Penjadwalan Process

Penjadwalan Process

- Konsep Dasar Penjadwalan Proses.
  - ✓ Preemptive & Non-Preemtive Scheduling.
  - ✓ Dispatcher.
- Kriteria Penjadwalan.
- Algoritma Penjadwalan.
  - ✓ FCFS (First Come First Server) Scheduling.
  - ✓ SJF (Shortest Job First) Scheduling.
  - ✓ Priority Scheduling.
  - ✓ Round Robin Scheduling.

Overview Penjadwalan (1)

- Penjadwalan adalah fungsi dasar dari sistem operasi → semua resources komputer dijadwalkan sebelum digunakan.
- Penjadwalan CPU adalah pemilihan proses dari *Ready Queue* untuk dapat dieksekusi.
- Penjadwalan CPU didasarkan pada sistem operasi yang menggunakan prinsip **Multiprogramming**.
- Penjadwalan bertugas memutuskan :
  - Proses yang harus berjalan.
  - Kapan dan selama berapa lama proses itu berjalan.

Overview Penjadwalan (2)

- Pada saat CPU Idle → SO harus memilih proses yang ada dalam memori utama (Ready Queue) dan mengalokasikan CPU untuk mengeksekusinya.

Preemtive & Non-Preemtive Schedulling

- Terdapat 2 strategi penjadwalan :
  - Penjadwalan **Non Preemptive**
    - Jika proses sedang menggunakan CPU → proses tersebut akan membawa CPU sampai proses tersebut melepaskannya (berhenti dalam keadaan wait).
  - Penjadwalan **Preemptive**
    - Pada saat proses sedang menggunakan CPU → CPU dapat diambil alih oleh proses lain.
    - Dalam hal ini harus selalu dilakukan perbaikan data.

Dispatcher

- **Dispatcher** adalah suatu modul yang akan memberikan kontrol pada CPU terhadap penyeleksian proses.
- **Dispatch Latency** adalah waktu yang dibutuhkan untuk menghentikan suatu proses dan menjalankan proses yang lain.

Kriteria Scheduling (1)

- Adil, proses-proses diperlakukan sama, dalam artian adil. Adil disini tidak berarti terdapat perlakuan yang sama kepada setiap process, melainkan terdapat beberapa variabel seperti prioritas, dll yang akan dipelajari nanti.
- CPU Utilization, diharapkan agar CPU selalu dalam keadaan sibuk, sehingga penggunaan CPU lebih optimal.
- Throughput, adalah banyaknya proses yang selesai dikerjakan dalam satu satuan waktu. Sasaran penjadwalan adalah memaksimalkan jumlah job yang diproses dalam satu satuan waktu.
- Turn Around Time, adalah banyaknya waktu yang diperlukan untuk mengeksekusi proses, dari mulai menunggu untuk meminta tempat di memori utama, menunggu di Ready Queue, eksekusi oleh CPU dan mengerjakan I/O.
  - Turn Around Time = waktu eksekusi + waktu tunggu.
  - Sasaran Penjadwalan adalah meminimalkan waktu Turn Around Time.

Kriteria Scheduling (2)

- Waiting-Time, adalah waktu yang diperlukan oleh suatu proses untuk menunggu di *ready queue.* Sasaran Penjadwalan : meminimalkan waiting time. Untuk membandingkan algoritma penjadwalan CPU adalah rata-rata waktu tunggu (Average Waiting Time) = AWT

- Response-Time, adalah waktu yang diperlukan oleh suatu proses dari minta dilayani hingga ada respon pertama menanggapi permintaan tersebut . Sasaran penjadwalan : meminimalkan waktu tanggap.

Scheduling Algorithm

- Proses yang belum mendapatkan jatah alokasi dari CPU akan mengantri di *ready queue.*
- Algoritma Penjadwalan diperlukan untuk mengatur giliran proses-proses tersebut.
- Algoritma-algoritma penjadwalan :
  - FCFS (First Come First Serve).
  - SJF (Sortest Job First).
  - Priority.
  - Round Robin.

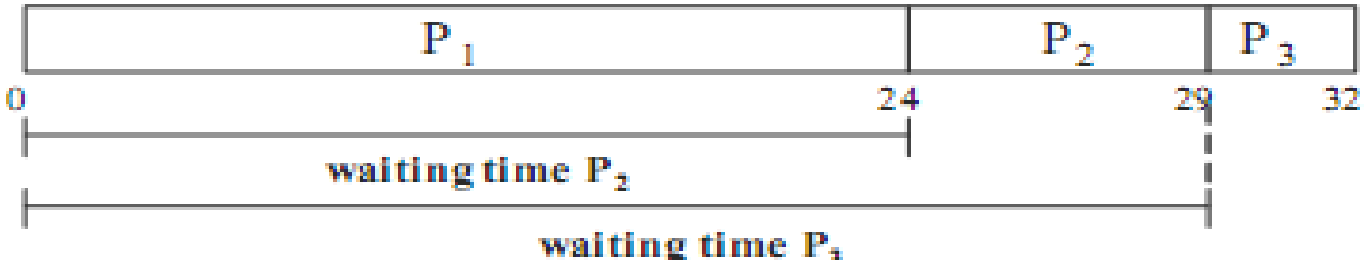
FCFS Algorithm (1)

