Scheduling Algorithms

Shortest Job First and Sortest Remaining Time First

Non-preemptive vs. Preemptive Scheduling

Under *non-preemptive scheduling*, each running process keeps the CPU until it completes or it switches to the waiting (blocked) state

Under *preemptive scheduling,* a running process may be also forced to release the CPU even though it is neither completed nor blocked.

- In time-sharing systems, when the running process reaches the end of its time *quantum*

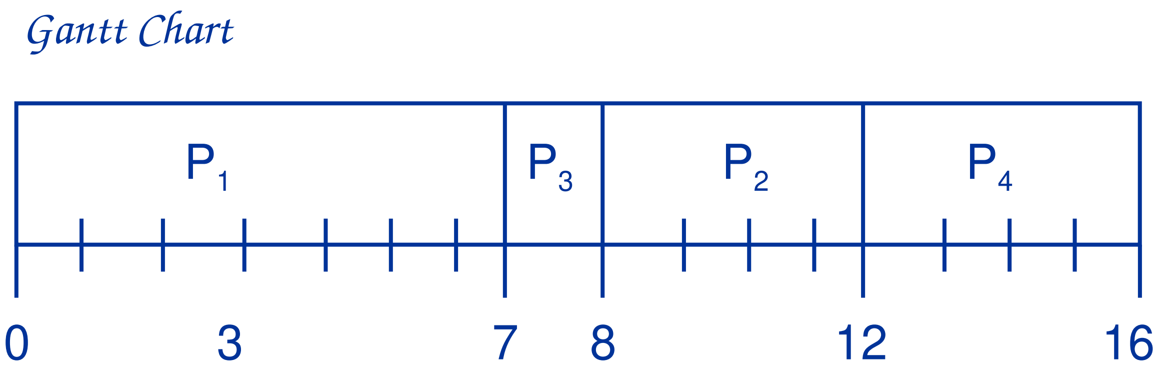
Algorithm 3: Non-Preemptive Shortest Job First (SJF)

Characteristics:

- Always assign job with shortest next CPU burst to CPU

Example:

Process	Arrival	Burst
P1	0	7
P2	2	4
P3	4	1
P4	5	4

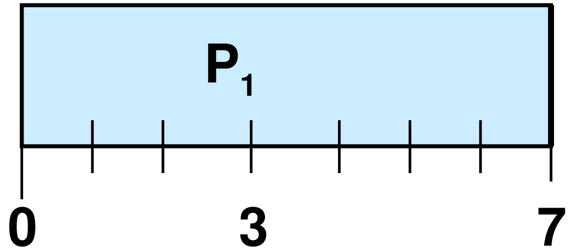


Avg wait time: (0 + 6 + 3 + 7) / 4 = 4 ms

Example for 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)

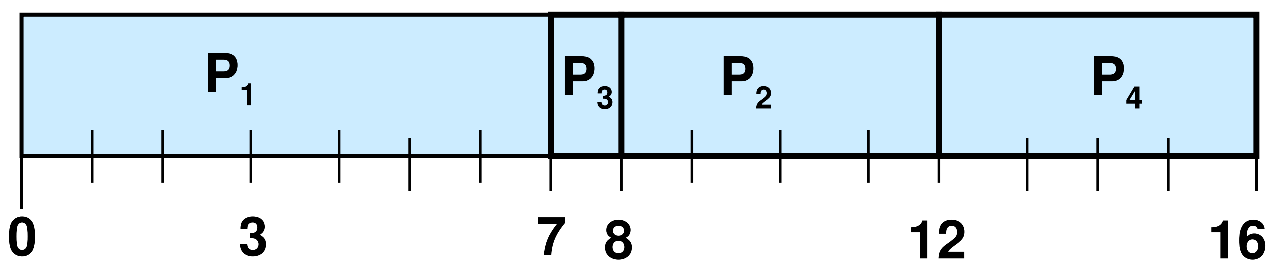


At time 0, P_1 is the only process, so it gets the CPU and runs to completion

Example for 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

Once P_1 has completed the queue now holds P_2, P_3 and P_4



P_3 gets the CPU first since it is the shortest. P_2 then P_4 get the CPU in turn (based on arrival time)

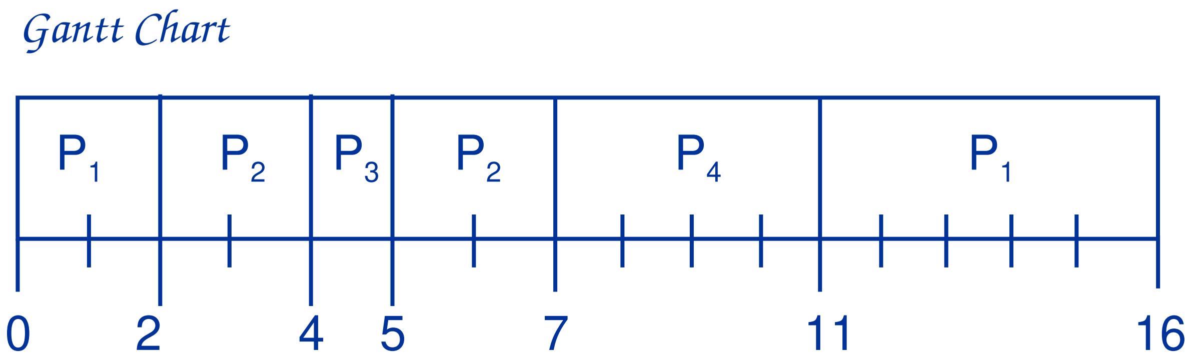
Algorithm 4: Preemptive Shortest Job First (SJF)-Shortest-Remaining-Time-First (SRTF)

Characteristics:

- Like Shortest Job First, but running job can be preempted.

