Available *online* at : http://bit.ly/InfoTekJar

InfoTekJar : Jurnal Nasional Informatika dan Teknologi Jaringan

ISSN (Print) 2540-7597 | ISSN (Online) 2540-7600

Implementasi Penjadwalan CPU Menggunakan Algoritma *First Come First Served* (FCFS)

Ahmad Rizal, Nurlayla Hasibuan

Universitas Sumatera Utara

KEYWORDS

Penjadwalan; Waktu tunggu; Algortima FCFS

CORRESPONDENCE

Phone:

E-mail: ahmadrizal@students.usu.ac.id

A B S T R A C T

Penerapan sistem operasi dengan *multitasking* dengan pemrosesan pekerjaan secara simultan memberi konsekuensi terhadap pemberian beban kepada prosesor semakin besar. Untuk melakukan pemrosesan CPU lebih optimal dilakukan penjadwalan terhadap job. Salah satu penjadwalan yang dipakai dengan algortima *First Come First Served* (FCFS)

PENDAHULUAN

Sistem operasi modern semakin berkembang dengan penerapan sistem *multitasking*, pada proses ini maka sistem operasi dapat melakukan banyak pekerjaan dalam waktu bersamaan. Penerapan multitasking ini membuat prosessor atau *CPU* melakukan proses penjadwalan untuk memaksimalkan kinerja *CPU*. Proses penjadwalan dibagi menjadi 2 hal, yaitu :

a. *I/O bound process*, proses yang lebih banyak menghabiskan waktu untuk mengerjakan proses *I/O* (*input/output*) daripada proses di *CPU* itu sendiri
b. *CPU bound process*, dimana proses lebih banyak menghabiskan waktu untuk penggunaan proses di *CPU* daripada permintaan I/O(*input/output*).

Sistem dengan kinerja yang terbaik akan memiliki kombinasi proses *CPU bound* dan *I/O bound* yang seimbang. Proses penjadwalan dalam melakukan proses mempertimbangkan sejumlah faktor :

a. Keadilan (*fairness*)
Proses penjadwalan harus memastikan bahwa setiap proses mendapatkan giliran yang sama, akan tetapi bukan berarti memperoleh jatah waktu yang sama. Perlu diperhatikan dan dipastikan tidak terjadi proses yang tidak dilayani dalam jangka waktu yang lama.
b. *Processor Utilization*
penggunaan waktu pemerosesan seoptimal mungkin, pada proses ini prosesor harus terus terpakai secara terus menerus selama masih ada proses antrian
c. Waktu tanggapan (*Response Time*)
waktu tanggapan adalah waktu antara CPU memberikan output atau umpan balik kepada pengguna
d. Waktu Tunggu (*Waiting Time*)
waktu tunggu merupakan lama waktu yang dihabiskan suatu proses dalam menyelesaikan antrian suatu proses.proses penjadwalan harus melakukan seminim mungkin waktu tunggu.
e. *Turn Around Time*
merupakan durasi waktu dari suatu proses aktif dalam sistem penjadwalan sampai dengan selesai proses.
f. Throughput
rata-rata proses yang dapat diselesaikan dalam satuan waktu. Nilai Thoughtput ini harus tinggi.

Rumusan Masalah

Rumusan masalah pada tulisan ini adalah analisa perancangan aplikasi perhitungan proses penjadwalan *CPU*, *waiting time*, *burst time*, *arrival time*, *average waiting time* dengan algoritma *First Come First Served* (FCFS).

Batasan masalah

Penulis memberikan batasan masalah sebagai berikut:

1. Proses penjadwalan *CPU* menggunakan metode *First Come First Served* (FCFS)

2. Analisa penjadwalan pada *burst time, arrival time, waiting time* dan *average waiting time*
3. Implementasi perhitungan penjadwalan FCFS menggunakan bahasa C++
4. Sistem Operasi Windows Xp

Tujuan Penelitian

1. Mengimplementasikan Analisa penjadwalan pada burst time, arrival time, waiting time dan average waiting time
2. Perancangan program aplikasi untuk menghitung proses penjadwalan dengan algortima FCFS menggunakan dengan bahasa pemograman C++.
3. Melakukan analisa penjadwalan dengan algortima FCFS

Manfaat Penelitian

Manfaat dari penelitian ini adalah:
Manfaat bagi penulis :

a. Menambah pengetahuan penulis tentang proses penjadwalan CPU menggunakan algoritma First Come First Served (FCFS)
b. Memahami proses penjadwalan CPU, waiting time, burst time, arrival time, average waiting time.

