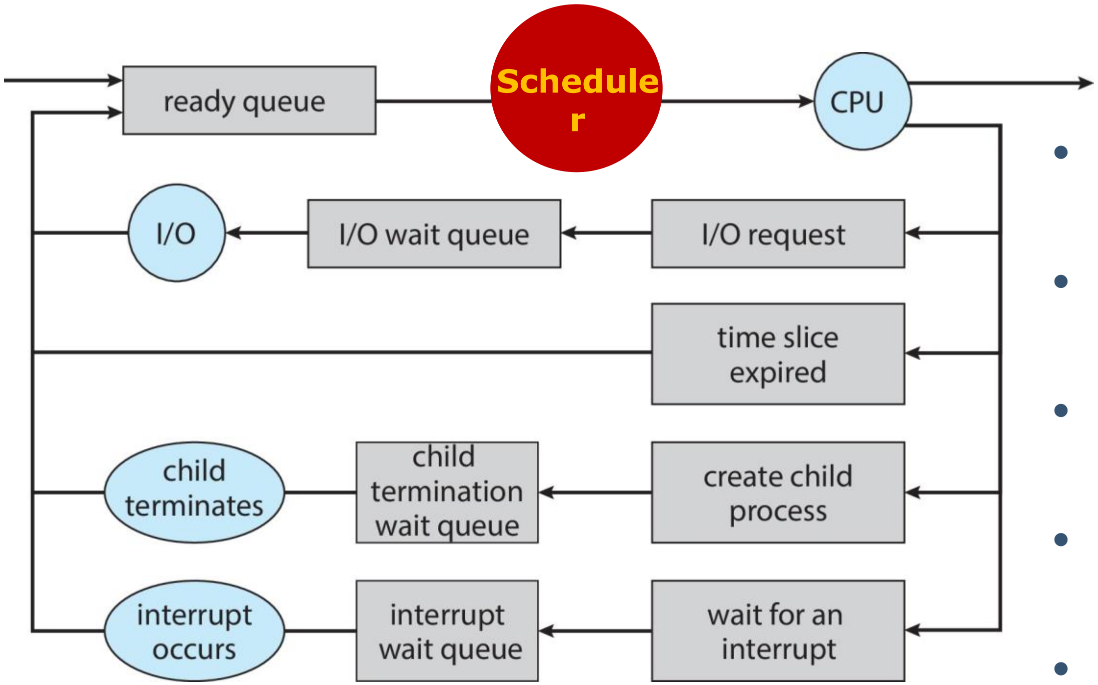
Process Management

- Scheduling Policies-I -

Slide acknowledgement: Dr. Sameh, Queen's

Recall: Basics and Criteria



- Maximizing CPU Utilization
- Maximizing Throughput
- Minimizing Turnaround Time
- Minimizing Waiting Time
- Minimizing Response Time

First Come First Serve (FCFS)

- Schedule process that requested execution first
  - Oldest process in the ready queue
  - Manage ready queue using a first-in, first-out policy.
    - When CPU idle, scheduler selects head of line (HOL)
    - Once scheduled, process is removed from the queue, next one become HOL
    - New process added to queue tail

FCFS Example

Process	Arrival Time	Burst Time
P_1	-3	24
P_2	3	3
P_3	9	3

The Gantt Chart for the schedule is:

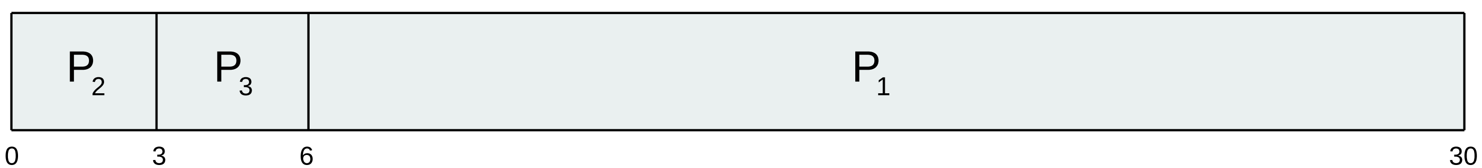
Process	Waiting Time
P_1	$0 - (-3) = 3$ ms
P_2	$24 - 3 = 21$ ms
P_3	$27 - 9 = 18$ ms
Average	14 ms

FCFS Scheduling (Cont.)

