**PROPOSAL TUGAS AKHIR**

**IMPLEMENTASI KOMPUTASI PARALEL PADA JARINGAN LAN MENGGUNAKAN ALGORITMA PEDJADWALAN TUGAS *WEIGHT ROUND ROBIN***

**KOMPETENSI JARINGAN KOMPUTER**

****

**I NYOMAN BUDAYASA**

**NIM. 1208605032**

**PROGRAM STUDI TEKNIK INFORMATIKA JURUSAN ILMU KOMPUTER FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM UNIVERSITAS UDAYANA**

**BUKIT JIMBARAN**

**2015**

###### LEMBAR PERSETUJUAN UJIAN PROPOSALTUGASAKHIR

Judul Skripsi : Implementasi Komputasi Parallel Pada Jaringan LAN Menggunakan Algoritma Penjadwalan Tugas Weight Round Robin.

Kompetensi : Jaringan Komputer

Nama : I Nyoman Budayasa

NIM : 1208605032

Tanggal disetujui :

###### KATA PENGANTAR

Proposal penelitian dengan judul “Implementasi Komputasi Parallel Pada Jaringan LAN Menggunakan Algoritma Penjadwalan Tugas Weight Round Robin” ini disusun dalam rangkaian kegiatan pelaksanaan Tugas Akhir di Jurusan Ilmu Komputer FMIPA UNUD. Sehubung dengan telah terselesaikannya proposal ini, maka diucapkan terima kasih dan penghargaan kepada berbagai pihak yang telah membantu pengusul, antara lain:

1. Bapak-bapak dan ibu-ibu dosen di Jurusan Ilmu Komputer yang telah meluangkan waktu turut memberikan saran dan masukan dalam penyempurnaan proposal ini.
2. Komisi Tugas Akhir Jurusan Ilmu Komputer FMIPA UNUD, yang telah memberikan petunjuk dalam penyusunan proposal tugas akhir ini.
3. Rekan-rekan mahasiswa yang telah memberi dukungan, motivasi, semangat dan kerja sama dalam pembuatan proposal tugas akhir ini.
4. Keluarga dan kerabat serta semua pihak yang turut serta memberi dukungan sehingga laporan ini dapat diselesaikan sesuai dengan waktu yang ditentukan.

Penulis menyadari bahwa proposal ini masih belum sempurna, untuk itu kritik dan saran yang bermanfaat demi kesempurnaan tugas akhir ini sangat penulis harapkan.

Denpasar, 24 Desember 2014

Penulis

###### DAFTAR ISI

[LEMBAR PERSETUJUAN UJIAN PROPOSALTUGASAKHIR ii](#_Toc416858899)

[KATA PENGANTAR iii](#_Toc416858900)

[DAFTAR ISI iv](#_Toc416858901)

[DAFTAR GAMBAR v](#_Toc416858902)

[1. Latar Belakang 6](#_Toc416858903)

[2. Rumusan Masalah 7](#_Toc416858904)

[3. Tujuan Penelitian 7](#_Toc416858905)

[4. Batasan Masalah 7](#_Toc416858906)

[5. Manfaat Penelitian 8](#_Toc416858907)

[6. Tinjauan Pustaka 8](#_Toc416858908)

[6.1 Bahasa C 8](#_Toc416858909)

[6.2 MPI 8](#_Toc416858910)

[6.3 Komputasi Paralel 10](#_Toc416858911)

[6.4 Penjadwalan CPU 11](#_Toc416858912)

[6.5 Algoritma Weight Round Robin 12](#_Toc416858913)

[7. Metodologi Penelitian 13](#_Toc416858914)

[7.1 Metode Pengumpulan Data 13](#_Toc416858915)

[7.1.1 Studi Pustaka 13](#_Toc416858916)

[7.1.2 Studi Literatur 13](#_Toc416858917)

[7.1.3 Rancangan Sistem 13](#_Toc416858918)

[DAFTAR PUSTAKA 15](#_Toc416858919)

