Single Processor Scheduling Algorithms

- Batch systems
  - First Come First Serve (FCFS)
  - Shortest Job First
- Interactive Systems
  - Round Robin
  - Priority Scheduling
  - Multi Queue & Multi-level Feedback
  - Shortest process time
  - Guaranteed Scheduling
  - Lottery Scheduling
  - Fair Sharing Scheduling

First Come First Serve (FCFS)

- Process that requests the CPU FIRST is allocated the CPU FIRST.
- Also called FIFO
- Preemptive or Non-preemptive?
- Used in Batch Systems
- Real life analogy?
  - Buying tickets?
- Implementation
  - FIFO queues
  - A new process enters the tail of the queue
  - The schedule selects from the head of the queue.
- Performance Metric: **Average Waiting Time**.
- Given Parameters:
  - Burst Time (in ms), Arrival Time and Order

FCFS Example

Process	Duration	Order	Arrival Time
P1	24	1	0
P2	3	2	0
P3	4	3	0

The final schedule (Gantt chart):

P1 waiting time: 0
P2 waiting time: 24
P3 waiting time: 27

The average waiting time:
(0+24+27)/3 = 17

What if P1 arrives at time 2

Problems with FCFS

- Non-preemptive
- Not optimal AWT
- Cannot utilize resources in parallel:
  - Assume 1 process CPU bounded and many I/O bounded processes
  - result: Convoy effect, low CPU and I/O Device utilization
  - Why?

Why Convoy Effects?

- Consider 100 I/O-bound processes and 1 CPU-bound job in the system.
- I/O-bound processes pass quickly through the ready queue and suspend themselves waiting for I/O.
- The CPU-bound process arrives at head of queue and executes the program until completion.
- I/O bound processes rejoin the ready queue and wait for the CPU-bound process releasing the CPU.
- I/O devices idle until the CPU-bound process completes.
- In general, a convoy effect happens when a set of processes need to use a resource for a short time, and one process holds the resource for a long time, blocking all of the other processes. Essentially, it causes poor utilization of the other resources in the system.

Shortest Job First (SJF)

- Schedule the job with the shortest duration time first
- Used in batch systems
- Two types:
  - Non-preemptive
  - Preemptive
- Requirement: the duration time needs to be known in advance
- Optimal if all jobs are available simultaneously (provable)
  - Gives the best possible AWT (average waiting time)

Non-preemptive SJF: Example

Process	Duration	Order	Arrival Time
P1	6	1	0
P2	8	2	0
P3	7	3	0
P4	3	4	0

Do it yourself

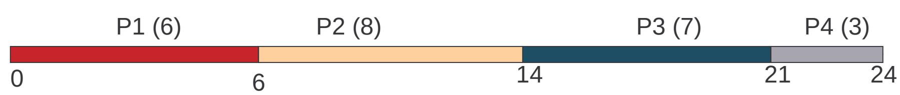
P4 waiting time: 0
P1 waiting time: 3
P3 waiting time: 9
P2 waiting time: 16

The total time is: 24
The average waiting time (AWT):
(0+3+9+16)/4 = 7

Comparing to FCFS

Process	Duration	Order	Arrival Time
P1	6	1	0
P2	8	2	0
P3	7	3	0
P4	3	4	0

Do it yourself



P1 waiting time: 0
P2 waiting time: 6
P3 waiting time: 14
P4 waiting time: 21

The total time is the same.
The average waiting time (AWT):
(0+6+14+21)/4 = 10.25
(comparing to 7)

SJF Is Not Always Optimal

- Is SJF optimal if all the jobs are not available simultaneously?

Process	Duration	Order	Arrival Time
P1	10	1	0
P2	2	2	2

Do it yourself

P1 waiting time: 0
P2 waiting time: 8

The average waiting time (AWT):
(0+8)/2 = 4

Preemptive SJF

- Also called Shortest Remaining Time First
  - Schedule the job with the shortest remaining time required to complete
- Requirement: the duration time needs to be known in advance

Preemptive SJF: Same Example

Process	Duration	Order	Arrival Time
P1	10	1	0
P2	2	2	2

P1 waiting time: 4-2 =2
P2 waiting time: 0

The average waiting time (AWT):
(0+2)/2 = 1

A Problem with SJF

- Starvation
  - In some scenarios, a job may wait for ever
  - Example: SJF
    - Process A with duration time of 1 hour arrives at time 0
    - But ever 1 minute, a shorter process with duration time of 2 minutes arrive
    - Result of SJF: A never gets to run
- What's the difference between starvation and a deadlock?

Interactive Scheduling Algorithms

- Usually preemptive
  - Time is **sliced** into quantum (time intervals)
  - Scheduling decision is also made at the beginning of each quantum
- Performance Criteria
  - Min Response time
  - best proportionality
- Representative algorithms:
  - Priority-based
  - Round-robin
  - Multi Queue & Multi-level Feedback
  - Shortest process time
  - Guaranteed Scheduling
  - Lottery Scheduling
  - Fair Sharing Scheduling

Priority Scheduling

- Each job is assigned a priority.
- FCFS within each priority level.
- Select highest priority job over lower ones.
- Rationale: higher priority jobs are more mission-critical
  - Example: DVD movie player vs. send email
- Problems:
  - May not give the best AWT
  - Starvation

Set Priority

- Two approaches
  - Static (for system with well known and regular application behaviors)
  - Dynamic (otherwise)
- Priority may be based on:
  - Cost to user.
  - Importance of user.
  - Aging
  - Percentage of CPU time used in last X hours.

Round-Robin (RR)

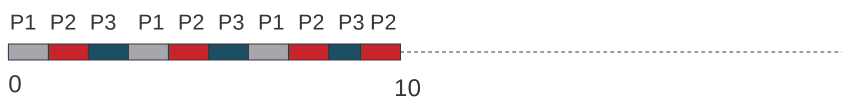
- One of the oldest, simple, commonly used scheduling algorithms
- Select process/thread from ready queue in a round-robin fashion (take turns)
- Problems:
  - Do not consider priority
  - More context switch overhead

Round-robin: Example

Process	Duration	Order	Arrival Time
P1	3	1	0
P2	4	2	0
P3	3	3	0

Suppose time quantum is: 1 unit, P1, P2 & P3 never block

Do it yourself



P1 waiting time: 4
P2 waiting time: 6
P3 waiting time: 6

The average waiting time (AWT):
(4+6+6)/3 = 5.33