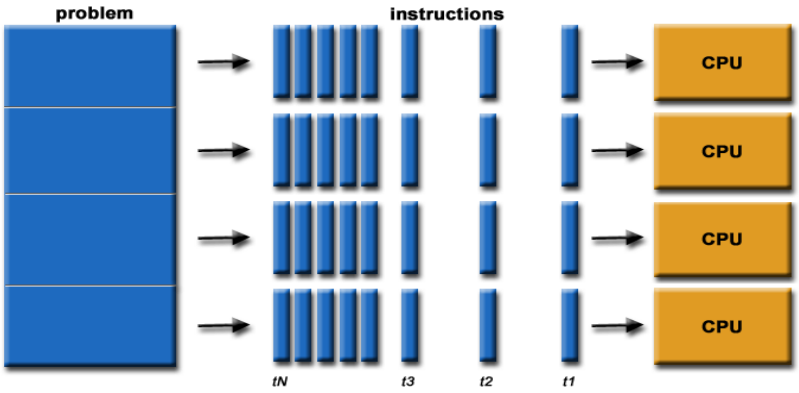

Fig. 2. Parallel Computing

C. *Message Passing Interface (MPI)*

MPI (Message Passing Interface) serves as a standard in parallel programming using memory distribution system [7] as shown at figure 1. The main function of MPI is to use data communication from one process to another [8]. MPI supports C programming language, and FORTRAN [9]. The other innovation of MPI is that it is possible to be used in shared memory. Illustration Distributed Memory show in Figure 3.

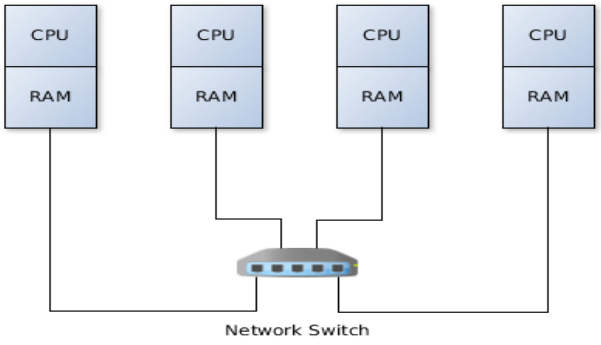

Fig. 3. Distributed Memory

III. Method

A. *Development Stages of Computer Cluster*

Design procedures of a computer cluster in this study consist of several stages, show in Figure 4:

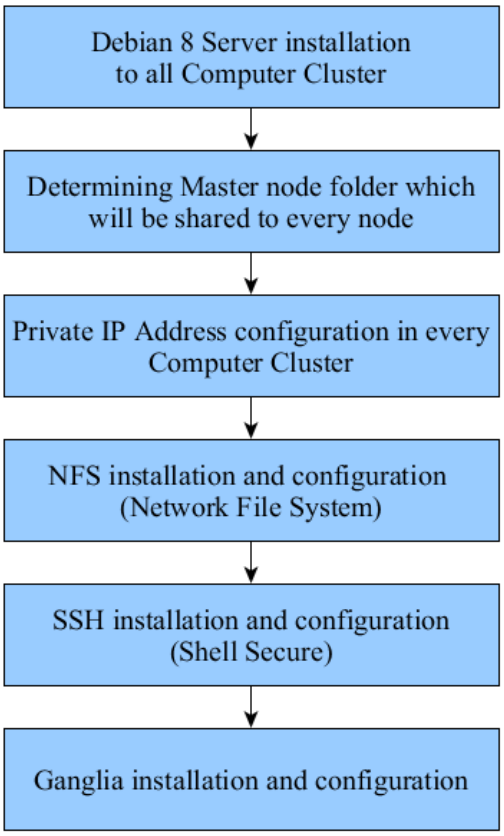

Fig. 4. Development Stages of Computer Cluster

B. *Computer Cluster Network Topology*

A *master node* computer connected by a 5 node computer client uses a switch as a connecting device as shown in Figure 5. The network cluster topology which was private by nature was created by using a star topology or commonly called star topology.

