PERTEMUAN 6:

PENJADWALAN CPU

A. TUJUAN PEMBELAJARAN

Pada bab ini akan dijelaskan mengenai penjadwalan CPU, Anda harus mampu:

1.1 Burst duration

1.2 Kriteria Penjadwalan

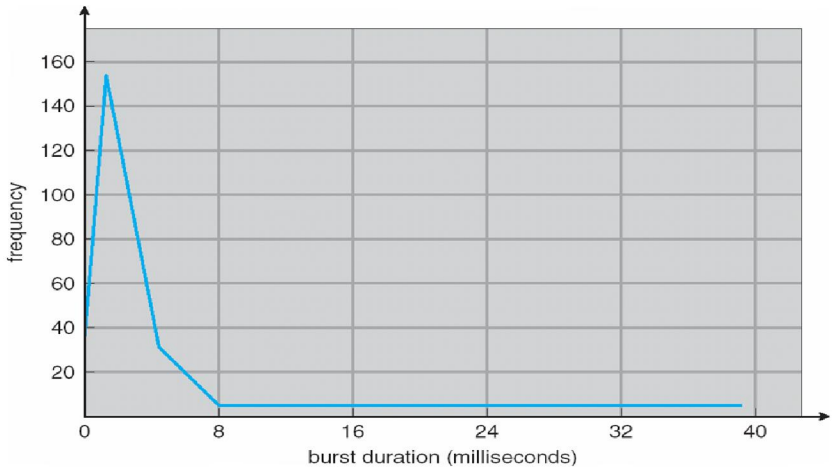
1.3 Algoritma Penjadwalan

B. URAIAN MATERI

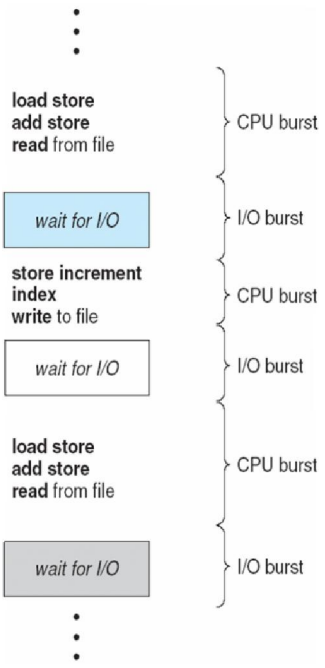
Tujuan Pembelajaran 1.1:
Penjadwalan CPU

- Maximum CPU utilization obtained with multiprogramming
- CPU–I/O Burst Cycle – Process execution consists of a cycle of CPU execution and I/O wait
- CPU burst distribution

Histogram of CPU-burst Times



Alternating Sequence of CPU And I/O Bursts



Selects from among the processes in memory that are ready to execute, and allocates the CPU to one of them

CPU scheduling decisions may take place when a process:

Switches from running to waiting state
Switches from running to ready state
Switches from waiting to ready
Terminates
Scheduling under 1 and 4 is nonpreemptive

All other scheduling is preemptive

- 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

- 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.
- 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 à CPU dapat diambil alih oleh proses lain.
- Dalam hal ini harus selalu dilakukan perbaikan data.

- 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 Penjadwalan

- 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.
- Waiting-Time, adalah waktu yang diperlukan oleh suatu proses untuk menunggu di ready queue. Sasaran Penjadwalan : meminimalkan waiting time.
- 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

ü Proses yang belum mendapatkan jatah alokasi dari CPU akan mengantri di ready queue.

ü Algoritma Penjadwalan diperlukan untuk mengatur giliran proses-proses tersebut.

ü Algoritma-algoritma penjadwalan :

- FCFS (First Come First Serve).
- SJF (Sortest Job First).
- Priority.
- Round Robin.

Algoritma FCFS

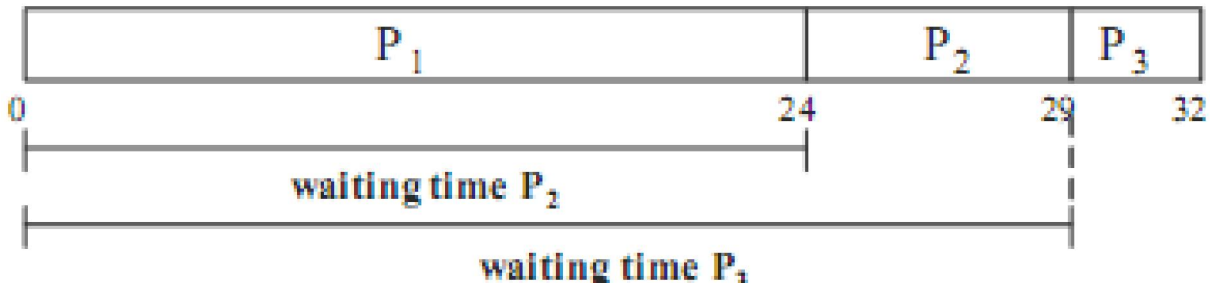
ü Penjadwalan ini merupakan penjadwalan Non Preemptive.