1. Manfaat bagi bidang ilmu :

a. Menambah pengetahuan tentang penjadwalan CPU menggunakan algoritma First Come First Served (FCFS)
b. Bahan referensi dan diskusi bagi kalangan penulis lain yang ingin mengembangkan dan merancang aplikasi perhitungan yang sama.

2. Manfaat bagi kalangan akademisi dan masyarakat adalah memberi bantuan kepada masyarakat untuk merancang dan perhitungan proses kerja penjadwalan CPU.

LANDASAN TEORI

Strategi proses penjadwalan

Ada beberapa strategi penjadwalan dasar antara lain :
1. Non Preemptive, dimana proses-proses yang sedang berjalan tidak bisa dihentikan sementara, dan harus berjalan terus sampai selesai. Strategi ini bisa membuat crash server.
2. Preemptive, dimana proses-proses yang sendang berjalan dapat dihentikan sementara. Strategi ini cocok diterapkan untuk proses yang dilakukan secara bersamaan dan *multi tasking* (2).

Kriteria Penjadwalan

Proses penjadwalan yang optimal harus memenuhi kriteria sebagai berikut :

- Memaksimumkan utilisasi CPU
- Memaksimumkan throughput
- Meminimumkan turnaroud time
- Meminimumkan waiting time
- Meminimumkan response time

Algoritma Penjadwalan First Come First Served

Algortima *First Come First Served* adalah suatu proses penjadwalan dimana permintaan yang paling awal akan dilayani lebih dahulu sedangkan permintaan yang datang belakangan dilayani belakangan, penjawalan ini bersifat sequensial (berurutan) (1). setiap proses diberi jadwal eksekusi berdasarkan urutan waktu kedatangan (*arrival time*).

HASIL DAN PEMBAHASAN

Analisa penjadwalan CPU dengan metode First Come First Served (FCFS)

Algortima *First Come First Served* adalah suatu proses penjadwalan dimana permintaan yang paling awal akan dilayani lebih dahulu sedangkan permintaan yang datang belakangan dilayani belakangan, penjawalan ini bersifat sequensial (berurutan). Setiap proses diberi jadwal eksekusi berdasarkan urutan waktu kedatangan (*arrival time*) (1). Proses penjadwalan ini mirip dengan proses algoritma FIFO pada stack. Pada skema ini proses permintaan ke *CPU* yang pertama kali akan mendapat layanan terlebih dahulu. Kita asumsikan terdapat empat proses permintaan yang dapat diurutkan P1,P2,P3,P4 dengan waktu sebagai berikut :
Berikut ini kita asumsikan 4 proses permintaan dengan urutan P1,P2,P3,P4 dengan waktu Burst Time dalam waktu mili detik sebagai berikut :

Tabel 1.Tabel Burst Time

Proses	Burst Time	Arrival Time
P1	2	0
P2	4	2
P3	7	6
P4	9	3

Berdasarkan Diagram Grant, arrival time yang paling kecil lah yang akan dikerjakan. Jadi urutan penjadwalan nya adalah sebagai berikut :

Tabel 2.Tabel Arrival Time

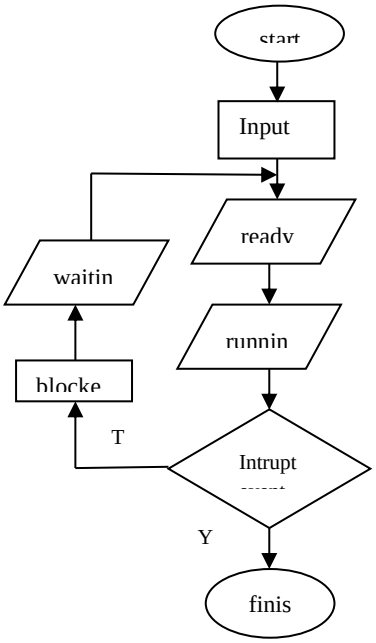
Proses	Burst Time	Arrival Time
P1	2	0
P2	2+4 = 6	2
P4	2+4+7=13	3
P3	2+4+7+9=22	6