Each node is given a class C private IP address in which the range of the corresponding IP is 192.168.1.1 – 192.168.1.6. A *master node* computer is given the IP 192.168.1.1, computer *node* 1 is given 192.168.1.2, *node* 2 192.168.1.3, *node* 3 192.168.1.4, *node* 4 192.168.1.5 and *node* 5 192.168.1.6.

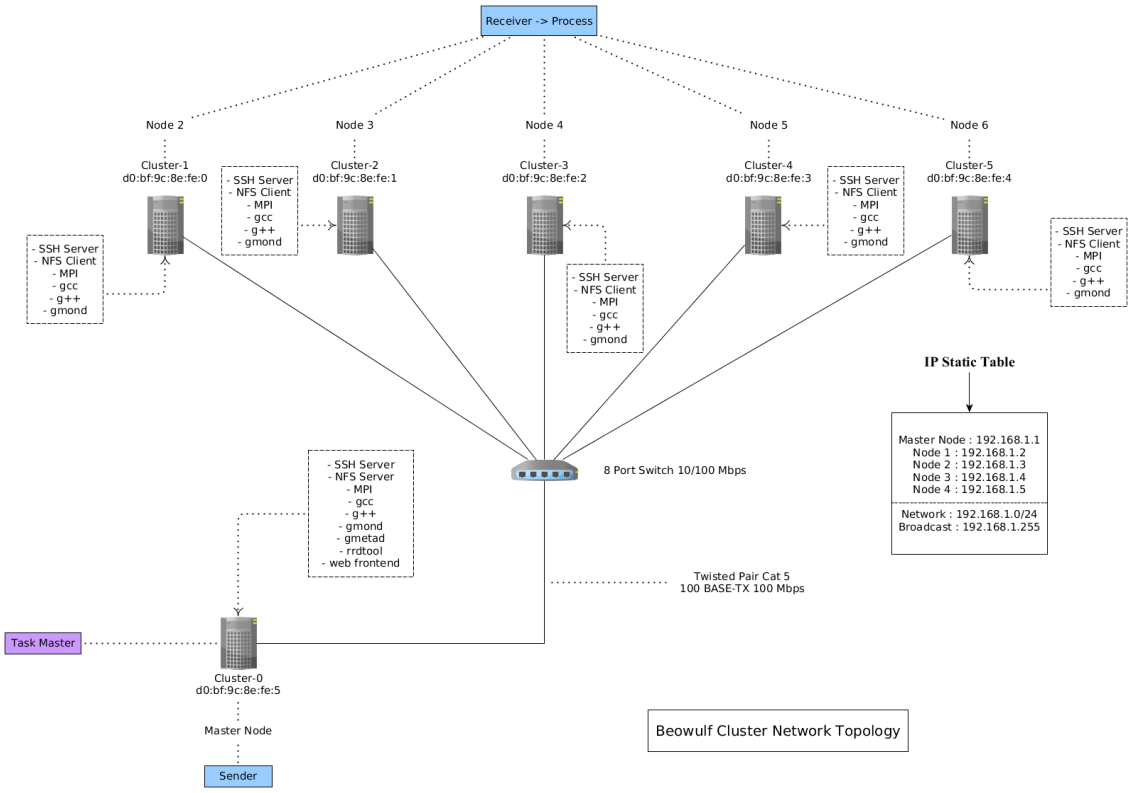

Fig. 5. Cluster Computer Network Topology

C. *Testing Scenario*

Stages of testing in this study are as follows:

1. Test of computing time with matrix multiplication on one computer and five computers (Parallel).
2. Monitoring computer cluster performance using Ganglia tools

IV. Results

Based on the results of sequential and parallel matrix computations testing, it indicates that the sequential matrix calculations are faster than parallel when the matrix size is small because the parallel matrix calculations seem to show longer computing time. Comparison graph execution time between sequential with parallel show in Figure 6.

This is due to the burden of data communication between nodes on the network is greater than the computational load of each node in calculating the size of tiny matrix data. Large Data size of a matrix with 30.000 x 30.000 shows that parallel computing is faster than sequential as indicated in Figure 6 and Table 1. Parallel computing matrix calculation can still perform computational matrix at the size of up to 38.000 x 38.000 whereas sequential computation is only capable of up to 30.000 x 30.000, this is shown in Figure 7.

