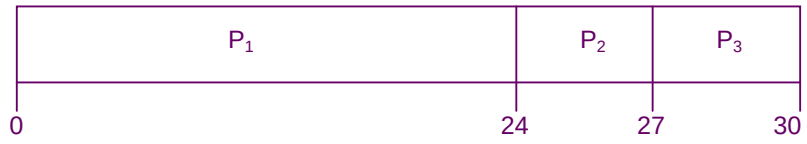
4.3.1 First-Come First-Served Scheduling (FCFS)

Proses yang pertama kali meminta jatah waktu untuk menggunakan CPU akan dilayani terlebih dahulu. Pada skema ini, proses yang meminta CPU pertama kali akan dialokasikan ke CPU pertama kali.

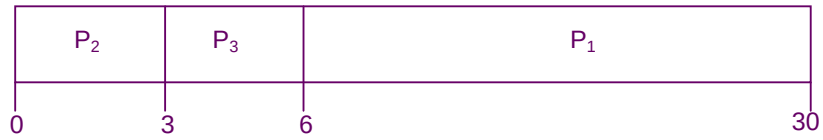
Misalnya terdapat tiga proses yang dapat dengan urutan P_1, P_2, dan P_3 dengan waktu CPU-burst dalam milidetik yang diberikan sebagai berikut :

Process	Burst Time
P_1	24
P_2	3
P_3	3

Gant Chart dengan penjadwalan FCFS adalah sebagai berikut :



Waktu tunggu untuk P_1 adalah 0, P_2 adalah 24 dan P_3 adalah 27 sehingga rata-rata waktu tunggu adalah (0 + 24 + 27)/3 = 17 milidetik. Sedangkan apabila proses datang dengan urutan P_2, P_3, dan P_1, hasil penjadwalan CPU dapat dilihat pada gant chart berikut :



Waktu tunggu sekarang untuk P_1 adalah 6, P_2 adalah 0 dan P_3 adalah 3 sehingga rata-rata waktu tunggu adalah (6 + 0 + 3)/3 = 3 milidetik. Rata-rata waktu tunggu kasus ini jauh lebih baik dibandingkan dengan kasus sebelumnya. Pada penjadwalan CPU dimungkinkan terjadi *Convoy effect* apabila proses yang pendek berada pada proses yang panjang.

Algoritma FCFS termasuk *non-preemptive*. karena, sekali CPU dialokasikan pada suatu proses, maka proses tersebut tetap akan memakai CPU sampai proses tersebut melepaskannya, yaitu jika proses tersebut berhenti atau meminta I/O.

4.3.2 Shortest Job First Scheduler (SJF)

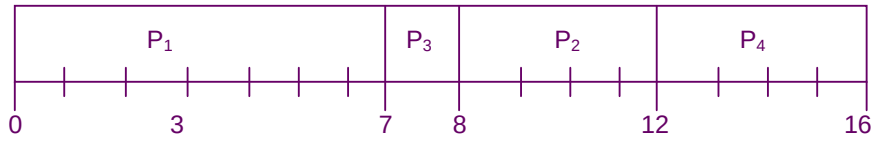
Pada penjadwalan SJF, proses yang memiliki CPU burst paling kecil dilayani terlebih dahulu. Terdapat dua skema :

1. Non preemptive, bila CPU diberikan pada proses, maka tidak bisa ditunda sampai CPU burst selesai.
2. Preemptive, jika proses baru datang dengan panjang CPU burst lebih pendek dari sisa waktu proses yang saat itu sedang dieksekusi, proses ini ditunda dan diganti dengan proses baru. Skema ini disebut dengan Shortest-Remaining-Time-First (SRTF).

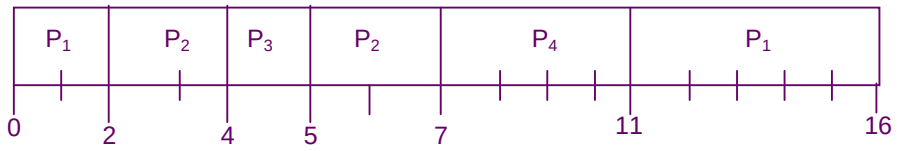
SJF adalah algoritma penjadwalan yang optimal dengan rata-rata waktu tunggu yang minimal. Misalnya terdapat empat proses dengan panjang CPU burst dalam milidetik.

