KONFIGURASI JARINGAN BERTINGKAT PADA CLUSTER PARALEL ORANGE PI

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI REKAYASA KOMPUTER

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

ARI BADIA CHRISTIAN
NIM. 125060300111070

UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK
MALANG
2017

LEMBAR PENGESAHAN

KONFIGURASI JARINGAN BERTINGKAT PADA CLUSTER PARALEL ORANGE PI

SKRIPSI

TEKNIK ELEKTRO KONSENTRASI REKAYASA KOMPUTER

Ditujukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

ARI BADIA CHRISTIAN
NIM. 125060300111070

Skripsi ini telah direvisi dan disetujui oleh dosen pembimbing
Pada tanggal 30 Januari 2017

Dosen Pembimbing I	**Dosen Pembimbing II**
Dr. Ir. M. Aswin, M.T.	**Waru Djuriatno, S.T., M.T.**
NIP. 19640626 199002 1 001	**NIP. 19690725 199702 1 001**

Mengetahui,
Ketua Jurusan Teknik Elektro

M. Aziz Muslim, S.T., M.T., Ph.D
NIP. 19741203 200012 1 001

JUDUL SKRIPSI :

KONFIGURASI JARINGAN BERTINGKAT PADA CLUSTER PARALEL ORANGE PI

Nama Mahasiswa : Ari Badia Christian

NIM : 125060300111070

Program Studi : Teknik Elektro

Konsentrasi : Rekayasa Komputer

KOMISI PEMBIMBING :

Ketua : Dr. Ir. M. Aswin, M.T. ……

Anggota : Waru Djuriatno, S.T., M.T. ……

TIM DOSEN PENGUJI :

Dosen Penguji 1: Adharul Muttaqin, S.T., M.T. ……

Dosen Penguji 2: R. Arief Setyawan, S.T., M.T. ……

Dosen Penguji 3: Dr. Ir. Bambang Siswojo, M.T. ……

Tanggal Ujian : 27 Januari 2017

SK Penguji :

PERNYATAAN ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya dan berdasarkan hasil penelusuran berbagai karya ilmiah, gagasan dan masalah ilmiah yang diteliti dan diulas di dalam Naskah Skripsi ini adalah asli dari pemikiran saya. Tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu Perguruan Tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis dikutip dalam naskah ini dan disebutkan dalam sumber kutipan dan daftar pustaka.

Apabila ternyata di dalam naskah Skripsi ini dapat dibuktikan terdapat unsur-unsur jiplakan, saya bersedia Skripsi dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, pasal 25 ayat 2 dan pasal 70).

Malang, 23 Januari 2017

Mahasiswa,

ARI BADIA CHRISTIAN
NIM. 125060300111070

Untuk keluargaku yang selalu mendukungku

RINGKASAN

Ari Badia Christian, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya, Januari 2017, *Konfigurasi Jaringan Bertingkat Pada Cluster Paralel Orange Pi*, Dosen Pembimbing: M. Aswin dan Waru Djuriatno.

Orange Pi merupakan salah satu PC namun berukuran jauh lebih kecil dan mengonsumsi daya yang jauh lebih rendah. Untuk meningkatkan performa komputasi, Orange Pi dapat disusun secara paralel, membemtuk suatu cluster paralel. Paralelisasi Orange Pi dapat didesain dengan menggunakan sistem paralel cluster Beowulf, dimana semua perangkat berada pada jaringan yang sama dan dapat bekerja bersama dalam menyelesaikan suatu permasalahan. Namun salah satu kekurangan pada metode Cluster Beowufl adalah klienan Bandwith yang terbatas. Pada saat pemrosesan paralel dilakukan, perangkat node dan main akan saling berkomunikasi. Komunikasi antar node dan main yang terjadi ini seringkali membebani Bandwith pada jaringan yang tersedia. Bandwith yang terbatas inilah yang seringkali mengurangi performa komputasi paralel. Oleh karena itu, dilakukan pengujian menggunakan Jaringan bertingkat, dimana pada satu cluster Beowulf, akan dibagi lagi menjadi Sub-Cluster dengan menggunakan 1 router ke setiap sub-cluster. Klienan konfigurasi jaringan bertingkat diharapkan dapat menambah Bandwith pada saat pemrosesan paralel, sehingga kecepatan komputasi secara keseluruhan dapat meningkat. Pada penelitian ini, digunakan 17 buah Orange Pi yang dibagi dalam 4 sub-cluster. Tiap sub-cluster terdiri atas 4 buah Orange Pi, dan 1 Orange Pi bekerja sebagai main. Setiap sub-cluster terhubung pada router yang berbeda. Proses komputasi paralel di implementasikan menggunakan OpenMPI. Pengujian dilakukan sebanyak 2 kali, pada sistem paralel yang sama dan sistem paralel bertingkat. Pengujian kecepatan dilakukan dengan menjalankan program perkalian matriks di kedua sistem paralel. Pengujian juga dilakukan pada dua kasus yang berbeda. Kasus pertama, pengujian dilakaukan dengan kondisi awal yang sama, dimana semua perangkat dalam kondisi baru dinyalakan. Pengujian kedua adalah dengan memandingkan kecepatan pemrosesan dengan berbagai ukuran program yang berbeda. Pengujian juga dilakukan dengan menggunakan 4 jumlah pemrosesan yang berbeda, yaitu 17, 34, 51, dan 68. Hasil dari penelitian ini menunjukkan bahwa peningkatan kecepatan didapat pada kasus pemrosesan paralel dengan kondisi awal baru dinyalakan, pada jumlah proses 17, 51, dan 68. Peningkatan kecepatan juga didapat pada pengujian dengan jumlah proses 34 dan ukuran program 100, 200, 600, dan 700, dan lebih cepat pada semua ukuran program dengan jumlah proses 51 dan 68.

Kata Kunci – Paralel, Cluster, Sub-Cluster, jumlah proses, MPI

SUMMARY

Ari Badia Christian, Department of Electrical Engineering, Faculty of Engineering, University of Brawijaya, January 2017, *Configuration of Multilevel Network in Orange Pi Parallel Cluster*, Academic Supervisor: M. Aswin and Waru Djuriatno.

Orange Pi is a type of PC but much smaller in size the and power consumption is much lower. To improve the computing performance, Orange Pi can be arranged in parallel, making a parallel cluster. Orange Pi parallelization can be designed using a Beowulf cluster parallel system, where all the devices are on the same network and can work together in solving a problem. But one drawback of the method is the use of cluster Beowufl limited Bandwidth. At the time of parallel processing is done, the node and the main will be communicating. Communication between nodes and main often overload the available Bandwidth on the network. Limited Bandwidth is often reduces performance in parallel computing. Therefore, testing is done using multilevel network, which at the Beowulf cluster, the Orange Pi will be subdivided into Sub-Cluster by using one router to each sub-cluster. With the using of multilevel network configuration in Orange Pi Cluster, it is expected to add Bandwidth at the time of parallel processing, so the overall computing speed can be increased. In this study, we use 17 units Orange Pi and is divided into four sub-clusters. Each sub-cluster consisting of four pieces of Orange Pi, and one Orange Pi worked as a Main. Each sub-cluster connected to a different router. Parallel computing process was implemented using OpenMPI. Testing was done 2 times, on the same network parallel system and system of multilevel network parallel. Testing was done by running matrix multiplication program in two parallel systems. Tests were also conducted on two different cases. The first case, testing done with the same initial conditions, where all the devices in fresh condition, boot for the first time. The second test is to compare processing speed with the various sizes of different programs. Tests were also done using four number of process, which is 17, 34, 51, and 68. The results of this study indicate that increased in computing performance obtained in the case of parallel processing with fresh condition, with the number of processes 17, 51, and 68. Speed increasing was also obtained in the test with the number of process 34 and the size of the program at 100, 200, 600, and 700, and more quickly on all measures program with the number of process 51 and 68.

***Keywords** – Parallel, Cluster, Sub-Cluster, number of process, MPI*

PENGANTAR

Syukur dan terima kasih penulis ucapkan kepada Yesus Kristus, karena hanya oleh karena kasih dan kebaikan-Nya lah skripsi ini dapat diselesaikan. Skripsi berjudul "Konfigurasi Jaringan Bertingkat pada Cluster Paralel Orange Pi" ini disusun untuk memenuhi persyaratan memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Universitas Brawijaya.

Penulis menyadari bahwa dalam penyusunan skripsi ini tidak terlepas dari bantuan berbagai pihak. Oleh karena itu, dengan ketulusan dan kerendahan hati penulis menyampaikan terima kasih kepada:

- Papa dan Mama, yang sudah menjadi pembimbing dan motivator terbesar dalam hidup saya.
- Kakak dan adik saya, yang selalu menyemangati saya.
- Keluarga besar yang juga sudah menyemangati dan mendoakan saya.
- Bapak M. Aziz Muslim, ST., MT., Ph.D selaku Ketua Jurusan Teknik Elektro Universitas Brawijaya.
- Bapak Hadi Suyono, ST., MT., Ph.D selaku Sekretaris Jurusan Teknik Elektro Universitas Brawijaya.
- Bapak Dr. Ir. M. Aswin, M.T. sebagai Dosen Pembimbing I dan Bapak Waru Djuriatno, S.T., M.T selaku Dosen Pembimbing II dan Ketua Kelompok Dosen Keahlian Rekayasa Komputer atas segala bimbingan, pengarahan, saran, dan kritik yang telah diberikan selama proses pengerjaan skripsi dan selama saya kuliah di Teknik Elektro.
- Bapak Eka Maulana, ST., MT., M.Eng. selaku Dosen Penasehat Akademik atas segala bimbingan, pengarahan, saran, dan kritik yang telah diberikan selama perkuliahan.
- Seluruh dosen pengajar Jurusan Teknik Elektro Universitas Brawijaya,
- Seluruh staff recording Jurusan Teknik Elektro Universitas Brawijaya.
- Ridha, Risto, dan Alin, sebagai teman – teman seperjuangan di konsentrasi Rekayasa Komputer Angkatan 2012 Teknik Elektro Universitas Brawijaya.
- Sahabat "Voltage" angkatan 2012 atas segala bantuan dan kebersamaan yang telah diberikan selama 4,5 tahun ini.
- Keluarga Besar Laboratorium Informasi dan Komputer atas segala pengalaman, kebersamaan dan bantuan selama menjadi asisten.
- Keluarga Besar RisTIE yang telah memberi pengalaman baru bagi saya.

- Teman- teman “Not Jones”, yang selalu sedih dalam kebahagiaanku, dan tertawa dalam kesukaranku.
- Keluarga Besar “GHF 97”, karena telah menjadi rumah bagi penulis selama menjalani studi di kota Malang.
- Keluarga Besar PMK Yehezkiel, yang telah menjadi tempat saya bertumbuh dalam iman.
- Keluarga PMK Yehezkiel 2012, yang telah menjadi sahabat dan menaruh kasih setiap waktu, dan menjadi saudara dalam kesukaran.
- Keluarga Besar Mahasiswa Teknik Elektro Universitas Brawijaya,
- Seluruh teman-teman serta semua pihak yang tidak mungkin bagi penulis untuk mencantumkan satu-persatu, terimakasih banyak atas bantuan dan dukungannya

Pada akhirnya, penulis menyadari bahwa skripsi ini masih belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun. Penulis berharap semoga skripsi ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi serta bagi masyarakat.

Malang, Januari 2017

Penulis

DAFTAR ISI

DAFTAR TABEL

Judul Halaman

DAFTAR GAMBAR

Judul Halaman

DAFTAR LAMPIRAN

Judul Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Dunia teknologi berkembang dengan sangat pesat. Perkembangan ini didukung dengan kemajuan dunia digital yang sangat pesat. Seiring dengan perkembangan teknologi yang semakin pesat, kebutuhan akan komputasi yang tinggi juga semakin meningkat. Dengan kecepatan komputasi yang tinggi, maka dapat diperoleh waktu proses komputasi yang lebih rendah pula, sehingga kita dapat menyelesaikan suatu permasalahan komputasi dengan waktu yang lebih singkat. Namun saat ini, kecepatan komputasi yang tinggi biasanya di iringi dengan harga perangkat mikroprosesor yang tinggi pula. Ini lah yang menyebabkan timbul ide untuk menggunakan beberapa prosesor dengan kecepatan rendah secara bersama-sama untuk mendapat hasil komputasi yang tinggi. Penggunaan prosesor secara bersama-sama untuk menyelesaikan suatu masalah tertentu ini disebut dengan komputasi paralel.

Platform komputer yang digunakan, komputer paralel (*parallel computer*), bisa berupa komputer dengan beberapa prosesor mapupun beberapa komputer yang terhubung dengan cara tertentu. Pendekatan tersebut seharusnya mampu meningkatkan kemampuan komputer secara signifikan. Maksudnya adalah p prosesor/komputer mampu menghasilkan hingga p kali kecepatan komputer dengan satu prosesor/komputer. Berapa pun kecepatan prosesor/komputer itu dengan harapan problem yang ad dapat diselesaikan denganwaktu 1/p. Tentu saja, situasi tersebut merupakan situasi ideal yang jarang sekali terjadi. Seringkali, problem tidak dapat dipecah menjadi bagian-bagian kecil. Selain itu, diperlukan interaksi antara masing-masing bagian, baik untuk transfer data atau sinkronisasi. Namun seberapa jauh pengingkatan kecepatan dapat dicapai bergantung pada masalah dan peran parallelisme. Satu hal yang menyebabkan kemampuan komputer paralel menjadi tak terbatas adalah peningkatan kecepatan eksekusi suatu proseso rsecara berkelanjutan akan meningkatkan pula kecepatan komputer paralel. Dengan demikian, akan selalu ada problem dengan tantangan besar yang tak bisa diselesaikan dalam waktu yang memadai menggunakan komputer saat ini

Komputasi paralel tidak hanya dapat digunakan dengan perangkat keras yang biasa digunakan pada komputer personal kita (motherboard, prosesor, ram, harddisk), tapi juga dapat menggunakan *open-source single board computer*, seperti Orange Pi. Orange Pi merupakan sebuah mini PC berbasis open-source yang sudah memiliki spesifikasi yang

cukup untuk menjalankan fungsinya sebagai komputer. Orange Pi sendiri memiliki ukuran yang lebih ringkas jika dibandingkan dengan komputer pada umumnya dan lebih hemat energi.

Pada klienan komputer paralel, biasanya digunakan sistem Beowulf Cluster, dimana kumpulan komputer umum yang identik yang digunakan secara bersama dalam satu jaringan lokal yang sama dengan *library* dan program yang sudah terinstal sehingga setiap komputer dapat saling berkomunikasi. Hasilnya adalah komputasi paralel dengan performansi yan tinggi dari perangkat komputer yang dapat didapat secara mudah di toko-toko komputer.

Pada penelitian ini akan dicoba untuk mengaplikasikan cluster Beowulf dengan menggunakan 17 buah Orange Pi yang berjalan pada 5 jaringan yang berbeda dengan menggunakan sistem operasi Lubuntu. Diharapkan dapat meningkatkan bandwith pada jaringan sehingga mempercepat waktu komputasi. Algoritma paralel yang digunakan akan diimplementasikan pada *Message Passing Interface* atau MPI. MPI adalah bahasa independen protokol komunikasi yang digunakan program paralel pada komputer.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang sudah dijelaskan, maka rumusan masalah yang disusun adalah sebagai berikut:

1. Bagaimana merancang jaringan pada perangkat Orange Pi?
2. Bagaimana mengatur konfigurasi pada setiap Orange Pi agar dapat saling berkomunikasi?
3. Apakah ada perbedaan kecepatan komputasi jika dibandingkan dengan metode jaringan yang sama?

1.3 Batasan Masalah

Berdasarkan rumusalan masaklah tersebut di atas, batasan masalah yang dirumuskan adalah sebagi berikut

1. Daya terpakai proses komputasi tidak dihitung.
2. Proses komputasi paralel dimplementasikan pada OpenMPI.
3. Pengujian dilakukan dengan menggunakan 1 aplikasi perkalian matriks.

1.4 Tujuan

Berdasarkan rumusan masalah yang tertera di atas, maka tujuan peneltian skripsi ini adalah sebagai berikut:

1. Mengetahui konfigurasi yang tepat untuk sistem jaringan yang diinginkan
2. Mengetahui seberapa besar peningkatan performa dengan menggunakan komputer paralel *cluster*.

3. Mengetahui seberapa besar perubahan *traffic* dengan menggunakan sistem jaringan bertingkat.

1.5 Manfaat

Manfaat pemecahan masalah dalam skripsi ini adalah sebagai berikut:

1. Untuk kepentingan ilmiah, konfigurasi jaringan ini dapat digunakan untuk penelitian selanjutnya yang membutuhkan komputasi paralel yang lebih kompleks.

BAB II
TINJAUAN PUSTAKA

2.1 Komputasi Paralel

Komputasi paralel adalah salah satu teknik melakukan komputasi secara bersamaan dengan memanfaatkan beberapa komputer independen secara bersamaan. Ini umumnya diperlukan saat kapasitas yang diperlukan sangat besar, baik karena harus mengolah data dalam jumlah besar (di industri keuangan, bioinformatika, dan lain lain) ataupun karena tuntutan proses komputasi yang banyak. Kasus kedua umum ditemui di kalkulasi numerik untuk menyelesaikan persamaan matematis di bidang fisika (fisika komputasi), kimia (kimia komputasi) dan lain lain.

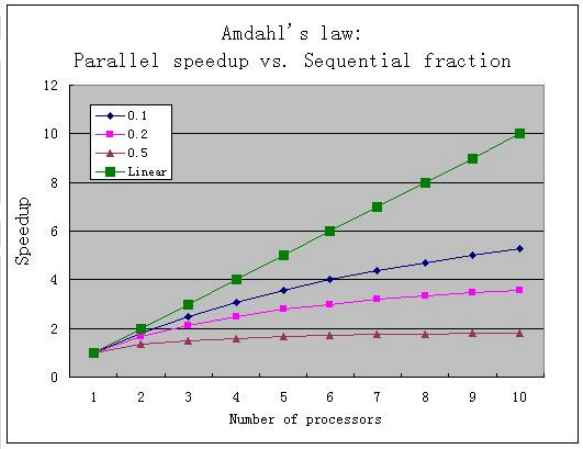
Untuk melakukan aneka jenis komputasi paralel ini diperlukan infrastruktur mesin paralel yang terdiri dari banyak komputer yang dihubungkan dengan jaringan dan mampu bekerja secara paralel untuk menyelesaikan satu masalah. Untuk itu diperlukan aneka perangkat lunak pendukung yang biasa disebut sebagai *middleware* yang berperan untuk mengatur distribusi pekerjaan antar node dalam satu mesin paralel. Selanjutnya pemakai harus membuat pemograman paralel untuk merealisasikan komputasi. Tidak berarti dengan mesin paralel semua program yang dijalankan diatasnya otomatis akan diolah secara parallel.

Di dalam komputasi paralel ada yang dinamakan dengan pemograman paralel. Pemograman paralel adalah teknik pemograman komputer yang memungkinkan eksekusi perintah/operasi secara bersamaan (komputasi paralel), baik dalam komputer dengan satu (prosesor tunggal) ataupun banyak (prosesor ganda dengan mesin paralel) CPU. Bila komputer yang digunakan secara bersamaan tersebut dilakukan oleh komputer-komputer terpisah yang terhubung dalam suatu jaringan komputer lebih sering istilah yang digunakan adalah sistem terdistribusi (*distributed computing*).

Tujuan utama dari pemograman paralel adalah untuk meningkatkan performa komputasi. Semakin banyak hal yang bisa dilakukan secara bersamaan (dalam waktu yang sama), semakin banyak pekerjaan yang bisa diselesaikan. Analogi yang paling gampang adalah, bila anda dapat merebus air sambil memotong-motong bawang saat anda akan memasak, waktu yang anda butuhkan akan lebih sedikit dibandingkan bila anda mengerjakan hal tersebut secara berurutan (serial). Atau waktu yg anda butuhkan memotong bawang akan lebih sedikit jika anda kerjakan berdua.

Performa dalam pemograman paralel diukur dari berapa banyak peningkatan kecepatan (*speed-up*) yang diperoleh dalam menggunakan tehnik paralel. Secara informal, bila anda memotong bawang sendirian membutuhkan waktu 1 jam dan dengan bantuan teman, berdua anda bisa melakukannya dalam 1/2 jam maka anda memperoleh peningkatan kecepatan sebanyak 2 kali.

Namun ada limitasi dalam usaha membuat suatu program komputer berjalan lebih efisien melalui peningkatan kecepatan. Hukum yang menetapkan batasan ini dikenal sebagai Hukum Amdahl.



Gambar 1 Hukum Amdahl

Ide dari hukum Amdahl ini adalah bahwa anda hanya akan bisa meningkatkan efisiensi program komputer anda, sebatas pada bagian tertentu dari program tersebut yang dapat di paralelkan. Sementara bagian yang memang harus dilaksanakan secara berurutan, akan menjadi penentu performa akhir. Semakin banyak prosesor akan meningkatkan kecepatan, tetapi pada suatu titik jumlah prosesor akan mencapai jenuh.