Maka rata-rata waktu tunggu : 22/4= 5,5 milidetik

Perancangan aplikasi penghitungan waktu tunggu (waiting time) pada proses penjadwalan dengan Algortima First Come First Served (FCFS)

Perancangan aplikasi ini dimulai dengan analisa penjadwalan dengan algortima *First Come First Served* (FCFS) dapat disusun berdasarkan urutan sebagai berikut :

1. Merancang *flowchart*. Berikut ini gambar *flowchart* sistem aplikasi perhitungan



Gambar 1. *Flowchart* FCFS

2. Analisa *flowchart* Algoritma *First Come First Served* (FCFS)

Analisa *Flowchart* dimulai dengan dilakukan penentuan jumlah proses penjadwalan langkah berikutnya input proses dimana disimbolkan dengan p1,p2,p3,p4,...,pn. Penggunaan algortima first Come First Served (FCFS) maka proses akan memprioritaskan proses penjadwalan sesuai waktu kedatangan (*arrival time*). Dalam *flowchart* diatas proses p1 tiba dibandingkan dengan p2 ,p3, dan p4, disusul perbandingan p2 dengan p3 dan p4 begitu seterusnya sampai p4. Proses alur pemograman p1 berada pada memory kemudian p1 status diarahkan ke status ready berarti siap untuk dieksekusi. Proses tersebut akan berlangsung hal yang sama dengan p2,p3,p4 semua proses penjadwalan berada pada memory sebelum proses tersebut dieksekusi. Proses p1 di eksekusi berdasarkan burst time atau lamanya eksekusi proses, apabila proses tersebut selesai kemudian p1 berada distatus finis atau selesai, jika ada proses interupt bisa karena ada proses priority maka p1 dialihkan ke status blocked dan kembali menjadi status waiting, untuk eksekusi proses p2, p3dan p4 selanjutnya dan proses ini akan berlangsung sama.

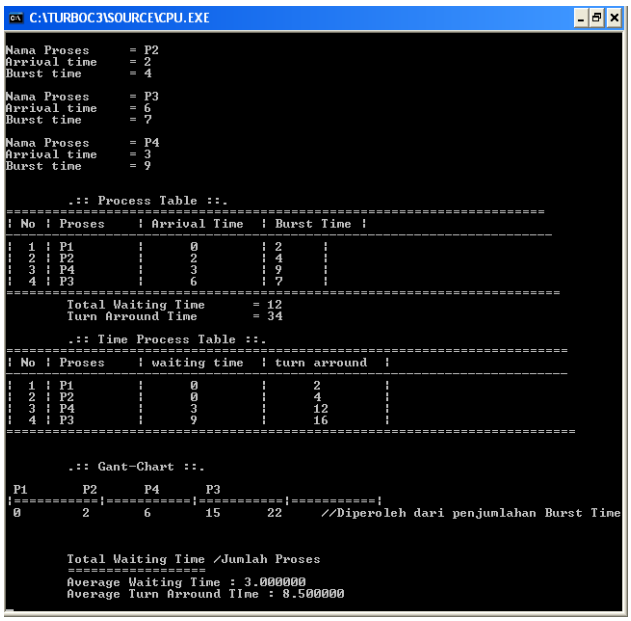
3. Perancangan kode Program

Perancangan kode proogram menggunakan bahasa pemograman C++, berikut analisa perancangan kode program

Pertama dilakukan input banyak proses yang akan dilakukan penjadwalan. Proses input banyak proses digunakan untuk melakukan perulangan inputan terhadap nama proses, arrival time dan burst time nilai –nilai tersebut disimpan dalam bentuk array. Untuk mengetahui nilai waiting time maka nilai arrival time dikurangi dengan nilai burst time. Menghitung nilai rata-rata waktu tunggu nilai waiting time dijumlahkan seluruh proses kemudian dibagi jumlah waiting time tersebut. Menghitung nilai rata-rata time arround nilai burst time dijumlahkan seluruhnya kemudian dibagi jumlah burst time tersebut.