###### DAFTAR GAMBAR

Gambar 6.1 Arsitektur MPICH…………………………………………………...4

Gambar 6.2 Tipe-tipe penjadwalan ………………………………………………6

Gambar 6.3 Rancangan Sistem …………………………………………………...9

# Latar Belakang

Pada saat ini perkembangan teknologi komputer sudah semakin pesat; tidak jarang pada teknologi yang terus berkembang ini, diperlukan perangkat yang handal dan cepat untuk mengatasi kinerja komputer yang semakin berat. Sebagai contoh untuk melakukan *rendering* sebuah video dengan ukuran 1GB dengan cepat pasti membutuhkan waktu yang lama pada sebuah komputer dengan kemampuan prosessing yang standar; untuk itulah dibutuhkan sebuah komputer dengan kemampuan *prosessing* sangat tinggi, yakni dengan *supercomputer.* Untuk mendapatkan sebuah *supercomputer,* dibutuhkan biaya yang sangat mahal. Mengatasi hal tersebut kita dapat menggunakan sebuah cara, yakni menggabungkan beberapa perangkat dengan kecepatan *prosessing* dan *Random Access Memory* (RAM) yang standar untuk melakukan suatu tugas yang berat tersebut, contohnya melakukan *rendering* video. Selain itu, teknologi yang sangat berkembang saat ini adalah teknologi jaringan komputer; di mana dengan teknologi jaringan ini, sejumlah komputer akan dapat saling berkomunikasi dan membentuk suatu kesatuan jaringan yang luas. Dengan adanya perkembangan tersebut, muncullah ide untuk melakukan komputasi dengan dilakukan secara paralel dengan memanfaatkan teknologi jaringan komputer tersebut.

*Parallel prosessing computation* adalah proses atau pekerjaan komputasi pada komputer dengan memakai suatu bahasa pemrograman; yang dijalankan secara paralel pada saat bersamaan dan dihubungkan melalui sebuah jaringan komputer. Secara umum komputasi paralel diperlukan untuk meningkatkan kecepatan komputasi bila dibandingkan dengan pemakaian komputasi pada komputer tunggal.

Penggunaan *parallel prosessing computation* atau komputasi pemrosesan paralel, merupakan pilihan tepat pada saat ini, untuk pengolahan data yang berukuran besar dan berjumlah banyak. Hal ini apabila dibandingkan dengan membeli suatu *superkomputer* yang harganya sangat mahal, maka penggunaan komputasi pemrosesan paralel merupakan pilihan yang sangat tepat untuk pengolahan data tersebut.

Kinerja dan pemrosesan komputasi paralel ini dipengaruhi oleh penjadwalan yang digunakan. Penjadwalan ini sangat dibutuhkan untuk mengatur tugas–tugas yang akan dikerjakan oleh *client*. Dengan penjadwalan yang tepat, maka tugas–tugas tersebut dapat dilayani sesuai dengan kebutuhan; dan pemakaian prosessor bisa dioptimalkan sehingga dapat meningkatkan performa dari komputasi.

# Rumusan Masalah

Permasalahan yang akan dibahas dalam proposal ini adalah “Bagaimana algoritma penjadwalan weight round robin dapat meningkatkan kinerja dari komputasi pemrosesan paralel berbasis multi agent?”

# Tujuan Penelitian

Tujuan penelitian yang dibahas dalam proposal ini adalah

Mengetahui peningkatan kinerja komputasi paralel dengan algoritma penjadwalan weight round robin

# Batasan Masalah

Adapun batasan masalah yang dibahas dalam proposal ini adalah sebagai berikut:

1. Bahasa pemrograman yang digunakan C.
2. Menggunakan perangkat bantu MPI (Message Passing Interface).
3. Titik berat pada pembuatan penjadwalan rangkaian tugas ke komputer simpul dengan mengguakan algoritma *weight round robin.*
4. Karena penekanan pada poin 3, spesifikasi dari setiap node yang melakukan prosessing berbeda.

# Manfaat Penelitian

Adapun manfaat yang nantinya diharapkan dari penelitian ini adalah sebagai berikut :

1. Dapat meningkatkan kinerja jaringan yang menerapkan komputasi parallel.
2. Memberikan penulis dalam melakukan komputasi parallel.

# Tinjauan Pustaka

## 6.1 Bahasa C

Bahasa pemrograman C merupakan salah satu bahasa pemrograman komputer. Dibuat pada tahun 1972 oleh Dennis Ritchie untuk Sistem Operasi Unix di Bell Telephone Laboratories.