ü 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 à proses dijalankan sampai selesai/ sampai proses tersebut melepaskannya, yaitu jika proses tersebut berhenti atau meminta I/O.

ü 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 :



SJF (Shortest Job First) Algorithm

- Mendahulukan proses dengan Burst-Time terkecil.
- Ada 2 Tipe :

Jika ada proses P1 yang datang pada saat P0 sedang berjalan à akan dilihat CPU burst P1 à

- Preemptive, Jika CPU burst P1 lebih kecil dari sisa waktu yang dibutuhkan oleh P0 à CPU ganti dialokasikan untuk P1.
- Non Preemptive, Akan tetap menyelesaikan P0 sampai habis CPU burstnya.

- Contoh SJF Scheduling à Non Preemptive
  - Waktu kedatangan sama

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

- Contoh SJF Scheduling à Non Preemptive
  - Waktu kedatangan tidak sama

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

- Contoh SJF Scheduling à Preemptive

- Waktu kedatangan tidak sama

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

Priority Scheduling

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

  Jika ada proses P1 yang datang pada saat P0 sedang berjalan à 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.
- Contoh Priority Scheduling

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

Round Robin Algorithm

- 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 à switching yang terjadi akan semakin sedikit (seperti FCFS)
  - Jika quantum time terlalu kecil à switching yang terjadi akan semakin banyak, sehingga banyak waktu yang terbuang
- Ketentuan Algoritma Round Robin adalah :
  - Jika proses memiliki CPU Burst < Quantum Time, maka proses akan melepaskan CPU, jika telah selesai digunakan à 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

- Contoh Round Robin Scheduling :

Proses	Durasi	Urutan	Kedatangan
P1	3	1	0
P2	4	2	0
P3	3	3	0

Asumsi: kuantum waktu=1 unit; P1, P2, & P3 tidak pernah diblokir

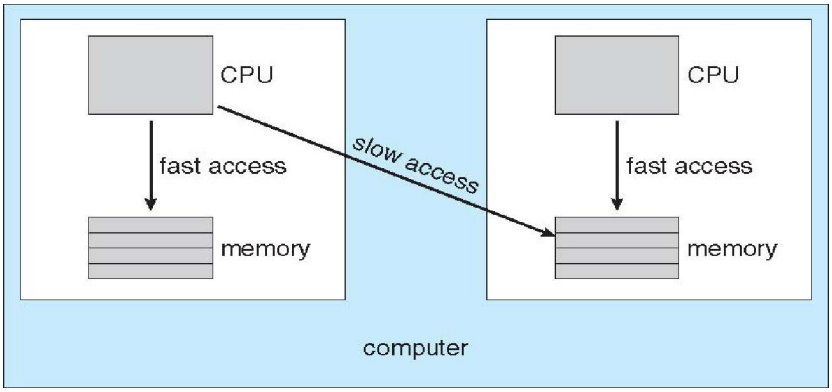
Thread Scheduling

- Distinction between user-level and kernel-level threads
- Many-to-one and many-to-many models, thread library schedules user-level threads to run on LWP

  Known as process-contention scope (PCS) since scheduling competition is within the process
- Kernel thread scheduled onto available CPU is system-contention scope (SCS) – competition among all threads in system

