PROCESS / TASK SCHEDULING ALGORITHMS

CHAPTER 5

MOTIVATION

Scheduling of processes is an important topic in operating systems… As most of the software are made so that processes can be scheduled accordingly.

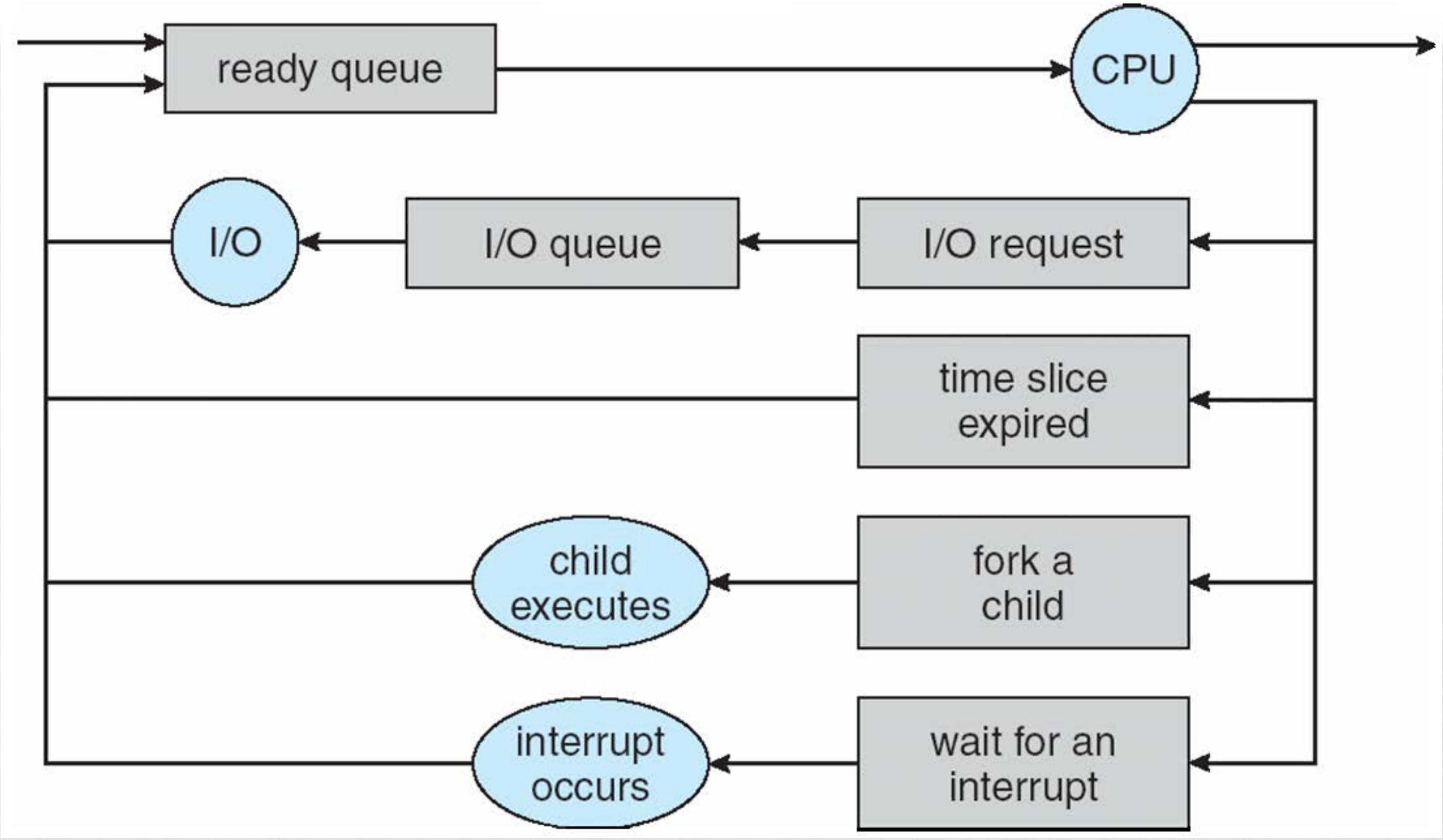
We should have a hands on these algorithms.

Therefore we have given a simple idea to revise these algos.

BACKGROUND

- In computer science, scheduling is the method by which threads, processes or data flows are given access to system resources (e.g. processor time).
- This is done to load balance a system effectively.
- The need for a scheduling algorithm arises from the requirement for modern systems to perform multitasking.