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

Penjadwalan proses dengan algoritma SJF (non-preemptive) dapat dilihat pada gant chart berikut :



Waktu tunggu untuk P_1 adalah 0, P_2 adalah 26, P_3 adalah 3 dan P_4 adalah 7 sehingga rata-rata waktu tunggu adalah (0 + 6 + 3 + 7)/4 = 4 milidetik. Sedangkan Penjadwalan proses dengan algoritma SRTF (preemptive) dapat dilihat pada gant chart berikut :



Waktu tunggu untuk P_1 adalah 9, P_2 adalah 1, P_3 adalah 0 dan P_4 adalah 4 sehingga rata-rata waktu tunggu adalah (9 + 1 + 0 + 4)/4 = 3 milidetik.

Meskipun algoritma ini optimal, namun pada kenyataannya sulit untuk diimplementasikan karena sulit untuk mengetahui panjang *CPU burst* berikutnya. Namun nilai ini dapat diprediksi. *CPU burst* berikutnya biasanya diprediksi sebagai suatu rata-rata eksponensial yang ditentukan dari *CPU burst* sebelumnya atau "*Exponential Average*".

$$\tau_{n+1} = \alpha t_0 + (1-\alpha)\tau_n \qquad (4.1)$$

dengan:

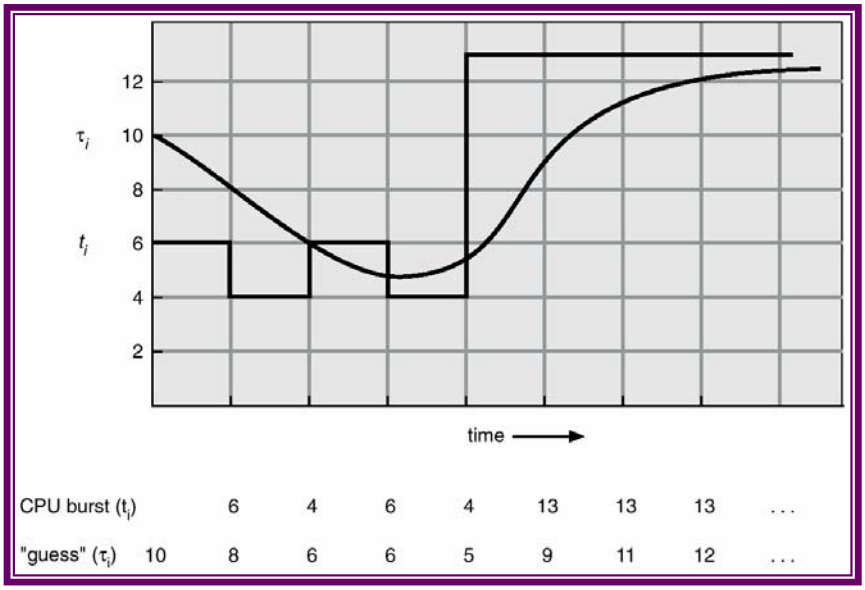
τ_{n+1} = panjang CPU burst yang diperkirakan

τ_0 = panjang CPU burst sebelumnya

τ_n = panjang CPU burst yang ke-*n* (yang sedang berlangsung)

α = ukuran pembanding antara τ_{n+1} dengan τ_n (0 sampai 1)

Grafik hasil prediksi CPU burst dapat dilihat pada Gambar 4-3.



Gambar 4-3 : Prediksi panjang CPU burst berikutnya

Sebagai contoh, jika $\alpha = 0{,}5$, dan:

CPU burst (τ_n) = 6 4 6 4 13 13 13 . . .

τ_n = 10 8 6 6 5 9 11 12 . . .

Pada awalnya τ_0 = 6 dan τ_n = 10, sehingga :

τ_2 = 0,5 * 6 + (1 - 0,5) * 10 = 8

Nilai yang dapat digunakan untuk mencari τ_3

τ_3 = 0,5 * 4 + (1 - 0,5) * 8 = 6

4.3.3 Priority Scheduling

Algoritma SJF adalah suatu kasus khusus dari penjadwalan berprioritas. Tiap-tiap proses dilengkapi dengan nomor prioritas (integer). CPU dialokasikan untuk proses yang memiliki prioritas paling tinggi (nilai integer terkecil biasanya merupakan prioritas terbesar). Jika beberapa proses memiliki prioritas yang sama, maka akan digunakan algoritma FCFS. Penjadwalan berprioritas terdiri dari dua skema yaitu non preemptive dan preemptive. Jika ada proses P_1 yang datang pada saat P_0 sedang berjalan, maka akan dilihat prioritas P_1. Seandainya prioritas P_1 lebih besar dibanding dengan prioritas P_0, maka pada *non-preemptive*, algoritma tetap akan menyelesaikan P_0 sampai habis *CPU burst*-nya, dan meletakkan P_1 pada posisi *head queue*. Sedangkan pada *preemptive*, P_0 akan dihentikan dulu, dan CPU ganti dialokasikan untuk P_1.