Suppose that the processes arrive in the order:

$$P_2 , P_3 , P_1$$

- The Gantt chart for the schedule is:



- Waiting time for P1 = 6; P2 = 0; P3 = 3
- Average waiting time: (6 + 0 + 3)/3 = 3
- Much better than previous case
- **Convoy effect** - short process behind long process
  - Consider one CPU-bound and many I/O-bound processes
  - Is there a convoy effect in the above Gantt chart?

FCFP Pros and Cons

- Pros:
  - Simple and easy to implement
  - Fair – every process will eventually run as long as CPU is not blocked by a process
- Cons:
  - Waiting time depends on arrival order
  - Short process stuck waiting for longer process – convoy effect
  - No differentiation between processes
  - Nonpreemptive

Shortest Job First (SJF)

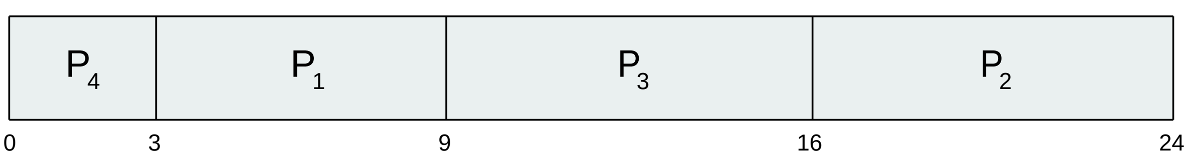
- Schedule process with shortest burst time first
  - Associate with each process the length of its next CPU burst **?**
  - Use these lengths to schedule the process with the shortest time
  - Waiting-time optimal, but poor on turnaround and response times

- Example (nonpreemptive SJF):

Process	Arrival Time	Burst Time
P_1	-4	6
P_2	-3	8
P_3	-2	7
P_4	-1	3

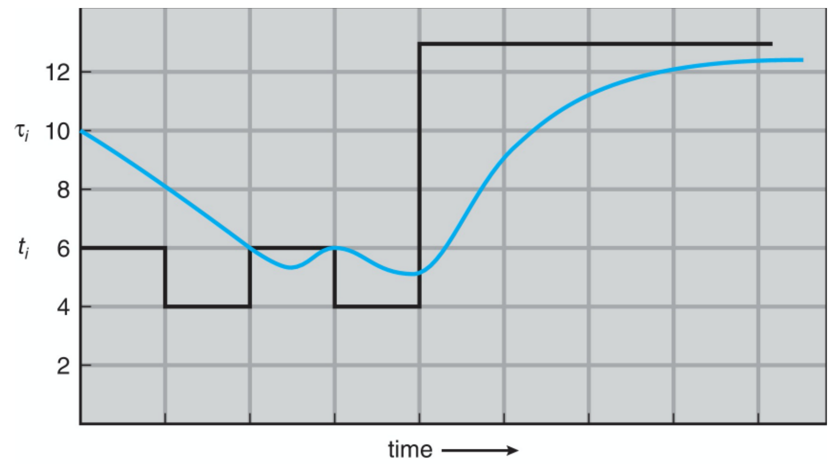
  - Average waiting time FCFS = (4 + 9+ 16 + 22)/4 = 12.75 ms
  - Average waiting time SJF = (1 + 7 + 11 + 19) / 4 = 9.5 ms