- Penjadwalan ini merupakan penjadwalan *Non Preemptive.*
- Dalam penjadwalan FCFS *(First Come First Serve) :*
  - Proses yang pertama kali minta jatah waktu untuk menggunakan CPU akan dilayani terlebih dahulu.
  - Begitu proses mendapatkan jatah waktu CPU → proses dijalankan sampai selesai/ sampai proses tersebut melepaskannya, yaitu jika proses tersebut berhenti atau meminta I/O.

FCFS Algorithm (2)

- Contoh FCFS Scheduling :

Ada 3 buah proses yang datang secara berurutan yakni, P_1 selama 24 ms, P_2 selama 5 ms, P_3 selama 3 ms. Maka *Gantt Chart*-nya :



Average Waiting Time = (0+24+29)/3 = 17,6 ms

Kelemahan FCFS

Kelemahan dari algoritma ini:

1. *Waiting time* rata-ratanya cukup lama.
2. Terjadinya *convoy effect,* yaitu proses-proses menunggu lama untuk menunggu 1 proses besar yang sedang dieksekusi oleh CPU. Algoritma ini juga menerapkan konsep non-preemptive, yaitu setiap proses yang sedang dieksekusi oleh CPU tidak dapat di-interrupt oleh proses yang lain.

SJF (Shortest Job First) Algorithm (1)

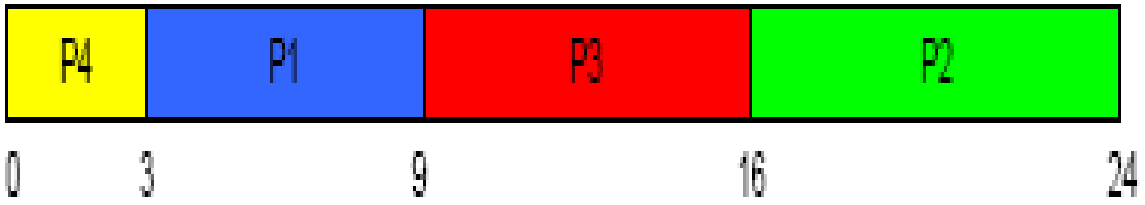
- Mendahulukan proses dengan Burst-Time terkecil.
  #Burst time = lama waktu kerja CPU
- Ada 2 Tipe :

  Jika ada proses P1 yang datang pada saat P0 sedang berjalan → akan dilihat CPU burst P1 →
  - **Preemptive**, Jika CPU burst P1 lebih kecil dari sisa waktu yang dibutuhkan oleh P0 → CPU ganti dialokasikan untuk P1. Biasa disebut dengan shortest Remaining Time First (SRTF)
  - **Non Preemptive**, Akan tetap menyelesaikan P0 sampai habis CPU burstnya.

SJF Algorithm (2)

- Contoh SJF Scheduling → **Non Preemptive**
  - Waktu kedatangan sama

Proses	Waktu
P1	6
P2	8
P3	7
P4	3

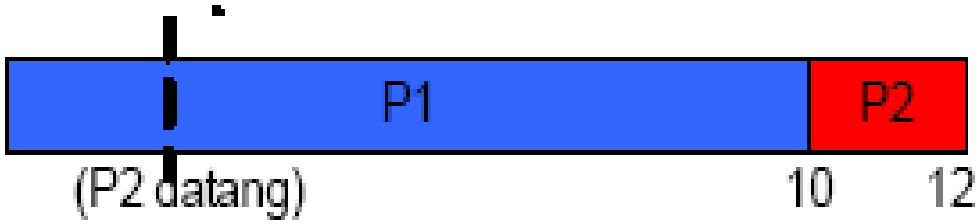


AWT = (o + 3 + 9 + 16)/4 = 7 ms

SJF Algorithm (3)