Multiple Processor Scheduling

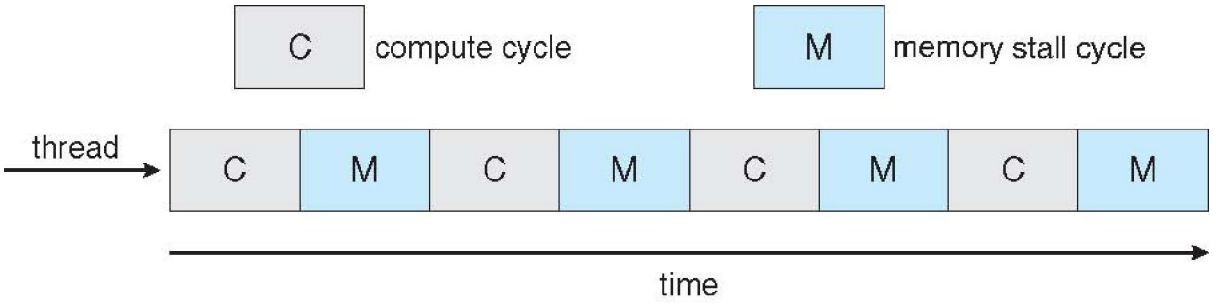
- CPU scheduling more complex when multiple CPUs are available
- Homogeneous processors within a multiprocessor
- Asymmetric multiprocessing – only one processor accesses the system data structures, alleviating the need for data sharing
- Symmetric multiprocessing (SMP) – each processor is self-scheduling, all processes in common ready queue, or each has its own private queue of ready processes
- Processor affinity – process has affinity for processor on which it is currently running
  - soft affinity
  - hard affinity



Multicore Processors

- Recent trend to place multiple processor cores on same physical chip
- Faster and consume less power
- Multiple threads per core also growing
    - § Takes advantage of memory stall to make progress on another thread while memory retrieve happens

Multithreaded Multicore System



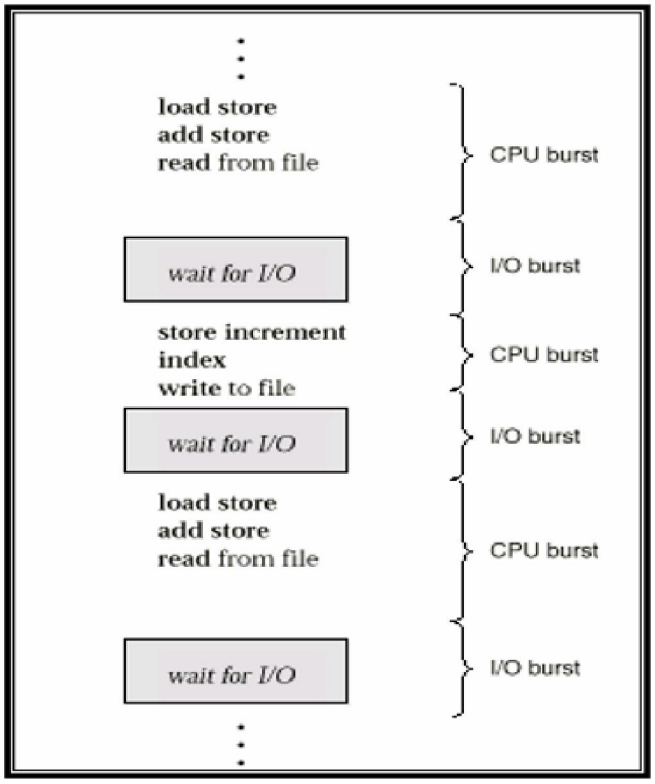
Penjadualan CPU

- Konsep Dasar
- Kriteria Penjadualan
- Algoritma Penjadualan
- Penjadualan Multiple-Processor
- Penjadualan Real-Time
- Evaluasi Algorithm