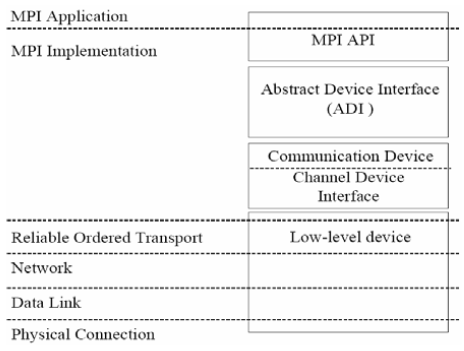
Meskipun C dibuat untuk memprogram sistem dan jaringan komputer namun bahasa ini juga sering digunakan dalam mengembangkan software aplikasi. C juga banyak dipakai oleh berbagai jenis platform sistem operasi dan arsitektur komputer, bahkan terdapat beberepa compiler yang sangat populer telah tersedia. C secara luar biasa memengaruhi bahasa populer lainnya, terutama C++ yang merupakan extensi dari C.

## 6.2 MPI

Dalam implementasinya MPI menggunakan fungsi-fungsi pustaka yang dapat dipanggil dari program C, C++, atau Fortran. Hampir sama dengan PVM, MPI juga portable untuk berbagai arsitektur. Salah satu implementasi terbaru pada saat penulisan makalah ini yang berjalan pada lingkungan Windows adalah MPICH. Versi terakhirnya adalah MPICH versi 1.2.5, yang tersedia secara bebas di ftp://ftp.mcs.anl.gov/pub/mpi/nt/mpich.nt.1.2.5.exe.

Ditinjau dari sisi aplikasi, MPI hanya dapat digunakan dengan model single program multiple data (SPMD), sedangkan PVM dapat digunakan dengan model SPMD maupun multiple program multiple data (MPMD). Model SPMD secara fisik ditunjukkan dengan program master dan slave yang menyatu, sedangkan MPMD ditunjukkan dengan program master dan slave yang terpisah, sehingga slave dapat mengerjakan tugas yang berbeda-beda antara satu node dengan node lainnya.

Arsitektur MPICH ditunjukkan pada Gambar 1. Abstraksi program aplikasi yang dibuat oleh user dinyatakan pada lapisan API. Fungsi-fungsi pustaka yang tersedia pada MPI dinyatakan dalam header mpi.h. Pengaktifan MPI dimulai dengan menjalankan perintah MPI\_Init(&argc, &argv); pada program utama, dilanjutkan dengan menentukan ranking dari tiap node yang menjalankan program aplikasi dengan perintah MPI\_Comm\_rank(MPI\_COMM\_WORLD, &my\_rank); my\_rank adalah bilangan bulat positif, bernilai nol berarti program berjalan pada komputer master, sebaliknya bernilai tidak sama dengan nol berarti program berjalan pada komputer slave. MPI\_COMM\_WORLD adalah konstanta yang telah terdefinisi untuk mengendalikan proses-proses yang ada pada saat MPI dimulai. Untuk mengetahui jumlah prosesor (node) yang aktif digunakan perintah MPI\_Comm\_size(MPI\_COMM\_WORLD, &p).



**Gambar 6.1 Arsitektur MPICH**

Komunikasi dilakukan secara berurutan dari lapisan teratas sampai ke lapisan fisik, pada sisi penerima berlaku sebaliknya, yaitu dari lapisan fisik ke atas. Misalnya ada perintah MPI\_Send(message, strlen(message)+1, MPI\_CHAR,dest, tag, MPI\_COMM\_WORLD); maka pesan (message) sepanjang strlen(message)+1 dan bertipe karakter (MPI\_CHAR) dikirim ke prosesor tujuan (dest). Pada lapisan ADI pesan tersebut diterima oleh Send\_handle yaitu pengendali pengiriman data pada lapisan ADI, kemudian pada lapisan Channel Device paket tersebut diterima dengan pengendali MPID\_SendControl dan MPID\_SendChannel [8]. Selanjutnya, pada lapisan lowlevel device komunikasi dilakukan dengan protokol yang tersedia. Misalnya MPICH berjalan pada Windows 2000, maka protokol yang digunakan adalah TCP/IP. Pada sisi penerima, lapisan komunikasi atau channel device interface memiliki pengendali MPID\_ControlMsgAvail dan MPID\_RecvAnyControl. Kedua pengendali ini meneruskan paket ke lapisan ADI. Pada lapisan ini, terdapat dua pengendali, yaitu: PostedRecv\_Handles dan UnexpectedRecv\_Handles, masing-masing digunakan untuk mengetahui pengiriman paket yang terkirim dengan benar dan yang salah. Selanjutnya, pada lapisan aplikasi, data diterima dengan perintah MPI\_Recv (message, len, MPI\_CHAR, source, tag, MPI\_COMM\_WORLD, &status); dan untuk mengakhiri MPI digunakan perintah MPI\_Finalize();

## 6.3 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, dll) 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) dll.

