Praktikum 10

Penjadwalan CPU 2

POKOK BAHASAN:

- ✓ Membuat program simuliasi Pendawalan CPU

TUJUAN BELAJAR:

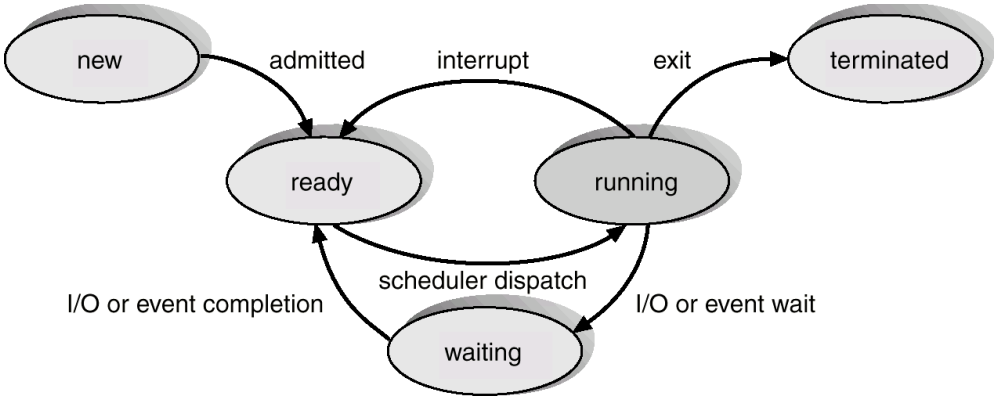
Setelah mempelajari materi dalam bab ini, mahasiswa diharapkan mampu:

- ✓ Memahami cara Penjadwalan CPU.
- ✓ Membuat Aplikasi simulasi Algoritma penjadwalan first come first serve
- ✓ Membuat Aplikasi simulasi Algoritma penjadwalan shortest job first
- ✓ Membuat Aplikasi simulasi Algoritma penjadwalan round robin

DASAR TEORI:

1 Penjadwalan CPU Preemptive

Penjadwalan CPU preemptive adalah penjadwalan CPU yang dapat disela walupun proses masih dalam proses pengerjaan. Perhatikan diagram proses pada gambar 10-1 :



Gambar 10 -1 . Diagram Proses.

Kapan pun CPU menjadi idle, sistem operasi harus memilih salah satu proses untuk masuk kedalam antrian ready (siap) untuk dieksekusi. Pemilihan tersebut dilakukan oleh penjadwalan short term. Penjadwalan memilih dari sekian proses yang ada di memori yang sudah siap dieksekusi, den mengalokasikan CPU untuk mengeksekusinya Penjadwalan CPU mungkin akan dijalankan ketika proses:

1. Berubah dari running ke waiting state.
2. Berubah dari running ke ready state.
3. Berubah dari waiting ke ready.
4. Terminates.

Penjadwalan dari no 1 sampai 4 ***non-preemptive sedangkan*** yang lain ***preemptive***. Dalam penjadwalan non-preemptive sekali CPU telah dialokasikan untuk sebuah proses, maka tidak bisa di ganggu, penjadwalan model seperti ini digunakan oleh Windows 3.x; Windows 95 telah menggunakan penjadwalan preemptive.

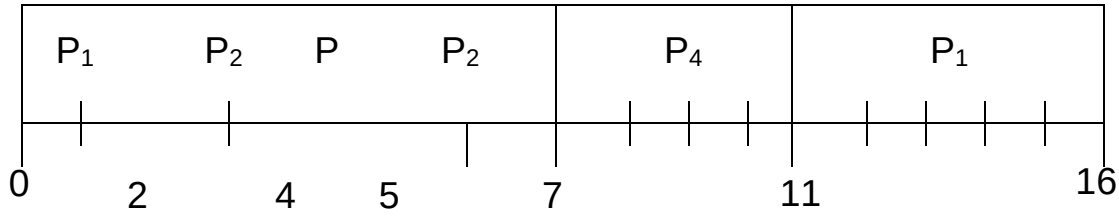
2 Penjadwalan Shortest Job First Premptive

SJF preventive bila sebuah proses datang dengan waktu prose lebih rendah dibandingkan dengan waktu proses yang sedang dieksekusi oleh CPU maka proses yang waktunya lebih rendah mendapatkan prioritas. Skema ini disebut juga Short - Remaining Time First (SRTF).

Contoh:

Arrival Time	Burst Time	proses
0.0	7	*P1*
2.0	4	*P2*
4.0	1	*P3*
5.0	4	*P4*

Gambar 10-4. Kedatangan Proses.