  - SJF



Predicting Burst Times

- Predicting ?
  - Knowing exact burst times is impossible, must be predicted
  - Typically, should have some correlation with previous bursts
- Exponential average based prediction:
  - τ_n: Predicted value for n-th burst
  - t_n = Actual time of n-th burst
  - $\tau_{n+1} = \alpha . t_n + (1 - \alpha) . \tau_n$
  - α must be in [0,1], typically = ½



CPU burst (t_i)	6	4	6	4	13	13	13	…
"guess" (τ_i)	10	8	6	6	5	9	11	12

Preemptive SJF

- Shortest Remaining Time First (SRTF)
  - Stop current process and schedule new one if burst time for latter shorter than remaining time former.
  - Now we add the concepts of varying arrival times and preemption to the analysis considering four processes and the length of CPU burst in milliseconds (msec).

Process	*Arrival* Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

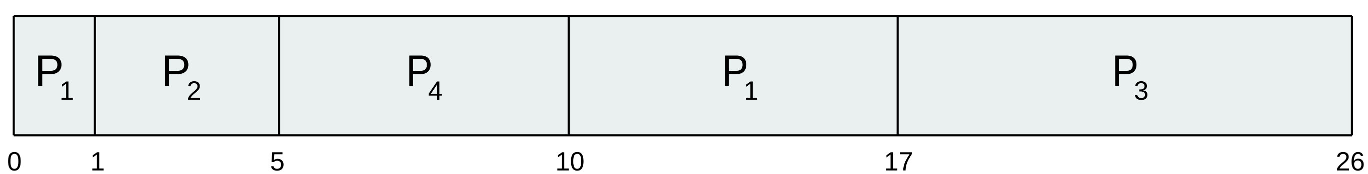
Waiting Time in SRTF

- **Waiting time = Start time of last executed segment – Arrival time – Scheduled time of all prior segments**

- Example

Process	*Arrival* Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

- Waiting time (P_1) = 10 – 0 – 1 = 9 ms
- Waiting time (P_2) = 1 – 1 – 0 = 0 ms
- Waiting time (P_3) = 17 – 2 – 0 = 15 ms
- Waiting time (P_4) = 5 – 3 – 0 = 2 ms



- Average waiting time = (9+0+15+2)/4 = 26/4 = 6.5 ms

Round Robin (RR)

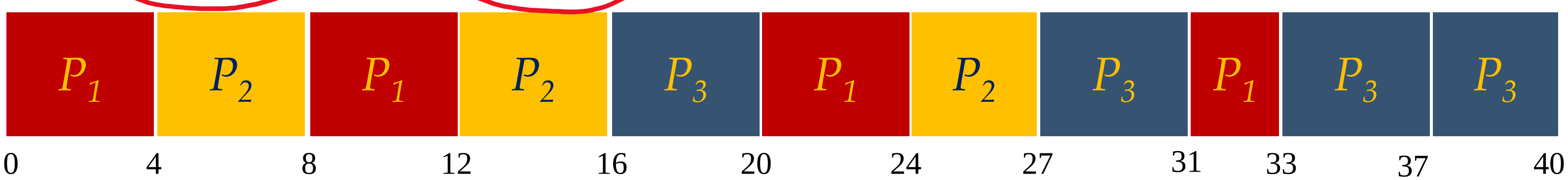
- FCFS with preemption to enable time sharing
  - Defines a time quantum/slice (10 - 100 ms)
  - **Circular operation**
    - Process bursts sequentially get up to 1 slice
    - First burst get another slice after slice of last one expires, and so on
- Implementable in a circular FIFO queue
  - Timer interrupts every quantum to schedule next process
  - Interrupt occurs = context switching, current process sent to tail
  - A terminating process releases CPU for next process
- Pros: Fast response time
- Con: Long waiting and turnaround times

Example

The process will complete at Round 5

Time slice: 4 ms

Process	Arrival Time	Burst Time	After R1	After R2	After R3	After R4
P_1	0	14	10	6	2	-
P_2	1	11	7	3	-	-
P_3	9	15	-	11	7	3