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 perangkat lunak pendukung yang biasa disebut sebagai *middleware* yang berperan untuk mengatur distribusi pekerjaan antar node dalam satu mesin paralel. Selanjutnya pemakai harus membuat pemrograman paralel untuk merealisasikan komputasi

Alasan utama penggunaan komputasi parallel adalah :

1. Dapat menghemat waktu
2. Dapat mengerjakan suatu tugas dengan kapasitas besar
3. Dapat menghemat biaya
4. Dapat mengatasi keterbatasan fisik dari komputasi serial

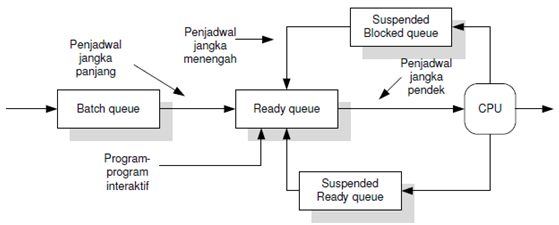
## 6.4 Penjadwalan CPU

Penjadwalan merupakan salah satu fungsi dasar dari sistem operasi. Hampir semua sumber daya komputer dijadwalkan sebelum digunakan. Ketika CPU mengalami waktu idle, sistem operasi harus memilih salah satu proses untuk masuk kedalam antrian yang akan untuk dieksekusi. Pemilihan tersebut dilakukan oleh penjadwal jangka pendek atau penjadwal CPU. Ada tiga tipe penjadwal yang berada bersama pada sistem operasi kompleks, yaitu:

a. Penjadwal jangka pendek yang bertugas menjadwalkan alokasi pemroses di antara proses-proses yang telah siap di memori utama.

b. Penjadwal jangka menengah akan menangani serta mengendalikan transisi dari *suspended-toready* dari proses-proses *swapping*.

c. Penjadwal jangka panjang bekerja terhadap antrian *batch* dan memilih *batch* berikutnya yang harus dieksekusi.



**Gambar 6.2 Tipe-tipe penjadwalan**

## 6.5 Algoritma Weight Round Robin

Penjadwalan Weight Round Robin dirancang untuk lebih menangani server dengan kapasitas pengolahan yang berbeda. Setiap server dapat diberikan berat, nilai integer yang menunjukkan kapasitas pemrosesan. Server dengan bobot yang lebih tinggi menerima sambungan baru pertama dibandingkan mereka yang kurang berat, dan server dengan bobot yang lebih tinggi mendapatkan lebih banyak koneksi dari mereka yang kurang berat dan server dengan bobot yang sama mendapatkan koneksi yang sama. Kode pseudo penjadwalan Weight Round Robin adalah sebagai berikut:

Supposing that there is a server set *S* = {S0, S1, …, Sn-1};

W(Si) indicates the weight of Si;

*i* indicates the server selected last time, and *i* is initialized with -1;

*cw* is the current weight in scheduling, and cw is initialized with zero;

max(S) is the maximum weight of all the servers in S;

gcd(S) is the greatest common divisor of all server weights in S;

while (true) {

i = (i + 1) mod n;

if (i == 0) {

cw = cw - gcd(S);

if (cw <= 0) {

cw = max(S);

if (cw == 0)

return NULL;

}

}

if (W(Si) >= cw)

return Si;

}

Sebagai contoh, server A, B dan C, memiliki bobot, 4, 3, 2 masing-masing, urutan penjadwalan akan menjadiAABABCABC dalam periode penjadwalan (mod sum (Wi)).

# Metodologi Penelitian

Dalam Implementasi Komputasi Parallel Pada Jaringan LAN Menggunakan Algoritma Penjadwalan Tugas Weight Round Robin, terdapat beberapa metode penelitian yang digunakan yakni :

## 7.1 Metode Pengumpulan Data

Dalam penyusunan proposal ini, diperlukan data-data informasi yang relative lengkap sebagi bahan yang dapat mendukung kebenaran materi uraian pembahasan. Oleh karena itu dilakukan pengumpulan data untuk mendapatkan informasi atau bahasa materi yang diperlukan. Adapun metode pengumpulan data-data dan informasi yang digunakan adalah sebagai berikut :

### 7.1.1 Studi Pustaka

Metode pustaka, yakni pengumpulan data dan informasi dengan cara membaca buku-buku referensi, *e-book* dan *web site*. Dokumen-dokumen yang termasuk didalamnya yaitu penelitian-penelitian terdahulu, buku, artikel dan jurnal yang berkaitan dengan objek penelitian.