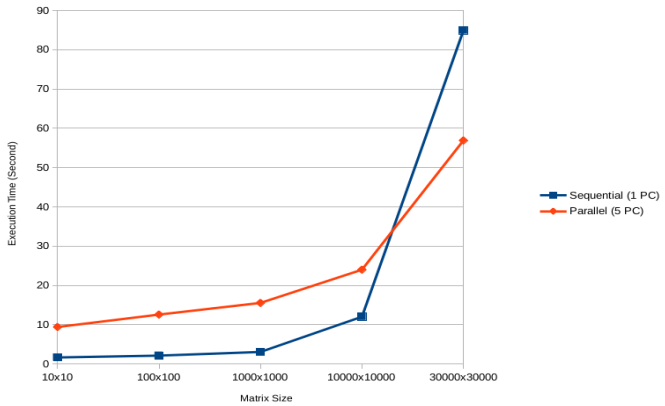

Fig. 6. Sequential and Parallel Testing Graph

Table 1. Computation Time

No	Matrix size	Computation time	
		Sequential (second)	Parallel (second)
1	10x10	1.66	9.42
2	100x100	2.13	12.58
3	1000x1000	3.06	15.55
4	10000x10000	12.03	24.01
5	30000x30000	84.94	56.91

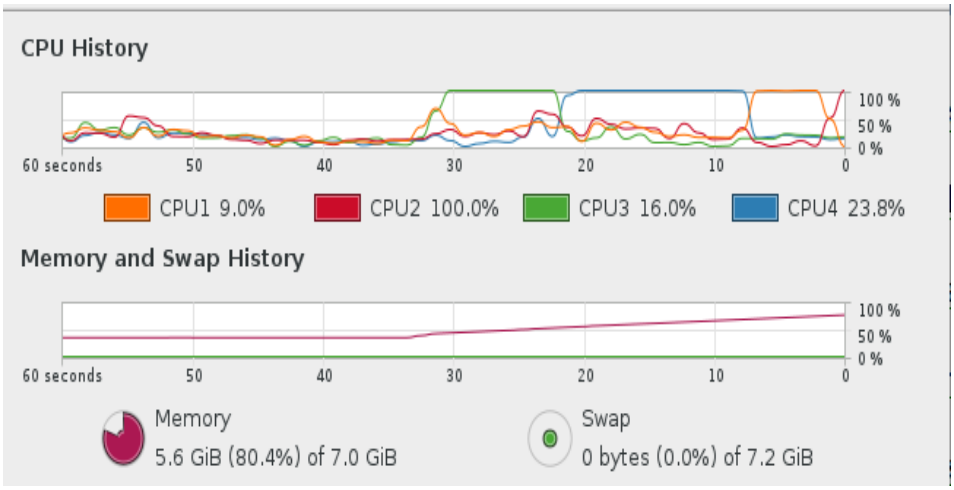

Fig. 7. CPU and Memory Graph For Matrix Size 30.000 x 30.000 (Sequential)

The computer cluster testing is monitored by Ganglia tools. Monitoring is performed for the purpose of observing all computer hardware node process when conducting the computation. Moreover, the hardware activity process being monitored is memory (RAM).

The process of memory (RAM) which will be monitored is useful to see how much the memory usage is when computing large amounts of matrix calculations. Matrix size of 38,000 x 38,000 is the size of the matrix used to perform monitoring experiments due to the size of the matrix computation using a computer (sequential) which were no longer able to process matrix total calculation.

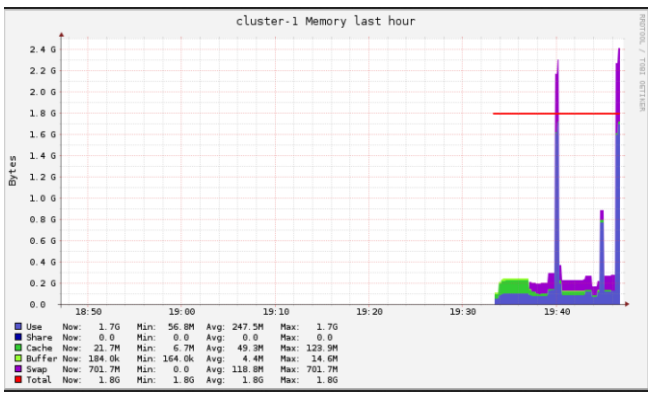

Fig. 8. *Computation Node*-1 process (Matrix Size 38.000 x 38.000)