Misalnya terdapat lima proses P_1, P_2, P_3, P_4 dan P_5 yang datang secara berurutan dengan *CPU burst* dalam milidetik.

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2

Penjadwalan proses dengan algoritma priority dapat dilihat pada gant chart berikut :

Waktu tunggu untuk P_1 adalah 6, P_2 adalah 0, P_3 adalah 16, P_4 adalah 18 dan P_5 adalah 1 sehingga rata-rata waktu tunggu adalah (6 + 0 +16 + 18 + 1)/5 = 8.2 milidetik.

4.3.4 Round-Robin Scheduling

Konsep dasar dari algoritma ini adalah dengan menggunakan time-sharing. Pada dasarnya algoritma ini sama dengan FCFS, hanya saja bersifat preemptive. Setiap proses mendapatkan waktu CPU yang disebut dengan waktu quantum (*quantum time)* untuk membatasi waktu proses, biasanya 1-100 milidetik. Setelah waktu habis, proses ditunda dan ditambahkan pada ready queue.

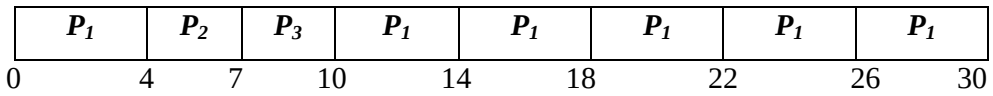
Jika suatu proses memiliki CPU burst lebih kecil dibandingkan dengan waktu quantum, maka proses tersebut akan melepaskan CPU jika telah selesai bekerja, sehingga CPU dapat segera digunakan oleh proses selanjutnya. Sebaliknya, jika suatu proses memiliki CPU burst yang lebih besar dibandingkan dengan waktu quantum, maka proses tersebut akan dihentikan sementara jika sudah mencapai waktu quantum, dan selanjutnya mengantri kembali pada posisi ekor dari ready queue, CPU kemudian menjalankan proses berikutnya.

Jika terdapat *n* proses pada ready queue dan waktu quantum *q*, maka setiap proses mendapatkan 1/*n* dari waktu CPU paling banyak *q* unit waktu pada sekali penjadwalan CPU. Tidak ada proses yang menunggu lebih dari (*n*-1)*q* unit waktu. Performansi algoritma round robin dapat dijelaskan sebagai berikut, jika q besar, maka yang digunakan adalah algoritma FIFO, tetapi jika q kecil maka sering terjadi context switch.

Misalkan ada 3 proses: P_1, P_2, dan P_3 yang meminta pelayanan CPU dengan *quantum-time* sebesar 4 milidetik.

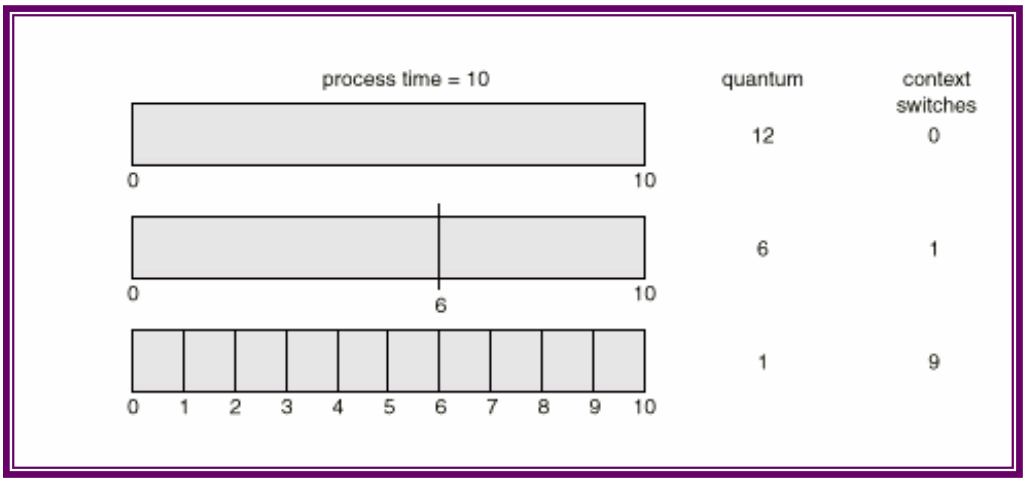
Process	Burst Time
P_1	24
P_2	3
P_3	3

Penjadwalan proses dengan algoritma round robin dapat dilihat pada gant chart berikut :