Yang menjadi focus utama dalam merencanakan solusi menggunakan multiprosesor adalah perkiraan akan seberapa besar peningkatan kecepatan multiprosesor dalam menyelesaikan suatu *problem*. Perbandingan tersebut dapat dilakukan dengan membandingkan algoritma sekuensial terbaik pada sistem prosesor tunggal dibandingkan dengan algortima paralel pada multiprosesor yang sedang di teliti. Faktor Pemercepat, S(p), adalah ukuran relative performa yang didefinisikan sebagai berikut

$$S(p) = \frac{waktu\ proses\ menggunakan\ n\ sistem\ prosesor\ tunggal}{waktu\ proses\ menggunakan\ multiprosesor\ dengan\ p\ prosesor}$$

Peningkatan kecepatan komputasi paralel dibatasi oleh 2 hal. Pertama, jika sebuah rutin tidak dapat dipecah menjadi unit yang lebih kecil dan harus dijalankan secara sekuensial di suatu prosesor. Kedua, peningkatan kecepatan komputasi paralel dibatasi performa

perangkat keras, penjadwalan system operasi, konfigurasi lingkungan komputasi dan *overhead* rutin program karena komunikasi dan sinkronisasi antar proses.

2.2 Komputer Cluster

Konsep klienan banyak komputer personal yang saling terkoneksi sebagai sebuah platform komputasi paralel semakin matang dengan ketika komputer personal menjadi sangat kuat dan tersedia di mana-mana. *Workstation*, yaitu komputer yang secara khusus ditujukan kearah klienan laboratorium, telah diganti dengan PC biasa, dan perbedaan antara *workstation* dan PC dalam hal peneltian mulai berkurang. Istilah "*network of workstations*' disisngkat dengan lebih sederhana dengan "cluster" komputer, dan klienan komputer di suatu cluster secara kolektif pada sebuah permasalahan tunggal disebut dengan istilah cluster computing.

Komputer Cluster terbagi ke dalam beberapa kategori:

- *Cluster* untuk ketersediaan yang tinggi (*High-availability cluster*)

High-availability cluster, yang juga sering disebut sebagai Failover Cluster pada umumnya diimplementasikan untuk tujuan meningkatkan ketersediaan layanan yang disediakan oleh *cluster* tersebut. Elmen *cluster* akan bekerja dengan memiliki node-node redundan, yang kemudian digunakan untuk menyediakan layanan saat salah satu elemen *cluster* mengalami kegagalan. Ukuran yang paling umum dari kategori ini adalah dua node, yang merupakan syarat minimum untuk melakukan redundansi. Implementasi *cluster* jenis ini akan mencoba untuk menggunakan redundansi komponen *cluster* untuk menghilangkan kegagalan di satu titik (*Single Point of Failure*).

Ada beberapa implementasi komersial dari sistem *cluster* kategori ini, dalam beberapa sistem operasi. Meski demikian, proyek Linux-HA adalah salah satu paket yang paling umum digunakan untuk sistem operasi GNU/Linux.

Dalam keluarga sistem operasi Microsoft Windows NT, sebuah layanan yang disebut dengan *Microsoft Cluster Service* (MSCS) dapat digunakan untuk menyediakan *cluster* kategori ini. MSCS ini diperbarui lagi dan telah diintegrasikan dalam *Windows 2000 Advanced Server* dan *Windows 2000 Datacenter Server*, dengan nama M*icrosoft Clustering Service*. Dalam Windows Server 2003, *Microsoft Clustering Service* ini ditingkatkan lagi kinerjanya.

- *Cluster* untuk pemerataan beban komputasi (*Load-balancing clusters*)

Cluster kategori ini beroperasi dengan mendistribusikan beban pekerjaan secara merata melalui beberapa node yang bekerja di belakang (*back-end node*). Umumnya *cluster* ini akan dikonfigurasikan sedemikian rupa dengan beberapa *front-end load-balancing* redundan.

Karena setiap elemen dalam sebuah *cluster load-balancing* menawarkan layanan penuh, maka dapat dikatakan bahwa komponen *cluster* tersebut merupakan sebuah *cluster* aktif/*cluster* HA aktif, yang bisa menerima semua permintaan yang diajukan oleh klien.

- *Cluster* hanya untuk komputasi (Compute clusters)

Seringnya, klienan utama *cluster* komputer adalah untuk tujuan komputasi, ketimbang penanganan operasi yang berorientasi I/O seperti layanan Web atau basis data. Sebagai contoh, sebuah *cluster* mungkin mendukung simulasi komputasional untuk perubahan cuaca atau tabrakan kendaraan. Perbedaan utama untuk kategori ini dengan kategori lainnya adalah seberapa eratkah penggabungan antar node-nya. Sebagai contoh, sebuah tugas komputasi mungkin membutuhkan komunikasi yang sering antar node--ini berarti bahwa *cluster* tersebut menggunakan sebuah jaringan terdedikasi yang sama, yang terletak di lokasi yang sangat berdekatan, dan mungkin juga merupakan node-node yang bersifat homogen. Desain *cluster* seperti ini, umumnya disebut juga sebagai Beowulf Cluster. Ada juga desain yang lain, yakni saat sebuah tugas komputasi hanya menggunakan satu atau beberapa node saja, dan membutuhkan komunikasi antar-node yang sangat sedikit atau tidak ada sama sekali. Desain *cluster* ini, sering disebut sebagai "*Grid*". Beberapa compute cluster yang dihubungkan secara erat yang didesain sedemikian rupa, umumnya disebut dengan "*Supercomputing*". Beberapa perangkat lunak *Middleware* seperti MPI atau Parallel Virtual Machine (PVM) mengizinkan program compute clustering agar dapat dijalankan di dalam *cluster-cluster* tersebut.

- *Grid computing*

Grid pada umumnya adalah compute cluster, tetapi difokuskan pada *throughput* seperti utilitas perhitungan ketimbang menjalankan pekerjaan-pekerjaan yang sangat erat yang biasanya dilakukan oleh *Supercomputer*. Seringnya, *Grid* memasukkan sekumpulan komputer, yang bisa saja didistribusikan secara geografis, dan kadang diurus oleh organisasi yang tidak saling berkaitan.

Grid computing dioptimalkan untuk beban pekerjaan yang mencakup banyak pekerjaan independen atau paket-paket pekerjaan, yang tidak harus berbagi data yang sama antar pekerjaan selama proses komputasi dilakukan. *Grid* bertindak untuk mengatur alokasi pekerjaan kepada komputer-komputer yang akan melakukan tugas tersebut secara independen. Sumber daya, seperti halnya media penyimpanan, mungkin bisa saja digunakan bersama-sama dengan komputer lainnya, tetapi hasil sementara dari sebuah tugas tertentu tidak akan memengaruhi pekerjaan lainnya yang sedang berlangsung dalam komputer lainnya.

Sebagai contoh *Grid* yang sangat luas digunakan adalah proyek *Folding@home*, yang menganalisis data yang akan digunakan oleh para peneliti untuk menemukan obat untuk beberapa penyakit seperti Alzheimer dan juga kanker. Proyek lainnya, adalah *SETI@home*, yang merupakan proyek *Grid* terdistribusi yang paling besar hingga saat ini. Proyek SETI@home ini menggunakan paling tidak 3 juta komputer rumahan yang berada di dalam komputer rumahan untuk menganalisis data dari teleskop radio observatorium Arecibo (*Arecibo Observatory radiotelescope*), mencari bukti-bukti keberadaan makhluk luar angkasa. Dalam dua kasus tersebut, tidak ada komunikasi antar node atau media penyimpanan yang digunakan bersama-sama.

2.3 Cluster Beowulf

Sebuah cluster Beowulf adalah cluster komputer dari apa yang biasanya identik, komputer kelas standar yang saling terhubung ke jaringan area lokal kecil dengan library dan program yang diinstal yang memungkinkan pengolahan untuk dibagi di antara mereka. Hasilnya adalah kinerja tinggi komputasi paralel klaster dari perangkat keras komputer pribadi yang murah.

Atribut kunci dalam penamaan Beowulf untuk suatu cluster adalah klienan komponen-komponen yang tersedia luas untuk menghasilkan biaya/kinerja terbaik. Istilah *commodity computer* digunakan untuk menyoroti bahwa komponen PC sekarang dapat di peroleh dimana-mana. Hal ini berlaku pada semua komponen di sekitar prosesor, seperti memori dan antarmuka jaringan.

Nama Beowulf awalnya merujuk pada komputer yang dibangun pada tahun 1994 oleh Thomas Sterling dan Donald Becker di NASA. Nama "Beowulf" berasal dari tokoh utama dalam puisi Inggris Tua yang indah, Beowulf, yang dipilih Sterling karena puisi itu menggambarkan kepahlawannya sebagai "bobot tiga puluh laki-laki dalam genggaman tangannya".

Jacek Radajewski and Douglas Eadline dalam Linux Documentation Project pada tahun 1998 menjelaskan Beowulf Cluster Sebagai berikut:

“Beowulf adalah arsitektur multi-komputer yang dapat digunakan untuk komputasi paralel. Ini adalah sebuah sistem yang biasanya terdiri dari satu server node, dan satu atau lebih node klien terhubung melalui Ethernet atau jaringan lainnya. Ini adalah sistem yang dibangun menggunakan komponen komoditas, seperti PC yang mampu menjalankan sistem operasi Unix, dengan adapter Ethernet standar, dan switch. Beowulf tidak mengandung komponen perangkat keras kustom dan jarang diproduksi. Beowulf juga menggunakan software komoditas seperti FreeBSD, Linux atau sistem operasi Solaris, *Parallel Virtual*

Machine (PVM) dan *Message Passing Interface* (MPI). Server mengontrol seluruh cluster dan melayani file ke klien. Server juga menjadi konsol cluster dan gateway ke dunia luar. Beowulf mesin besar mungkin memiliki lebih dari satu server node, dan node lain mungkin didedikasikan untuk tugas-tugas tertentu, misalnya konsol atau stasiun pemantauan. Dalam kebanyakan kasus node klien dalam sistem Beowulf bodoh, semakin bodoh semakin baik. Node dikonfigurasi dan dikendalikan oleh server node, dan hanya melakukan apa yang diperintahkan untuk dilakukan. Dalam konfigurasi klien yang tidak menggunakan disk, node klien bahkan tidak tahu alamat IP atau namanya sampai server mengatakan itu.

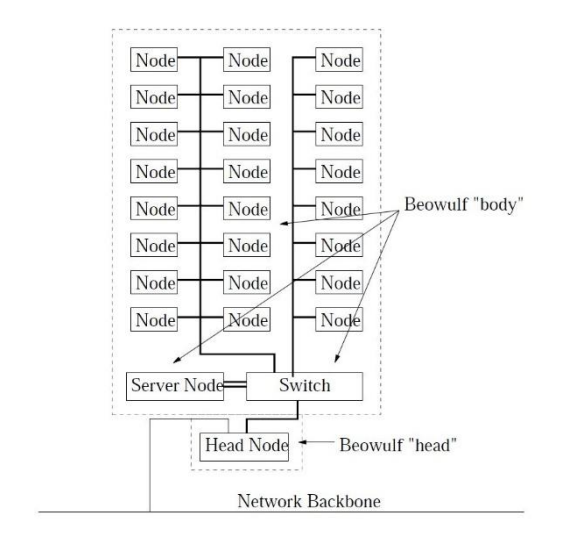
Salah satu perbedaan utama antara Beowulf dan Cluster Workstation (KK) adalah bahwa Beowulf berperilaku lebih seperti mesin tunggal daripada banyak workstation. Dalam kebanyakan kasus node klien tidak memiliki keyboard atau monitor, dan diakses hanya melalui remote login atau mungkin terminal serial. Beowulf node dapat dianggap sebagai paket CPU + memori yang dapat dipasang ke cluster, seperti CPU atau memori modul dapat dipasang ke motherboard.

Beowulf bukan paket software khusus, topologi jaringan baru, atau kernel hack terbaru. Beowulf adalah teknologi clustering komputer untuk membentuk paralel, superkomputer virtual. Meskipun ada banyak paket perangkat lunak seperti modifikasi kernel, PVM dan library MPI, dan alat-alat konfigurasi yang membuat arsitektur Beowulf lebih cepat, lebih mudah untuk mengkonfigurasi, dan jauh lebih bermanfaat, seseorang dapat membangun sebuah mesin kelas Beowulf menggunakan distribusi Linux standar tanpa tambahan perangkat lunak. Jika Anda memiliki dua komputer jaringan yang bsetidaknya berbagi file sistem /home melalui NFS, dan saling percaya untuk mengeksekusi remote shells (rsh), maka bisa dikatakan bahwa Anda memiliki dua nodel mesin Beowulf secara sederhana."

Robert G. Brown dalam bukunya, *Engineering a Beowulf-Style Compute Cluster*, menjelaskan bahwa Cluster Beowulf adalah kumpulan dari komputer yang terhubung dalam jaringan dengan karakteristik berikut:

- Node yang ada didedikasikan untuk Beowulf dan tidak melayani tujuan lain.
- Jaringan di mana node berada adalah didedikasikan untuk Beowulf dan tidak melayani tujuan lain.
- Node berupa M2COTS (*Mass Market Commodity-Off-The-Shelf*) komputer. Bagian penting dari pengertian Beowulf (yang membedakannya dari, misalnya produksi vendor massively paralel prosesor – MPP -system) adalah bahwa node yang digunakan diproduksi massal, tersedia dari "rak", dan karenanya relatif murah.

- Perangkat jaringannya juga merupakan COTS (*Commodity-Off-The-Shelf*), setidaknya sejauh itu bisa diintegrasikan dengan M2COTS komputer dan karenanya harus terhubung melalui bus (misalnya PCI) standar. Sekali lagi, ini untuk membedakannya dari sistem MPP yand di produksi vendor di mana jaringan dan CPU dibuat secara khusus dan terpadu dengan biaya yang sangat tinggi.
- Semua node dijalankan menggunakan perangkat lunak open source.
- Cluster yang dihasilkan digunakan untuk *High Performance Computing* (HPC, juga disebut "*parallel supercomputing*" dan nama lain).



Gambar 2 Cluster Beowulf

Tidak ada bagian dari perangkat lunak mendefinisikan cluster sebagai Beowulf. Cluster Beowulf biasanya dijalankan pada sistem operasi mirip Unix, seperti BSD, Linux, atau Solaris, biasanya dibangun dari perangkat lunak bebas dan open source. Umumnya digunakan pengolahan library paralel mencakup Message Passing Interface (MPI) dan *Parallel Virtual Machine* (PVM). Kedua hal ini memungkinkan programmer untuk membagi tugas di antara sekelompok komputer jaringan, dan mengumpulkan hasil pengolahan. Contoh perangkat lunak MPI adalah OpenMPI atau MPICH. Ada tambahan implementasi MPI yang tersedia

Pada 2014 sistem Beowulf beroperasi di seluruh dunia, terutama dalam mendukung komputasi ilmiah.

2.4 Message Passing Interface (MPI)

MPI (*Message Passing Interface*) adalah spesifikasi API (*Application Programming Interface*) yang memungkinkan terjadinya komunikasi antar komputer pada network dalam usaha untuk menyelesaikan suatu tugas. Paradigma *Message - Passing* dengan implementasi MPI memberikan suatu pendekatan yang unik dalam membangun suatu software dalam

domain fungsi tertentu, yang dalam hal ini pada lingkungan sistem terdistribusi, sehingga memberikan kemampuan pada produk software yang dibangun diatas *middleware* tersebut untuk dapat mengeksploitasi kemampuan jaringan komputer dan komputasi secara paralel.

MPI adalah standar *interface* dari model message - passing yang didefenisikan oleh sebuah grup yang terdiri dari 60 orang yang berasal dari 40 organisasi baik vendor komersil maupun dari kalangan peneliti akademisi yang berada di Amerika Serikat dan Eropa. Dalam grup tersebut mereka mencoba merumuskan dan membuat sebuah "*standard by consensus*" untuk pustaka message - passing yang dapat digunakan dalam komputasi parallel.

2.5 Orange Pi

Orange Pi adalah sebuah perangkat komputer seukuran kartu kredit. Sistem operasinya dipasang pada sebuah SD Flash Card yang menjadikannya sangat mudah untuk diganti dan ditukar. Potensinya luar biasa, dari yang sudah maupun belum pernah dieksplorasi, tetapi telah diuji sebagai multimedia player dengan kemampuan streaming, sebagai perangkat mesin game, Internet dan sebagai mainboard pengembangan perangkat keras. Orange Pi merupakan sebuah mini PC berbasis ARM, yang menjadikan perangkat ini sebagai perangkat yang hemat energi.

2.6 Router

Router berfungsi sebagai penghubung 2 jaringan atau lebih untuk meneruskan data dari satu jaringan ke jaringan lainnya. Router berbeda dengan switch. Switch merupakan penghubung beberapa alat untuk membentuk *suatu Local Area Network* (LAN). Sebagai ilustrasi perbedaan fungsi dari router dan switch, switch merupakan suatu jalan, sedangkan router merupakan penghubung antar jalan. Masing-masing rumah berada pada jalan yang memiliki alamat dalam suatu urutan tertentu. Dengan cara yang sama, switch menghubungkan berbagai macam alat, dimana masing-masing alat memiliki alamat IP sendiri pada sebuah LAN.

Router sangat banyak digunakan dalam jaringan berbasis teknologi protokol TCP/IP, dan router jenis itu disebut juga dengan IP Router. Selain IP Router, ada lagi AppleTalk Router, dan masih ada beberapa jenis router lainnya. Internet merupakan contoh utama dari sebuah jaringan yang memiliki banyak router IP. Router dapat digunakan untuk menghubungkan banyak jaringan kecil ke sebuah jaringan yang lebih besar, yang disebut dengan internetwork, atau untuk membagi sebuah jaringan besar ke dalam beberapa subnetwork untuk meningkatkan kinerja dan juga mempermudah manajemennya. Router juga kadang digunakan untuk mengoneksikan dua buah jaringan yang menggunakan media yang berbeda (seperti halnya router wireless yang pada umumnya selain ia dapat

menghubungkan komputer dengan menggunakan radio, ia juga mendukung penghubungan komputer dengan kabel UTP), atau berbeda arsitektur jaringan, seperti halnya dari Ethernet ke Token Ring.

2.7 Sistem Operasi Lubuntu

Gambar 3 Logo Lubuntu

Lubuntu adalah sebuah proyek yang dimaksudkan untuk menghasilkan suatu turunan resmi dari sistem operasi Ubuntu yang "lebih ringan, lebih sedikit menggunakan sumber daya dan efisien energi", menggunakan lingkungan desktop LXDE.

Desktop LXDE menggunakan window manager Openbox dan dimaksudkan untuk menjadi sistem yang rendah persyaratan, menggunakan RAM sedikit untuk netbook, mobile devices dan PC (komputer) tua. Dalam tugas ini Lubuntu akan bersaing dengan Xubuntu.

Nama Lubuntu adalah kombinasi dari LXDE dan Ubuntu. LXDE adalah singkatan dari Lightweight X11 Desktop Environment, sedang Ubuntu berarti "perikemanusian terhadap sesama manusia" dalam bahasa Zulu dan bahasa Xhosa.

Yang digunakan pada semua Orange PI adalah Lubuntu versi 14.04 LTS yang di unduh dari website resmi Orange Pi.

2.8 SSH (*Secure Shell*)

SSH merupakan protocol di dalam jaringan komputer yang berfungsi untuk membantu klien komputer di dalam bertukar data, transfer data, serta mengendalikan komputer jarak jauh secara lebih aman. SSH menyediakan keamanan yang lebih baik dibandingkan protocol Telnet dan FTP. Didalam SSH terjadi proses enkripsi, otorisasi, dan otentikasi. SSH menjanjikan keamanan pada jaringan komputer, terutamanya pada kasus serangan MITM (*Man In The Middle*). SSH menggunakan port 22 untuk koneksi di dalam jaringan komputer.

Pada SSH terdapat SSH *server*, SSH klien, dan SSH *user*. SSH *server* merupakan komputer yang akan dieksekusi (dikendalikan jarak jauh termasuk juga untuk proses tukar dan transfer data). SSH Klien merupakan klien yang menggunakan SSH untuk melakukan otentikasi dan kendali ke komputer yang menjadi SSH server. Baik SSH server dan SSH klien keduanya diimplementasikan ke dalam aplikasi. Pada sistem operasi Linux Anda dapat menggunakan SSH server dan SSH klien secara mudah karena tersedia secara default

ataupun melalui repository. SSH dapat digunakan pada jaringan public (internet) maupun jaringan private (intranet), baik wired maupun wireless. SSH user merupakan klien yang melakukan kendali ke SSH server melalui SSH klien.