CPU SCHEDULING



<u>Scheduling</u> : **deciding which threads are given access to resources from moment to moment .**

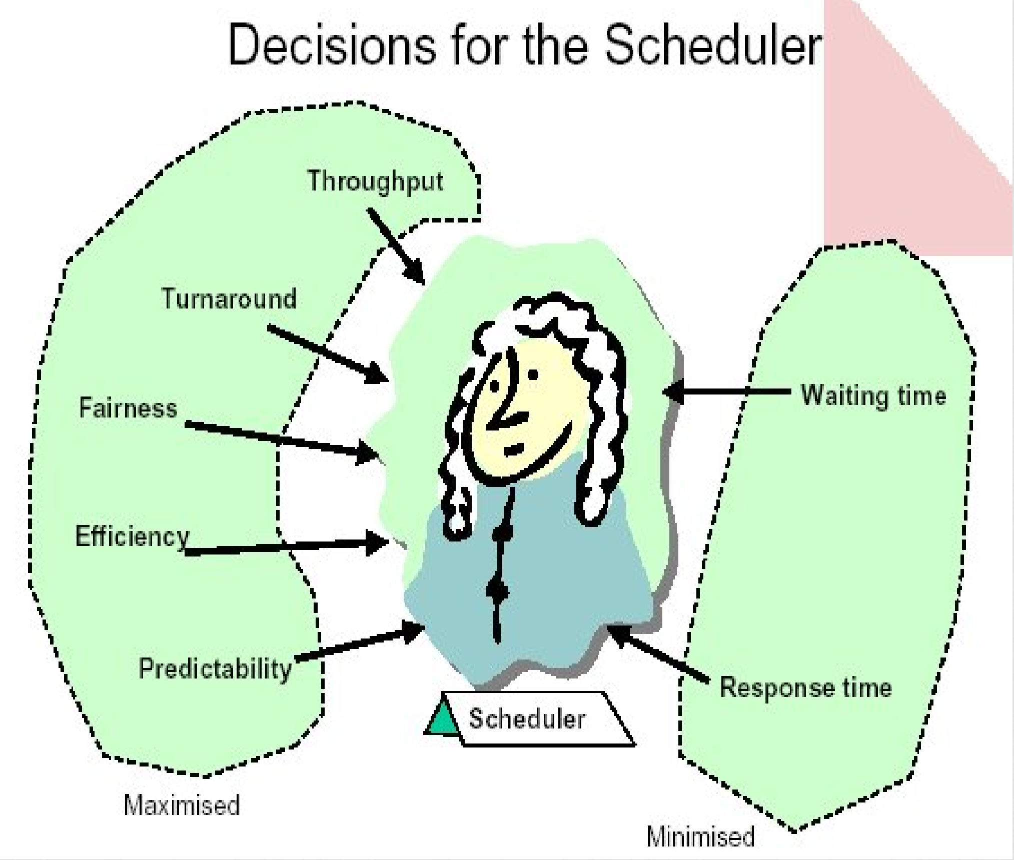
ASSUMPTIONS ABOUT SCHEDULING

Many implicit assumptions for CPU scheduling are:

- One program per user
- One thread per program
- Programs are independent
- These are unrealistic but simplify the problem

Does " fair " mean fairness among users or programs?

- If I run one compilation job and you run five, do you get five times as much CPU?
  - Often times, yes!
  - Goal: dole out CPU time to optimize some desired parameters of the system.



TYPES OF SCHEDULERS

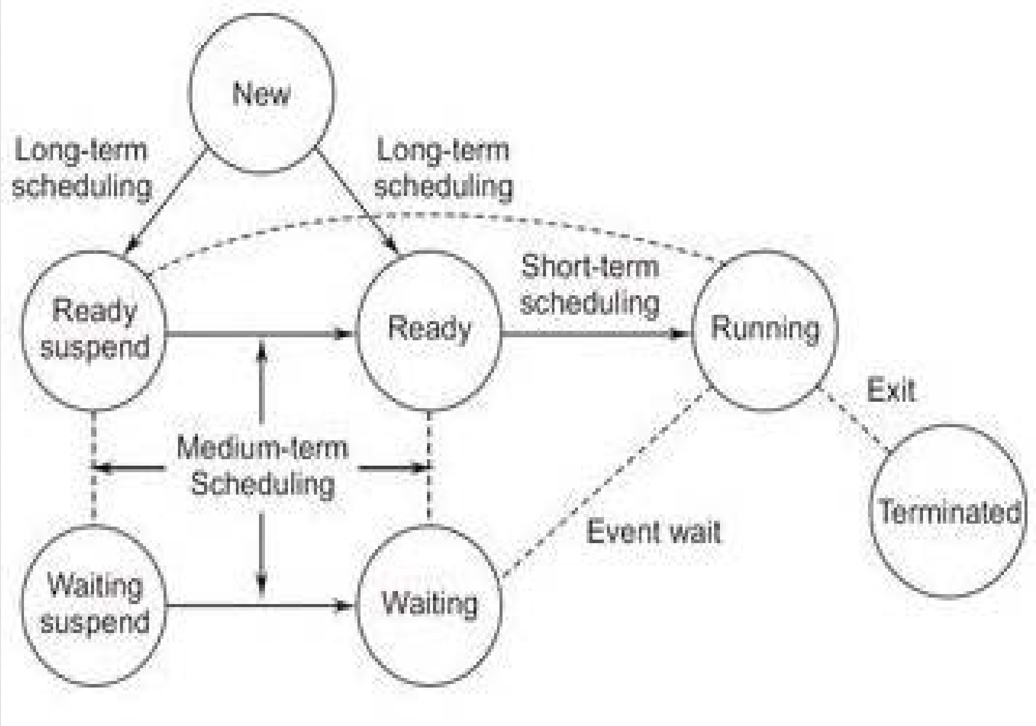
- **LONG TERM SCHEDULER:**

Long term scheduler determines which programs are admitted to the system for processing.