Rata-rata waktu tunggu = (9 + 1 + 0 +2)/4 = 3

3 Penjadwalan Prioritas

Penjadwalanan SJF (Shortest Job First) adalah kasus khusus untuk algoritma penjadwalan Prioritas. Prioritas dapat diasosiasikan masing-masing proses dan CPU dialokasikan untuk proses dengan prioritas tertinggi. Untuk proritas yang sama dilakukan dengan FCFS.

Ada pun algoritma penjadwalan prioritas adalah sebagai berikut:

- Setiap proses akan mempunyai prioritas (bilangan integer). Beberapa sistem menggunakan integer dengan urutan kecil untuk proses dengan prioritas rendah, dan sistem lain juga bisa menggunakan integer urutan kecil untuk proses dengan prioritas tinggi. Tetapi dalam teks ini diasumsikan bahwa integer kecil merupakan prioritas tertinggi.
- CPU diberikan ke proses dengan prioritas tertinggi (integer kecil adalah prioritas tertinggi).
- SJF adalah contoh penjadwalan prioritas dimana prioritas ditentukan oleh waktu pemakaian CPU

Beikutnya. Permasalahan yang muncul dalam penjadwalanan prioritas adalah indefinite blocking atau starvation.

- Kadang-kadang untuk kasus dengan prioritas rendah mungkin tidak pernah dieksekusi. Solusi untuk algoritma penjadwalan prioritas adalah aging
- Prioritas akan naik jika proses makin lama menunggu waktu jatah CPU.

4 Penjadwalan Round Robin

Algoritma Round Robin (RR) dirancang untuk sistem time sharing. Algoritma ini mirip denganpenjadwalan FCFS, namun preemption ditambahkan untuk switch antara proses. Antrian ready diperlakukan atau dianggap sebagai antrian sirkular. CPU menglilingi antrian ready dan mengalokasikan masing-masing proses untuk interval waktu tertentu sampai satu time slice/ quantum.

Berikut algritma untuk penjadwalan Round Robin:

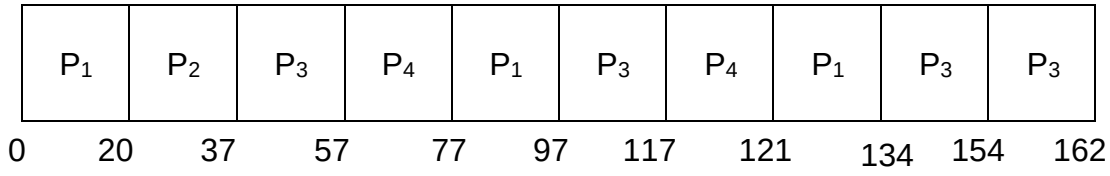
- Setiap proses mendapat jatah waktu CPU (time slice/ quantum) tertentu Time slice/quantum umumnya antara 10 - 100 milidetik.
    1. Setelah time slice / quantum maka proses akan di-preempt dan dipindahkan ke antrian ready.
    2. Proses ini adil dan sangat sederhana.

- Jika terdapat n proses di "antrian ready" dan waktu quantum q (milidetik), maka:
    1. Maka setiap proses akan mendapatkan 1/n dari waktu CPU.
    2. Proses tidak akan menunggu lebih lama dari: (n-1)q time units.

- Kinerja dari algoritma ini tergantung dari ukuran time quantum
    1. Time Quantum dengan ukuran yang besar maka akan sama dengan FCFS
    2. Time Quantum dengan ukuran yang kecil maka time quantum harusdiubah ukurannya lebih besar dengan respek pada alih konteks sebaliknya akan memerlukan ongkos yang besar.

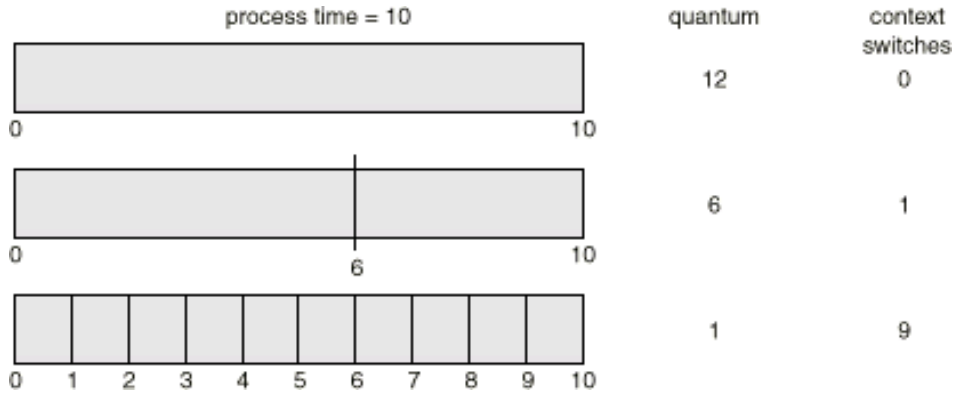
Tipikal: lebih lama waktu rata-rata turnaround dibandingkan SJF, tapi mempunyai response terhadap user lebih cepat.