Berikut kode program perhitungan waiting time, rata-rata waiting time dan rata-rata time arroud.

```
#include<stdio.h>
#include<string.h>
#include<conio.h>
main()
{
int n, ar[256], b[256], i, j, tmp, wt[256], ta[256], time[256];
int totWT=0, totTA=0;
float AvWT, AvTA;
char name[50][50], tmpName[50];
clrscr();
printf("Banyak Proses\t= "); scanf("%d", &n);
puts("");
// Masukkan data yang diproses
for(i=0; i<n; i++){
fflush(stdin);
printf("Nama Proses\t= "); gets(name[i]);
printf("Arrival time\t= "); scanf("%d", &ar[i]);
printf("Burst time\t= "); scanf("%d", &b[i]);
puts("");
}
// SORTING Data
for(i=0; i<n; i++){
for(j=i+1; j<n; j++)
if(ar[i]>ar[j]){
//tukar nama
strcpy(tmpName, name[i]);
strcpy(name[i], name[j]);
strcpy(name[j], tmpName);
//tukar arrival time
tmp=ar[i];
ar[i]=ar[j];
ar[j]=tmp;
//tukar burst time
tmp=b[i];
b[i]=b[j];
b[j]=tmp;
}
}
time[0]=ar[0];
puts("\n\t.:: Process Table ::.");
puts("=========================================
============================");
printf("| No | Proses\t | Arrival Time\t | Burst Time |\n");
puts("-----------------------------------------------------------------
-");
for (i=0; i<n; i++){
printf("| %2d | %s\t |  \t%d\t | %d\t |\n", i+1, name[i], ar[i], b[i]);
time[i+1]=time[i]+b[i]; //menghitung time pada ganchart
wt[i]=time[i]-ar[i];
ta[i]=time[i+1]-ar[i];
totWT+=wt[i];
totTA+=ta[i];
}
printf("\tTotal Waiting Time\t= %d \n", totWT);
printf("\tTurn Arround Time\t= %d \n", totTA);
puts("\n\t.:: Time Process Table ::.");
printf("| No | Proses\t | waiting time\t | turn arround\t |\n");
puts("-----------------------------------------------------------------
---");
for(i=0; i<n; i++){
printf("| %2d | %s\t |  \t%d\t | \t%d\t |\n", i+1, name[i], wt[i],
ta[i]);
}
AvWT=(float)totWT/n; //perhitungan rata-rata waktu tunggu
AvTA=(float)totTA/n; //average turn arround time
printf("\n");
printf("\tTotal Waiting Time /Jumlah Proses \n");
printf("\t==================\n");
printf("\tAverage Waiting Time : %f\n", AvWT);
printf("\tAverage Turn Arround TIme : %f\n", AvTA);
getch();
}
```



Gambar 2. Output program

KESIMPULAN

Adapun kesimpulan dari tulisan ini adalah sebagai berikut :

1. Proses penjadwalan dengan algoritma First Come First Served (FCFS) dilaksanakan berdasarkan urutan waktu kedatangan (arrival time). Proses ini merupakan proses berurutan atau sequensial.
2. Penjadwalan dengan algoritma FCFS terdapat kelemahan diantaranya :
   - Proses yang penting harus menunggu proses yang kurang penting karena harus menunggu berdasarkan waktu kedatangan.
   - Proses yang pendek dan cepat harus menunggu proses yang panjang dan lama
3. Sebagai bahan masukan dan telaah bagi penulis yang lain dalam pengembangan proses penjadwalan CPU dengan metode atau algoritma yang sama atau lainnya
4. Pengembangan masih perlu dilakukan untuk metode algoritma penjadwalan berbeda.

REFERENSI

[1] Abraham Silberschatz, Peter Baer Galvin and Greg Gagne, *operating system concept* , 7 edition, : Jhon-Willey & Son, 2005.

[2] *Haryanto Bambang, Ir Sistem Operasi* , edisi 2, Informatika Bandung, 2002

[3] Kadir Abdul, *Pemograman Lanjutan Turbo C untuk IBM-PC, jilid 1* , Andi Yogyakarta, 1993

[4] Osana Robert, 2007,*Rate-monotonic Analysis Keeps Real-time Systems on Schedule,* www.reed-electronics.com/ednmag/article/CA81193

[5] Chen, H., Chen, R., Zhang, F., Zang, B., dan Yew, P. *Live Updating Operating Systems Using Virtualization Prosiding 2nd international Conference on Virtual Execution Environment(VEE'06). Ottawa,2006*