- **MEDIUM TERM SCHEDULER:**

Medium term scheduling is part of the swapping. It removes the processes from the memory.

It reduces the degree of multiprogramming. The medium term scheduler is in-charge of handling the swapped out-processes



TYPES OF SCHEDULERS

- **SHORT TERM SCHEDULER:**

It is the change of ready state to running state of the process. Short term scheduler also known as dispatcher, execute most frequently and makes the fine grained decision of which process to execute next. Short term scheduler is faster than long term scheduler.

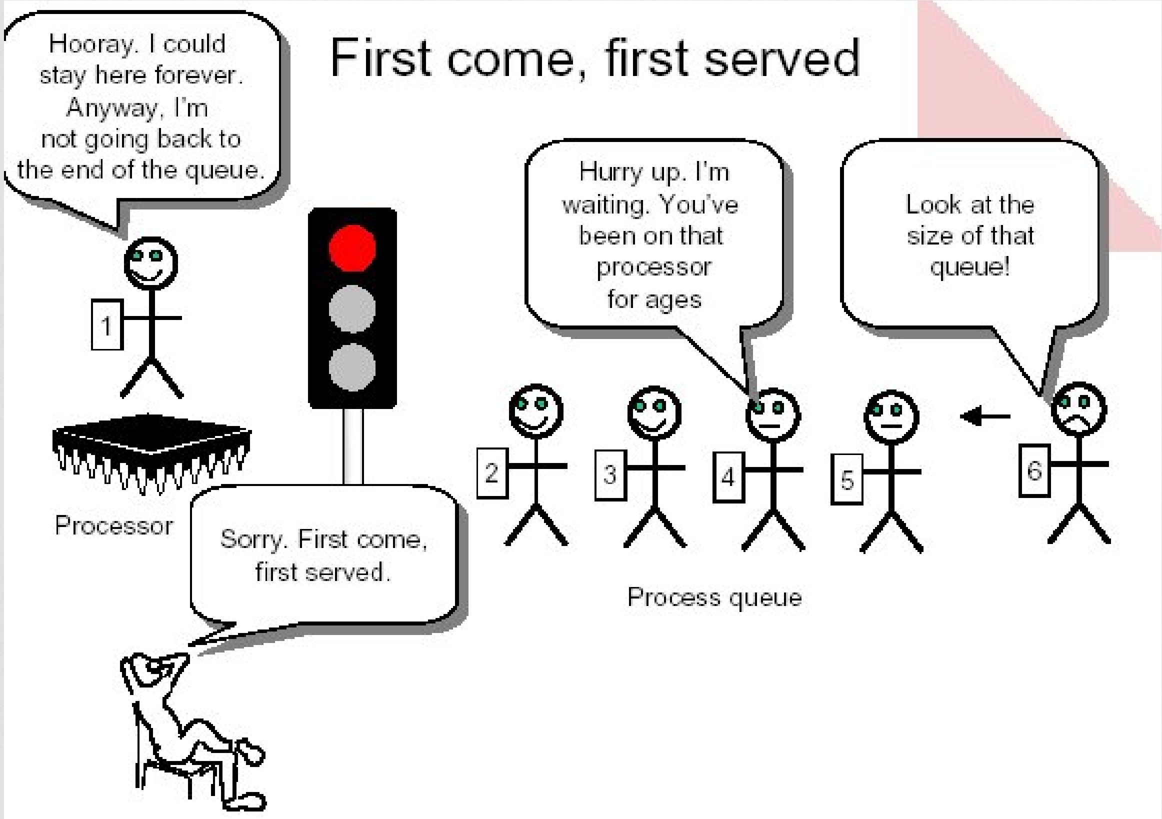
- **DISPATCHER:**
- The dispatcher is the module that gives control of the CPU to the process selected by the short-term scheduler.
- This function involves the following:
- Switching context
- Switching to user mode
- Jumping to the proper location in the user program to restart that program

TYPES OF SCHEDULING ALGORITHMS

- **FIRST COME, FIRST SERVED**
- **SHORTEST JOB FIRST (Pre-emptive & Non Pre-emptive)**
- **Round-robin scheduling**
- **Fixed-priority pre-emptive scheduling**
- **Multilevel Queue Scheduling**

FIRST-COME-FIRST-SERVED (FCFS) SCHEDULING

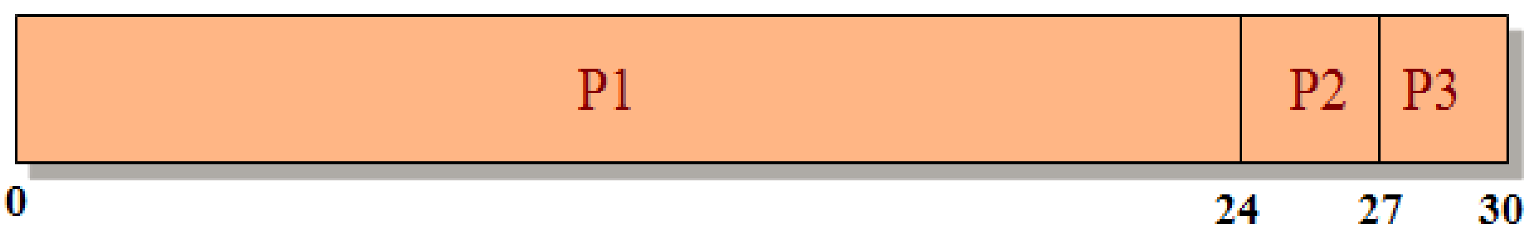
- Processes get the CPU in the order they request it and run until they release it
- Ready processes form a FIFO queue