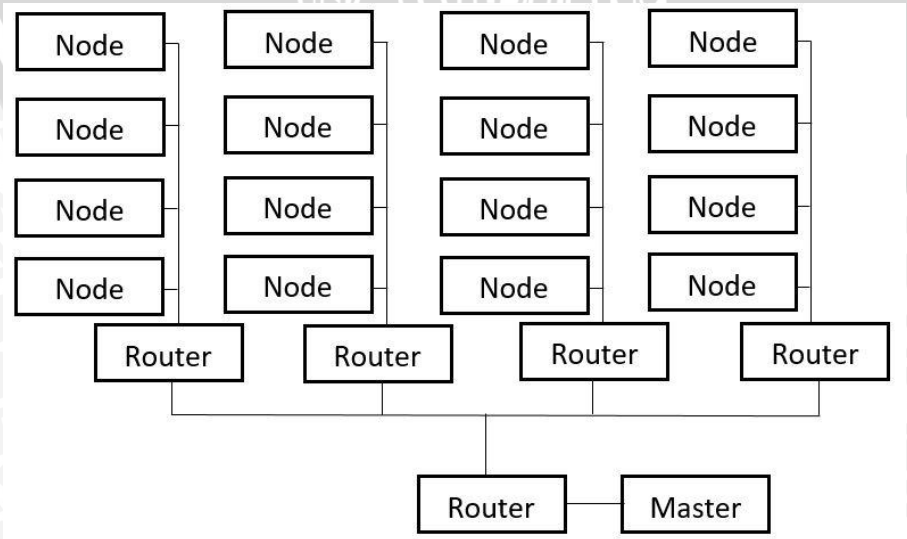
Gambar 4 Cara Kerja SSH

BAB III
METODE PENELITIAN

Penyusunan skripsi ini didasarkan pada masalah yang bersifat aplikatif, yaitu perencanaan dan perealisasian alat agar dapat bekerja sesuai dengan yang direncanakan dengan mengacu pada rumusan masalah. Oleh karena itu dibutuhkan suatu metode penelitian agar perencanaan dan perealisasian alat dapat dilakukan.

Pada Cluster Beowulf yang sering digunakan pada umumnya, sekumpulan node digabung dalam satu jaringan yang sama, dan dihubungkan pada satu jraingan yang sama. Hal ini mempermudah dalam memprogram dan pengonfigurasian system jaringan pada komputer paralel Cluster Beowulf. Namun ketika rutin program paralel dijalankan, bandwith yang tersedia tidak mencukupi traffic yang sedang berlangsung, sehingga proses komputasi berjalan tidak secepat yang diperkirakan karena terhambat pada proses komunikasi antar node.

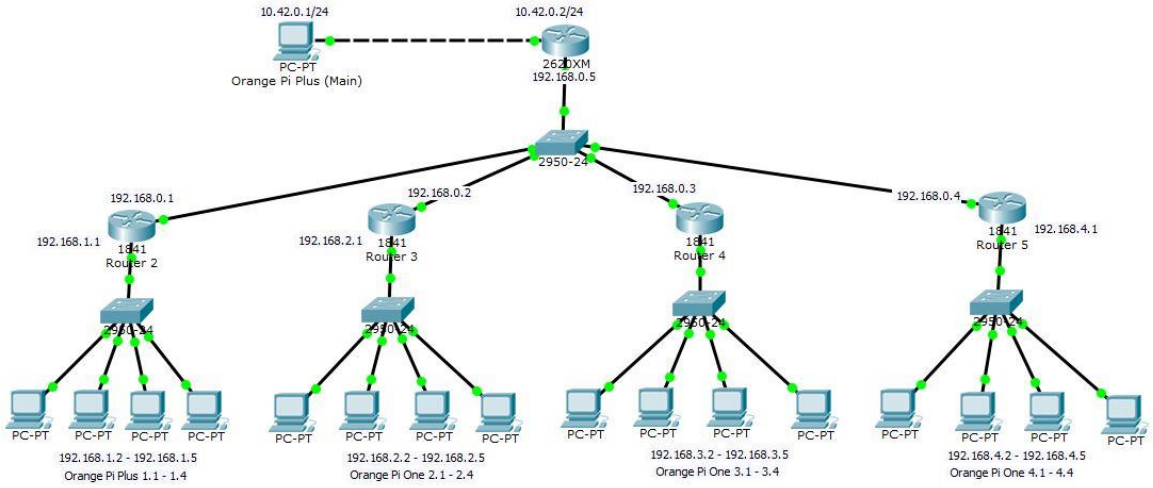
Pada sistem konfigurasi Cluster Beowulf yang di ajukan pada skripsi ini, Cluster Beowulf tidak berjalan pada satu jaringan yang sama, tetapi menggunakan 5 jaringan yang berbeda. Terdapat 4 jaringan untuk node, dan 1 jaringan untuk master seperti pada gambar 5. Dalam setiap 1 jaringan node terdapat 4 buah Orange Pi. Diharapkan dengan menggunakan model jaringan seperti ini, bandwith yang tersedia saat rutin program paralel dijalankan menjadi meningkat, dan proses komputasi pun dapat diselesaikan dengan lebih cepat.



Gambar 6 Model Jaringan Sub-Cluster

3.1 Perancangan dan Pembuatan Sistem

Perancangan sistem ditunjukkan dalam gambar 6 berikut



6. *Gambar 7 Perancangan Sistem*

Perangkat yang digunakan diuraikan seperti berikut:

3.1.1 Perangkat Keras

1. 5 unit Orange Pi PC (CPU H3 Quad-core Cortex-A7, Memori 1GB DDR3), 1 sebagai Main, 4 sebagai bagian Cluster
2. 12 Unit Orange Pi One (CPU H3 Quad-core Cortex-A7, Memori 512MB DDR3)
3. 17 Unit MicroSD Card 8 GB sebagai media penyimpanan
4. 5 Router TP-Link TL-R470T+
5. Kabel UTP Cat 5e
6. Power Supply

3.1.2 Perangkat Lunak

1. GNU/Linux Lubuntu 14.04 LTS
2. OpenMPI
3. OpenSSH
4. GCC Compiler

Tahapan perancangan sistem sub-cluster adalah sebagai berikut:

1. Merangkai peralatan yang digunakan.
2. Melakukan instalasi sistem Operasi Lubuntu 14.04 pada setiap Orange Pi.
3. Melakukan konfigurasi IP dan Routing pada setiap router yang digunakan.
4. Memasang semua paket perangkat lunak yang digunakan (SSH, MPI)
5. Melakukan konfigurai hosts dan hostname pada setiap perangkat.
6. Pengujian PING, apakah perangkat sudah saling terhubung atau tidak.

Konfigurasi dilakukan pada ke lima router. Setiap router yang digunakan meiliki konfigurasi yang berbeda, karena setiap router meimilki jaringan yang berbeda. Konfigurasi pada setiap router di jelaskan sebagai berikut.

3.1.3 Konfigurasi IP.

Konfigurasi IP statik dilakukan dengan menggunakan MAC pada setiap perangkat.

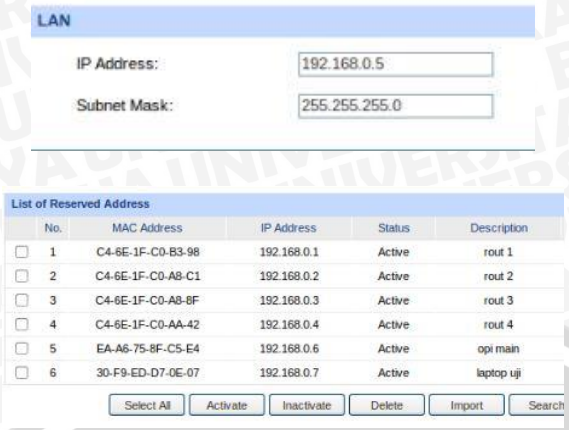
3.1.3.1 Router 1

Router 1 merupakan router yang menyambungkan antara master dengan router lainnya. Jaringan local antara router dengan master menggunakan alamat 10.42.0.0/24, sedangkan jaringan lokal antar ke lima router adalah 192.168.0.0/24.

a. 1 port WAN dan 4 port LAN. Port WAN disambung ke Orange Pi Master, dan ke empat port LAN disaimbungkan dengan port WAN pada ke empat router lainnya, sebagai berikut
   - Port WAN ke Master
   - Port LAN 1 ke Router 2
   - Port LAN 2 ke Router 3
   - Port LAN 3 ke Router 4
   - Port LAN 4 ke Router 5

b. Konfigurasi IP pada router 1
   - IP Master 10.42.0.2
     - IP Router 10.42.0.1
     - IP Router 1 192.168.0.5
     - IP Router 2 192.168.0.1
     - IP Router 3 192.168.0.2
     - IP Router 4 192.168.0.3
     - IP Router 5 192.168.0.4

Konfigurasi IP dilakukan dengan menggunakan MAC Address tiap perangkat.



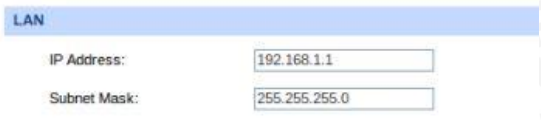
Gambar 8 Konfigurasi IP Router 1

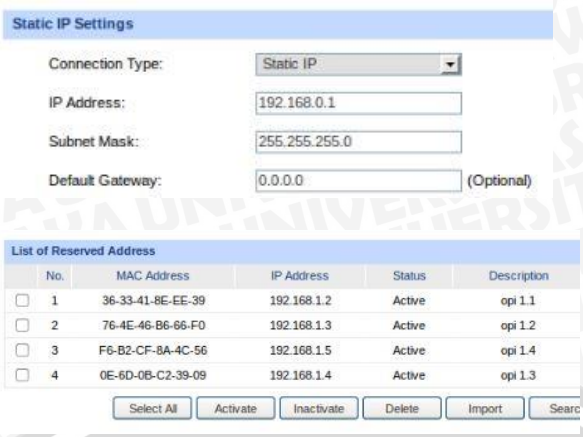
3.1.3.2 Router 2

Router 2 merupakan router yang menyambungkan antara master, router 1, dan jaringan 1. Jaringan 1 menggunakan alamat 192.168.1.0/24.

a. 1 port WAN dan 4 port LAN. Port WAN disambungkan ke Port LAN 1 pada Router 1, dan ke empat port LAN disambungkan dengan 4 Orange Pi untuk menjadikan sub cluster 1, sebagai berikut
   - Port WAN ke Router 1
   - Port LAN 1 ke Orange Pi 1.1
   - Port LAN 2 ke Orange Pi 1.2
   - Port LAN 3 ke Orange Pi 1.3
   - Port LAN 4 ke Orange Pi 1.4

b. Konfigurasi IP pada Router 2
   - IP Router 2 192.168.0.1
   - IP Router 2 192.168.1.1
   - IP Orange Pi 1.1 192.168.1.2
   - IP Orange Pi 1.2 192.168.1.3
   - IP Orange Pi 1.3 192.168.1.4
   - IP Orange Pi 1.4 192.168.1.5





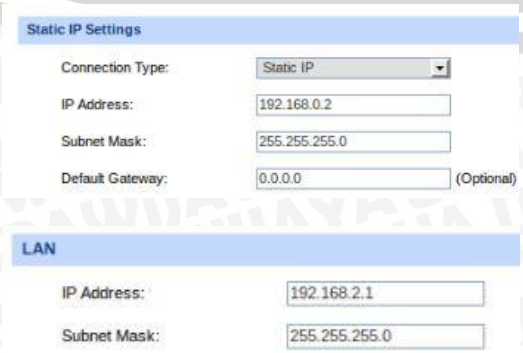
Gambar 9 Konfigurasi IP Router 2

3.1.3.3 Router 3

Router 3 merupakan router yang menyambungkan antara master, router 1, dan jaringan 2. Jaringan 2 menggunakan alamat 192.168.2.0/24.

a. 1 port WAN dan 4 port LAN. Port WAN disambungkan ke Port LAN 1 pada Router 1, dan ke empat port LAN disambungkan dengan 4 Orange Pi untuk menjadikan sub cluster 2, sebagai berikut
   - Port WAN ke Router 1
   - Port LAN 1 ke Orange Pi 2.1
   - Port LAN 2 ke Orange Pi 2.2
   - Port LAN 3 ke Orange Pi 2.3
   - Port LAN 4 ke Orange Pi 2.4

b. Konfigurasi IP pada Router 3
   - IP Router 3 192.168.0.2
   - IP Router 3 192.168.2.1
   - IP Orange Pi 2.1 192.168.2.2
   - IP Orange Pi 2.2 192.168.2.3
   - IP Orange Pi 2.3 192.168.2.4
   - IP Orange Pi 2.4 192.168.2.5



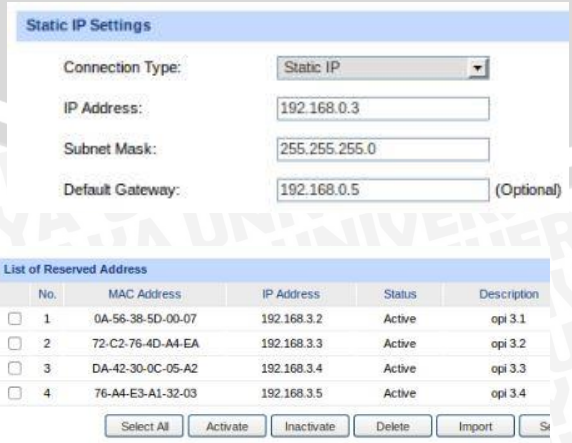
Gambar 10 Konfigurasi IP Router 3

3.1.3.4 Router 4

Router 4 merupakan router yang menyambungkan antara master, router 1, dan jaringan 3. Jaringan 3 menggunakan alamat 192.168.3.0/24.

a. 1 port WAN dan 4 port LAN. Port WAN disambungkan ke Port LAN 1 pada Router 1, dan ke empat port LAN disambungkan dengan 4 Orange Pi untuk menjadikan sub cluster 3, sebagai berikut
   - Port WAN ke Router 1
   - Port LAN 1 ke Orange Pi 3.1
   - Port LAN 2 ke Orange Pi 3.2
   - Port LAN 3 ke Orange Pi 3.3
   - Port LAN 4 ke Orange Pi 3.4

b. Konfigurasi IP pada Router 2
   - IP Router 4 192.168.0.3
   - IP Router 4 192.168.3.1
   - IP Orange Pi 3.1 192.168.3.2
   - IP Orange Pi 3.2 192.168.3.3
   - IP Orange Pi 3.3 192.168.3.4
   - IP Orange Pi 3.4 192.168.3.5



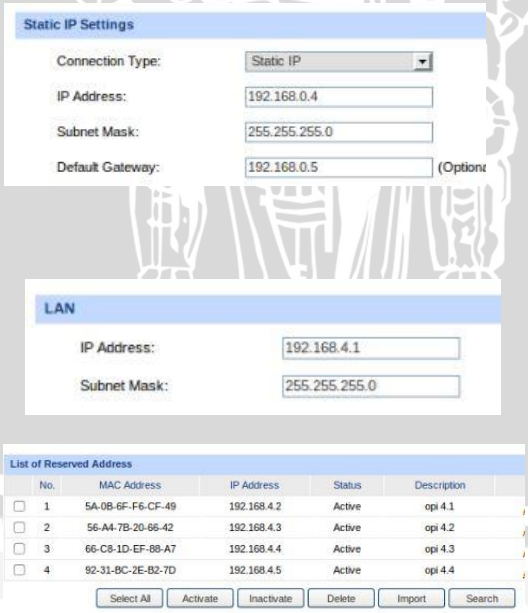
Gambar 11 Konfigurasi IP Router 4

3.1.3.5 Router 5

Router 5 merupakan router yang menyambungkan antara master, router 1, dan jaringan 4. Jaringan 4 menggunakan alamat 192.168.4.0/24.

a. 1 port WAN dan 4 port LAN. Port WAN disambungkan ke Port LAN 1 pada Router 1, dan ke empat port LAN disambungkan dengan 4 Orange Pi untuk menjadikan sub cluster 4, sebagai berikut
   - Port WAN ke Router 1
   - Port LAN 1 ke Orange Pi 4.1
   - Port LAN 2 ke Orange Pi 4.2
   - Port LAN 3 ke Orange Pi 4.3
   - Port LAN 4 ke Orange Pi 4.4

b. Konfigurasi IP pada Router 2
   - IP Router 5 192.168.0.4
   - IP Router 5 192.168.4.1
   - IP Orange Pi 4.1 192.168.4.2
   - IP Orange Pi 4.2 192.168.4.3
   - IP Orange Pi 4.3 192.168.4.4
   - IP Orange Pi 4.4 192.168.4.5



Gambar 12 Konfigurasi IP Router 5

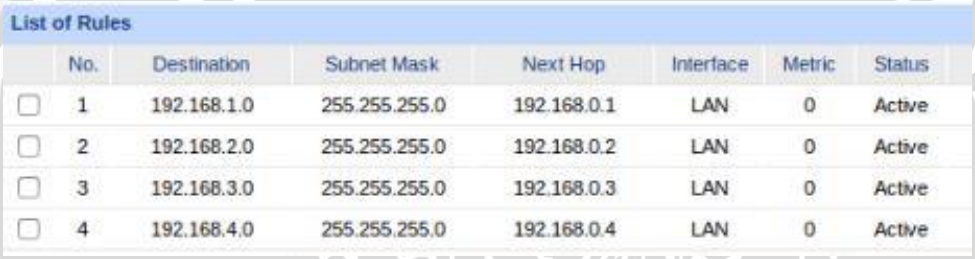
3.1.4 Konfigurasi Static Routing

Routing diperlukan agar setiap sub-cluster saling terhubung, dan router mengetahui jalur yang diperlukan untuk mengirim data.

3.1.4.1 Routing Router 1

Tabel 1 Routing Router 1

No	Destination	Subnet Mask	Next Hoop	Interface
1	192.168.1.0	/24	192.168.0.1	LAN
2	192.168.2.0	/24	192.168.0.2	LAN
3	192.168.3.0	/24	192.168.0.3	LAN
4	192.168.4.0	/24	192.168.0.4	LAN



Gambar 13 Routing Router 1

3.1.4.2 Routing Router 2

Tabel 2 Routing Router 2

No	Destination	Subnet Mask	Next Hoop	Interface
1	10.42.0.0	/24	192.168.0.5	WAN1
2	192.168.2.0	/24	192.168.0.5	WAN1
3	192.168.3.0	/24	192.168.0.5	WAN1
4	192.168.4.0	/24	192.168.0.5	WAN1

Gambar 14 Routing Router 2

3.1.4.3 Routing Router 3

Tabel 3 Routing Router 3

No	Destination	Subnet Mask	Next Hoop	Interface
1	10.42.0.0	/24	192.168.0.5	WAN1
2	192.168.1.0	/24	192.168.0.5	WAN1
3	192.168.3.0	/24	192.168.0.5	WAN1
4	192.168.4.0	/24	192.168.0.5	WAN1


List of Rules

No.	Destination	Subnet Mask	Next Hop	Interface	Metric	Status
1	192.168.1.0	255.255.255.0	192.168.0.5	WAN1	0	Active
2	192.168.3.0	255.255.255.0	192.168.0.5	WAN1	0	Active
3	192.168.4.0	255.255.255.0	192.168.0.5	WAN1	0	Active
4	10.42.0.0	255.255.255.0	192.168.0.5	WAN1	0	Active

Gambar 15 Routing Router 3

3.1.4.4 Routing Router 4

Tabel 4 Routing Router 4

No	Destination	Subnet Mask	Next Hoop	Interface
1	10.42.0.0	/24	192.168.0.5	WAN1
2	192.168.1.0	/24	192.168.0.5	WAN1
3	192.168.2.0	/24	192.168.0.5	WAN1
4	192.168.4.0	/24	192.168.0.5	WAN1


List of Rules

No.	Destination	Subnet Mask	Next Hop	Interface	Metric	Status
1	192.168.1.0	255.255.255.0	192.168.0.5	WAN1	0	Active
2	192.168.2.0	255.255.255.0	192.168.0.5	WAN1	0	Active
3	192.168.4.0	255.255.255.0	192.168.0.5	WAN1	0	Active
4	10.42.0.0	255.255.255.0	192.168.0.5	WAN1	0	Active

Gambar 16 Routing Router 4

3.1.4.5 Routing Router 5

Tabel 5 Routing Router

No	Destination	Subnet Mask	Next Hoop	Interface
1	10.42.0.0	/24	192.168.0.5	WAN1
2	192.168.1.0	/24	192.168.0.5	WAN1
3	192.168.2.0	/24	192.168.0.5	WAN1
4	192.168.3.0	/24	192.168.0.5	WAN1

Gambar 17 Routing Router 5

3.1.5 Instalasi Perangkat Lunak

1. Melakukan instalasi OS (Lubuntu 14.04) pada setiap perangkat Orange Pi.
2. Melakukan pembaharuan daftar paket yang tersedia dan versi terbarunya, dengan menjalankan perintah

```
orangepi@orangepi:~$ sudo apt-get update
```

3. Melakukan instalasi paket-paket yang dibutuhkan, dengan menjalankan perintah

```
orangepi@orangepi:~$ sudo apt-get isntall openmpi-bin openmpi-common openssh-
client openssh-server libopenmpi-dev -y
```

3.1.6 Konfigurasi pada main-node

1. Melakukan konfigurasi file hosts pada main.

```
orangepi@orangepi:~$ cat /etc/hosts
127.0.0.1 localhost orangepi

#mpi cluster setup
10.42.0.1 master
192.168.1.2 1.1
192.168.1.3 1.2
192.168.1.4 1.3
192.168.1.5 1.4
192.168.2.2 2.1
192.168.2.3 2.2
192.168.2.4 2.3
192.168.2.5 2.4
192.168.3.2 3.1
192.168.3.3 3.2
192.168.3.4 3.3
192.168.3.5 3.4
192.168.4.2 4.1
192.168.4.3 4.2
192.168.4.4 4.3
192.168.4.5 4.4
```

2. Melakukan konfigurasi file hosts pada node. Sama, tapi pada node hanya disimpan IP main. IP node lain tidak disimpan.
3. Melakukan Konfigurasi Secure Shell (SSH)

   Konfigurasi SSH diperlukan supaya node dapat saling berkomunikasi dengan lebih mudah.