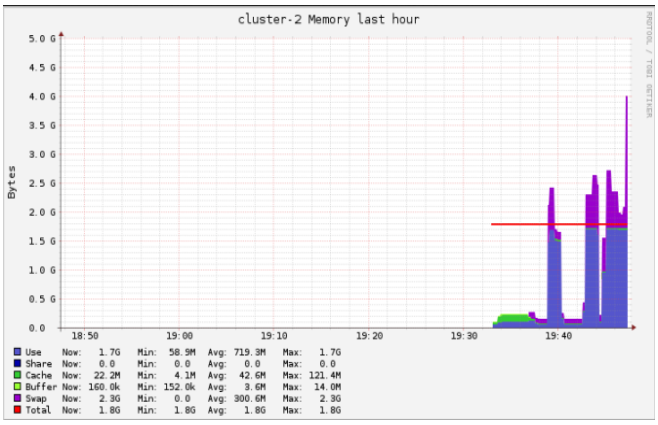

Figure 9. *Computation Node*-2 process (Matrix Size 38.000 x 38.000)

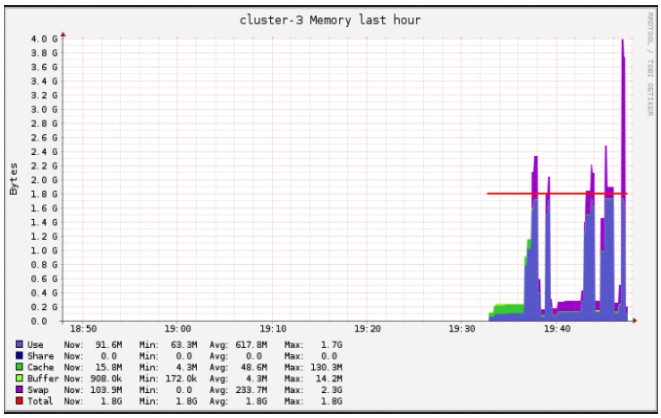

Figure 10. *Computation Node-3* process (Matrix Size 38.000 x 38.000)

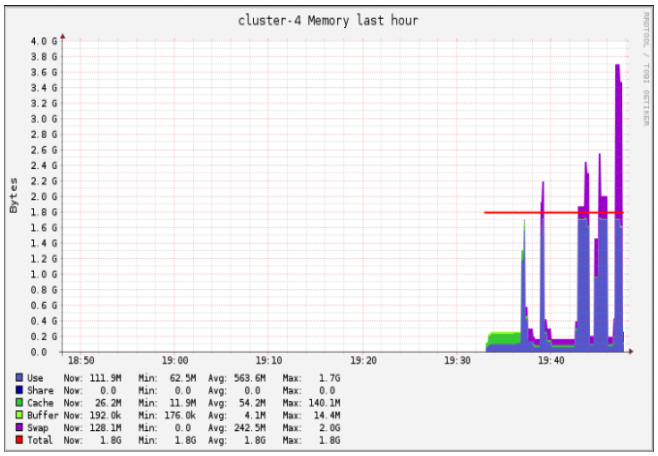

Figure 11. *Computation Node-4* process (Matrix Size 38.000 x 38.000)

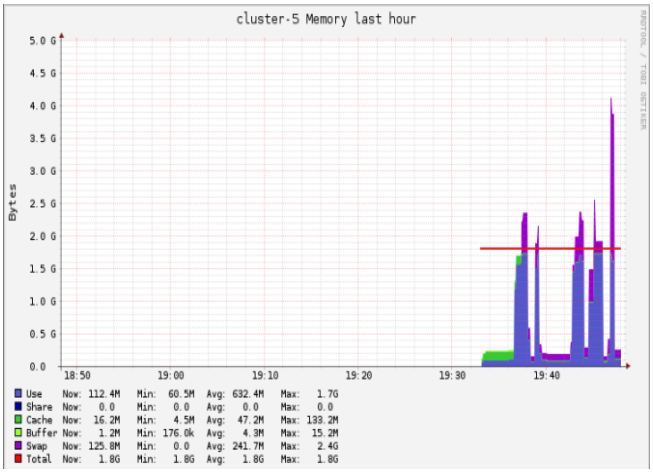

Figure 12. *Computation Node-5* process (Matrix Size 38.000 x 38.000)