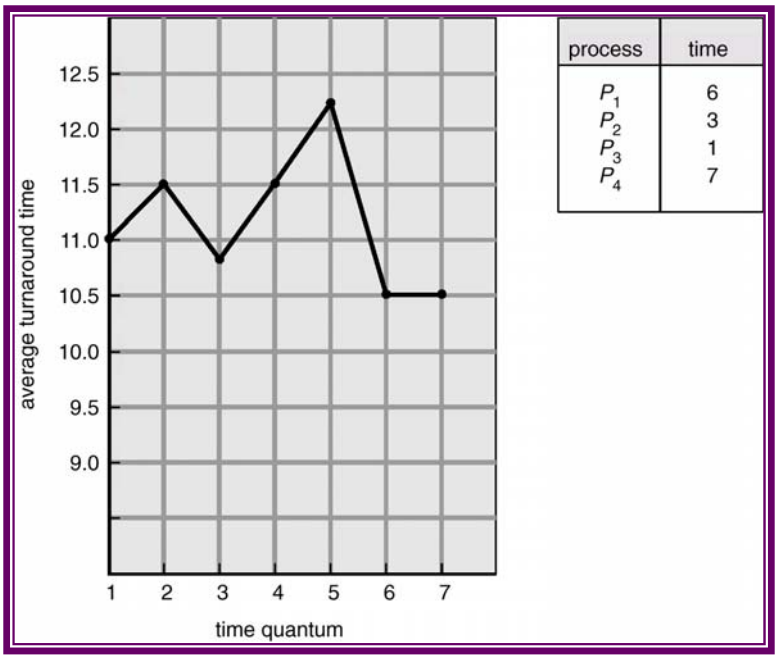
Waktu tunggu untuk P_1 adalah 6, P_2 adalah 4, dan P_3 adalah 7 sehingga rata-rata waktu tunggu adalah (6 + 4 + 7)/3 = 5.66 milidetik.

Algoritma Round-Robin ini di satu sisi memiliki keuntungan, yaitu adanya keseragaman waktu. Namun di sisi lain, algoritma ini akan terlalu sering melakukan switching seperti yang terlihat pada Gambar 4-4. Semakin besar quantum-timenya maka switching yang terjadi akan semakin sedikit.



Gambar 4-4 : Menunjukkan waktu kuantum yang lebih kecil meningkatkan context switch

Waktu turnaround juga tergantung ukuran waktu quantum. Seperti pada Gambar 4-5, rata-rata waktu turnaround tidak meningkat bila waktu quantum dinaikkan. Secara umum, rata-rata waktu turnaround dapat ditingkatkan jika banyak proses menyelesaikan CPU burst berikutnya sebagai satu waktu quantum. Sebagai contoh, terdapat tiga proses masing-masing 10 unit waktu dan waktu quantum 1 unit waktu, rata-rata waktu turnaround adalah 29. Jika waktu quantum 10, sebaliknya, rata-rata waktu turnaround turun menjadi 20.



Gambar 4-5 : Menunjukkan waktu turnaround berbeda pada waktu quantum yang berbeda

LATIHAN SOAL :

1. Sebutkan perbedaan antara penjadwalan *preemptive* dan *nonpreemptive.*
2. Terdapat 5 job yang datang hampir pada saat yang bersamaan. Estimasi waktu eksekusi (*burst time*) masing-masing 10, 6, 2, 4 dan 8 menit dengan prioritas masing-masing 3, 5, 2, 1 dan 4, dimana 5 merupakan prioritas tertinggi. Tentukan rata-rata *waktu turnaround* untuk penjadwalan CPU dengan menggunakan algoritma
   a. Round Robin (quantum time = 2)
   b. Priority
   c. Shortest job first
3. Diketahui proses berikut :

Proses	***Arrival Time***	***Burst Time***

*P*1	0.0	8
*P*2	0.4	4
*P*3	1.0	1

Tentukan rata-rata waktu tunggu dan rata-rata waktu turnaround dengan algoritma penjadwalan

a. FCFS

b. SJF non preemptive

c. SJF preemptive / SRTF

d. Round Robin dengan quantum time = 1

4. Suatu algoritma penjadwalan CPU kemungkinan melibatkan algoritma yang lain, contohnya algoritma FCFS adalah algoritma RR dengan waktu quantum tertentu. Apakah ada hubungan antara pasangan algoritma berikut ?

a. Priority dan SJF

b. Priority dan FCFS

c. RR dan SJF