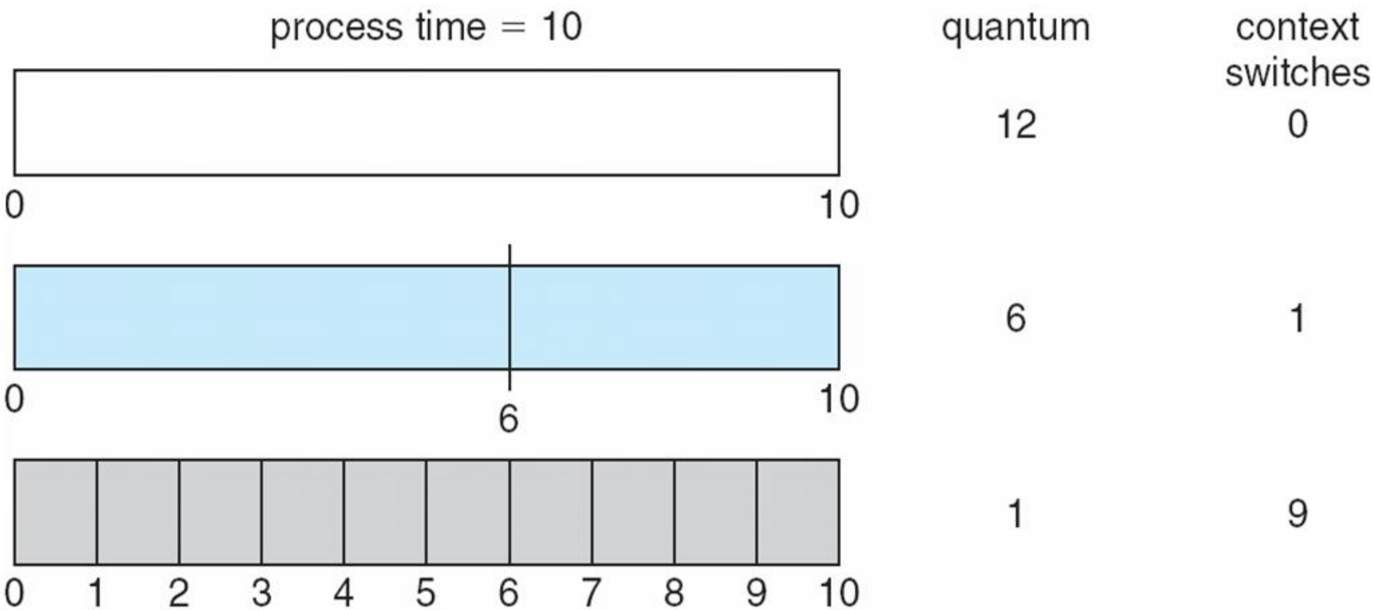
Process	Waiting Time	Turnaround Time	Response Time
P_1	31 – 0 – 12 = 19 ms	33 – 0 = 33 ms	0 – 0 = 0 ms
P_2	?	?	4 – 1 = 3 ms
P_3	?	?	16 – 9 = 7 ms
Average	?	?	3.33 ms

RR Facts and Considerations

- N bursts in ready queue, t_s ms per slice
  - Each burst gets $\leq 1/N$ of the time
  - No burst waits more than $(N - 1) \times t_s$ without executing
  - Response time $\leq (N - 1) \times t_s$
- t_s must be carefully selected
  - $t_s \gg \quad \Rightarrow$ FCFS
  - $t_s \ll \quad \Rightarrow$ Too many context switching $\Rightarrow$ Low CPU utilization
  - $t_s \ll \quad \Rightarrow$ Typically higher average turnaround and waiting times

80% of CPU burst times $\leq t_s$

Time Quantum and Context Switch Time



Assume, a process is of 10 time unit:

- If q = 12 time unit, process finishes in 1 time q, no context switch
- If q = 6 time unit, process finishes in 2 time q, 1 context switch
- If q = 1 time unit, process finishes in 10 time q, 9 context switch