   • Membuat kunci, dengan perintah

   ```
   orangepi@orangepi:~$ ssh-keygen -t rsa
   ```

   • Menambahkan kunci ke setiap node

   ```
   orangepi@orangepi:~$ ssh-copy-id 1.1
   ```

   • Lakukan SSH ke setiap node

   • Jalankan perintah berikut

   eval `ssh-agent`

   ssh-add ~/.ssh/id_rsa

3.2 Pengujian Sistem

Ada 2 jenis pengujian yang dilakukan pada penelitian ini, yaitu:

3.2.1 Pengujian *Fresh Start*

Pengujian dengan kondisi awal yang sama yaitu baru saja di nyalakan, dan setiap perangkat menjalankan program dengan *size* 100 dalam 4 kondisi, yaitu dengan *number of process* 17, 34, 51 dan 68.

Tabel 6 Rencana Pengujian Fresh Start

Number of process	Waktu (Sec)	
	Jaringan Sama	Sub-Cluster
17		
34		
51		
68		

3.2.2 Pengujian Program dengan beragam *size.*

Pengujian program dengan ukuran 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000 dengan *number of process* 17, 34, 51 dan 68.

Tabel 7 Rencana Pengujian Beragam Size

	Waktu (Sec)			
Ukuran	Np 17	Np 34	Np 51	Np 68
100				
200				
300				
400				
500				
600				
700				
800				
900				
1000				

3.2.3 Program Pengujian

Program pengujian menggunakan algoritma "*manager/worker"* atau pendekatan *"task parallelism".* Idenya adalah bahwa pekerjaan yang perlu dilakukan dapat dibagi oleh "*manager*" menjadi potongan-potongan terpisah dan potongan dapat diberikan ke masing-masing proses "*worker*". Dengan demikian *manager* mengeksekusi algoritma berbeda dari para *worker*, tetapi semua *worker* mengeksekusi algoritma yang sama. Sebagian besar implementasi MPI memungkinkan proses MPI yang akan menjalankan program yang berbeda (*file executable*), tetapi lebih mudah (dan dalam beberapa kasus diperlukan) untuk menggabungkan kode *manager* dan *worker* ke satu program dengan struktur seperti berikut

Kadang-kadang pekerjaan dapat merata dibagi menjadi persis sebanyak *worker* yang ada, tetapi pendekatan yang lebih fleksibel adalah membuat *manager* untuk menjaga sekumpulan unit pekerjaan yang akan dilakukan lebih besar dibanding dengan jumlah *worker*, dan menetapkan kerja baru secara dinamis untuk *worker* setelah tugas mereka telah diselesaikan dan mengirim hasilnya kembali ke *manager*. Pendekatan ini, disebut

penjadwalan diri *(self-scheduling)*, bekerja dengan baik dengan adanya tugas dari berbagai ukuran dan / atau *worker* dari kecepatan yang bervariasi.

Teknik ini digambarkan dengan program paralel untuk mengalikan matriks dengan vektor. Program ini bukan cara yang sangat baik untuk melaksanakan operasi ini, tetapi ini menggambarkan pendekatan komputasi paralel dan cukup sederhana untuk ditampilkan secara keseluruhan. Program ini mengalikan matriks persegi dengan vektor b dan menyimpan hasilnya dalam c. Satuan kerja adalah titik produk individual dari baris dengan vektor b. Dengan demikian pada kode *manager*, dimulai dengan menginisialisasi. *Manager* kemudian mengirimkan unit awal kerja, satu baris untuk setiap *worker*. Tag MPI digunakan pada setiap pesan tersebut untuk mengkodekan nomor baris yang dikirim. Sejak nomor baris mulai dari 0 tapi ingin dipesan 0 sebagai tag dengan arti khusus dari "tidak ada pekerjaan lagi yang harus dilakukan", tag ditetapkan ke posisi baris +1. Ketika salah satu *worker* mengirim kembali produk titik, disimpan di tempat yang tepat di c dan mengirim baris lain kepada *worker* itu untuk kembali dikerjakan. Setelah semua baris telah ditetapkan, *worker* yang menyelesaikan tugas dikirim pesan "tidak ada pekerjaan lagi", ditunjukkan dengan pesan dengan tag 0.

BAB IV
HASIL DAN PEMBAHASAN

Pada bab ini akan dibahas hasil pengujian dari penelitian yang dilakukan.

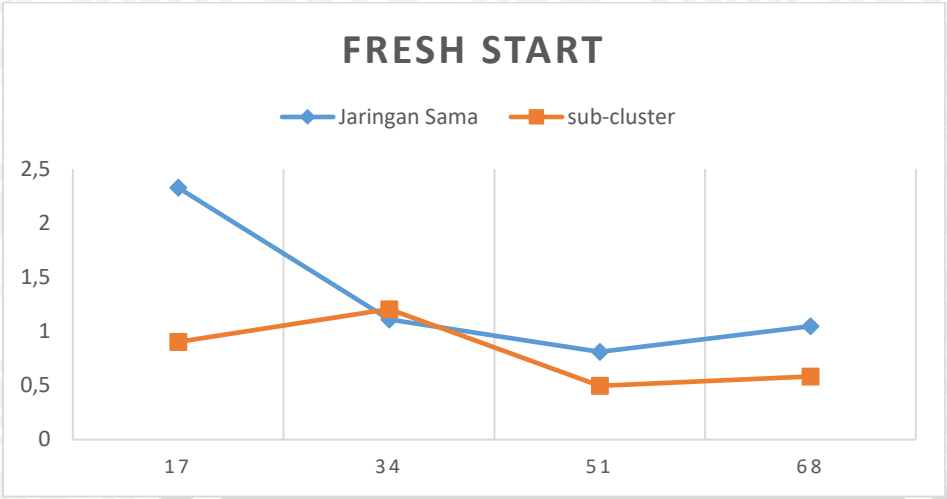
1. Pengujian dengan kondisi awal yang sama (*Fresh Start*) yaitu baru saja di nyalakan, dan setiap perangkat menjalankan program dengan *size* 100 dalam 4 kondisi, yaitu dengan *number of process* 17, 34, 51 dan 68.
2. Pengujian program dengan ukuran 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000 dengan *number of process* 17, 34, 51 dan 68.

4.1 Hasil Pengujian *Fresh Start*

Tabel 8 Hasil Pengujian Fresh Start

Number of process	Waktu (Sec)		*Speedup*
	Jaringan Sama	Sub-Cluster	
17	2,3254	0,9025	2,6803
34	1,1103	1,2048	0,9215
51	0,8101	0,4963	1,6322
68	1,0460	0,5844	1,7898

Dari tabel diatas, dapat dilihat waktu yang rata-rata yang dibutuhkan pada jaringan sama untuk menyelesaikan program dengan *size* 100 dan *number of process* 17 adalah 2,3254 sec, *number of process* 34 adalah 1,1103 sec, *number of process* 51 adalah 0,8101 sec dan *number of process* 68 adalah 1,0460 sec. Sedangkan pada metode sub-cluster, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,9025 sec, *number of process* 34 adalah 1,2048 sec, *number of process* 51 adalah 0,4963, dan *number of process* 68 adalah 0,5844 sec.



Gambar 18 Grafik Hasil Pengujian Fresh Start

Dari gambar di atas dapat dilihat bahwa jaringan sub-cluster lebih cepat pada saat pengujian *fresh start* dengan *number of process* 17, 51 dan 68. Sedangkan pada *number of process* 34, sub-cluster sedikit lebih lambat.

4.2 Pengujian program dengan *number of process* dan *size* yang beragam.

4.2.1 Hasil Pengujian Jaringan Sama

Tabel 9 Hasil Pengujian Jaringan Sama

	Waktu (Sec)			
Ukuran	Np 17	Np 34	Np 51	Np 68
100	0,1842	1,1075	0,5420	0,9304
200	0,1934	0,9475	1,3002	0,7484
300	0,2301	0,2980	0,6046	0,8511
400	0,2978	0,2973	1,4725	0,9496
500	0,3301	0,3406	0,5945	0,8765
600	0,4813	0,8180	1,5488	0,8616
700	0,6331	0,6822	0,7848	1,0588
800	0,6277	0,3744	0,8675	1,0385
900	0.7594	0,3999	0,7366	1,2032
1000	0,9607	0,4554	0,8248	1,0496

Dari tabel diatas, dapat dilihat waktu yang rata-rata yang dibutuhkan pada jaringan sama untuk menyelesaikan program dengan *size* 100 dan *number of process* 17 adalah 0,1842 sec, *number of process* 34 adalah 1,1075 sec, *number of process* 51 adalah 0,5420 sec dan *number of process* 68 adalah 0,9304 sec.

Untuk program dengan *size* 200, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,1934 sec, *number of process* 34 adalah 0,9475 sec, *number of process* 51 1,3002 sec dan *number of process* 68 adalah 0,7484 sec. Untuk program dengan *size* 300, waktu yang dibutuhkan untuk menyelesaikan program dengan

number of process 17 adalah 0,2301 sec, *number of process* 34 adalah 0,2980 sec, *number of process* 51 0,6046 sec dan *number of process* 68 adalah 0,8511 sec.

Untuk program dengan *size* 400, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,2978 sec, *number of process* 34 adalah 0,2973 sec, *number of process* 51 1,4725 sec dan *number of process* 68 adalah 0,9496 sec.

Untuk program dengan *size* 500, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,1934 sec, *number of process* 34 adalah 0,9475 sec, *number of process* 51 1,3002 sec dan *number of process* 68 adalah 0,7484 sec.

Untuk program dengan *size* 600, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,4813 sec, *number of process* 34 adalah 0,8180 sec, *number of process* 51 1,5488 sec dan *number of process* 68 adalah 0,8616 sec.

Untuk program dengan *size* 700, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,6331 sec, *number of process* 34 adalah 0,6822 sec, *number of process* 51 0,7848 sec dan *number of process* 68 adalah 1,0588 sec.

Untuk program dengan *size* 800, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,6277 sec, *number of process* 34 adalah 0,3744 sec, *number of process* 51 0,8675 sec dan *number of process* 68 adalah 1,0385 sec.

Untuk program dengan *size* 900, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,7594 sec, *number of process* 34 adalah 0,3999 sec, *number of process* 51 0,7366 sec dan *number of process* 68 adalah 1,2032 sec.

Untuk program dengan *size* 1000, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,9607 sec, *number of process* 34 adalah 0,4554 sec, *number of process* 51 0,8248 sec dan *number of process* 68 adalah 1,0496 sec.

4.2.2 Hasi Pengujian Sub-Cluster

Tabel 10 Hasil Pengujian Sub-Cluster

	Waktu (Sec)			
Ukuran	Np 17	Np 34	Np 51	Np 68
100	0,2165	0,3707	0,3244	0,4994
200	0,2513	0,3901	0,3142	0,6042
300	0,4244	0,3273	0,3265	0,6231
400	0,2969	0,3622	0,4681	0,5525
500	0,3482	0,7010	0,4314	0,6158
600	0,5149	0,4455	0,4679	0,5662
700	0,6281	0,4471	0,4837	0,7561
800	0,6402	0,4200	0,4348	0,7086
900	0,7568	0,4242	0,4456	0,6632
1000	0,9171	0,4661	0,5072	0,6725

Dari tabel diatas, dapat dilihat waktu yang rata-rata yang dibutuhkan pada jaringan bertingkat (sub-cluster) untuk menyelesaikan program dengan *size* 100 dan *number of process* 17 adalah 0,2165 sec, *number of process* 34 adalah 0,3707 sec, *number of process* 51 adalah 0,3244 sec dan *number of process* 68 adalah 0,4994 sec.

Untuk program dengan *size* 200, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,2513 sec, *number of process* 34 adalah 0,3901 sec, *number of process* 51 adalah 0,3142 sec dan *number of process* 68 adalah 0,6042 sec.

Untuk program dengan *size* 300, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,4244 sec, *number of process* 34 adalah 0,3273 sec, *number of process* 51 adalah 0,3265 sec dan *number of process* 68 adalah 0,6231 sec.

Untuk program dengan *size* 400, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,2969 sec, *number of process* 34 adalah 0,3622 sec, *number of process* 51 adalah 0,4681 sec dan *number of process* 68 adalah 0,5525 sec.

Untuk program dengan *size* 500, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,3482 sec, *number of process* 34 adalah 0,7010 sec, *number of process* 51 adalah 0,4314 sec dan *number of process* 68 adalah 0,6158 sec.

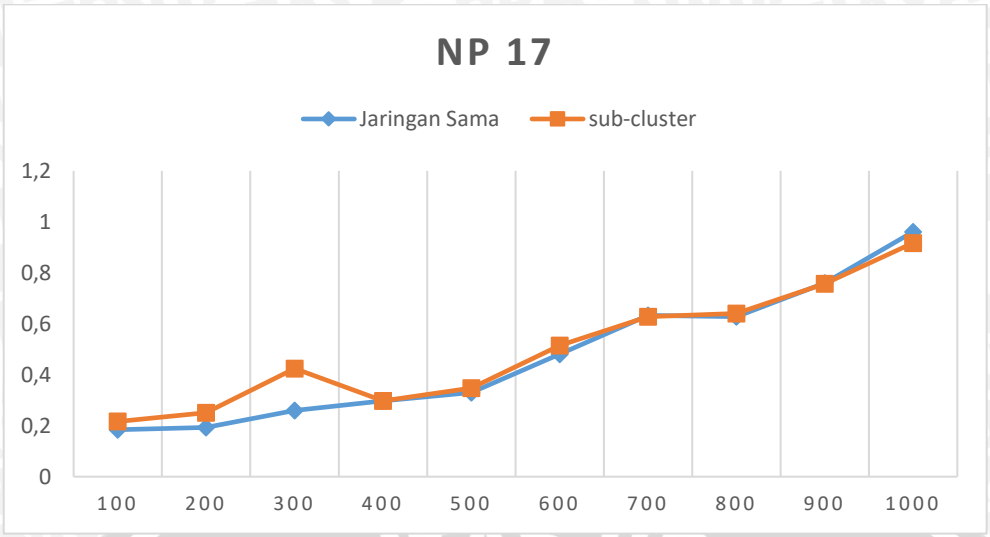
Untuk program dengan *size* 600, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,5149 sec, *number of process* 34 adalah 0,4455 sec, *number of process* 51 adalah 0,4679 sec dan *number of process* 68 adalah 0,5662 sec.

Untuk program dengan *size* 700, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,6281 sec, *number of process* 34 adalah 0,4471 sec, *number of process* 51 adalah 0,4837 sec dan *number of process* 68 adalah 0,7561 sec.

Untuk program dengan *size* 800, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,6402 sec, *number of process* 34 adalah 0,4200 sec, *number of process* 51 adalah 0,4348 sec dan *number of process* 68 adalah 0,7086 sec.

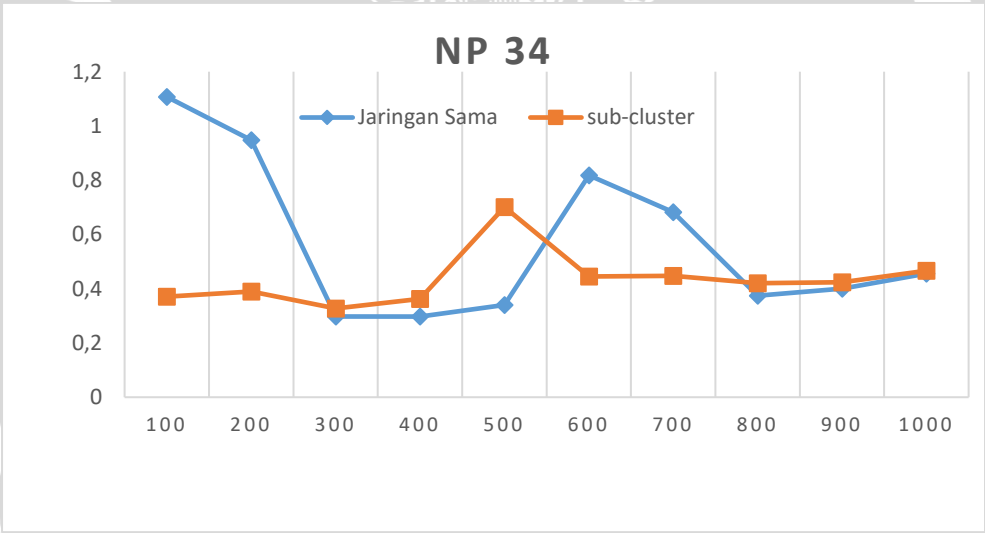
Untuk program dengan *size* 900, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,7568 sec, *number of process* 34 adalah 0,4242 sec, *number of process* 51 adalah 0,4456 sec dan *number of process* 68 adalah 0,6632 sec.

Untuk program dengan *size* 1000, waktu yang dibutuhkan untuk menyelesaikan program dengan *number of process* 17 adalah 0,9171 sec, *number of process* 34 adalah 0,4661 sec, *number of process* 51 adalah 0,6725 sec dan *number of process* 68 adalah 0,6725 sec.



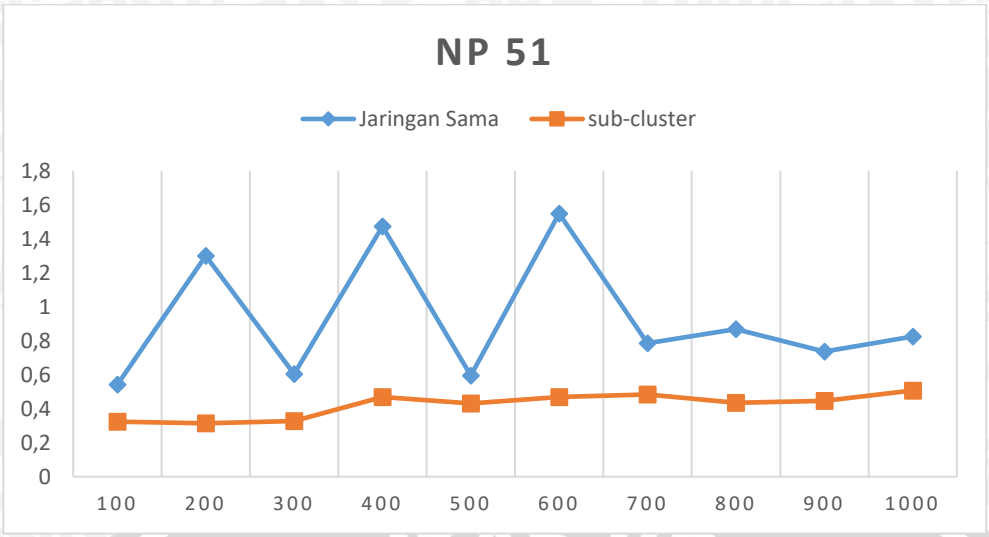
Gambar 19 Hasil Pengujian Dengan NP 17

Dari gambar di atas dapat dilihat bahwa waktu yang dibutuhkan untuk mengeksekusi program dengan *number of process* 17 hampir sama antara jaringan sama dan sub-cluster.



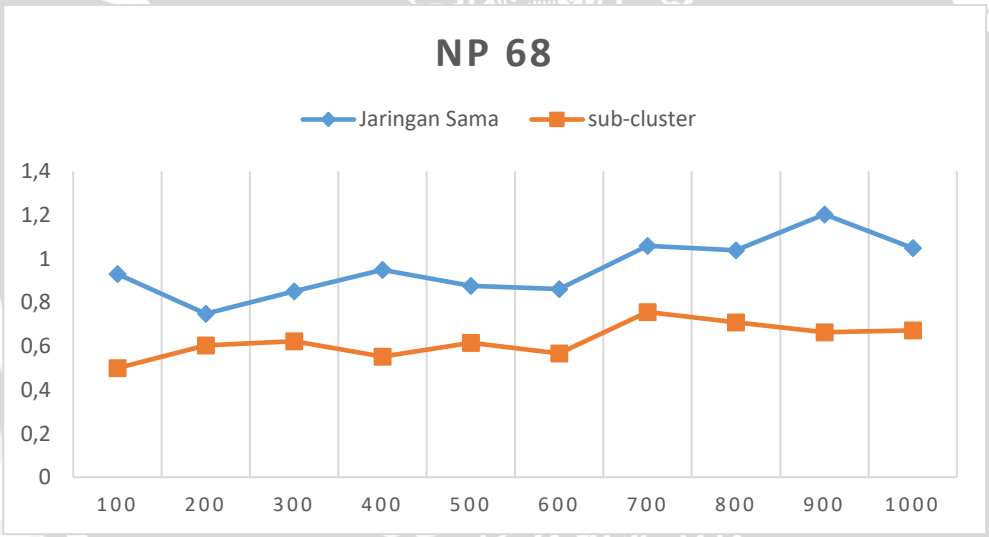
Gambar 20 Hasil Pengujian Dengan NP 34

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *number of process* 34, metode jaringan sama lebih cepat pada *size* 100, 200, 600 dan 700.