FCFS

Process	Burst time (milli)
P1	24
P2	3
P3	3

If they arrive in the order P1, P2, P3, we get:



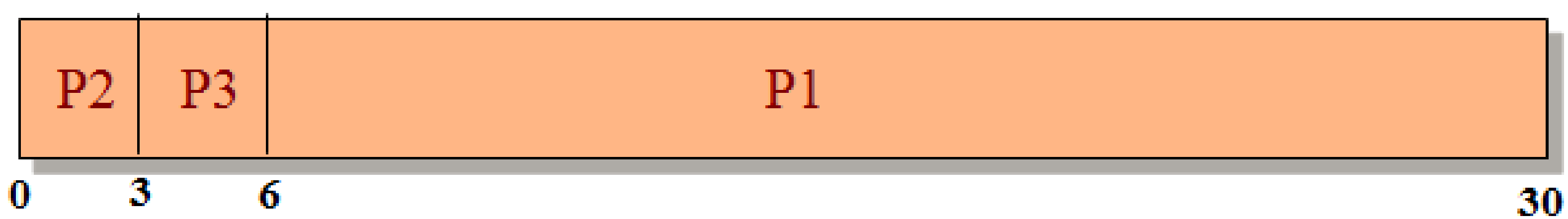
Average waiting time: (0+24+27) / 3 = 17

Gant Charts & Waiting Time for FCFS

Process	Burst time (milli)
P1	24
P2	3
P3	3

If they arrive in the order P1, P2, P3, average waiting time 17

What if they arrive in the order P2, P3, P1?



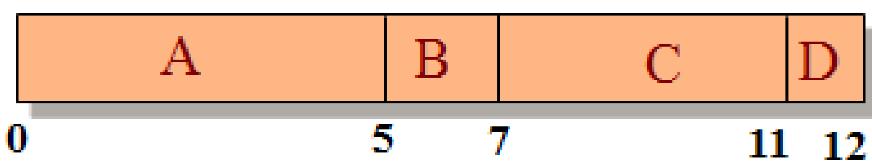
Average waiting time: (0+3+6) / 3 = 3

(NON PREEMPTIVE) SHORTEST JOB FIRST (SJF) SCHEDULING

- The CPU is assigned to the process that has the smallest next CPU burst
- In some cases, this quantity is known or can be approximated

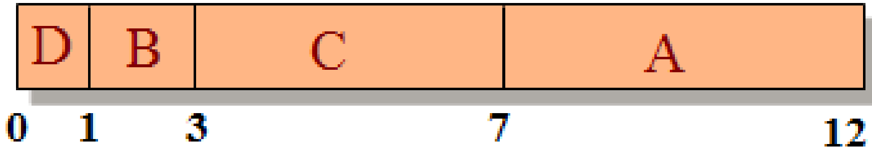
Process	Burst time (milli)
A	5
B	2
C	4
D	1

FCFS schedule



Average waiting time: **8.75**

SJF schedule



Average waiting time: **5.75**

APPROXIMATING NEXT CPU-BURST DURATION

Can be done by using the length of previous CPU bursts

- t_n = ***actual*** length of n^{th} CPU burst
- T_{n+1} = ***predicted*** value for the next CPU burst
- for $\mathbf{0} \le \alpha \le \mathbf{1}$ define the exponential averag

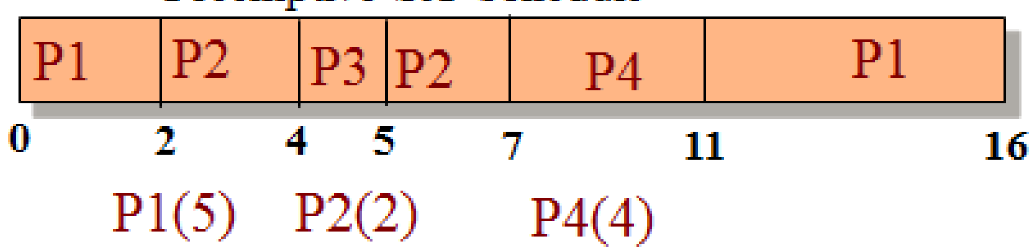
$$T_{n+1} = \underline{\alpha}\, t_n + (1 - \alpha)\, T_n$$

- α determines the relative weight of recent bursts

PREEMPTIVE SJF (SHORTEST REMAINING TIME FIRST)