The results of monitoring by using Ganglia tools indicate there is a sign of memory usage of each node (computer) on a computer cluster. The process of matrix calculation 38,000 x 38,000 consumes a lot of memory resources. Of the total 2 Gigabytes memory nodes 1.7 GB is used to perform calculations and swap memory (memory backup) is also used, this is shown in Figure 8-12.

V. Conclusion and Recommendation

The conclusions of this study are as follows:

1. Parallel computing speed is strongly influenced by the network interconnection among the nodes.
2. Linux Debian operating system used in this study is very suitable for building a computer cluster, due to the Linux operating system possessing applications that could be used to integrate multiple computers. These applications are NFS, SSH, GANGLIA and MPI.
3. Time speed obtained in the study is that if the value of the matrix is small then the sequential computing is faster than parallel computing. Matrix size of 35.000 x 35.000 can no longer be processed with a single computer (sequential) whereas using multiple computers (parallel) can still perform matrix computation 35.000 x 35.000 to the size of 38.000 x 38.000.
4. Matrix calculation for sequential is merely capable of performing size 30.000 x 30.000 matrix computation.
5. The result of node computer monitoring by using ganglia shows that all node cluster computers are used in the matrix calculation at the size of 38.000 x 38.000 *resource memory*.

Recommendation in this study is to work on further development with several things to consider:

1. To test the performance of parallel computing for total nodes of more than 5 computers.
2. Improvement of cluster into a grid resulting in cluster able to be integrated with other clusters.
3. Selection of cases to be solved is adjusted with the computing cluster.

References

[1] G. Widyaputra. "Peracangan *cluster* linux komputasi paralel *octave*". Universitas Negeri Gadjah mada. Yogyakarta, 2008.

[2] I. H. Syifullah, W. Djuriatno, M. Aswin. "Implementasi pemrosesan paralel pada permainan catur di cluster Beowulf". Univeristas Brawijaya. Malang, 2014.

[3] A. Igumenov dan J. 'Zilinskas. "Electrical Energy Aware Parallel Computing with MPI and CUDA". Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing, 2013.

[4] M. Z. Aliyansyah, M Shidiq, M. Aswin. "Komputasi paralel integral definitif rangkap tiga dengan metode monte carlo di cluster Beowulf". Universitas Brawijaya, Malang, 2003.

[5] Y. Yao, J. Chang, dan K. Xia. "A Case of Parallel EEG Data Processing Upon a Beowulf Cluster". International Conference on Parallel and Distributed Systems, 2009.

[6] R. Satra, W. A. Kusuma, dan H. Sukoco. "Accelerating Computation of DNA Multiple Sequence Alignment in Distributed Environment". TELKOMNIKA Indonesian Journal of Electrical Engineering, 2014.

[7] H. Jin, D. Jespersen, P. Mehrotra, R. Biswas, L. Huang, dan B. Chapman. "High performance computing using MPI and OpenMP on multi-core parallel systems. Parallel Computing". ELSEVIER, USA, 2011.

[8] R. Hempel dan D. W. Walker. "The emergence of the MPI message passing standard for parallel computing". Computer Standards & Interfaces. ELSEVIER, 1999.

[9] YC. Chou, S. S. Nestinger, H. H. Cheng. "Ch MPI: Interpretive Parallel Computing in C". IEEE, 2010.

[10] T. Hoefler, J. Dinan, D. Buntinas, P. Balaji, B. Barrett, R. Brightwell, W. Gropp, V. Kale, dan R. Thakur. MPI + MPI: a new hybrid approach to parallel programming with MPI plus shared memor. Springer-Verlag, 2013.