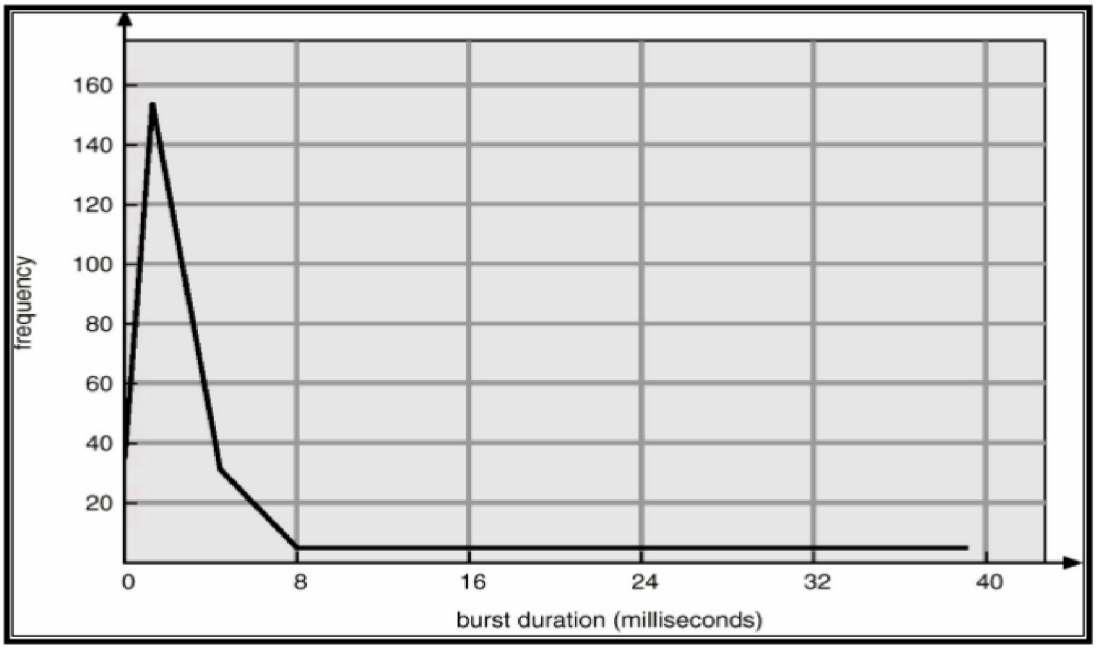
Konsep Dasar

- Memaksimalkan kinerja CPU melalui multiprogramming
- CPU–I/O Burst Cycle – Eksekusi proses terdiri dari siklus eksekusi CPU dan I/O wait.
- Pendistribusian CPU burst

Penggantian Rangkaian Urutan CPU dan I/O Burst



Histogram CPU-burst Times



Penjadual CPU

- Algoritma scheduling:
  - Memilih dari proses-proses yang berada di memori (ready to execute) dan memberikan jatah CPU ke salah satu proses tersebut.
- Kapan keputusan untuk algoritma dilakukan:
  - Saat suatu proses:
    1. Switch dari status running ke waiting.
    2. Switch dari status running ke ready.
    3. Switch dari status waiting ke ready.
    4. Terminates.
  - Penjadualan 1 dan 4 termasuk nonpreemptive
  - Penjaudualan lainnya termasuk preemptive

Jenis Penjadualan

- Preemptive: OS dapat mengambil (secara interrupt, preempt) CPU dari satu proses setiap saat.
- Non-preemptive: setiap proses secara sukarela

- (berkala) memberikan CPU ke OS.
- Contoh:
    - Penjadualan untuk switch dari running ke wait atau terminate: non-preemptive.
        - Penjadualan proses dari running ke ready: pre-emptive. Prasyarat untuk OS real-time system.

Dispatcher

- Modul Dispatcher: mengatur dan memberikan control CPU kepada proses yang dipilih oleh “short-term scheduler”:
    - switching context
    - switching ke user mode
    - Melompat ke lokasi yang lebih tepat dari user program untuk memulai kembali program
- Dispatch latency – terdapat waktu yang terbuang (CPU idle) dimana dispatcher menghentikan satu proses dan menjalankan proses lain.
    - Save (proses lama) dan restrore (proses baru).

Kriteria Penjadualan

- Utilisasi CPU: menjadikan CPU terus menerus sibuk (menggunakan CPU semaksimal mungkin).
- Throughput: maksimalkan jumlah proses yang selesai dijalankan (per satuan waktu).
- Turn around time: minimalkan waktu selesai eksekusi suatu proses (sejak di submit sampai selesai).
- Waiting time: minimalkan waktu tunggu proses (jumlah waktu yang dihabiskan menunggu di ready queue).
- Response time: minimalkan waktu response dari sistim terhadap user (interaktif, time-sharing system), sehingga interaksi dapat berlangsung dengan cepat.

Kriteria Penjadualan yang Optimal

- Memaksimumkan utilisasi CPU
- Memaksimumkan throughput
- Meminimukan turnaround time
- Meminimumkan waiting time
- Meminimumkan response time

Algoritma Penjadualan

- First-come, first-served (FCFS)
- Shortest-Job-First (SJF)
- Priority
- Round-Robin (RR)
- Multilevel Queue
- Multilevel Feedback Queue

First-Come, First-Served (FCFS)

- Algoritma:
    - Proses yang request CPU pertama kali akan mendapatkan jatah CPU.
    - Sederhana – algoritma maupun struktur data: menggunakan FIFO queue (ready queue).
- FIFO: Non preemptive
    - Timbul masalah “waiting time” terlalu lama jikadidahului oleh proses yang waktu selesainya lama.
    - Tidak cocok untuk time-sharing systems.
    - Digunakan pada OS dengan orientasi batch job.