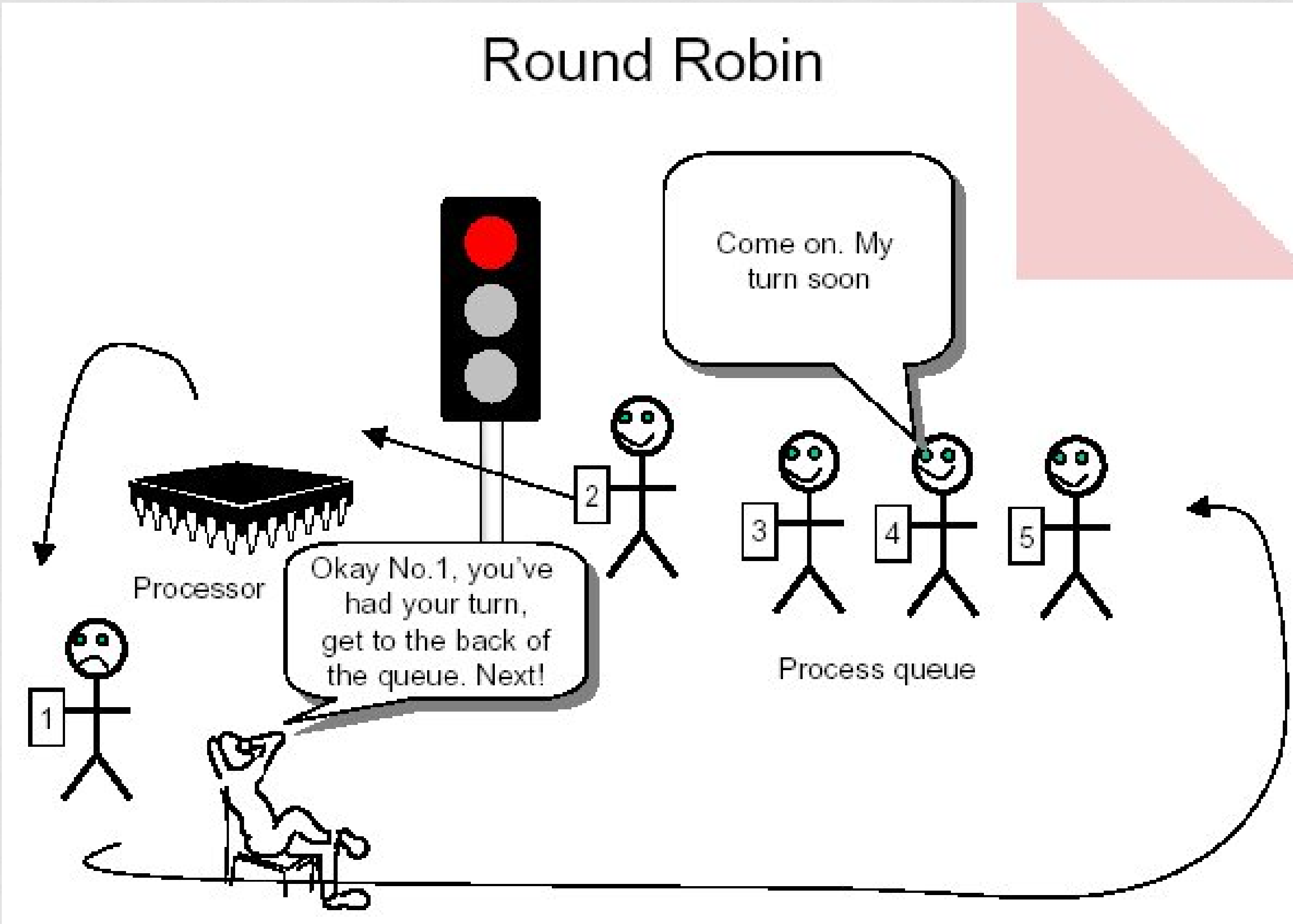
Process	Arrival time	Burst time
P1	0	7
P2	2	4
P3	4	1
P4	5	5

