Information Technology Institute

Operating System Fundamentals

Chapter Five

CPU SCHEDULING

Table of Content

- Basic Concepts
- Scheduling Criteria
- Scheduling Algorithms

BASIC CONCEPTS

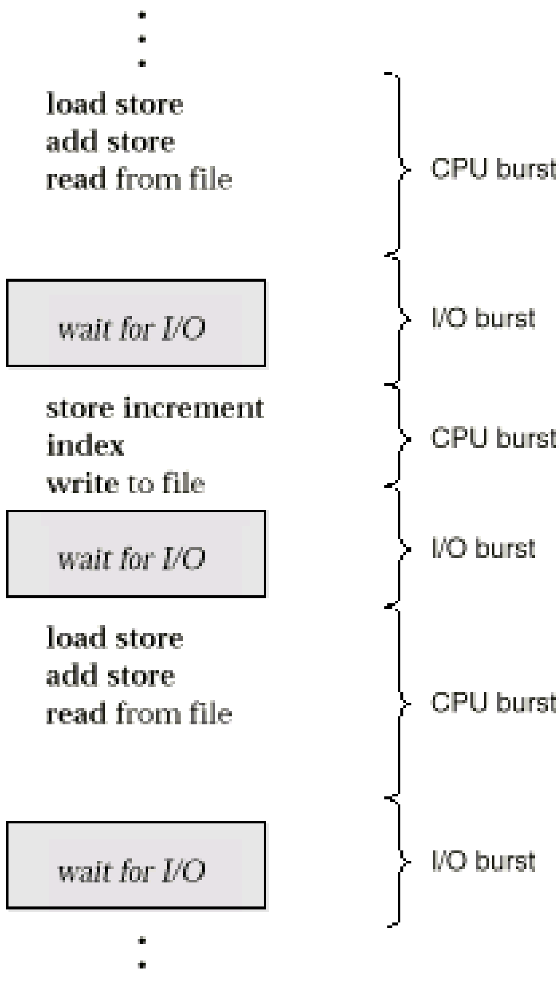
Basic Concepts

- Maximum CPU utilization obtained with multitasking
- CPU–I/O Burst Cycle
  - Process (1) execution consists of a cycle of CPU execution and (2) I/O wait.

كل بروسيز وقتها بتقضيه لاما وقت جوه السي بي يو او وقت مستنيه الانبوت اوتبوت

وانا عايز مشكل من ده وده لان لو نفس النوع كده هيقف

Alternating Sequence of CPU And I/O Bursts



CPU Scheduler

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

(4)

- CPU scheduling decisions may take place when a process:

لو البروسيز محتاجه انبوت اوتبوت

1. Switches from running to waiting state.

لو البروسيز جه عليها انتربت او عدت وقتها

2. Switches from running to ready state.
3. Switches from waiting to ready. ده لما يدخل الانبوت اوتبوت
4. Terminates. دي الحاله الطبيعيه لما البروسيز تخلص

CPU Scheduler Cont'd

- (1) Preemptive هنا البروسيز مش تخلص عادي وواحده تانيه تدخل
  و هنا احيانا بيضيع وقت اكتر من النون برايميتيف
  - Process release the CPU before it finish execution
  - Example: Modern OS: Unix, Linux, Windows7

- (2) Non-preemptive هنا البروسيز مش هتخرج الا لما تخلص
  - Process release CPU when:
    - Running → Waiting هتخرج لو عايزه انبوت اوتبوت
    - Running → Terminated
  - Example: MS Windows 3.1

Dispatcher

ده الي بيوزع ويحكم مين الي هيدخل

- Gives control of the CPU to the process selected by the short-term scheduler:
  - switching context
  - switching to suitable mode (User or Monitor)
  - بيحط الي هيطلعها ولسه مخلصتش في مكان محدد علشان ندخلها تاني
  - jumping to the proper location in the user program to restart that program
- Dispatch latency انا عايز اقلل الوقت ده
  - time taken by dispatcher to stop one process and start another running. ده الوقت الي بدخل وخرج فيه