Example:

Process	Arrival	Burst
P1	0	7
P2	2	4
P3	4	1
P4	5	4



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

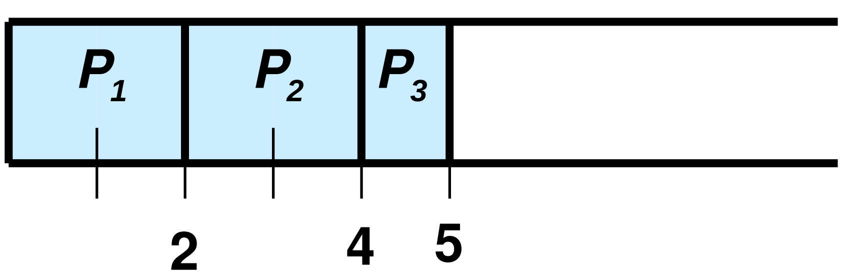
Example for Preemptive SJF (SRTF)

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

Time 0 – P_1 gets the CPU Ready = [(P_1,7)]

Time 2 – P_2 arrives – CPU has P_1 with time=5, Ready = [(P_2,4)] – P_2 gets the CPU

Time 4 – P_3 arrives – CPU has P_2 with time = 2, Ready = [(P_1,5),(P_3,1)] – P_3 gets the CPU



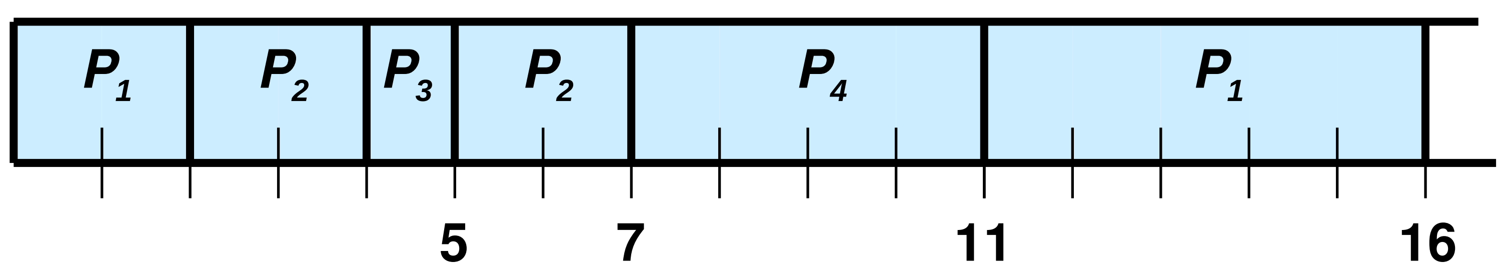
Example for Preemptive SJF (SRTF)

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

Time 5 – P_3 completes and P_4 arrives - Ready = [(P_1,5),(P_2,2),(P_4,4)] – P_2 gets the CPU

Time 7 – P_2 completes – Ready = [(P_1,5),(P_4,4)] – P_4 gets the CPU

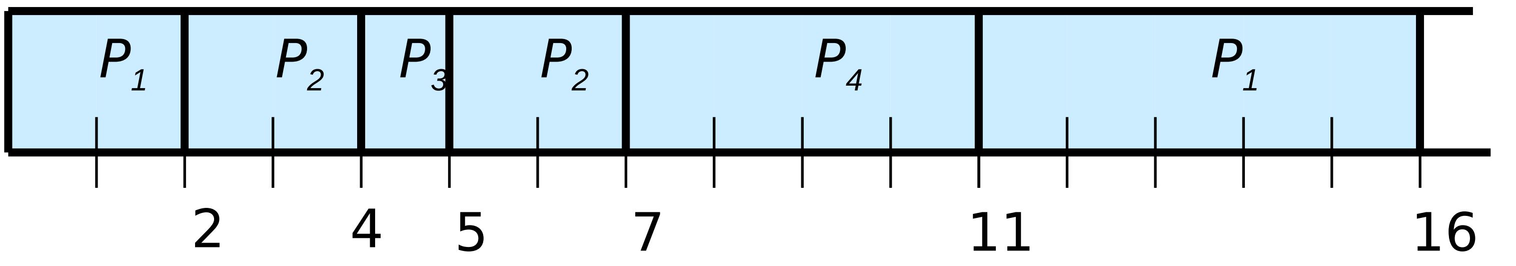
Time 11 – P_4 completes, P_1 gets the CPU



Example for Preemptive SJF (SRTF)

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

Implementing SJF

SJF is great, but how do we implement it?

- We don’t know a priori how long a job’s burst time is
- We have to try to *predict* the burst time

Priority-Based Scheduling

A priority number (integer) is associated with each process

The CPU is allocated to the process with the highest priority (smallest integer $\equiv$ highest priority).

- Preemptive
- Non-preemptive

SJF is a priority scheme with the priority the remaining time.

Example for Non Preemptive Priority-based Scheduling

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