Preemptive SJF schedule

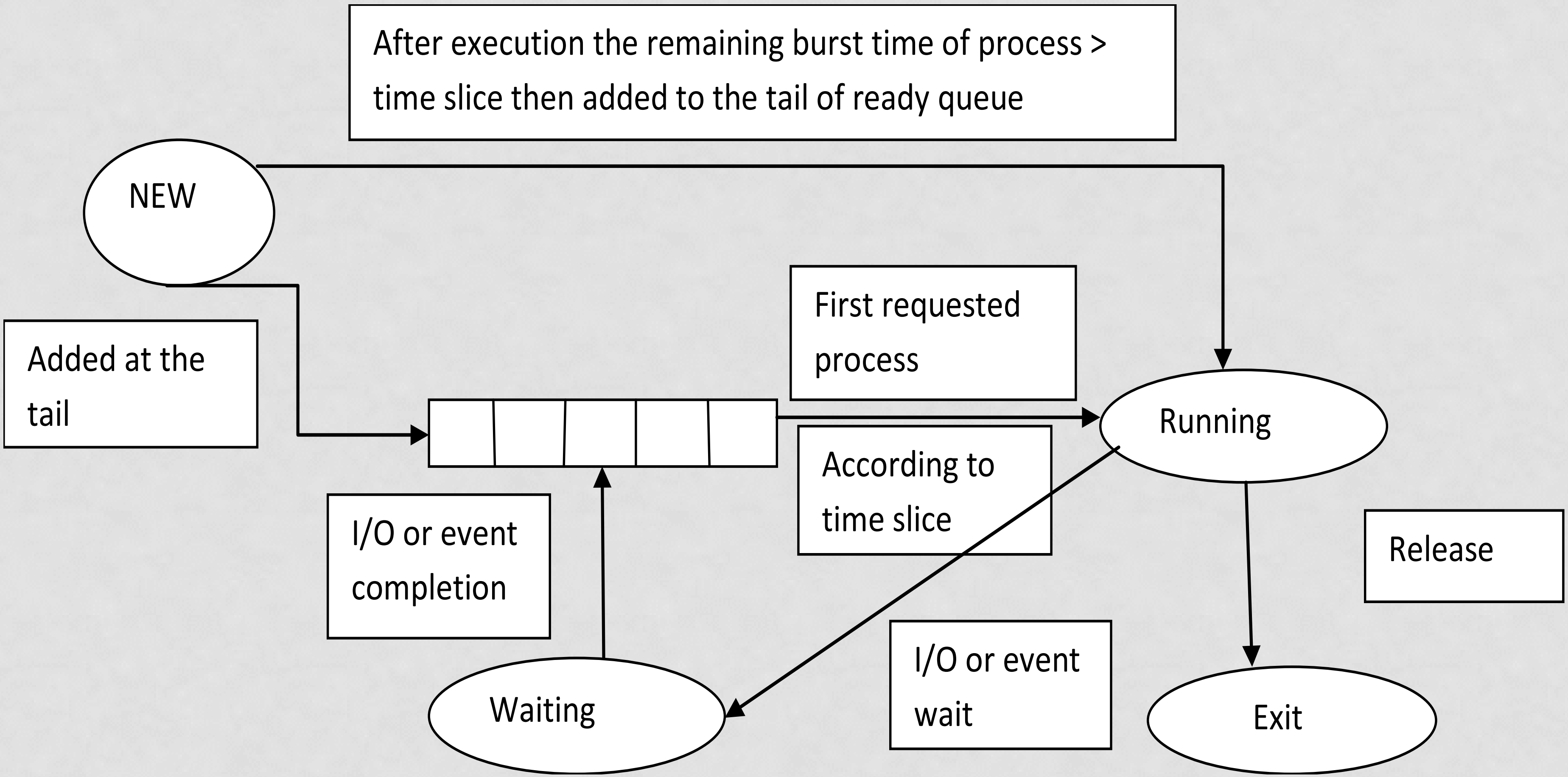


Average waiting time: $(9+1+0+2) / 4 = 3$

ROUND ROBIN SCHEDULING ALGORITHM (PICTORIAL REPRESENTATION)



ROUND ROBIN SCHEDULING ALGORITHM (FLOW DIAGRAM)



