BITS, PILANI – K. K. BIRLA GOA CAMPUS

Operating Systems

by

Mrs. Shubhangi Gawali

Dept. of CS and IS

OPERATING SYSTEMS

LECTURE 12: CPU SCHEDULING

Scheduling Algorithms

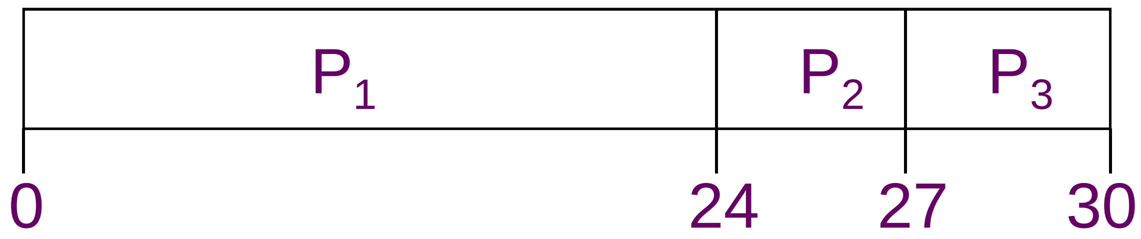
First-Come-First-Served (FCFS)

- Selection function: the process that has been waiting the longest in the ready queue – hence called First-Come First-Served (FCFS).
  - Each process joins the Ready queue
  - When the current process finishes its execution, the oldest process in the Ready queue is selected
- Decision mode: Nonpreemptive.
  - A short process may have to wait a very long time before it can execute
  - Favors CPU-bound processes
  - I/O processes have to wait until CPU-bound process completes

First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_1 , P_2 , P_3
  The Gantt Chart for the schedule is:



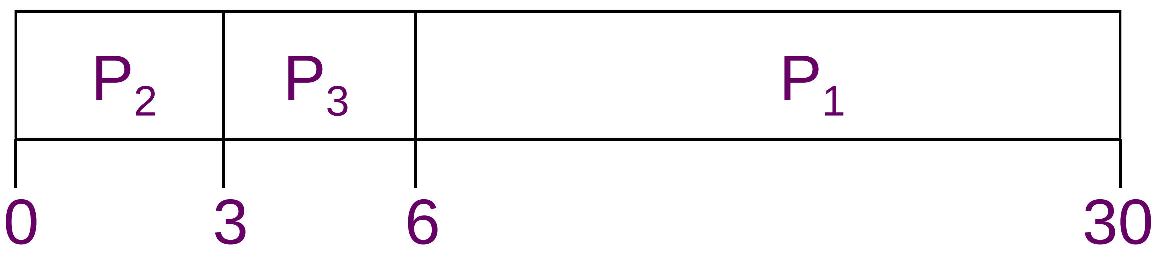
- Waiting time for P_1 = 0; P_2 = 24; P_3 = 27
- Average waiting time: (0 + 24 + 27)/3 = 17
- Turn around time P1 = 24; P2 = 27; P3 = 30
- Average Turn around time = (24 + 27 + 30)/3 = 27

A twist in FCFS Scheduling example

Suppose that the processes arrive in the order

$$P_2 , P_3 , P_1$$

- The Gantt chart for the schedule is:

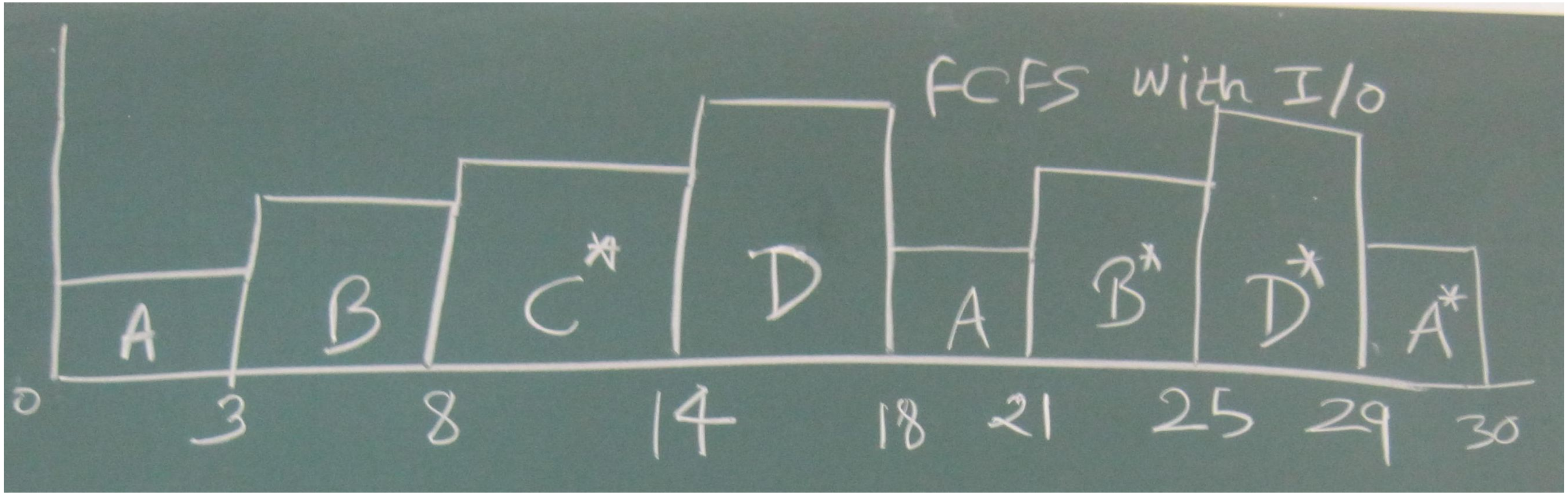


- Waiting time for $P_1 = 6; P_2 = 0, P_3 = 3$
- Average waiting time: (6 + 0 + 3)/3 = 3
- Much better than previous case
- ***Convoy effect*** short process behind long process