SCHEDULING CRITERIA

Scheduling Criteria

- CPU utilization هنا البروسيسر قعد قد ايه شغال على الوقت الكلي وده عايزين نزوده
  - Keep the CPU as busy as possible
- Throughput ده عدد الناس او البروسيز ال بيخرج خلال وقت معين وده عايزين نزوده
  - Number of processes that complete their execution per time unit
- Turnaround time ده الوقت الكلي الي البروسيس بتاخده علشان تتنفذ يعني الوقت ده يشمل وقت الانتظار
  - Amount of time to execute a particular process
- Waiting time وقت الانتظار
  - Amount of time a process has been waiting in the ready queue
- Response time يعني البروسيز قعدت وقت قد ايه قبل ما البروسيسور يعبرها ويستجيب ليها وعايز الوقت ده يقل
  - Amount of time it takes from when a request was submitted until the first response is produced, **not** output (for time-sharing environment)

Optimization Criteria

- Maximize
  - 1 CPU Utilization
  - 2 Throughput
- Minimize
  - 1 Turnaround time
  - 2 Waiting time
  - 3 Response time

- Considerations
  - Minimize maximum response time
  - Minimize the variance of response times

SCHEDULING ALGORITHMS

Scheduling Algorithms

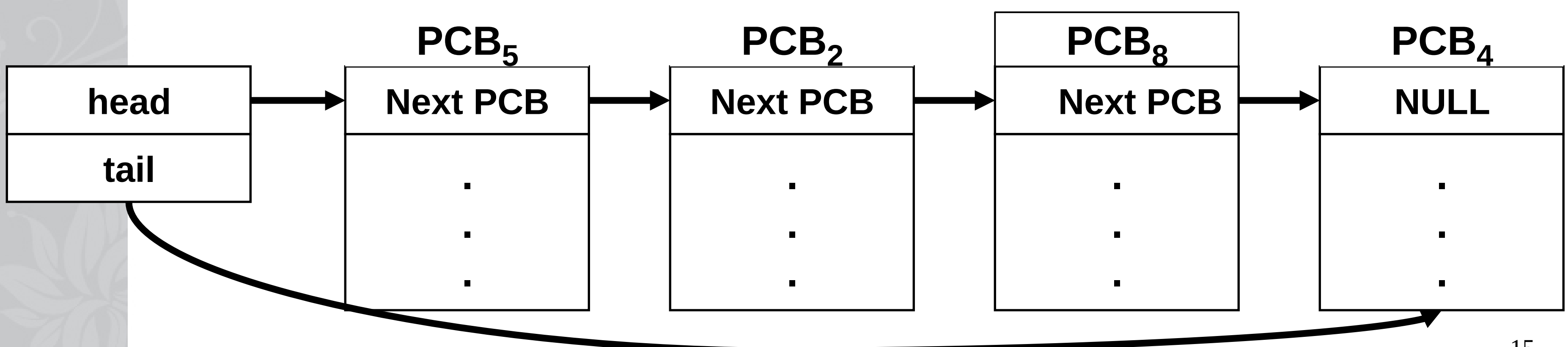
عندي لوغاريتم وعايزين نشوف مين الاحسن فيهم وممكن مش يكون في احسن ولكن لكل واحده مزايا

- 1 First-Come First Served FCFS
- 2 Shortest-Job First SJF
- 3 Priority
- 4 Round-Robin

First-Come, First-Served (FCFS) Scheduling

لو حظي حلو هنا القصيرهو الي هييجي الاول

- Easily implemented
- Ready queue is FIFO First input first output
- P_n ready → P_n PCB is linked to tail of queue
- Process at head of ready queue → CPU
- Average waiting time is long!