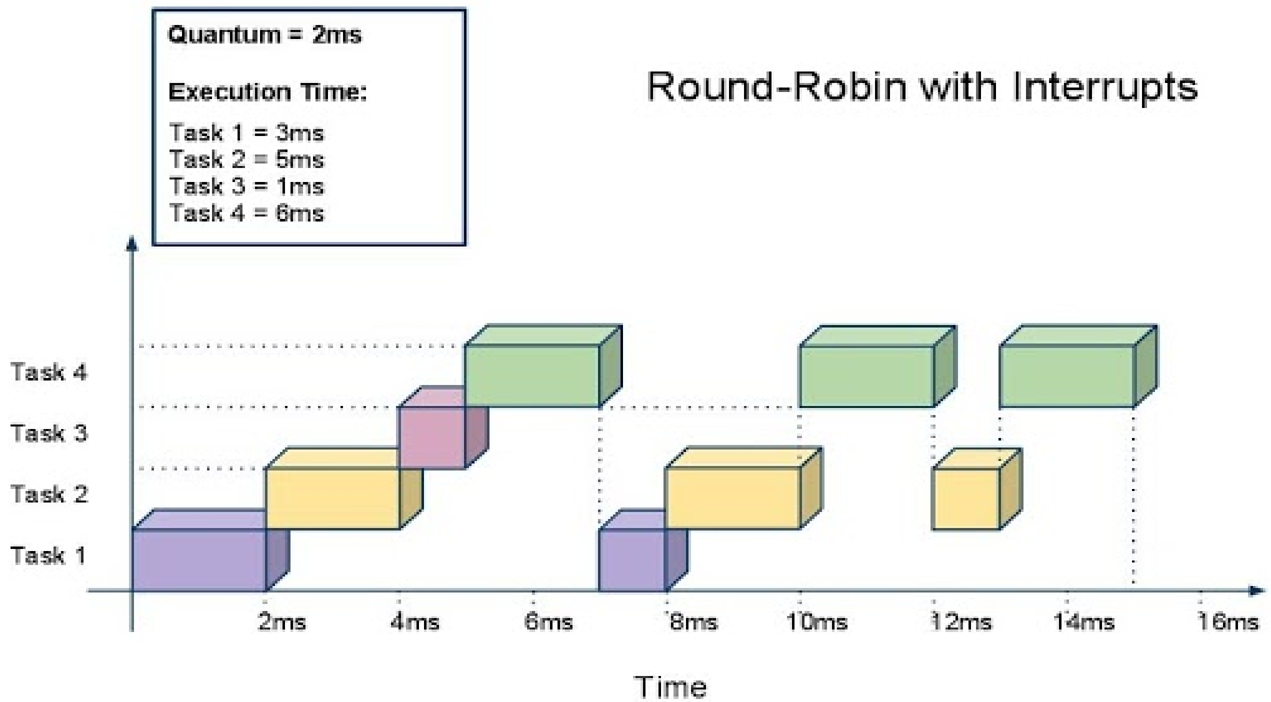
ROUND ROBIN SCHEDULING ALGORITHM

Process	Arrival Time	Execute Time
P0	0	5
P1	1	3
P2	2	8
P3	3	6

Quantum = 3

P0	P1	P2	P3	P0	P2	P3	P2

ROUND ROBIN SCHEDULING ALGORITHM



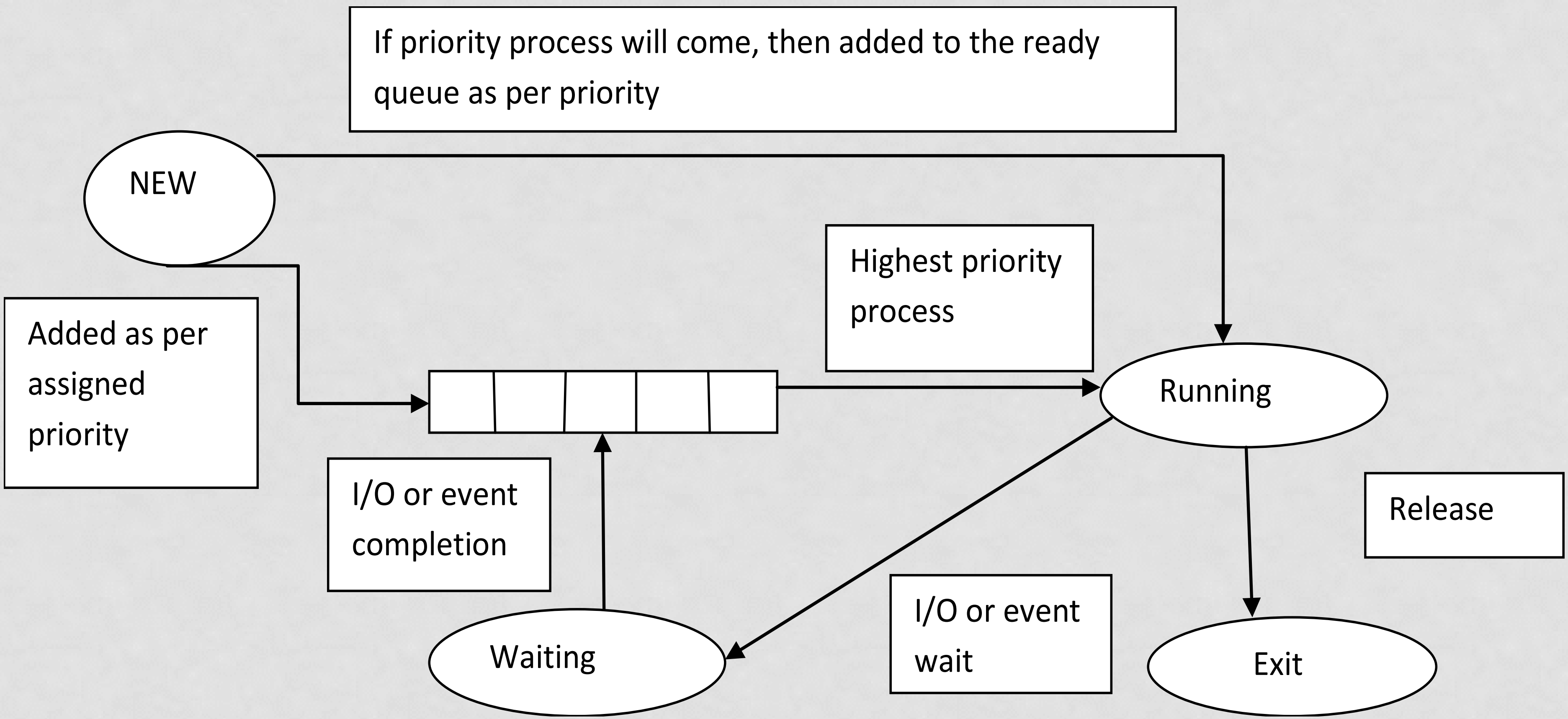
FIXED-PRIORITY PRE-EMPTIVE SCHEDULING

- Each process is assigned a priority. Process with highest priority is to be executed first and so on. Processes with same priority are executed on first come first serve basis. Priority can be decided based on memory requirements, time requirements or any other resource requirement.