proses	Burst Time
P1	53
P2	17
P3	68
P4	24



Gambar 2-38. Round Robin

Time Quantum Vs Alih Konteks



Gambar 2-39. Time Quantum dan Alih Konteks.

TUGAS PENDAHULUAN:

1. Lihat Tabel dibawah,,Gambarkan 4 diagram Chart yang mengilustrasikan eksekusi dari proses-proses tersebut menggunakan FCFS, SJF nonpreemptive, round robin.

Tabel Proses

Proses	Burst Time	Prioritas
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

Keterangan:
Misal diberikan beberapa proses dibawah ini dengan panjang CPU burst (dalam milidetik)
Semua proses diasumsikan datang pada saat t=0 dan quantum time= 3

PERCOBAAN

1. Login ke sistem GNU/Linux kemudian buka terminal.
2. Lakukan percobaan terhadap coding-coding Round Robin yang ada di bawah.

```
 1 #include<stdio.h>
 2 int main()
 3 {
 4   int i,j,n,time,remain,flag=0,tq;
 5   int TotWT=0,TotTA=0,AT[100],b[100],rt[100];
 6   printf("Masukkan jumlah proses : ");
 7   scanf("%d",&n);
 8   remain=n;
 9   for(i=0;i<n;i++)
10   {
11     printf("Masukkan arrival time untuk Proses P%d :",i+1);
12     scanf("%d",&AT[i]);
13     printf("Masukkan burst time untuk Proses P%d :",i+1);
14     scanf("%d",&b[i]);
15     rt[i]=b[i];
16   }
17   printf("Masukkan time quantum ");
18   scanf("%d",&tq);
19   printf("\n\nProcess\t|Turnaround time|waiting time\n\n");
20   for(time=0,i=0;remain!=0;)
21   {
22     if(rt[i]<=tq && rt[i]>0)
23     {
24       time+=rt[i];
25       rt[i]=0;
26       flag=1;
27     }
28     else if(rt[i]>0)
29     {
30       rt[i]-=tq;
31       time+=tq;
32     }
33     if(rt[i]==0 && flag==1)
34     {
35       remain--;
36       printf("P[%d]\t|\t%d\t|\t%d\n",i+1,time-AT[i],time-AT[i]-
   b[i]);
37       TotWT+=time-AT[i]-b[i];
38       TotTA+=time-AT[i];
39       flag=0;
40     }
41     if(i==n-1)
42       i=0;
43     else if(AT[i+1]<=time)
44       i++;
45     else
46       i=0;
47   }
48   printf("\nAverage Waiting Time = %f\n",TotWT*1.0/n);
49   printf("Average Turnaround Time = %f\n",TotTA*1.0/n);
50   return 0;
51 }
```

3. Compile menggunakan **`gcc –o rr rr.c`** lalu run menggunakan **`./rr`**
4. Masukkan variabel berikut dengan: jumlah proses sebanyak 3, time quantum 2, dan

Nama Proses	Arrival Time	Burst Time
p1	0	3
p2	1	2
p3	2	1

5. Amati hasil outputnya untuk buat kesimpulannya!

6. Apa yang menjadi perbedaan dengan algoritma Round Robin ini dengan percobaan sebelumnya yaitu FCFS? Jelaskan!

7. Jika variabel pada pada nomor 4 di atas diubah menjadi: jumlah proses sebanyak 3, time quantum 2, dan

Nama Proses	Arrival Time	Burst Time
p1	0	5
p2	20	4
p3	25	5

8. Amati hasil outputnya dan buat kesimpulannya pada nomor 7. Apa yang menjadi perbedaan dengan hasil pada nomor 5, jelaskan!

LAPORAN RESMI:

1. Analisa hasil percobaan yang Anda lakukan.
2. Buatlah Program di atas.
3. Berikan kesimpulan dari praktikum ini.