FCFS (Cont.)

- Example: Process Burst Time

P_1	24
P_2	3
P_3	3

- Diketahui proses yang tiba adalah P_1, P_2, P_3. Gant chart-nya adalah :

- Waiting
- Average waiting time: (0 + 24 + 27)/3 = 17

FCFS (Cont.)

Diketahui proses yang tiba adalah P 2 , P 3 , P 1 .Gant chart-nya adalah :

- Waiting time untuk P1 = 6; P2 = 0; P3 = 3
- Average waiting time: (6 + 0 + 3)/3 = 3
- Lebih baik dari kasus sebelumnya
- Convoy effect proses yang pendek diikuti proses yang panjang

Shortest-Job-First (SJR)

- Penggabungan setiap proses merupakan panjang dari burst CPU berikutnya. Panjang tersebut digunakan untuk penjadualan proses pada waktu terpendek
- Terdapat 2 skema :
    - nonpreemptive – CPU hanya satu kali diberikan pada suatu proses, maka proses tersebut tetap akan memakai CPU hingga proses tersebut melepaskannya
    - preemptive –jika suatu proses tiba dengan panjang CPU burst lebih kecil dari waktu yang tersisa pada ekseksusi proses yang sedang berlangsung, maka dijalankan preemtive. Skema ini dikenal dengan Shortest-Remaining-Time-

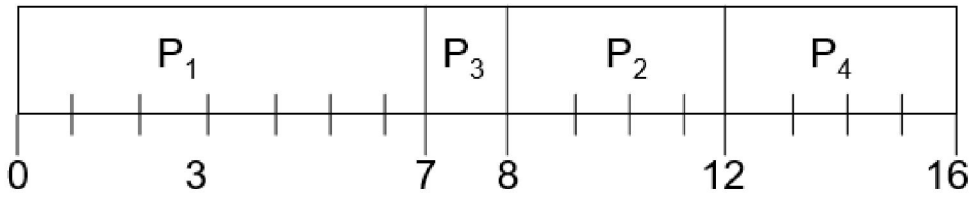
First (SRTF).

- SJF akan optimal, keteika rata-rata waktu tunggu minimum untuk set proses yang diberikan

Contoh Non-Preemptive SJF

Process	Arrival Time	Burst Time
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (non-preemptive)

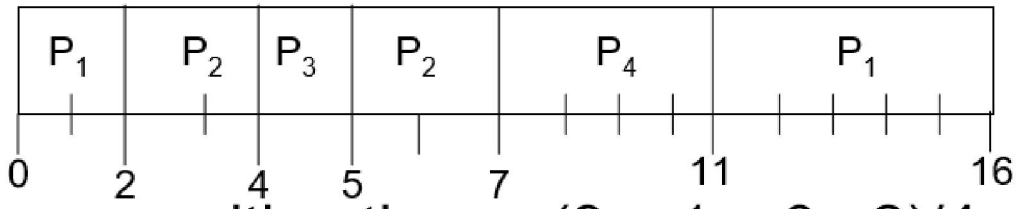


- Average waiting time = (0 + 6 + 3 + 7)/4 - 4

SJF (non-preemptive)

Process	Arrival Time	Burst Time
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (preemptive)



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

Penjadualan Prioritas

- Algoritma
  - Setiap proses akan mempunyai prioritas (bilangan integer).
  - CPU diberikan ke proses dengan prioritas tertinggi (smallest integer o highest priority).
  - Preemptive: proses dapat di interupsi jika terdapat prioritas lebih tinggi yang memerlukan CPU.
  - Nonpreemptive: proses dengan prioritas tinggi akan mengganti pada saat pemakain time-slice habis.
  - SJF adalah contoh priority scheduling dimana prioritas ditentukan oleh waktu pemakaian CPU berikutnya.
- Problem = Starvation
  - Proses dengan prioritas terendah mungkin tidak akan pernah dieksekusi
  - Solution = Aging
    - Prioritas akan naik jika proses makin lama menunggu waktu jatah CPU.