FCFS with I/O

Process	Arrival Time	Execution Time
A	0	7
B	2	9
C	4	6
D	6	8

- Assume process A goes for I/O for 5 units of time after every 3 unit execution in CPU
- Assume B goes for I/O for 2 units after 5 units of execution in CPU
- Process C is a CPU bound process with no I/O
- Process D goes for I/O for 1 unit after 4 units of execution in CPU.



Process	AT	CT	TA = CT-AT	EP	BP	WP = TA-EP-BP
A	0	30	30	7	10	13
B	2	25	23	9	2	12
C	4	14	10	6	0	4
D	6	29	23	8	1	14
AVG						(13+12+4+14)/4=10.75

FCFS Drawbacks

- A process that does not perform any I/O will monopolize the processor (Convoy Effect).
- Favors CPU-bound processes:
  - I/O-bound processes have to wait until CPU-bound process completes.
  - They may have to wait even when their I/O are completed (poor device utilization).
  - We could have kept the I/O devices busy by giving a bit more priority to I/O bound processes.

Quick lookup about FCFS

- Which process to select? (scheduling policy)

- How long to schedule (Preemptive/NonPreemptive)

  OR

  When to invoke Scheduler?

  (Calculation of Next decision point)

- Implementation

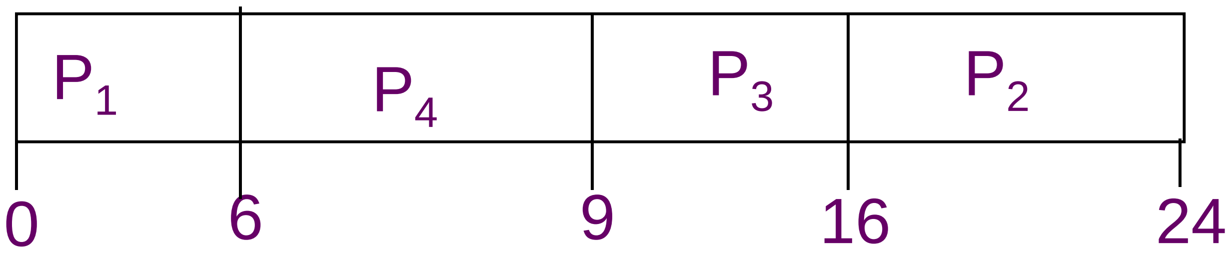
Shortest-Job-First (SJF) Scheduling

- Associate with each process the length of its next CPU burst. Use these lengths to schedule the process with the shortest time
- SJF is optimal – gives minimum average waiting time for a given set of processes
  - The difficulty is knowing the length of the next CPU request
- Nonpreemptive policy
- Process with shortest expected processing time is selected next
- Short process jumps ahead of longer processes

Example of SJF

Process	Arrival Time	Burst Time
P_1	0.0	6
P_2	1.0	8
P_3	2.0	7
P_4	3.0	3

- SJF scheduling chart



- Average waiting time = (0 + 15 + 7 + 3) / 4 = 6.25

Shortest-Job-First (SJF) Scheduling

- SJF is optimal – gives minimum average waiting time for a given set of processes
  - The difficulty is knowing the length of the next CPU request
- Non preemptive policy
- Process with shortest expected processing time is selected next
- Short process jumps ahead of longer processes

Shortest-Job-First (SJF) Scheduling

- If estimated time for process not correct, the operating system may abort it
- Possibility of starvation for longer processes

Determining Length of Next CPU Burst

- Can only estimate the length
- Can be done by using the length of previous CPU bursts, using exponential averaging

1. $\boldsymbol{t_n}$ = actual length of $\boldsymbol{n^{th}}$ CPU burst
2. $\tau_{\boldsymbol{n}+1}$ = predicted value for the next CPU burst
3. $\alpha, 0 \le \alpha \le 1$
4. Define:

$$\tau_{\boldsymbol{n}+1} = \alpha \, \boldsymbol{t}_{\boldsymbol{n}} + (1 - \alpha) \tau_{\boldsymbol{n}}.$$

Examples of Exponential Averaging

- $\alpha = 0$
  - $\tau_{n+1} = \tau_n$
  - Recent history does not count
- $\alpha = 1$
  - $\tau_{n+1} = \alpha\, t_n$
  - Only the actual last CPU burst counts
- If we expand the formula, we get:

$$\tau_{n+1} = \alpha\, t_n + (1 - \alpha)\alpha\, t_n - 1 + \ldots$$
$$+ (1 - \alpha)^j \alpha\, t_{n-j} + \ldots$$
$$+ (1 - \alpha)^{n+1} \tau_0$$

- Since both α and $(1 - \alpha)$ are less than or equal to 1, each successive term has less weight than its predecessor