Example 1

Process	Burst Time
P1	24
P2	3
P3	3

FCFS AND NON-PREEMITIVE

هو هنا افترض وقت الوصول بي واحد ثم اتنين ثم تلاته

- Suppose that the processes arrive in the order: *P1* , *P2* , *P3* The Gantt Chart for the schedule is:

P1	P2	P3

0 24 27 30

- Waiting time for *P1* = 0; *P2* = 24; *P3* = 27
- Average waiting time: (0 + 24 + 27)/3 = 17

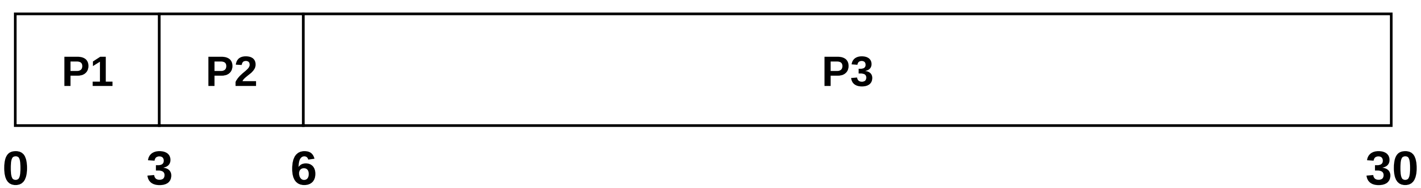
Example 2

FCFS AND NON-PREEMITIVE

هنا من حظي ان الصغيرين وصلو الاول

Process	Burst Time
P1	3
P2	3
P3	24

- Suppose that the processes arrive in the order: *P1* , *P2* , *P3* The Gantt Chart for the schedule is:



- Waiting time for *P1* = 0; *P2* =3; *P3* = 6
- Average waiting time: (0 + 3 + 6)/3 = 3

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.
- Two schemes:
  - 1 Non-preemptive – once CPU given to the process it cannot be preempted until completes its CPU burst.
  - 2 Preemptive – if a new process arrives with CPU burst length less than remaining time of current executing process, preempt. This scheme is know as the Shortest-Remaining-Time-First (SRTF).
- SJF is optimal
  - Gives minimum average waiting time for a given set of processes.

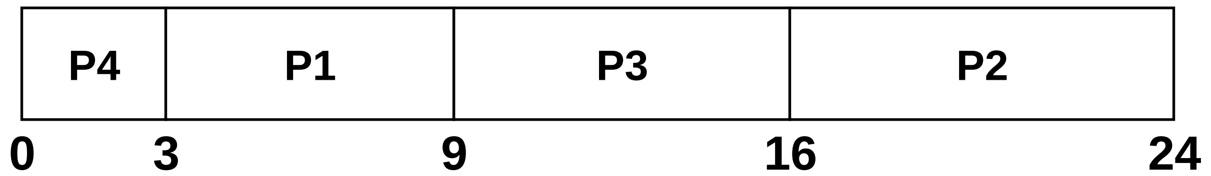
Example 1

Process	Burst Time
P1	6
P2	8
P3	7
P4	3

SJF

NON-PRE

- Suppose that all processes arrive at the same time: The Gantt Chart for the schedule is:



- Waiting time for *P1* = 3; *P2* =16; *P3* =9; P4=0
- Average waiting time: (3 + 16 + 9 + 0)/4 = 7

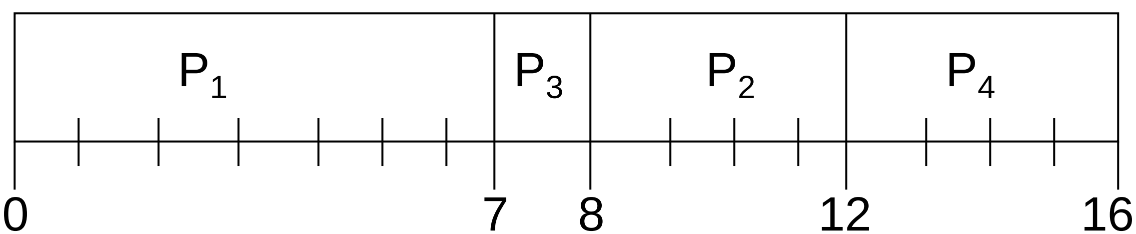
Example of Non-Preemptive SJF

هنا بي واحد جت الاول ف لازم يشتغل فيها علشان مش يقعد فاضي

Process	Arrival Time	Burst Time
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