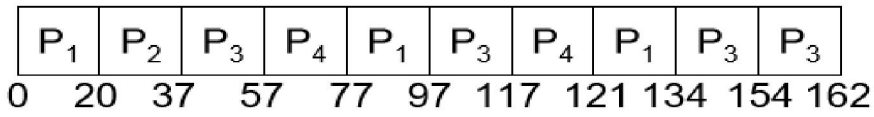
Round Robin (RR)

- Setiap proses mendapat jatah waktu CPU (time slice/quantum) tertentu misalkan 10 atau 100 milidetik.
    - Setelah waktu tersebut maka proses akan di-preempt dan dipindahkan ke ready queue.
    - Adil dan sederhana.
- Jika terdapat n proses di "ready queue" dan waktu quantum q (milidetik), maka:
    - Maka setiap proses akan mendapatkan 1/n dari waktu CPU.
    - Proses tidak akan menunggu lebih lama dari: (n-1) q time units.
- Performance
    - q besar ⇒FIFO
    - q kecil ⇒q harus lebih besar dengan mengacu pada context switch, jika tidak overhead akan terlalu besar

Contoh RR (Q= 20)

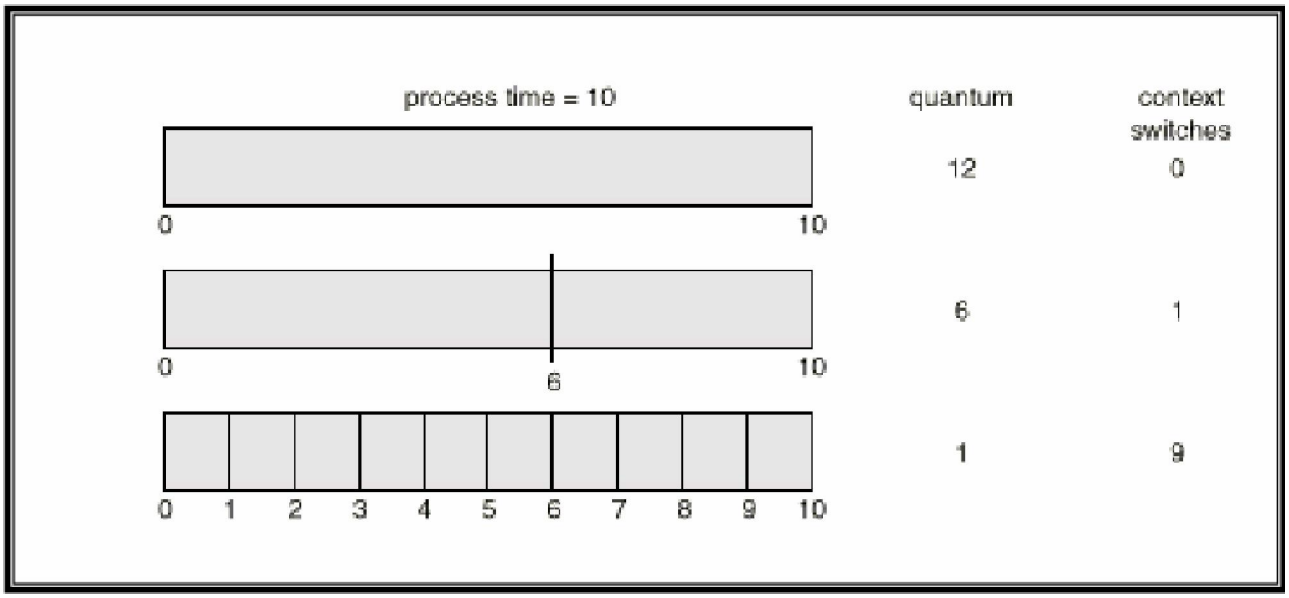
Process	Burst Time
P_1	53
P_2	17
P_3	68
P_4	24