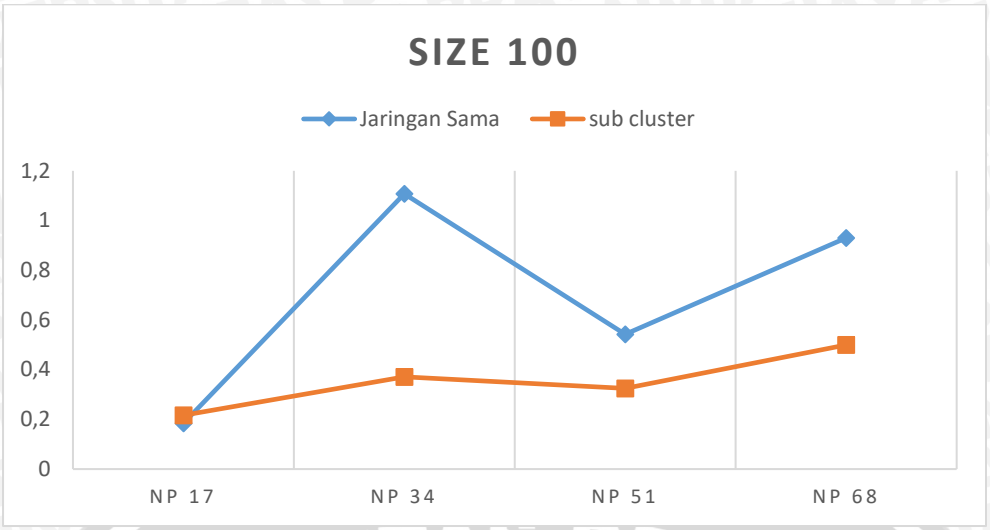
Gambar 21 Hasil Pengujian Dengan NP 51

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *number of process* 51, metode jaringan sama lebih cepat pada semua *size* program yang digunakan.



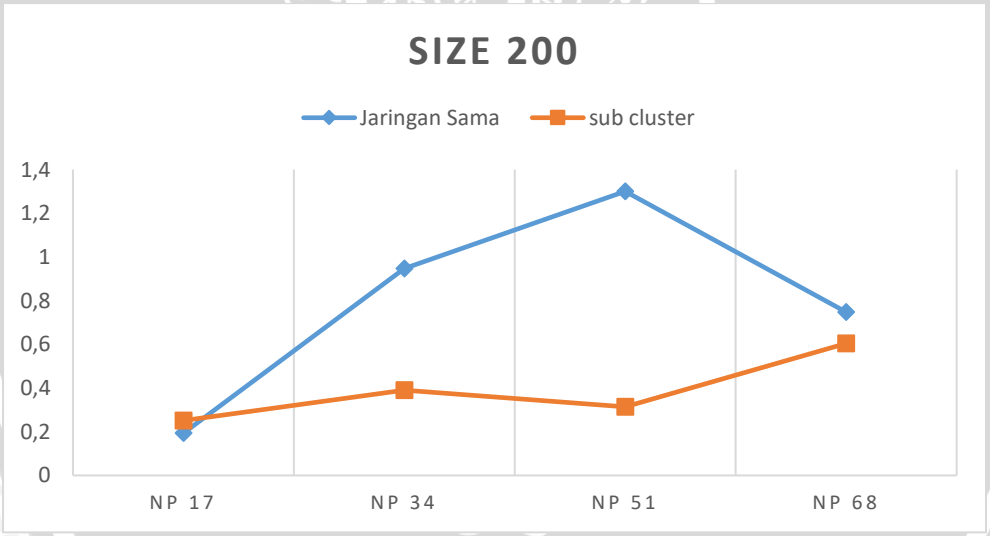
Gambar 22 Hasil Pengujian Dengan NP 68

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *number of process* 68 , metode jaringan sama lebih cepat pada semua *size* program yang digunakan.



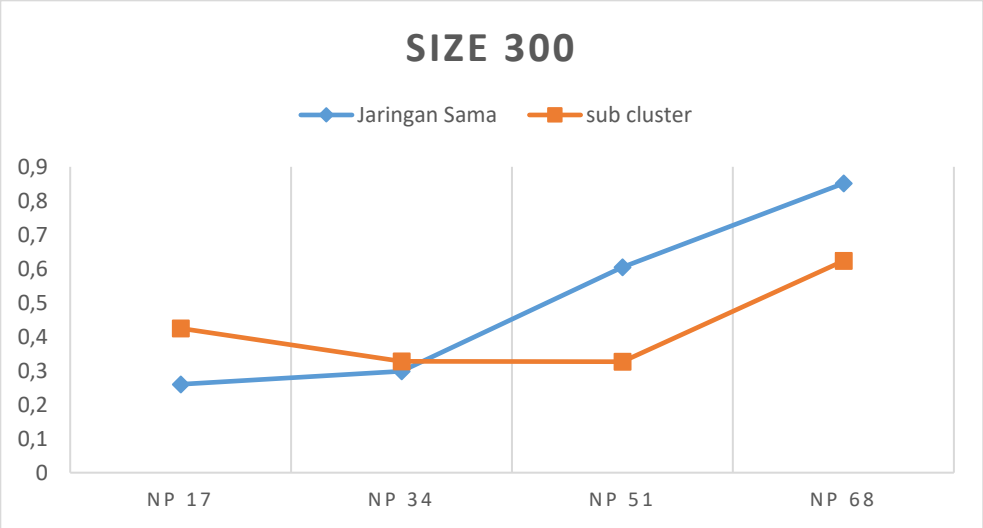
Gambar 23 Hasil Pengujian Size 100

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 100, metode sub-cluster lebih cepat pada *number of process* 34, 51 dan 68. Sedangkan pada *number of process* 17 waktu yang dibutuhkan nyaris sama.



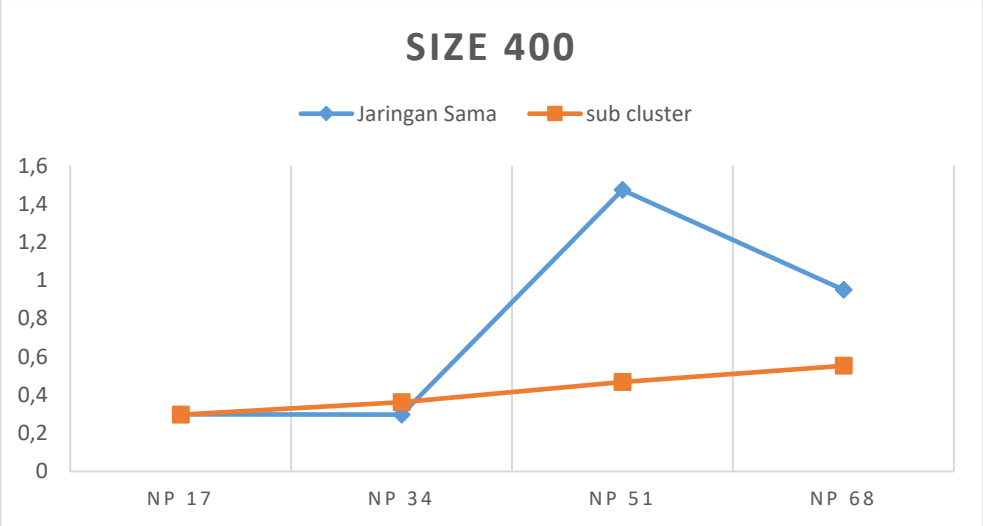
Gambar 24 Hasil Pengujian Size 200

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 200, metode sub-cluster lebih cepat pada *number of process* 34, 51 dan 68. Sedangkan pada *number of process* 17 waktu yang dibutuhkan nyaris sama.



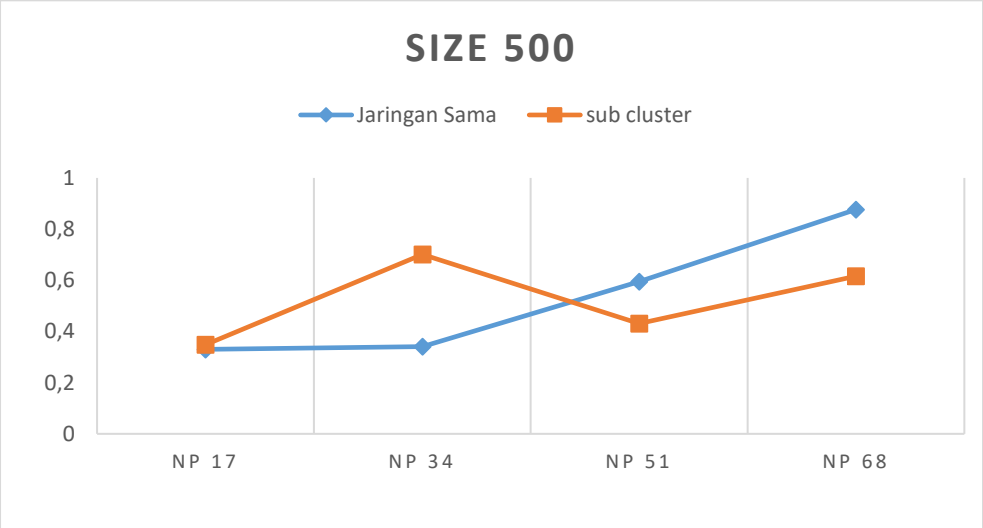
Gambar 25 Hasil Pengujian Size 300

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 300, metode sub-cluster lebih cepat pada *number of process* 51 dan 68. Sedangkan pada *number of process* 17 waktu yang dibutuhkan lebih lama dibanding dengan jaringan sama, dan hampir sama pada *number of process* 34.



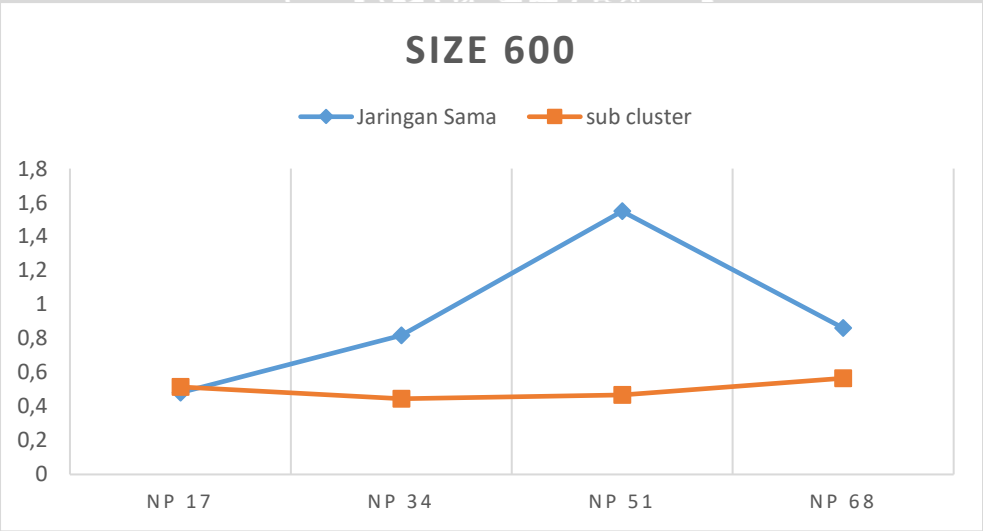
Gambar 26 Hasil Pengujian Size 400

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 400, metode sub-cluster lebih cepat pada *number of process* 51 dan 68. Sedangkan pada *number of process* 17 dan 34 waktu yang dibutuhkan hampir sama dengan metode jaringan sama.



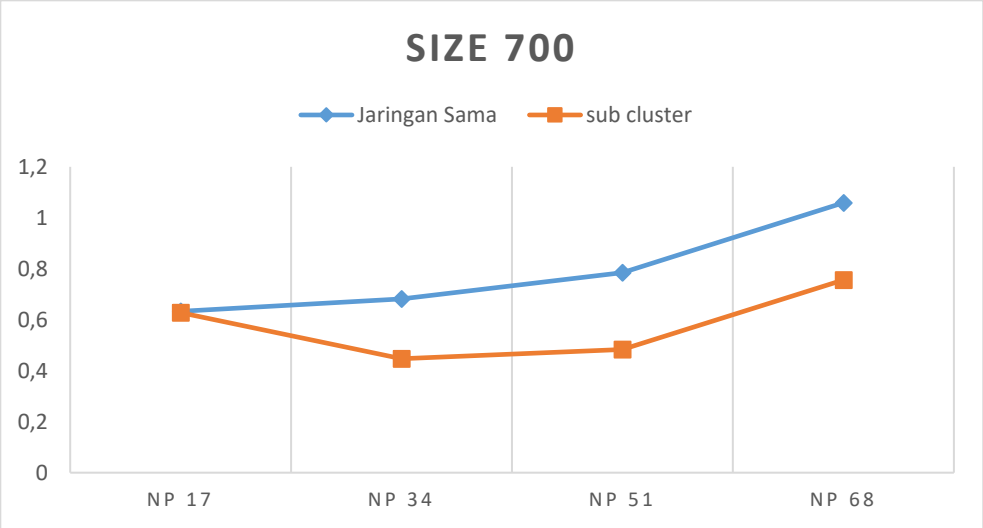
Gambar 27 Hasil Pengujian Size 500

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 500, metode sub-cluster lebih cepat pada *number of process* 51 dan 68. Sedangkan pada *number of process* 34 waktu yang dibutuhkan lebih lama dibanding dengan jaringan sama, dan waktu yang dibutuhkan hampir sama pada *number of process* 17.



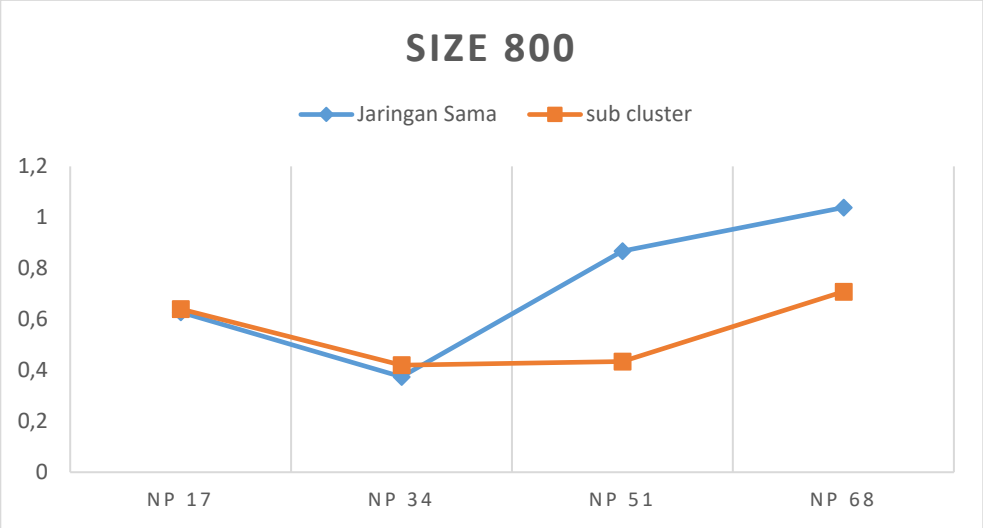
Gambar 28 Hasil Pengujian Size 600

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 600, metode sub-cluster lebih cepat pada *number of process* 34, 51 dan 68. Sedangkan pada *number of process* 17 waktu yang dibutuhkan nyaris sama.



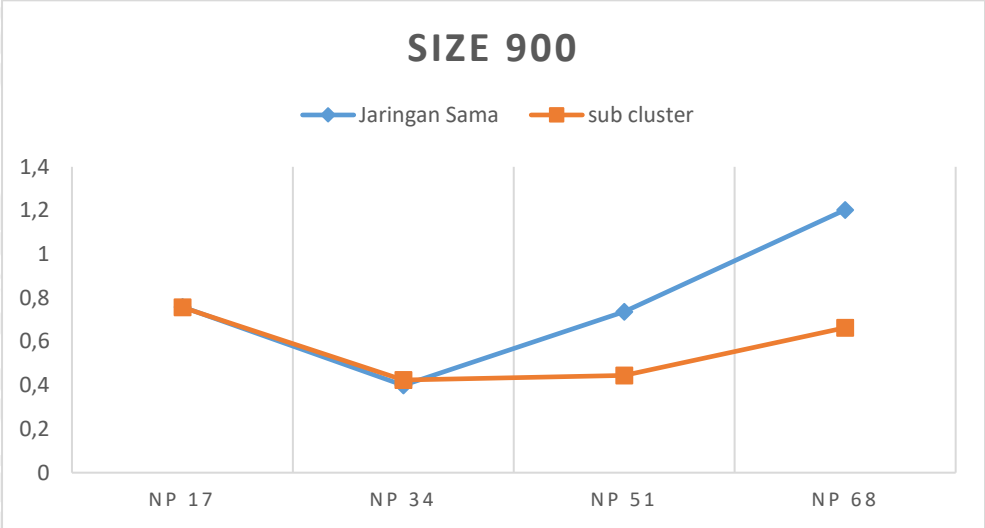
Gambar 29 Hasil Pengujian Size 700

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 700, metode sub-cluster lebih cepat pada *number of process* 34, 51 dan 68. Sedangkan pada *number of process* 17 waktu yang dibutuhkan nyaris sama.



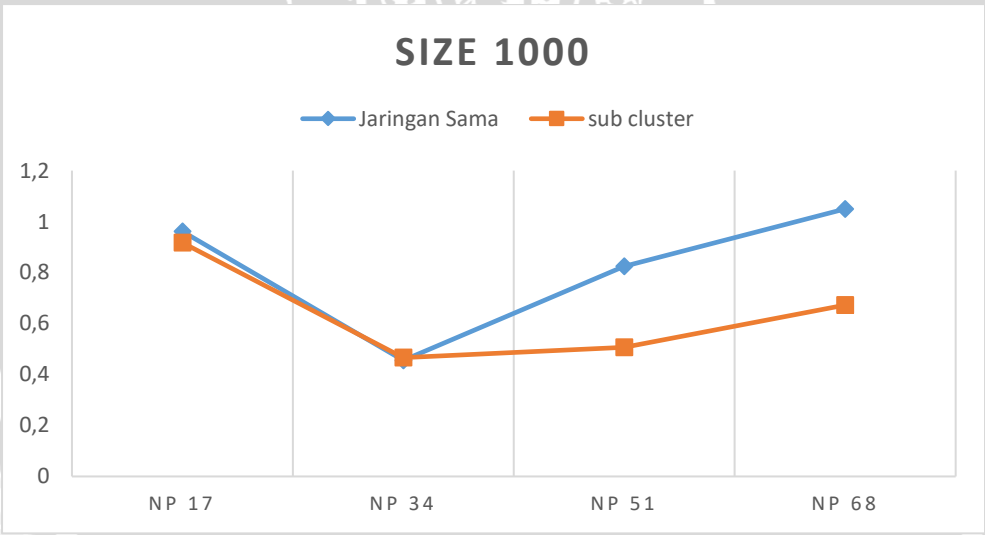
Gambar 30 Hasil Pengujian Size 800

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 800, metode sub-cluster lebih cepat pada *number of process* 51 dan 68. Sedangkan pada *number of process* 17 dan 34 waktu yang dibutuhkan hampir sama dengan metode jaringan sama.



Gambar 31 Hasil Pengujian Size 900

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 900, metode sub-cluster lebih cepat pada *number of process* 51 dan 68. Sedangkan pada *number of process* 17 dan 34 waktu yang dibutuhkan hampir sama dengan metode jaringan sama.



Gambar 32 Hasil Pengujian Size 1000

Dari gambar di atas dapat dilihat bahwa pada saat mengeksekusi program dengan *size* 1000, metode sub-cluster lebih cepat pada *number of process* 51 dan 68. Sedangkan pada *number of process* 17 dan 34 waktu yang dibutuhkan hampir sama dengan metode jaringan sama.

Dalam tabel dibawah dapat dilihat *speedup* dari tiap *number of process* yang digunakan. Speedup memperlihatkan seberapa besar perubahan kecepatan. *Speedup* didapat dengan membagi hasil pada jaringan yang sama dengan jaringan sub-cluster.

Tabel 11 Speedup pada number of process 17

Size	Waktu (sec)		*Speedup*
	Jaringan Sama	Jaringan Sub-Cluster	
100	0,1842	0,2165	0,8509
200	0,1934	0,2513	0,7696
300	0,2301	0,4244	0,6129
400	0,2978	0,2969	1,0029
500	0,3301	0,3482	0,9479
600	0,4813	0,5149	0,9347
700	0,6331	0,6281	1,0079
800	0,6277	0,6402	0,9805
900	0.7594	0,7568	1,0034
1000	0,9607	0,9171	1,0476

Tabel 12 Speedup pada number of process 34

Size	Waktu (sec)		*Speedup*
	Jaringan Sama	Jaringan Sub-Cluster	
100	1,1075	0,3707	2,9875
200	0,9475	0,3901	2,4287
300	0,2980	0,3273	0,9105
400	0,2973	0,3622	0,8208
500	0,3406	0,7010	0,4859
600	0,8180	0,4455	1,8360
700	0,6822	0,4471	1,5259
800	0,3744	0,4200	0,8913
900	0,3999	0,4242	0,9428
1000	0,4554	0,4661	0,9771

Tabel 13 Speedup pada number of process 51

Size	Waktu (sec)		*Speedup*
	Jaringan Sama	Jaringan Sub-Cluster	
100	0,5420	0,3244	1,6706
200	1,3002	0,3142	4,1382
300	0,6046	0,3265	1,8516
400	1,4725	0,4681	3,1453
500	0,5945	0,4314	1,3780
600	1,5488	0,4679	3,3097
700	0,7848	0,4837	1,6225
800	0,8675	0,4348	1,9952
900	0,7366	0,4456	1,6531
1000	0,8248	0,5072	1,6263

Tabel 14 Speedup pada number of process 68

Size	Waktu (sec)		*Speedup*
	Jaringan Sama	Jaringan Sub-Cluster	
100	0,9304	0,4994	1,8630
200	0,7484	0,6042	1,2388
300	0,8511	0,6231	1,3660
400	0,9496	0,5525	1,7186
500	0,8765	0,6158	1,4235
600	0,8616	0,5662	1,5218
700	1,0588	0,7561	1,4005
800	1,0385	0,7086	1,4655
900	1,2032	0,6632	1,8142
1000	1,0496	0,6725	1,5607

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil pengujian tiap bagian dan keseluruhan sistem yang telah dilaksanakan didapatkan kesimpulan:

1. Metode sub-cluster merupakan salah satu metode yang bisa digunakan untuk merancang suatu komputer paralel.
2. Metode sub-cluster dapat meningkatkan kecepatan komputasi dalam menjalankan program perkalian matriks, meskipun tidak di semua jenis pengujian terlihat adanya peningkatan kecepatan.