- Contoh SJF Scheduling → **Non Preemptive**
  - Waktu kedatangan tidak sama

Proses	Waktu	Urutan	Kedatangan
P1	10	1	0
P2	2	2	2



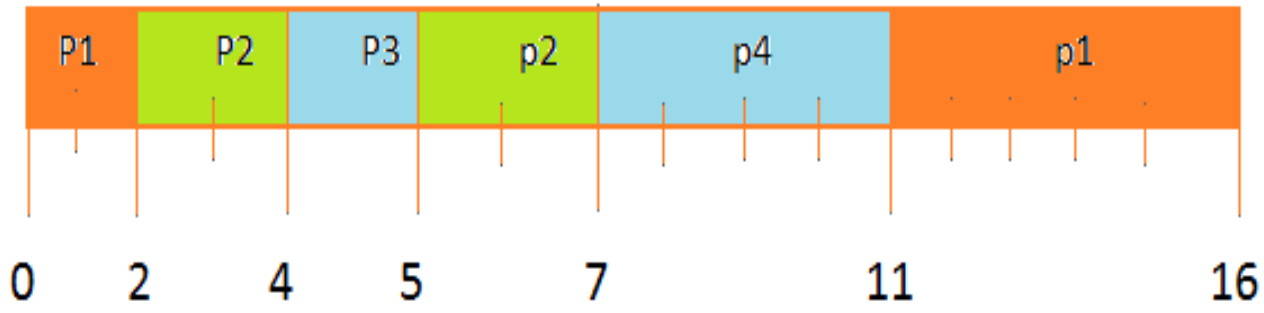
SJF Preemptive atau shortest Remaining Time First (SRTF)

- Contoh SJF Scheduling → **Preemptive**
  - Waktu kedatangan tidak sama

Proses	Waktu	Urutan	Kedatangan
P1	10	1	0
P2	2	2	2

Contoh SJF Scheduling → **Preemptive**

Proses	Arrival time	Burst Time
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4



- Average waiting time = (9 + 1 + 0 + 2)/4 =3

Round Robin Algorithm (1)

- Konsep dasar algoritma ini menggunakan time sharing
- Pada dasarnya, prinsip hampir sama dengan FCFS, tapi bersifat *preemptive*
- Tiap proses akan dibatasi waktu prosesnya, yang disebut *quantum time*
- Keuntungan algoritma round robin :
  - Adanya keseragaman waktu
- Kelemahannya :
  - Jika quantum time sangat besar → switching yang terjadi akan semakin sedikit (seperti FCFS)
  - Jika quantum time terlalu kecil → switching yang terjadi akan semakin banyak, sehingga banyak waktu yang terbuang

Round Robin Algorithm (2)

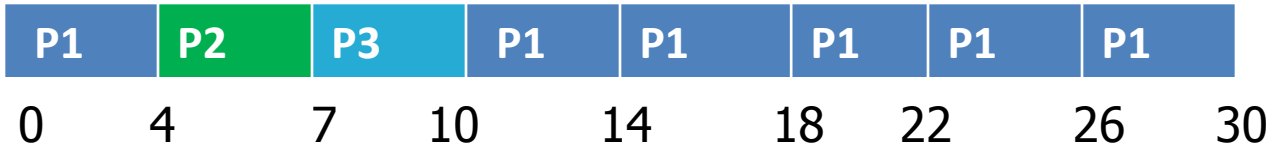
- Ketentuan Algoritma *Round Robin* adalah :
  - Jika proses memiliki *CPU Burst* < *Quantum Time*, maka proses akan melepaskan CPU, jika telah selesai digunakan → CPU dapat segera digunakan oleh proses selanjutnya
  - Jika proses memiliki *CPU Burst* > *Quantum Time*, maka proses tersebut akan dihentikan jika sudah mencapai *quantum time* dan selanjutnya mengantri kembali pada posisi *tail queue* (ekor dari *ready queue*), CPU kemudian menjalankan proses berikutnya
  - Jika *quantum time* belum habis dan proses menunggu suatu kejadian (selesainya operasi I/O), maka proses menjadi blocked dan CPU dialihkan ke proses lain

Round Robin Algorithm (3)

- Contoh Round Robin Scheduling :

Quantum time = 4 ms

Proses	Burst Time (ms)
P1	24
P2	3
P3	3



Proses	Waiting Time (ms)
P1	6
P2	4
P3	7

Average Waiting Time =
(6 + 4 + 7)/3 = 5.66

Priority Scheduling (1)

- Tiap proses diberi skala prioritas, proses yang mendapatkan prioritas tertinggi mendapat jatah waktu pemroses
- Jika beberapa proses memiliki prioritas yang sama akan digunakan algoritma FCFS
- Prioritas meliputi :
  - Waktu
  - Memori yang dibutuhkan
  - Banyaknya file yang dibuka
  - Perbandingan antara rata-rata I/O Burst dengan rata-rata CPU Burst

Priority Scheduling (2)

- Algoritma Priority Scheduling dapat bersifat *Preemptive* atau *Non Preemptive.*

  Jika ada proses P1 yang datang pada saat P0 sedang berjalan → akan dilihat prioritas P1, Jika prioritas P1 > P0, maka :

  - Pada *Non Preemptive,* Algoritma tetap akan menyelesaikan P0 sampai habis CPU burstnya dan meletakkan P1 pada posisi head queue.
  - Pada *Preemptive,* P0 akan dihentikan dulu dan CPU ganti dialokasikan untuk P1.

Priority Scheduling (3)

- Contoh Priority Scheduling

Proses	Waktu	Prioritas	Kedatangan
P1	6	4	0
P2	8	1	0
P3	7	3	0
P4	3	2	0

Waiting Time P2=0, P4=8, P3=11, P1=18
AWT = (0+8+11+18)/4 = 9.2 ms