وقت بي اتنين واربعه قد بعض يبقى هناخد الي واصل بدري الاول

- SJF (non-preemptive)



لما الدقيقه السابعه كان جت كان التلاته بروسيسس جم يعني لو فى واحده بس هي الى وصلت في الدقيقه السابعه كنت هاخدها هي بغض النظر عن وقتها قد ايه

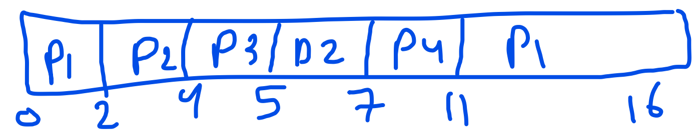
- Average waiting time = $(0 + 6 + 3 + 7)/4 = 4$

8-2 7-4 12-5

Example of Preemptive SJF

P1→5 P2→2

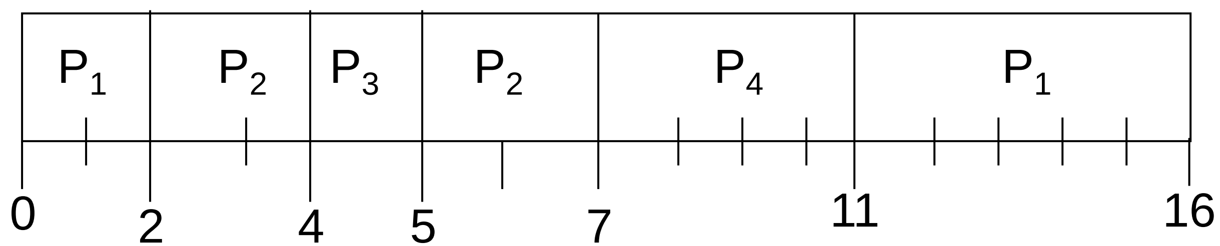
Process	Arrival Time	Burst Time
P1	0.0	5 ~~7~~
P2	2.0	2 ~~4~~
P3	4.0	~~1~~
P4	5.0	4



لما وصلنا للدقيقه خمسه كان عندي تلاته بروسيس وهما

$P_1 \to 5$, $P_2 \to 2$, $P_4 \to 4$

- SJF (preemptive)



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

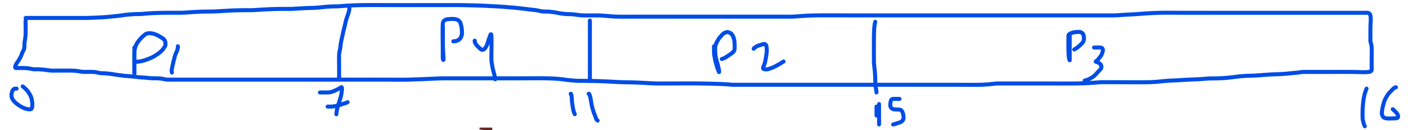
11-2

7-5

Priority Scheduling

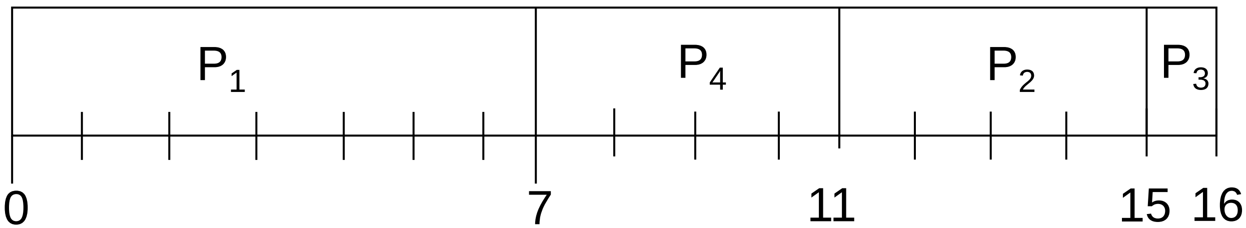
- Priority number (integer) is associated with each process
- The CPU is allocated to the process with the highest priority (smallest integer ≡ highest priority).
  - Preemptive
  - Non-preemptive
- SJF is a priority scheduling where priority is the predicted next CPU burst time.
  - Problem ≡ Starvation – low priority processes may never execute.
  - Solution ≡ Aging – as time progresses increase the priority of the process.