- Gantt Chart



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

Waktu Kuantum dan Waktu Context Switch

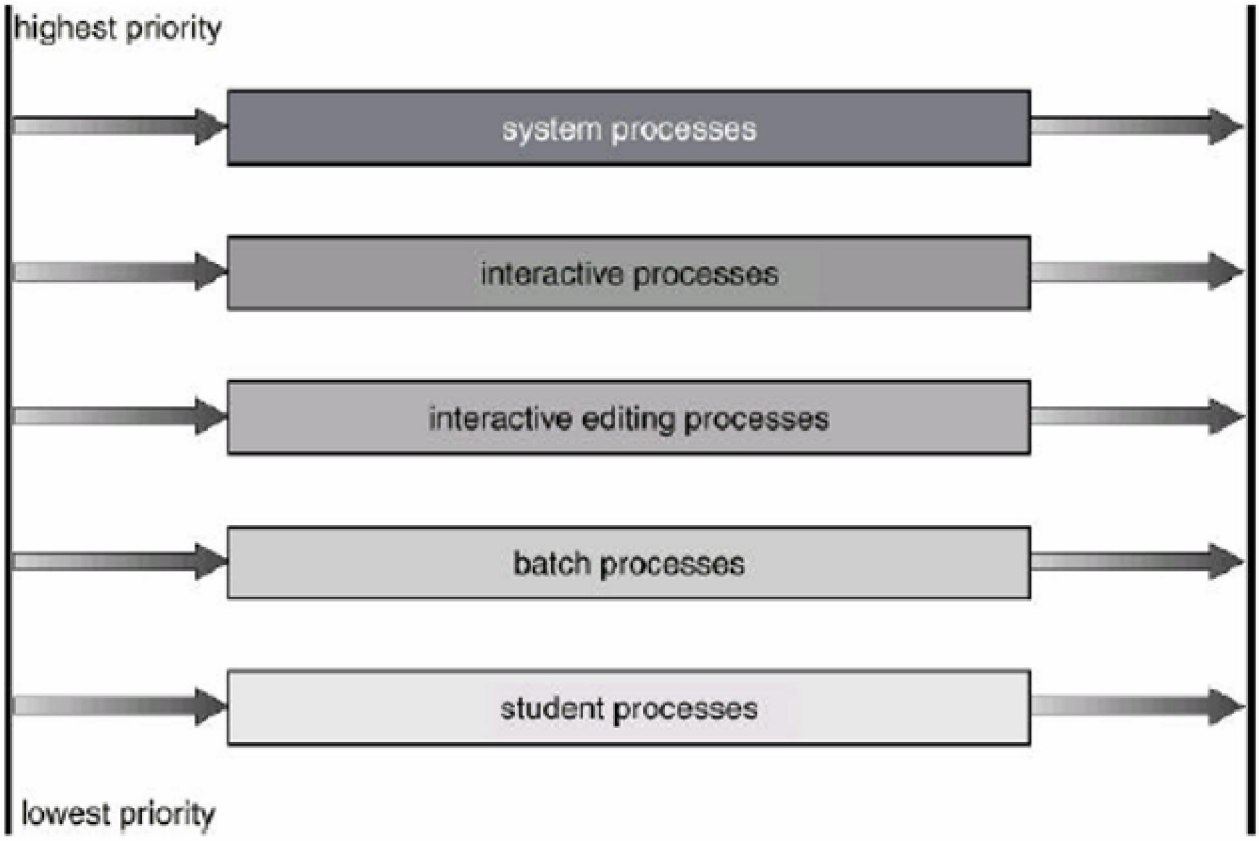


Penjadualan Antrian Multitingkat

- Kategori proses sesuai dengan sifat proses:
  - Interaktif (response cepat)
  - Batch dll

- Partisi “ready queue” dalam beberapa tingkat (multilevel) sesuai dengan proses:
  - Setiap queue menggunakan algoritma schedule sendiri
  - Foreground proses (interaktif, high prioritiy): R
  - Background proses (batch, low priority): FCFS
  - Setiap queue mempunyai prioritas yang fixed.