5.2 Saran

Saran yang dapat digunakan dalam peningkatkan kerja sistem dapat diuraikan sebagai berikut:

1. Metode sub-cluster dipakai dan di uji dengan menggunakan permasalahan dan penulisan program lain yang lebih sesuai
2. Penggunaan NFS untuk mempermudah dalam menjalankan program.
3. Penggunaan *batch* saat menjalankan program agar memudahkan dalam pengambilan data.

DAFTAR PUSTAKA

Agus Kurniawan. 2010. *Pemrograman Paralel dengan MPI dan C*. Yogyakarta: Penerbit ANDI.

Barry Wilkinson & Michael Allen. 2010. *Parallel Programing*. Yogyakarta: Penerbit ANDI.

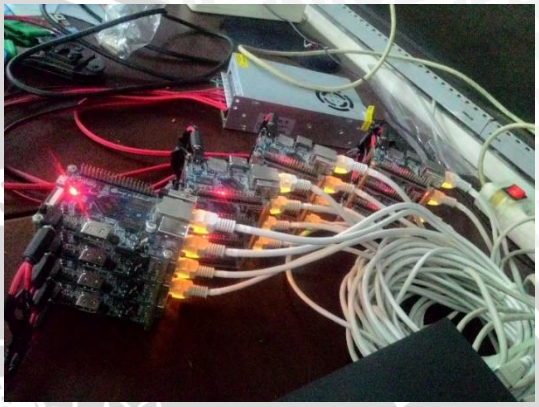
Georga S. Almasi & Allan Gottlieb. 1994. *Highly Parallel Computing*. California: The Benjamin/Cummings Publishing Company, Inc.

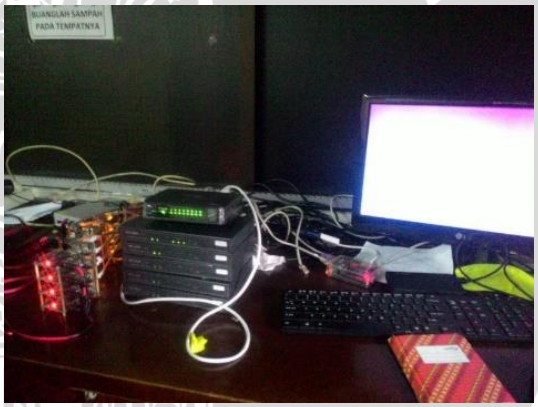
I Putu Agus Eka Pratama. 2014. *Handbook Jaringan Komputer*. Bandung: Penerbit Informatika.

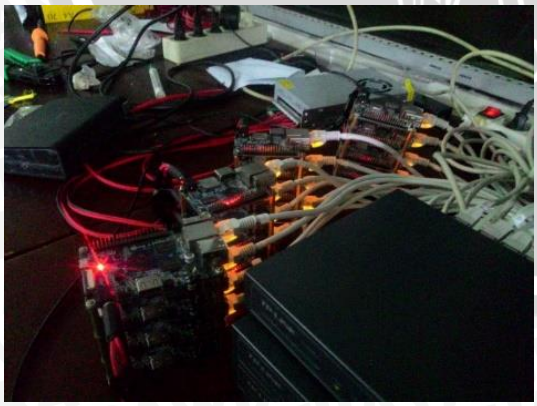
William Gropp & Ewing Lusk. 2003. *Beowulf Cluster Computing with Linux*. Massachusetts: MIT Press.

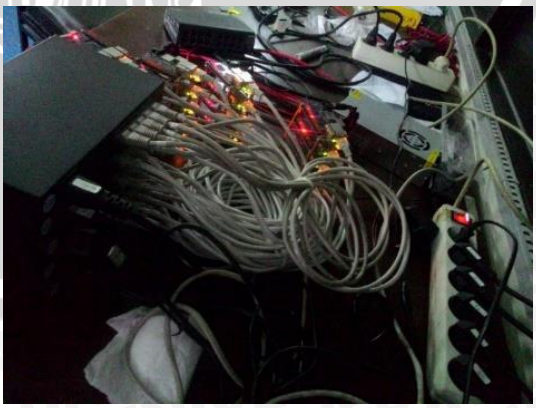
LAMPIRAN 1
DOKUMENTASI ALAT

Berikut adalah gambar hasil instalasi perangkat yang digunakan.



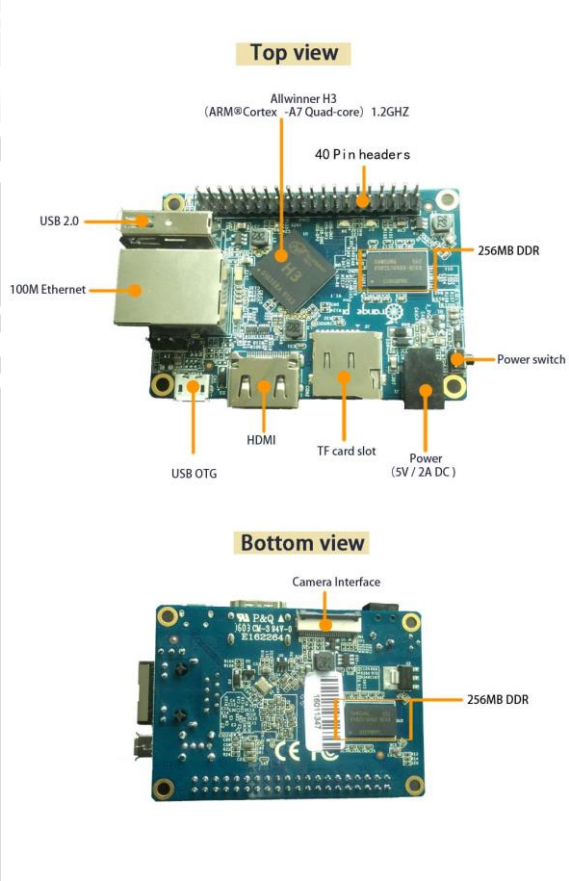






LAMPIRAN 2
DATASHEET

1. Orange Pi One

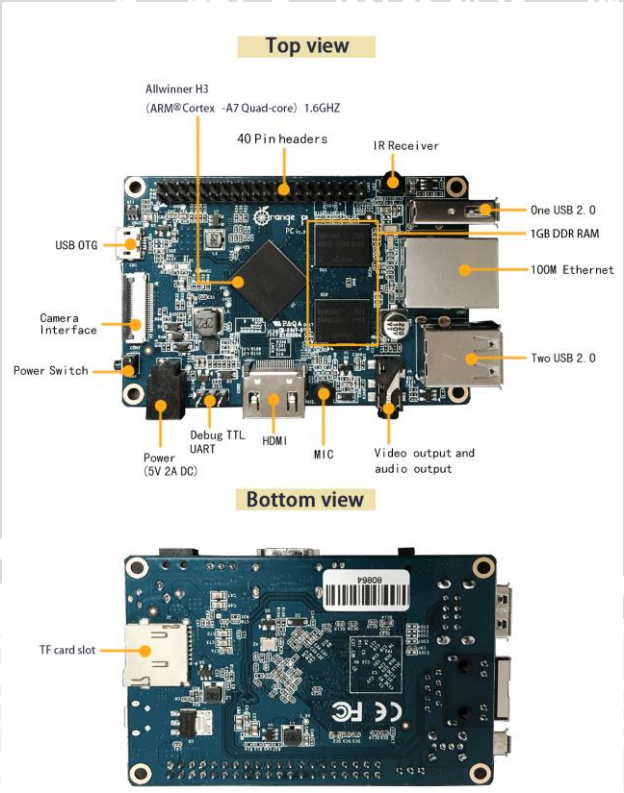


Spesifikasi:

1. CPU: H3 Quad-core Cortex-A7 H.265/HEVC 4K
2. GPU: Mali400MP2 GPU @600MHz, Supports OpenGL ES 2.0
3. Memory (SDRAM): 512MB DDR3 (shared with GPU)
4. Onboard storage: TF Card (Max 64 GB) / MMC Card Slot
5. Onboard Network: 10/100M Ethernet RJ45
6. Video Input: A CSI input connector Camera:Supports 8-bit YUV422 CMOS sensor interface, Supports CCIR656 protocol for NTSC and PAL, Supports SM pixel camera sensor, Supports video capture solution up to 1080p@30fps
7. Audio Input: MIC

8. Video Outputs: Supports HDMI output with HDCP, Supports HDMI CEC, Supports HDMI 30 function, Integrated CVBS, Supports simultaneous output of HDMI and CVBS
9. Audio Output: 3.5 mm Jack and HDMI
10. Power Source: DC input can supply power, but USB OTG input don't supply power
11. USB 2.0 Ports: Hanya 1 USB 2.0, 1 USB 2.0 OTG
12. Buttons: Power Button(SW4)
13. Low-level peripherals: 40 Pins Header, Key:vcompatible with Raspberry Pi B+
14. GPIO(1x3) pin: UART, ground.
15. LED: Power led & Status led
16. Key: POWER
17. Supported OS: Android Ubuntu, Debian, Rasberry Pi Image
18. Ukuruan Produk: 69 mm x 48 mm
19. Berat: 36 gr

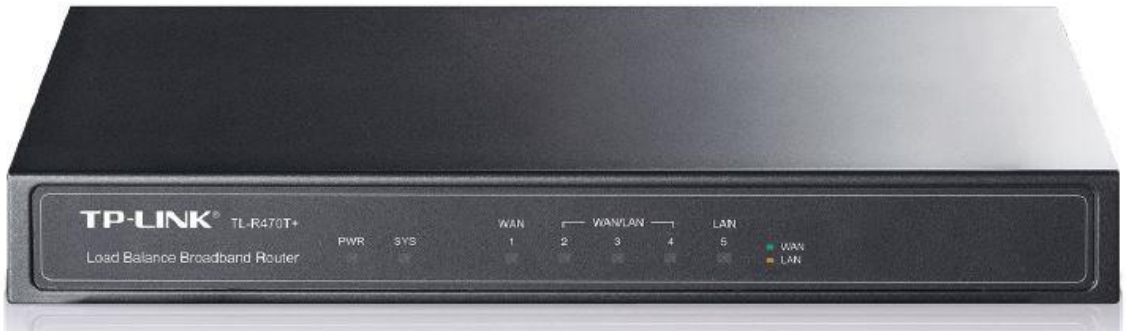
2. Orange Pi PC



1. CPU: H3 Quad-core Cortex-A7 H.265/HEVC 4K
2. GPU: Mali400MP2 GPU @600MHz, Supports OpenGL ES 2.0
3. Memory (SDRAM): 1GB DDR3 (shared with GPU)
4. Onboard storage: TF Card (Max 64 GB) / MMC Card Slot
5. Onboard Network: 10/100M Ethernet RJ45
6. Video Input: A CSI input connector Camera:Supports 8-bit YUV422 CMOS sensor interface, Supports CCIR656 protocol for NTSC and PAL, Supports SM pixel camera sensor, Supports video capture solution up to 1080p@30fps
7. Audio Input: MIC
8. Video Outputs: Supports HDMI output with HDCP, Supports HDMI CEC, Supports HDMI 30 function, Integrated CVBS, Supports simultaneous output of HDMI and CVBS
9. Audio Output: 3.5 mm Jack and HDMI
10. Power Source: DC input can supply power, but USB OTG input don't supply power
11. USB 2.0 Ports: Three USB 2.0 HOST, one USB 2.0 OTG
12. Buttons: Power Button(SW4)
13. Low-level peripherals: 40 Pins Header, Key:vcompatible with Raspberry Pi B+
14. GPIO(1x3) pin: UART, ground.
15. LED: Power led & Status led
16. Key: IR input, POWER
17. Supported OS: Android Ubuntu, Debian, Rasberry Pi Image
18. Ukuruan Produk: 85 mm x 55 mm
19. Berat: 38 gr

Datasheet Router TP-LINK TL-R470+

TP-LINK® Datasheet



TP-LINK

Load Balance Broadband Router

TL-R470+

Highlights

- Up to 4 WAN ports equipped with advanced load balance to guarantee maximum bandwidth and backup capabilities
- Provides extensive client account and network management for administrators with supported PPPoE Server
- Marshals bandwidth resource to specific clients based on their unique application environments
- Smaller case for cost-effective solution in the small office or Internet cafes network
- Professional 4KV lightning protection keeps your investments as safe as possible

www.tp-link.com

TP-LINK®

Datasheet

Specifications

Hardware Features

- Standards and Protocols: IEEE 802.3, 802.3u, 802.3x, TCP/IP, DHCP, ICMP, NAT, PPPoE, SNTP, HTTP, DDNS
- Interface: 1 Fixed Ethernet WAN Port, 1 Fixed Ethernet LAN Port and 3 Changeable Ethernet WAN/LAN Ports
- Network Media: 10BASE-T: UTP category 3, 4, 5 cable (Max 100m), 100BASE-TX: UTP category 5, 5e cable (Max 100m)
- Flash/DRAM: 4MB/64MB
- LEDs: PWR, SYS, WAN, LAN, WAN/LAN
- Button: Reset
- Dimensions: 209*126*26 mm
- Power Supply: Internal Universal Power Supply,
- AC100-240V~50/60Hz input

Performance

- Concurrent Session: 10000

Basic Function

- WAN Connection Type: Static/Dynamic IP, PPPoE, PPTP, L2TP, Dual Access, Bigpond Cable
- MAC Clone: Modify WAN/LAN MAC Address
- DHCP: DHCP Server/Client, DHCP Address Reservation
- Switch Setting: Port Mirror, Rate Control, Port Configuration, Port VLAN

Advanced Function

- Load Balance: Policy Routing, Link Backup
- Forwarding: Virtual Server, Port Triggering, DMZ
- Security: FTP/SIP/PPTP/IPsec/H.323 ALG, DoS Defence, Ping of Death
- ACL: IP/MAC/URL/WEB Filtering
- Services: PPPoE Server, E-Bulletin, Dynamic DNS, UPnP,
- Static Routing
- Bandwidth Control
- IP/MAC Binding
- Session Limit
- IGMP: IGMP Proxy/IGMP Snooping

Maintenance

- Time Setting
- Diagnostics
- Firmware Upgrade
- Factory Defaults/Reboot
- Backup/Restore
- System Log
- Remote Management
- Statistics
- Daylight Saving Time
- SNMP

Others

- Certification: CE, FCC, RoHS
- Package Contents: TL-R470T+, Power Cord, RJ-45 Ethernet Cable, Quick Installation Guide, Resource CD
- System Requirements: Microsoft® Windows 8/7/Vista/XP/2000, MAC® OS, NetWare®, UNIX® or Linux
- Environment:
  Operating Temperature: 0℃~40℃ (32℉~104℉), Storage Temperature: -40℃~70℃ (-40℉~158℉),
  Operating Humidity: 10%~90% non-condensing,
  Storage Humidity: 5%~90% non-condensing

Ordering Information

Product Model	Description
TL-R470T+	Load Balance Broadband Router

LAMPIRAN 3

LISTING PROGRAM PENGUJIAN

```
1.  #include "mpi.h"
2.  #include <sys/time.h>
3.  #include <stdio.h>
4.
5.  void manager_code();
6.  void worker_code();
7.  struct timeval start_time, stop_time;
8.  int main( int argc, char *argv[] )
9.  {
10. int numprocs, myrank;
11. char hostname[256];
12. gettimeofday(&start_time,NULL);
13. MPI_Init( &argc, &argv );
14. MPI_Comm_size( MPI_COMM_WORLD, &numprocs );
15. MPI_Comm_rank( MPI_COMM_WORLD, &myrank );
16. gethostname(hostname,255);
17.
18.
19. //printf("waktu: start = %d.%d\n",start_time.tv_sec,start_time.tv_usec);
20. if ( myrank == 0 )      /* manager process */
21. manager_code ( numprocs );
22. else                   /* worker process */
23. worker_code ( );
24.
25.
26.
27. MPI_Finalize( );
28. gettimeofday(&stop_time,NULL);
29.
30.
31. //printf("waktu: finish = %d.%d\n", stop_time.tv_sec,stop_time.tv_usec);
32. printf("waktu: elapsed = %f sec\n", stop_time.tv_sec-
    start_time.tv_sec+ (double)(stop_time.tv_usec-start_time.tv_usec)/1000000);
33.
34. return 0;
35. }
36.
37. #define SIZE 100
38. #define MIN( x, y ) ((x) < (y) ? x : y)
39. void manager_code( int numprocs )
40. {
41. double a[SIZE][SIZE], c[SIZE];
42. int i, j, sender, row, numsent = 0;
43. double dotp;
44. MPI_Status status;
45. /* (arbitrary) initialization of a */
46. for (i = 0; i < SIZE; i++ )
47. for ( j = 0; j < SIZE; j++ )
48. a[i][j] = ( double ) j;
49. for ( i = 1; i < MIN( numprocs, SIZE ); i++ ) {
50. MPI_Send( a[i-1], SIZE, MPI_DOUBLE, i, i, MPI_COMM_WORLD );
51. numsent++;
52. }
53. /* receive dot products back from workers */
54. for ( i = 0; i < SIZE; i++ ) {
```

```
55. MPI_Recv( &dotp, 1, MPI_DOUBLE, MPI_ANY_SOURCE, MPI_ANY_TAG, MPI_COMM_WORLD, &sta
    tus );
56. sender = status.MPI_SOURCE;
57. row = status.MPI_TAG - 1;
58. c[row] = dotp;
59. /* send another row back to this worker if there is one */
60. if ( numsent < SIZE ) {
61. MPI_Send( a[numsent], SIZE, MPI_DOUBLE, sender, numsent + 1, MPI_COMM_WORLD );
62. numsent++;
63. }
64. else /* no more work */
65. MPI_Send( MPI_BOTTOM, 0, MPI_DOUBLE, sender, 0, MPI_COMM_WORLD );
66. }
67. }
68.
69. void worker_code( void )
70. {
71. double b[SIZE], c[SIZE];
72. int i, row, myrank;
73. double dotp;
74. MPI_Status status;
75. for ( i = 0; i < SIZE; i++ ) /* (arbitrary) b initialization */
76. b[i] = 1.0;
77. MPI_Comm_rank( MPI_COMM_WORLD, &myrank );
78. if ( myrank <= SIZE ) {
79. MPI_Recv( c, SIZE, MPI_DOUBLE, 0, MPI_ANY_TAG, MPI_COMM_WORLD, &status );
80. while ( status.MPI_TAG > 0 ) {
81. row = status.MPI_TAG - 1;
82. dotp = 0.0;
83. for ( i = 0; i < SIZE; i++ )
84. dotp += c[i] * b[i];
85. MPI_Send( &dotp, 1, MPI_DOUBLE, 0, row + 1, MPI_COMM_WORLD );
86. MPI_Recv( c, SIZE, MPI_DOUBLE, 0, MPI_ANY_TAG, MPI_COMM_WORLD, &status );
87. }
88. }
89. }
```