Example of Non-Preemptive Priority



Process	Arrival Time	Burst Time	Priority
P1	0.0	7	3
P2	2.0	4	2
P3	4.0	1	4
P4	5.0	4	1

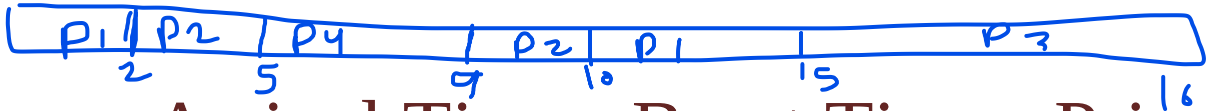
- Priority (Non-Preemptive) الدقيقه ا السابعه كان الباقي وصل



- Average waiting time = (0 + 9 + 11 +2)/4 = 22/4

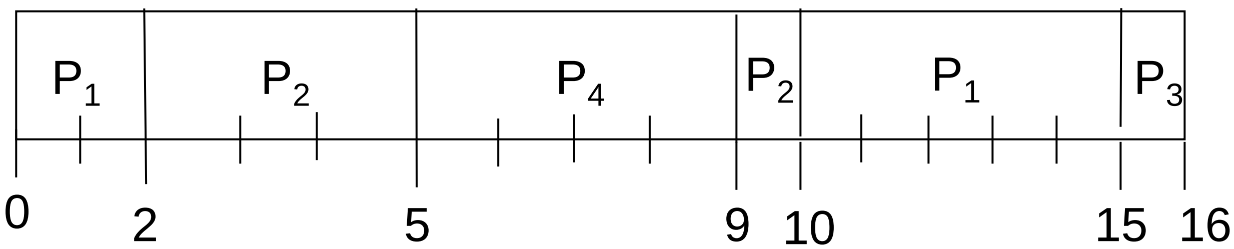
11-2 15-4 7-5

Example of Preemptive Priority



Process	Arrival Time	Burst Time	Priority
P1	0.0	5 ~~7~~	3
P2	2.0	x ~~4~~	2
P3	4.0	~~1~~	4
P4	5.0	~~4~~	1

- Priority (Preemptive)



- Average waiting time = (8 + 4 + 11 +0)/4 = 23/4

10-2 9-5 15-4

4- Round Robin (RR)

- Each process gets a small unit of CPU time (*time quantum*), usually 10-100 milliseconds. After this time has elapsed, the process is preempted and added to the end of the ready queue.
- If there are *n* processes in the ready queue and the time quantum is *q*, then each process gets 1/*n* of the CPU time in chunks of at most *q* time units at once. No process waits more than (*n*-1)*q* time units.
- Performance
  - *q* large $\rightarrow$ FIFO
  - *q* small $\rightarrow$ *q* must be large with respect to context switch, otherwise overhead is too high.

Example of RR, Time Quantum = 20

Process			Burst Time
P1	13	~~33~~	~~53~~
P2	0	0	17
P3	28	~~48~~	68
P4	0	~~4~~	24

$P_1 = 81$, $P_2 = 20$

wait

$P_3 = 94$, $P_4 = 97$

$\frac{81 + 20 + 94 + 97}{4} = 73$

- The Gantt chart is:

P_1	P_2	P_3	P_4	P_1	P_3	P_4	P_1	P_3	P_3

0 20 37 57 77 97 117 121 134 154 162

*Note: higher average turnaround than SJF, but better response.


Thank you