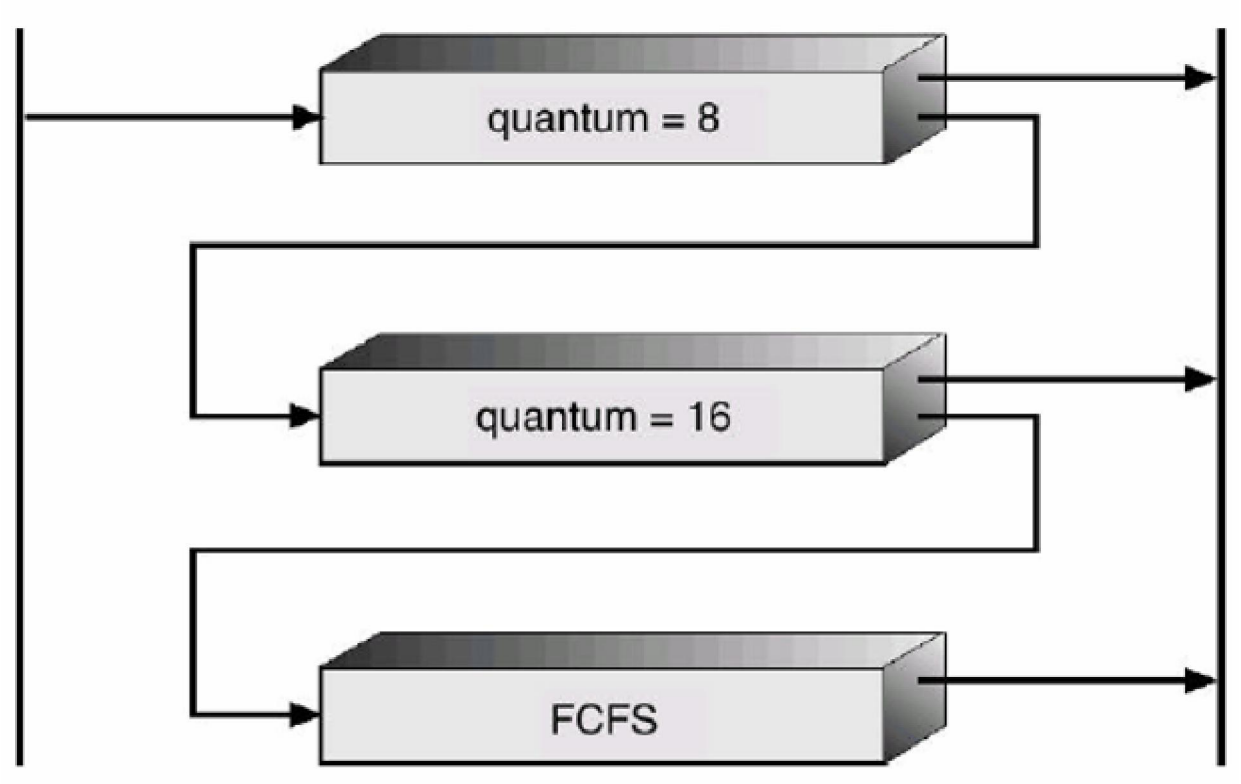
Penjadualan Antrian Multitingkat



Antrian Multitingkat Berbalikan

- Suatu proses dapat berpindah diantara beragam antrian;
- Perlu feedback untuk penentuan proses naik/turun prioritasnya (dinamis):
    - Aging dapat diimplementasikan sesuai dengan lama proses pada satu queue.
    - Suatu proses yang menggunakan CPU sampai habis (tanpa I/O wait) => CPU-bound (bukan proses interaktif) dapat dipindahk ke queue dengan prioritas lebih rendah

Antrian Multitingkat Berbalikan



Penjadualan Multiple-Processor

- Penjadualan CPU lehih kompleks ketika terdapat multiple Processor
- Processor yang homogen termasuk ke dalam multiprocessor
- Homogeneous processors within a multiprocessor.
- Load sharing
- Asymmetric multiprocessing – hanya ada satu processor yang dapat mengakses struktur system data,only one processor accesses the system data structures,sehingga meringankan kebutuhan
- sharing data

Penjadualan Real-Time

- Hard real-time systems
    - Task kritis harus selesai dengan garansi waktu tertentu

- OS akan melacak lamanya task tersebut dieksekusi (real time):
  - Mengetahui lama waktu system call, fungsi dan response dari hardware
  - Melakukan prediksi apakah task tersebut dapat dijalankan.
  - Mudah dilakukan untuk OS khusus pada peralatan/ pemakaian khusus (single task: control system)
- Sulit untuk time-sharing sistim, virtual memory (factor latency sebagian program aktif ada di disk).

Penjadualan Real-Time

- Soft real-time systems
  - Membutuhkan penggunaan skema prioritas
  - Multimedia, highly interactive graphics
  - Prioritas tidak menurunkan over time
  - Dispancy latency yang rendah :
    - Penyisipan point preemsi sepanjang waktu system calls

Membuat keseluruhan kernel preemptable

C. SOAL LATIHAN/TUGAS

1. Jelaskan algoritma SJF?
2. Jelaskan algoritma FCFS?
3. Jelaskan algoritma Roundrobin?

D. DAFTAR PUSTAKA

Buku

Bambang Hariyanto. 1997. Sistem Operasi, Bandung:Informatika Bandung.

Dali S. Naga. 1992. Teori dan Soal Sistem Operasi Komputer,Jakarta: Gunadarma.

Operating System Concepts (6th or 7th Edition). Silberschatz, Galvin, Gagne, ISBN: 0-471-25060-0. Wiley

Silberschatz Galvin. 1995. 4 Edition Operating System Concepts: Addison Wesley.

Sri Kusumadewi. 2000. Sistem Operasi. Yogyakarta: J&J Learning.

Tanenbaum, A.1992. Modern Operating Systems.New York: Prentice Hall

Link and Sites:

http://www.ilmukomputer.com
http://vlsm.bebas.org
http://www.wikipedia.com