LAMPIRAN 4
HASIL EKSEKUSI

1. Hasil pengujian *Fresh start* menggunakan metode jaringan sama

Np 17	Np 34		Np 51			Np 68			
2, 371952 sec	1, 107055 sec	1, 101995 sec	0, 789183 sec	0, 789174 sec	0, 790319 sec	1, 055343 sec	1, 052609 sec	1, 056345 sec	1, 040552 sec
2, 070519 sec	1, 126347 sec	1, 104682 sec	0, 820274 sec	0, 820247 sec	0, 796069 sec	1, 004855 sec	1, 003100 sec	1, 006268 sec	1, 048508 sec
2, 361819 sec	1, 102278 sec	1, 107753 sec	0, 777667 sec	0, 777634 sec	0, 816389 sec	1, 003040 sec	1, 037096 sec	1, 005623 sec	1, 035506 sec
2, 367664 sec	1, 122264 sec	1, 099174 sec	0, 780235 sec	0, 780234 sec	0, 796904 sec	1, 036820 sec	1, 029938 sec	1, 039035 sec	1, 059009 sec
2, 357869 sec	1, 115502 sec	1, 123651 sec	0, 794313 sec	0, 794490 sec	0, 784216 sec	1, 029957 sec	1, 022657 sec	1, 025386 sec	1, 049533 sec
2, 373515 sec	1, 121892 sec	1, 114637 sec	0, 814271 sec	0, 814862 sec	0, 826551 sec	1, 054325 sec	1, 054297 sec	1, 046630 sec	1, 029593 sec
2, 361470 sec	1, 122480 sec	1, 086316 sec	0, 789529 sec	0, 789538 sec	0, 797166 sec	1, 017550 sec	1, 017495 sec	1, 007912 sec	1, 025290 sec
2, 362522 sec	1, 101469 sec	1, 109937 sec	0, 815648 sec	0, 795466 sec	0, 797282 sec	1, 043081 sec	1, 042928 sec	1, 054885 sec	1, 075121 sec
2, 356435 sec	1, 126966 sec	1, 090207 sec	0, 795858 sec	0, 827721 sec	0, 792546 sec	1, 035179 sec	1, 035071 sec	1, 006384 sec	1, 042503 sec
2, 374315 sec	1, 102956 sec	1, 104992 sec	0, 783218 sec	0, 812968 sec	0, 799369 sec	1, 017328 sec	1, 030978 sec	1, 045723 sec	1, 044651 sec
2, 009624 sec	1, 123031 sec	1, 115358 sec	0, 812563 sec	0, 797186 sec	0, 817796 sec	1, 064797 sec	1, 024965 sec	1, 046761 sec	1, 075619 sec
2, 334414 sec	1, 116148 sec	1, 106628 sec	0, 786843 sec	0, 789767 sec	0, 820215 sec	1, 040396 sec	1, 054868 sec	1, 055739 sec	1, 078692 sec
2, 308644 sec	1, 085670 sec	1, 103376 sec	0, 815617 sec	0, 820910 sec	0, 820227 sec	1, 015837 sec	1, 018137 sec	1, 037914 sec	1, 080323 sec
2, 349559 sec	1, 122147 sec	1, 109902 sec	0, 795852 sec	0, 778427 sec	0, 834439 sec	1, 067016 sec	1, 045368 sec	1, 060857 sec	1, 136131 sec
2, 386600 sec	1, 109849 sec	1, 108666 sec	0, 783191 sec	0, 780612 sec	0, 961308 sec	1, 040155 sec	1, 035883 sec	1, 073259 sec	1, 143010 sec
2, 391028 sec	1, 089388 sec	1, 110864 sec	0, 812554 sec	0, 797296 sec	0, 961307 sec	1, 055333 sec	1, 018789 sec	1, 059096 sec	1, 161235 sec
2, 394401 sec	1, 122843 sec	1, 133819 sec	0, 786849 sec	0, 814432 sec	0, 968710 sec	1, 006048 sec	1, 070435 sec	1, 022737 sec	1, 172699 sec

2. Hasil pengujian *Fresh start* menggunakan metode jaringan bertingkat (sub-cluster)

Np 17	Np 34		Np 51			Np 68			
0, 958504 sec	1, 262362 sec	0, 519430 sec	0, 525237 sec	0, 527494 sec	0, 487718 sec	0, 546841 sec	0, 583819 sec	0, 571932 sec	0, 600156 sec
0, 923761 sec	1, 259966 sec	1, 238100 sec	0, 524322 sec	0, 507360 sec	0, 532698 sec	0, 546756 sec	0, 581483 sec	0, 572735 sec	0, 617072 sec
0, 958106 sec	1, 258993 sec	1, 231796 sec	0, 528669 sec	0, 505099 sec	0, 513247 sec	0, 565180 sec	0, 583825 sec	0, 567552 sec	0, 617621 sec
0, 896600 sec	1, 262943 sec	1, 260274 sec	0, 479046 sec	0, 506773 sec	0, 433628 sec	0, 543874 sec	0, 583703 sec	0, 584357 sec	0, 607670 sec
0, 952472 sec	1, 258085 sec	1, 227160 sec	0, 501857 sec	0, 433152 sec	0, 518463 sec	0, 565219 sec	0, 551202 sec	0, 574296 sec	0, 608915 sec
0, 942811 sec	1, 263012 sec	1, 255304 sec	0, 432871 sec	0, 494802 sec	0, 508660 sec	0, 546665 sec	0, 571549 sec	0, 547957 sec	0, 583699 sec
0, 938746 sec	1, 200559 sec	1, 238532 sec	0, 505851 sec	0, 508550 sec	0, 505579 sec	0, 555608 sec	0, 583244 sec	0, 546593 sec	0, 610322 sec
0, 939823 sec	1, 250237 sec	0, 519888 sec	0, 503037 sec	0, 512505 sec	0, 507595 sec	0, 555592 sec	0, 548677 sec	0, 607445 sec	0, 599903 sec
0, 949349 sec	1, 254027 sec	1, 238607 sec	0, 504790 sec	0, 455390 sec	0, 433747 sec	0, 566347 sec	0, 570468 sec	0, 586045 sec	0, 646494 sec
0, 282618 sec	1, 258688 sec	1, 254524 sec	0, 432951 sec	0, 487684 sec	0, 495541 sec	0, 544958 sec	0, 548914 sec	0, 576114 sec	0, 610789 sec
0, 927540 sec	1, 263594 sec	1, 268718 sec	0, 494364 sec	0, 511639 sec	0, 509039 sec	0, 556392 sec	0, 560395 sec	0, 570832 sec	0, 617718 sec
0, 950162 sec	1, 249077 sec	1, 249614 sec	0, 508559 sec	0, 432818 sec	0, 518921 sec	0, 579979 sec	0, 572345 sec	0, 577406 sec	0, 604477 sec
0, 957432 sec	1, 200548 sec	1, 241971 sec	0, 512535 sec	0, 517945 sec	0, 514032 sec	0, 581631 sec	0, 586537 sec	0, 604020 sec	0, 605615 sec
0, 953030 sec	1, 231155 sec	1, 269397 sec	0, 479581 sec	0, 513265 sec	0, 518957 sec	0, 568586 sec	0, 587337 sec	0, 619096 sec	0, 592766 sec
0, 936773 sec	1, 251000 sec	1, 260397 sec	0, 452470 sec	0, 483684 sec	0, 532101 sec	0, 568929 sec	0, 588074 sec	0, 603744 sec	0, 599903 sec
0, 945499 sec	1, 226881 sec	1, 255932 sec	0, 505048 sec	0, 457273 sec	0, 519676 sec	0, 579623 sec	0, 571150 sec	0, 637598 sec	0, 658258 sec
0, 928810 sec	1, 238068 sec	1, 242737 sec	0, 487699 sec	0, 509140 sec	0, 520413 sec	0, 582854 sec	0, 603563 sec	0, 606720 sec	0, 669444 sec

3. Hasil pengujian dengan *size* 100 – 1000 dengan *number of process* 17

100	200	300	400	500	600	700	800	900	1000
0, 175596 sec	0, 197766 sec	0, 262840 sec	0, 298811 sec	0, 326811 sec	0, 486375 sec	0, 636009 sec	0, 636894 sec	0, 750359 sec	0, 928080 sec
0, 195886 sec	0, 201421 sec	0, 265338 sec	0, 308578 sec	0, 338364 sec	0, 493805 sec	0, 636098 sec	0, 627394 sec	0, 761004 sec	0, 939523 sec
0, 180359 sec	0, 194921 sec	0, 261394 sec	0, 286591 sec	0, 338649 sec	0, 480972 sec	0, 624608 sec	0, 621009 sec	0, 756100 sec	0, 934315 sec
0, 175403 sec	0, 188969 sec	0, 247627 sec	0, 287255 sec	0, 331210 sec	0, 482577 sec	0, 626302 sec	0, 637449 sec	0, 761269 sec	0, 928254 sec
0, 180944 sec	0, 201905 sec	0, 267935 sec	0, 308984 sec	0, 326075 sec	0, 482555 sec	0, 610995 sec	0, 626839 sec	0, 753469 sec	0, 927021 sec
0, 165450 sec	0, 191188 sec	0, 259215 sec	0, 298048 sec	0, 330141 sec	0, 494636 sec	0, 619326 sec	0, 626273 sec	0, 761160 sec	0, 939788 sec
0, 175642 sec	0, 184216 sec	0, 259222 sec	0, 297174 sec	0, 338842 sec	0, 475607 sec	0, 624589 sec	0, 626171 sec	0, 748904 sec	0, 928334 sec
0, 186051 sec	0, 196347 sec	0, 259954 sec	0, 292248 sec	0, 326007 sec	0, 493968 sec	0, 521165 sec	0, 625947 sec	0, 755766 sec	0, 923600 sec
0, 187155 sec	0, 193786 sec	0, 261557 sec	0, 292453 sec	0, 328734 sec	0, 489247 sec	0, 625079 sec	0, 637498 sec	0, 748503 sec	0, 933588 sec
0, 182270 sec	0, 194340 sec	0, 262073 sec	0, 297827 sec	0, 320955 sec	0, 389764 sec	0, 608618 sec	0, 631801 sec	0, 739593 sec	0, 939722 sec
0, 191313 sec	0, 189227 sec	0, 268111 sec	0, 297420 sec	0, 329667 sec	0, 475610 sec	0, 632828 sec	0, 616038 sec	0, 757818 sec	0, 928412 sec
0, 172341 sec	0, 189304 sec	0, 254350 sec	0, 297012 sec	0, 334031 sec	0, 474973 sec	0, 620875 sec	0, 632349 sec	0, 750176 sec	0, 923534 sec
0, 199816 sec	0, 195890 sec	0, 257437 sec	0, 303492 sec	0, 325492 sec	0, 482823 sec	0, 617533 sec	0, 620925 sec	0, 743828 sec	0, 929051 sec
0, 187578 sec	0, 193169 sec	0, 262038 sec	0, 292002 sec	0, 327877 sec	0, 493466 sec	0, 623715 sec	0, 621084 sec	0, 752966 sec	0, 923095 sec
0, 196796 sec	0, 193228 sec	0, 253509 sec	0, 292367 sec	0, 333985 sec	0, 472616 sec	0, 622857 sec	0, 627165 sec	0, 753730 sec	0, 921819 sec
0, 187777 sec	0, 194349 sec	0, 268843 sec	0, 308816 sec	0, 330565 sec	0, 490040 sec	0, 688367 sec	0, 625165 sec	0, 799013 sec	0, 985875 sec
0, 190935 sec	0, 187421 sec	0, 250102 sec	0, 303595 sec	0, 323778 sec	0, 522525 sec	0, 822893 sec	0, 631446 sec	0, 816151 sec	1, 397467 sec

4. Hasil pengujian dengan *size* 100 – 1000 dengan *number of process* 34

100	200	300	400	500	600	700	800	900	1000
1, 114778 sec	0, 945895 sec	0, 312935 sec	0, 298952 sec	0, 347218 sec	0, 759942 sec	0, 697243 sec	0, 373296 sec	0, 411657 sec	0, 461900 sec
1, 118918 sec	0, 945267 sec	0, 311320 sec	0, 312231 sec	0, 234212 sec	0, 766602 sec	0, 683512 sec	0, 389221 sec	0, 406928 sec	0, 467889 sec
1, 120801 sec	0, 961315 sec	0, 294442 sec	0, 297866 sec	0, 340021 sec	0, 747925 sec	0, 696940 sec	0, 389393 sec	0, 382814 sec	0, 464774 sec
1, 105306 sec	0, 958727 sec	0, 295605 sec	0, 295420 sec	0, 234703 sec	0, 757331 sec	0, 673159 sec	0, 389931 sec	0, 412141 sec	0, 448929 sec
1, 105987 sec	0, 945241 sec	0, 312066 sec	0, 299061 sec	0, 355657 sec	0, 760881 sec	0, 688852 sec	0, 384234 sec	0, 388589 sec	0, 449670 sec
1, 120631 sec	0, 943150 sec	0, 279049 sec	0, 282812 sec	0, 344577 sec	0, 763842 sec	0, 681903 sec	0, 373779 sec	0, 401478 sec	0, 451426 sec
1, 121111 sec	0, 952066 sec	0, 293947 sec	0, 313633 sec	0, 347461 sec	0, 767103 sec	0, 697968 sec	0, 389733 sec	0, 397795 sec	0, 445992 sec
1, 090727 sec	0, 959797 sec	0, 290097 sec	0, 306875 sec	0, 349224 sec	0, 758780 sec	0, 670377 sec	0, 375344 sec	0, 404231 sec	0, 443944 sec
1, 104734 sec	0, 937475 sec	0, 298458 sec	0, 305922 sec	0, 340161 sec	0, 760416 sec	0, 681707 sec	0, 389651 sec	0, 407539 sec	0, 468506 sec
1, 120662 sec	0, 929052 sec	0, 299761 sec	0, 305640 sec	0, 329001 sec	0, 761729 sec	0, 696921 sec	0, 384246 sec	0, 407953 sec	0, 448075 sec
1, 122001 sec	0, 939251 sec	0, 291945 sec	0, 233131 sec	0, 339310 sec	0, 759052 sec	0, 681941 sec	0, 359811 sec	0, 394666 sec	0, 468044 sec
1, 099398 sec	0, 962193 sec	0, 284496 sec	0, 304305 sec	0, 350947 sec	0, 768859 sec	0, 690386 sec	0, 389152 sec	0, 412139 sec	0, 457695 sec
1, 099672 sec	0, 936122 sec	0, 297514 sec	0, 292996 sec	0, 340672 sec	0, 752815 sec	0, 672525 sec	0, 373335 sec	0, 400903 sec	0, 468047 sec
1, 098420 sec	0, 951235 sec	0, 307471 sec	0, 304896 sec	0, 363424 sec	0, 764278 sec	0, 677650 sec	0, 378008 sec	0, 406801 sec	0, 441283 sec
1, 105756 sec	0, 946339 sec	0, 315194 sec	0, 308495 sec	0, 345751 sec	0, 756456 sec	0, 681072 sec	0, 383177 sec	0, 403642 sec	0, 452020 sec
1, 097532 sec	0, 957943 sec	0, 298070 sec	0, 312861 sec	0, 341382 sec	0, 746540 sec	0, 680726 sec	0, 371790 sec	0, 401660 sec	0, 446386 sec
1, 106104 sec	0, 945643 sec	0, 304908 sec	0, 289178 sec	0, 334270 sec	0, 752629 sec	0, 682800 sec	0, 353735 sec	0, 407081 sec	0, 437836 sec
1, 104684 sec	0, 961896 sec	0, 294935 sec	0, 312931 sec	0, 347661 sec	0, 758605 sec	0, 683683 sec	0, 375867 sec	0, 401491 sec	0, 456719 sec
1, 121149 sec	0, 959198 sec	0, 297573 sec	0, 292013 sec	0, 359029 sec	0, 764085 sec	0, 698374 sec	0, 356090 sec	0, 410206 sec	0, 468468 sec
1, 092262 sec	0, 945740 sec	0, 279327 sec	0, 294829 sec	0, 352829 sec	0, 759377 sec	0, 664495 sec	0, 355243 sec	0, 403889 sec	0, 458150 sec
1, 106680 sec	0, 943544 sec	0, 294574 sec	0, 299633 sec	0, 330959 sec	0, 769640 sec	0, 679484 sec	0, 366645 sec	0, 402170 sec	0, 468359 sec
1, 098454 sec	0, 952693 sec	0, 290722 sec	0, 283088 sec	0, 351912 sec	0, 753244 sec	0, 675023 sec	0, 390031 sec	0, 397131 sec	0, 451003 sec
1, 094010 sec	0, 960137 sec	0, 298900 sec	0, 312332 sec	0, 341236 sec	0, 756782 sec	0, 690920 sec	0, 373791 sec	0, 376018 sec	0, 453485 sec
1, 116213 sec	0, 930360 sec	0, 300284 sec	0, 314696 sec	0, 363467 sec	0, 747713 sec	0, 661149 sec	0, 370822 sec	0, 377355 sec	0, 465195 sec
1, 111322 sec	0, 929715 sec	0, 278851 sec	0, 306312 sec	0, 345243 sec	0, 743333 sec	0, 680717 sec	0, 376960 sec	0, 412482 sec	0, 450659 sec
1, 100383 sec	0, 958453 sec	0, 300435 sec	0, 304085 sec	0, 337450 sec	0, 746610 sec	0, 693517 sec	0, 359638 sec	0, 389185 sec	0, 450112 sec
1, 114045 sec	0, 944526 sec	0, 297838 sec	0, 233704 sec	0, 331446 sec	0, 750447 sec	0, 696887 sec	0, 367266 sec	0, 402029 sec	0, 465302 sec
1, 097654 sec	0, 926185 sec	0, 310434 sec	0, 304583 sec	0, 347754 sec	0, 757645 sec	0, 684674 sec	0, 372987 sec	0, 398113 sec	0, 457863 sec
1, 107658 sec	0, 960239 sec	0, 298531 sec	0, 298428 sec	0, 355893 sec	0, 744020 sec	0, 682291 sec	0, 360312 sec	0, 396167 sec	0, 440373 sec
1, 099073 sec	0, 945363 sec	0, 305239 sec	0, 305497 sec	0, 344556 sec	0, 758048 sec	0, 663901 sec	0, 355365 sec	0, 391757 sec	0, 451399 sec

1, 113656 sec	0, 928768 sec	0, 311600 sec	0, 289401 sec	0, 347692 sec	0, 751238 sec	0, 677286 sec	0, 366840 sec	0, 379339 sec	0, 450782 sec
1, 102060 sec	0, 947861 sec	0, 289889 sec	0, 290066 sec	0, 349111 sec	0, 770020 sec	0, 658457 sec	0, 372031 sec	0, 398724 sec	0, 439306 sec
1, 109750 sec	0, 955353 sec	0, 290854 sec	0, 297014 sec	0, 368885 sec	1, 785033 sec	0, 682121 sec	0, 380049 sec	0, 408998 sec	0, 473213 sec
1, 112421 sec	0, 947757 sec	0, 305728 sec	0, 304518 sec	0, 367182 sec	1, 789948 sec	0, 685060 sec	0, 380221 sec	0, 404754 sec	0, 460397 sec

5. Hasil pengujian dengan *size* 100 – 1000 dengan *number of process* 51

100	200	300	400	500	600	700	800	900	1000
0, 528837 sec	1, 350609 sec	0, 607476 sec	1, 473562 sec	0, 608064 sec	1, 566047 sec	0, 803565 sec	0, 902277 sec	0, 733805 sec	0, 818556 sec
0, 538977 sec	1, 340069 sec	0, 609237 sec	1, 474870 sec	0, 605130 sec	1, 537823 sec	0, 792321 sec	0, 892165 sec	0, 727406 sec	0, 820072 sec
0, 558146 sec	1, 329761 sec	0, 607489 sec	1, 476700 sec	0, 617164 sec	1, 555065 sec	0, 802903 sec	0, 902274 sec	0, 734695 sec	0, 798934 sec
0, 557611 sec	1, 345945 sec	0, 613980 sec	1, 485969 sec	0, 591003 sec	1, 546515 sec	0, 801778 sec	0, 898361 sec	0, 715002 sec	0, 828375 sec
0, 544263 sec	0, 854098 sec	0, 576387 sec	1, 474058 sec	0, 599065 sec	1, 566055 sec	0, 792851 sec	0, 894361 sec	0, 733766 sec	0, 827559 sec
0, 557715 sec	1, 343053 sec	0, 591327 sec	1, 472605 sec	0, 599141 sec	1, 548957 sec	0, 793266 sec	0, 886634 sec	0, 704022 sec	0, 807357 sec
0, 535281 sec	1, 336987 sec	0, 600589 sec	1, 475660 sec	0, 581432 sec	1, 526317 sec	0, 765197 sec	0, 868371 sec	0, 723852 sec	0, 793454 sec
0, 546012 sec	1, 328052 sec	0, 608273 sec	1, 485868 sec	0, 570877 sec	1, 540434 sec	0, 783759 sec	0, 892954 sec	0, 725231 sec	0, 817329 sec
0, 550424 sec	1, 206612 sec	0, 589358 sec	1, 472605 sec	0, 558590 sec	1, 560226 sec	0, 804357 sec	0, 896243 sec	0, 720921 sec	0, 814692 sec
0, 541143 sec	1, 342465 sec	0, 583369 sec	1, 464113 sec	0, 607556 sec	1, 551549 sec	0, 803178 sec	0, 863628 sec	0, 719115 sec	0, 821785 sec
0, 519928 sec	1, 313468 sec	0, 605558 sec	1, 469069 sec	0, 561495 sec	1, 541193 sec	0, 794113 sec	0, 866376 sec	0, 713663 sec	0, 828785 sec
0, 550586 sec	1, 319522 sec	0, 570719 sec	1, 477721 sec	0, 590991 sec	1, 552867 sec	0, 794110 sec	0, 887571 sec	0, 724744 sec	0, 820774 sec
0, 548718 sec	1, 319869 sec	0, 591283 sec	1, 478311 sec	0, 618027 sec	1, 554927 sec	0, 791036 sec	0, 902302 sec	0, 730141 sec	0, 806473 sec
0, 548718 sec	1, 337487 sec	0, 578171 sec	1, 469288 sec	0, 619505 sec	1, 529206 sec	0, 782948 sec	0, 857585 sec	0, 729179 sec	0, 826116 sec
0, 539423 sec	1, 336078 sec	0, 593658 sec	1, 463813 sec	0, 611235 sec	1, 568957 sec	0, 769242 sec	0, 878067 sec	0, 676003 sec	0, 808352 sec
0, 508596 sec	1, 341858 sec	0, 597738 sec	1, 487557 sec	0, 617533 sec	1, 556542 sec	0, 774307 sec	0, 817058 sec	0, 716162 sec	0, 818973 sec
0, 548705 sec	1, 331898 sec	0, 578620 sec	1, 488466 sec	0, 617137 sec	1, 560518 sec	0, 765883 sec	0, 732407 sec	0, 725918 sec	0, 808247 sec
0, 530593 sec	1, 206858 sec	0, 591561 sec	1, 468342 sec	0, 585614 sec	1, 547244 sec	0, 799925 sec	0, 879033 sec	0, 733956 sec	0, 815276 sec
0, 531663 sec	1, 321932 sec	0, 600508 sec	1, 466804 sec	0, 580644 sec	1, 537000 sec	0, 776522 sec	0, 854804 sec	0, 722984 sec	0, 813818 sec
0, 540796 sec	1, 350609 sec	0, 600960 sec	1, 489200 sec	0, 599777 sec	1, 552861 sec	0, 773032 sec	0, 865628 sec	0, 716570 sec	0, 799230 sec
0, 559865 sec	1, 340082 sec	0, 589564 sec	1, 468182 sec	0, 600030 sec	1, 554915 sec	0, 775888 sec	0, 888149 sec	0, 729328 sec	0, 821819 sec
0, 555324 sec	1, 329764 sec	0, 566135 sec	1, 459836 sec	0, 581526 sec	1, 529203 sec	0, 802012 sec	0, 888287 sec	0, 694914 sec	0, 828782 sec
0, 540718 sec	1, 352498 sec	0, 602722 sec	1, 459827 sec	0, 561974 sec	1, 568841 sec	0, 791461 sec	0, 852468 sec	0, 725879 sec	0, 807478 sec
0, 559431 sec	0, 854084 sec	0, 619682 sec	1, 459829 sec	0, 558790 sec	1, 547448 sec	0, 795462 sec	0, 869847 sec	0, 712860 sec	0, 818536 sec
0, 538291 sec	1, 343906 sec	0, 573412 sec	1, 493096 sec	0, 607555 sec	1, 537813 sec	0, 765676 sec	0, 860012 sec	0, 696722 sec	0, 798850 sec