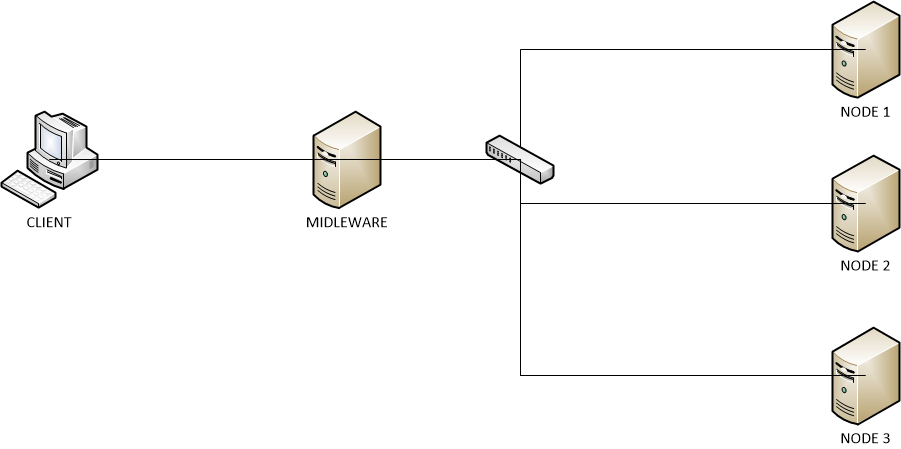
### 7.1.2 Studi Literatur

Melakukan studi perbandingan dan analisis antara aplikasi yang pernah dibuat oleh seseorang dengan aplikasi yang penulis akan buat. Terlampir.

### 7.1.3 Rancangan Sistem

Berikut adalah rancangan sistem yang penulis akan buat. Adapun langkah-langkah melakukan pemrosesan tugas adalah sebagai berikut :

1. Klien mengirimkan request tugas kepada middleware.
2. Middleware melakukan penjadwalan tugas berdasarkan algoritma weight round robin.
3. Setelah menjalankan algoritma weight round robin, middleware akan mengirimkan tugas-tugas ke node sesuai dengan hasil algoritma weight round robin.
4. Pada node-node memproses tugas tersebut dan setelah selesai, mengirimkan kembali hasilnya ke middleware untuk diteruskan lagi ke klien.



**Gambar 7.1 Rancangan Sistem**

# DAFTAR PUSTAKA

1. Rumagit, Arthur Mourits. Hariadi, Moch. 2009. MANAJEMEN GRID UNTUK RENDER ANIMASI 3 DIMENSI. Jurusan Teknik Elektro Institut Teknologi Sepuluh Nopember Kampus Sukolilo. Surabaya.
2. M. Fikri Muhtadi H. 2010. RANCANG BANGUN PENJADWALAN TUGAS (TASK) PADA KOMPUTASI PARALEL DENGAN MENGGUNAKAN ALGORITMA COST-TIME OPTIMIZATION. Jurusan Teknik Informatika Fakultas Sains dan Teknologi UIN. Malang
3. Ahmad Makhsun, Idris Winarno .IMPLEMENTASI GRID COMPUTING DENGAN MENGGUNAKAN PENGALAMATAN IPv6. Jurusan Teknik Informatika Politeknik Elektronika Negeri Surabaya
4. Giovanni Sakti Nugraha, Odit Ekwardo, Fata Mukhlish.2006. Penggunaan Algoritma Divide and Conquer Dalam Parallel Computing Untuk Melakukan 3D Rendering. Departemen Teknik Informatika, Institut Teknologi Bandung
5. Udo H¨onig, Wolfram Schiffmann. A Parallel Branch–and–Bound Algorithm for Computing Optimal Task Graph Schedules. FernUniversit¨at Hagen, Lehrgebiet Technische Informatik I, 58084 Hagen, Germany.
6. Anonim. 2005. Wighted Round Robin Scheduling.http://kb.linuxvirtualserver.org/wiki/Weighted\_Round-Robin\_Scheduling