0, 551186 sec	1, 313445 sec	0, 591693 sec	1, 472590 sec	0, 561979 sec	1, 555150 sec	0, 779157 sec	0, 817062 sec	0, 725392 sec	0, 810368 sec
0, 547361 sec	1, 336010 sec	0, 583949 sec	1, 461939 sec	0, 590027 sec	1, 546531 sec	0, 757691 sec	0, 732717 sec	0, 705819 sec	0, 828399 sec
0, 537931 sec	1, 320120 sec	0, 593649 sec	1, 472963 sec	0, 590661 sec	1, 567049 sec	0, 803389 sec	0, 857083 sec	0, 727658 sec	0, 795193 sec
0, 518934 sec	1, 336543 sec	0, 598063 sec	1, 493277 sec	0, 616819 sec	1, 548863 sec	0, 794112 sec	0, 900639 sec	0, 735723 sec	0, 828692 sec
0, 547056 sec	1, 338798 sec	0, 570883 sec	1, 472931 sec	0, 618159 sec	1, 540469 sec	0, 794022 sec	0, 886629 sec	0, 693573 sec	0, 807229 sec
0, 527495 sec	1, 338334 sec	0, 594666 sec	1, 451956 sec	0, 605120 sec	1, 540411 sec	0, 792920 sec	0, 867565 sec	0, 714421 sec	0, 828440 sec
0, 529922 sec	1, 331816 sec	0, 601431 sec	1, 457893 sec	0, 607739 sec	1, 564709 sec	0, 772954 sec	0, 890129 sec	0, 707727 sec	0, 807718 sec
0, 537045 sec	1, 207605 sec	0, 604905 sec	1, 476444 sec	0, 562498 sec	1, 562293 sec	0, 782823 sec	0, 899081 sec	0, 707481 sec	0, 818137 sec
0, 556435 sec	1, 320411 sec	0, 590436 sec	1, 474128 sec	0, 581095 sec	1, 553001 sec	0, 797093 sec	0, 893714 sec	0, 726947 sec	0, 817389 sec
0, 560337 sec	1, 351278 sec	0, 576023 sec	1, 494596 sec	0, 618986 sec	1, 553720 sec	0, 764176 sec	0, 899975 sec	0, 733695 sec	0, 815623 sec
0, 555813 sec	1, 330944 sec	0, 602924 sec	1, 454485 sec	0, 615970 sec	1, 556033 sec	0, 782184 sec	0, 867429 sec	0, 696501 sec	0, 797232 sec
0, 536933 sec	1, 346802 sec	0, 608688 sec	1, 450813 sec	0, 605123 sec	1, 547440 sec	0, 762754 sec	0, 867244 sec	0, 721763 sec	0, 822757 sec
0, 550110 sec	0, 851864 sec	0, 603781 sec	1, 446221 sec	0, 611007 sec	1, 548418 sec	0, 803086 sec	0, 884674 sec	0, 716219 sec	0, 828572 sec
0, 528078 sec	1, 316088 sec	0, 574238 sec	1, 455062 sec	0, 616379 sec	1, 527411 sec	0, 794267 sec	0, 887280 sec	0, 693545 sec	0, 806267 sec
0, 531419 sec	1, 313808 sec	0, 591254 sec	1, 452770 sec	0, 579094 sec	1, 540984 sec	0, 794021 sec	0, 847684 sec	0, 725882 sec	0, 809991 sec
0, 553540 sec	1, 312746 sec	0, 584000 sec	1, 474282 sec	0, 600512 sec	1, 546083 sec	0, 787157 sec	0, 871098 sec	0, 725328 sec	0, 798633 sec
0, 527404 sec	1, 325708 sec	0, 594156 sec	1, 465010 sec	0, 600522 sec	1, 552842 sec	0, 765588 sec	0, 875955 sec	0, 702424 sec	0, 813677 sec
0, 531832 sec	1, 320355 sec	0, 597249 sec	1, 487768 sec	0, 581511 sec	1, 553287 sec	0, 783193 sec	0, 817843 sec	0, 723073 sec	0, 795696 sec
0, 551880 sec	1, 339784 sec	0, 602040 sec	1, 455138 sec	0, 574555 sec	1, 529569 sec	0, 776682 sec	0, 731030 sec	0, 717681 sec	0, 802699 sec
0, 523499 sec	1, 330842 sec	0, 617366 sec	1, 496040 sec	0, 556719 sec	1, 568986 sec	0, 765993 sec	0, 877202 sec	0, 690363 sec	0, 795199 sec
0, 531028 sec	1, 345974 sec	0, 607178 sec	1, 477935 sec	0, 608772 sec	1, 556043 sec	0, 777895 sec	0, 904678 sec	0, 767416 sec	0, 806727 sec
0, 548631 sec	1, 308228 sec	0, 611088 sec	1, 476198 sec	0, 612778 sec	1, 549040 sec	0, 801101 sec	0, 878085 sec	0, 767597 sec	0, 821390 sec
0, 552368 sec	1, 315568 sec	0, 611323 sec	1, 476988 sec	0, 589826 sec	1, 543370 sec	0, 787570 sec	0, 863249 sec	0, 772065 sec	0, 816474 sec
0, 551803 sec	1, 406974 sec	0, 758017 sec	1, 487690 sec	0, 575440 sec	1, 527729 sec	0, 765254 sec	0, 888949 sec	1, 003322 sec	1, 005784 sec
0, 543626 sec	1, 386130 sec	0, 759065 sec	1, 478868 sec	0, 596522 sec	1, 543013 sec	0, 761966 sec	0, 854553 sec	0, 982337 sec	1, 008055 sec
0, 529526 sec	1, 384251 sec	0, 760367 sec	1, 477664 sec	0, 591229 sec	1, 526134 sec	0, 781790 sec	0, 883237 sec	0, 993942 sec	1, 008549 sec

6. Hasil pengujian dengan *size* 100 – 1000 dengan *number of process* 68

100	200	300	400	500	600	700	800	900	1000
0. 947832 sec	0. 750741 sec	0. 864016 sec	0. 958163 sec	0. 800190 sec	0. 870312 sec	1. 060409 sec	0. 970367 sec	1. 067394 sec	1. 046733 sec
0. 925330 sec	0. 751170 sec	0. 860120 sec	0. 961623 sec	0. 812791 sec	0. 842115 sec	1. 061870 sec	0. 988704 sec	1. 094780 sec	1. 023008 sec
0. 945924 sec	0. 731938 sec	0. 853753 sec	0. 922195 sec	0. 888007 sec	0. 855302 sec	1. 061615 sec	0. 949603 sec	1. 093200 sec	1. 047252 sec
0. 908655 sec	0. 744617 sec	0. 779801 sec	0. 946090 sec	0. 887335 sec	0. 871117 sec	1. 048954 sec	0. 970213 sec	1. 094431 sec	1. 045546 sec

0.937187 sec	0.740188 sec	0.804058 sec	0.949549 sec	0.863478 sec	0.851619 sec	1.048011 sec	0.991164 sec	1.064784 sec	0.997961 sec
0.894536 sec	0.730501 sec	0.821072 sec	0.954554 sec	0.847212 sec	0.869494 sec	1.028453 sec	0.978869 sec	1.092745 sec	1.022702 sec
0.915644 sec	0.718761 sec	0.860101 sec	0.924105 sec	0.890356 sec	0.855044 sec	1.058772 sec	0.969053 sec	1.088609 sec	1.023589 sec
0.918715 sec	0.755945 sec	0.792466 sec	0.932151 sec	0.862564 sec	0.844238 sec	1.046169 sec	0.992789 sec	1.086789 sec	1.031184 sec
0.919731 sec	0.740790 sec	0.863498 sec	0.964650 sec	0.868357 sec	0.846100 sec	1.059540 sec	0.948754 sec	1.082674 sec	0.997931 sec
0.905736 sec	0.753470 sec	0.864809 sec	0.959115 sec	0.873890 sec	0.855625 sec	1.058011 sec	0.964913 sec	1.089721 sec	1.023905 sec
0.920342 sec	0.741522 sec	0.842829 sec	0.909537 sec	0.869705 sec	0.870977 sec	1.059668 sec	0.987094 sec	1.079095 sec	1.044020 sec
0.932628 sec	0.740077 sec	0.860251 sec	0.950779 sec	0.876713 sec	0.850024 sec	1.047052 sec	0.982023 sec	1.090746 sec	1.028887 sec
0.926346 sec	0.730447 sec	0.793879 sec	0.952131 sec	0.875887 sec	0.832556 sec	1.017298 sec	0.996416 sec	1.068772 sec	1.039775 sec
0.943419 sec	0.716023 sec	0.862852 sec	0.943514 sec	0.798895 sec	0.868771 sec	1.030473 sec	0.976676 sec	1.076400 sec	1.017511 sec
0.928258 sec	0.754875 sec	0.865036 sec	0.962524 sec	0.809599 sec	0.855014 sec	1.013371 sec	0.968322 sec	1.063931 sec	1.045197 sec
0.924155 sec	0.740631 sec	0.842699 sec	0.921928 sec	0.883487 sec	0.842436 sec	1.028453 sec	0.989460 sec	1.055189 sec	1.016808 sec
0.941930 sec	0.738666 sec	0.847978 sec	0.934314 sec	0.884430 sec	0.846040 sec	1.042943 sec	0.995346 sec	1.089381 sec	1.048915 sec
0.941877 sec	0.751418 sec	0.776229 sec	0.954268 sec	0.865601 sec	0.864883 sec	1.022887 sec	0.949841 sec	1.082257 sec	1.034716 sec
0.930229 sec	0.752122 sec	0.803990 sec	0.907146 sec	0.842324 sec	0.842358 sec	1.063101 sec	0.973975 sec	1.089891 sec	1.042259 sec
0.940556 sec	0.735006 sec	0.850510 sec	0.939893 sec	0.861326 sec	0.871798 sec	1.044197 sec	0.994112 sec	1.058169 sec	0.994577 sec
0.892295 sec	0.753092 sec	0.821066 sec	0.906474 sec	0.867316 sec	0.855003 sec	1.056502 sec	0.967738 sec	1.072813 sec	1.022521 sec
0.892826 sec	0.730663 sec	0.854227 sec	0.950055 sec	0.879584 sec	0.871062 sec	1.055957 sec	0.992589 sec	1.052536 sec	1.047289 sec
0.912541 sec	0.742009 sec	0.863907 sec	0.946863 sec	0.852831 sec	0.846262 sec	1.057127 sec	0.960993 sec	1.092638 sec	1.022331 sec
0.894335 sec	0.750749 sec	0.851174 sec	0.923847 sec	0.845151 sec	0.845439 sec	1.042922 sec	0.996301 sec	1.092672 sec	1.023287 sec
0.922369 sec	0.752156 sec	0.778251 sec	0.946002 sec	0.890222 sec	0.832508 sec	1.047418 sec	0.971190 sec	1.077023 sec	1.024538 sec
0.902063 sec	0.732708 sec	0.804765 sec	0.927595 sec	0.872600 sec	0.857730 sec	1.025032 sec	0.980375 sec	1.098579 sec	1.022665 sec
0.915431 sec	0.753739 sec	0.852843 sec	0.962368 sec	0.857732 sec	0.863803 sec	1.039945 sec	0.971823 sec	1.080221 sec	1.047602 sec
0.929080 sec	0.722087 sec	0.823269 sec	0.918507 sec	0.873824 sec	0.850089 sec	1.030092 sec	0.947027 sec	1.092054 sec	1.025216 sec
0.929046 sec	0.728544 sec	0.794176 sec	0.915937 sec	0.855046 sec	0.845069 sec	1.051986 sec	0.975254 sec	1.078511 sec	1.013769 sec
0.942159 sec	0.741223 sec	0.827739 sec	0.959308 sec	0.867214 sec	0.846570 sec	1.041758 sec	0.972837 sec	1.058672 sec	1.037022 sec
0.937956 sec	0.741522 sec	0.863547 sec	0.958598 sec	0.873261 sec	0.874148 sec	1.062537 sec	0.995107 sec	1.089263 sec	1.000399 sec
0.920181 sec	0.739822 sec	0.837774 sec	0.911764 sec	0.865287 sec	0.856679 sec	1.062007 sec	0.971952 sec	1.075285 sec	1.027070 sec
0.937272 sec	0.730359 sec	0.844847 sec	0.960380 sec	0.888519 sec	0.844132 sec	1.026595 sec	0.954937 sec	1.072519 sec	1.022316 sec
0.893245 sec	0.721144 sec	0.851600 sec	0.958303 sec	0.879512 sec	0.850989 sec	1.059748 sec	0.986060 sec	1.098540 sec	1.047189 sec
0.904257 sec	0.745652 sec	0.783255 sec	0.922017 sec	0.783655 sec	0.875739 sec	1.004869 sec	0.992851 sec	1.088085 sec	1.021784 sec
0.917675 sec	0.738866 sec	0.807554 sec	0.933699 sec	0.815406 sec	0.842962 sec	1.029242 sec	0.997664 sec	1.055221 sec	1.023879 sec

0.906743 sec	0.743993 sec	0.853844 sec	0.965514 sec	0.884795 sec	0.874500 sec	1.055748 sec	0.929829 sec	1.093898 sec	1.052816 sec
0.924134 sec	0.734798 sec	0.824697 sec	0.966551 sec	0.879552 sec	0.859072 sec	1.011694 sec	0.949588 sec	1.071905 sec	1.017223 sec
0.903648 sec	0.751607 sec	0.830264 sec	0.909600 sec	0.860230 sec	0.875614 sec	1.043868 sec	0.982880 sec	1.075075 sec	1.026269 sec
0.911865 sec	0.735920 sec	0.866558 sec	0.903276 sec	0.801262 sec	0.850068 sec	1.059856 sec	0.952685 sec	1.093250 sec	1.022243 sec
0.952510 sec	0.756081 sec	0.830891 sec	0.945786 sec	0.812917 sec	0.842976 sec	1.062642 sec	0.996194 sec	1.075816 sec	1.040686 sec
0.924777 sec	0.720814 sec	0.868241 sec	0.957458 sec	0.864843 sec	0.834985 sec	1.033961 sec	0.958209 sec	1.085575 sec	1.004505 sec
0.951166 sec	0.734412 sec	0.857316 sec	0.929939 sec	0.850785 sec	0.831493 sec	1.062823 sec	1.025585 sec	1.128167 sec	0.978067 sec
0.923963 sec	0.729552 sec	0.851538 sec	0.950520 sec	0.878071 sec	0.860050 sec	1.050780 sec	1.023803 sec	1.105567 sec	1.047494 sec
0.899173 sec	0.741827 sec	0.865322 sec	0.945163 sec	0.861856 sec	0.860397 sec	1.059972 sec	1.029094 sec	1.137591 sec	1.012091 sec
0.949202 sec	0.743423 sec	0.790331 sec	0.921616 sec	0.878247 sec	0.845865 sec	1.051681 sec	1.005063 sec	1.143442 sec	1.028741 sec
0.936821 sec	0.733518 sec	0.862889 sec	0.936719 sec	0.879355 sec	0.871635 sec	1.041427 sec	1.003955 sec	1.134785 sec	1.056691 sec
0.932414 sec	0.758948 sec	0.855302 sec	0.898683 sec	0.869913 sec	0.846685 sec	1.034797 sec	1.008018 sec	1.097016 sec	1.020000 sec
0.949049 sec	0.743938 sec	0.848317 sec	0.954965 sec	0.888888 sec	0.832148 sec	1.063747 sec	1.032577 sec	1.109484 sec	1.054979 sec
0.901883 sec	0.749389 sec	0.869185 sec	0.945056 sec	0.877107 sec	0.833109 sec	1.067583 sec	1.006115 sec	1.122129 sec	1.039215 sec
0.891747 sec	0.755598 sec	0.827883 sec	0.937775 sec	0.893246 sec	0.875652 sec	1.036987 sec	1.030474 sec	1.139624 sec	1.102555 sec
0.915450 sec	0.731905 sec	0.868007 sec	0.914646 sec	0.873628 sec	0.841839 sec	1.062020 sec	1.017870 sec	1.136410 sec	1.066413 sec
0.909698 sec	0.715501 sec	0.846829 sec	0.954997 sec	0.885320 sec	0.835429 sec	1.081103 sec	1.009860 sec	1.137319 sec	1.073225 sec
0.922178 sec	0.743408 sec	0.871229 sec	0.955887 sec	0.852446 sec	0.854477 sec	1.102474 sec	1.000247 sec	1.145784 sec	1.028920 sec
0.904211 sec	0.727713 sec	0.851022 sec	0.956046 sec	0.876582 sec	0.828817 sec	1.104129 sec	1.035993 sec	1.123454 sec	1.076720 sec
0.928576 sec	0.714382 sec	0.829837 sec	0.952258 sec	0.863722 sec	0.874023 sec	1.067451 sec	1.034843 sec	1.140112 sec	1.057969 sec
0.934754 sec	0.755926 sec	0.896735 sec	0.969010 sec	0.914926 sec	0.849580 sec	1.101372 sec	0.992453 sec	1.144858 sec	1.061418 sec
0.936553 sec	0.744051 sec	0.896760 sec	0.956965 sec	0.893891 sec	0.874036 sec	1.095676 sec	0.999156 sec	1.147285 sec	1.062333 sec
0.921141 sec	0.761832 sec	0.896342 sec	0.962678 sec	0.929447 sec	0.852057 sec	1.091476 sec	0.994600 sec	1.162088 sec	1.095316 sec
0.902965 sec	0.753068 sec	0.898568 sec	0.955638 sec	0.906335 sec	0.865709 sec	1.100834 sec	1.025015 sec	1.181054 sec	1.010149 sec
0.931598 sec	0.793940 sec	0.895801 sec	0.995472 sec	0.964882 sec	0.888667 sec	1.097416 sec	1.050151 sec	1.994908 sec	1.059968 sec
0.952564 sec	0.785818 sec	0.898148 sec	0.992198 sec	0.933733 sec	0.907368 sec	1.071486 sec	1.023802 sec	2.001270 sec	1.108430 sec
0.988630 sec	0.786084 sec	0.888008 sec	0.982046 sec	0.986063 sec	0.880859 sec	1.097020 sec	1.046677 sec	2.004890 sec	1.144128 sec
0.985899 sec	0.796029 sec	0.900246 sec	0.994430 sec	0.988808 sec	0.921153 sec	1.112104 sec	1.097525 sec	2.004107 sec	1.137812 sec
0.997147 sec	0.823918 sec	0.919865 sec	1.005149 sec	0.958630 sec	0.899532 sec	1.097202 sec	1.803287 sec	1.981806 sec	1.217940 sec
1.018640 sec	0.807956 sec	0.932122 sec	1.023680 sec	0.956307 sec	0.943940 sec	1.100609 sec	1.798896 sec	2.003996 sec	1.219969 sec
1.021361 sec	0.815944 sec	0.947541 sec	1.032985 sec	0.958843 sec	0.938413 sec	1.126214 sec	1.776467 sec	1.986052 sec	1.206260 sec
1.043004 sec	0.858571 sec	0.967835 sec	1.049078 sec	0.951422 sec	0.972234 sec	1.158967 sec	1.805215 sec	2.024982 sec	1.239576 sec