FIXED-PRIORITY PRE-EMPTIVE SCHEDULING

- **Advantages**: The pre-emptive scheduler has a clock interrupt task that can provide the scheduler with options to switch after the task has had a given period—the time slice. This scheduler makes sure no task hogs the processor for any time longer than the time slice.
- **Disadvantages**: Overhead is not minimal, nor is it significant. Processes in lower-priority queues are selected only when all of the higher-priority queues are empty. Starvation of lower priority processes is possible with large amounts of high priority processes queuing for CPU time.

FIXED-PRIORITY PRE-EMPTIVE SCHEDULING (FLOW DIAGRAM)



FIXED-PRIORITY PRE-EMPTIVE SCHEDULING

Process	Arrival time	Execute Time	Priority	Service Time
P0	0	5	1	0
P1	1	3	2	3
P2	2	8	1	8
P3	3	6	3	16

P3	P1	P0	P2

0 6 9 14 22

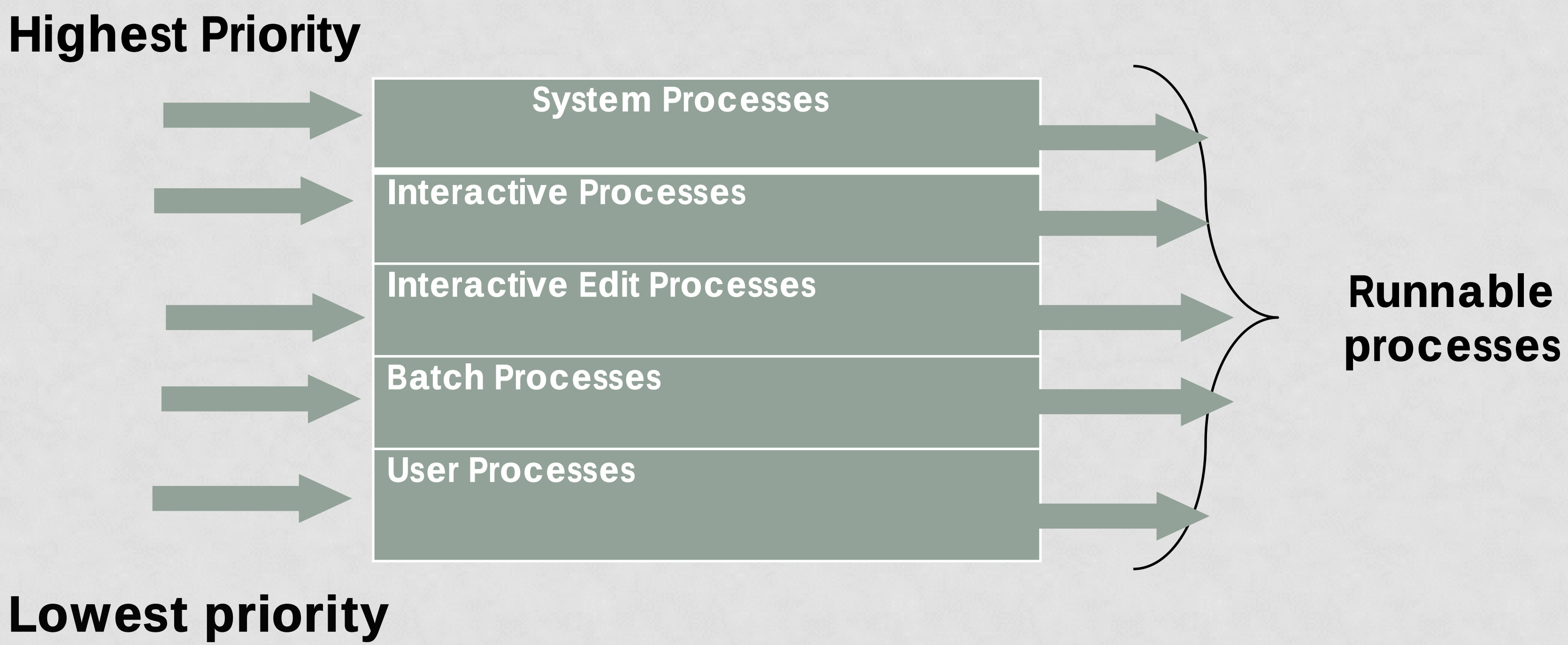
MULTILEVEL QUEUE SCHEDULING

- This is used for situations in which processes are easily divided into different groups. Multiple queues are maintained for processes. Each queue can have its own scheduling algorithms. Priorities are assigned to each queue.
- Ready queue is partitioned into separate queues: foreground (interactive) and background (batch)
- Each queue has its own scheduling algorithm: foreground – RR and background – FCFS

MULTILEVEL QUEUE SCHEDULING

- **Advantages**: As each type of process is in different queues it is easier to execute more important processes rather than lower priority ones without more overhead.
- **Disadvantages**: No process in the batch queue could run unless the queue for system processes and interactive processes were all empty. If an interactive process enters the ready queue while a batch process was running, the batch would be pre-empted. This would result in starvation.

MULTILEVEL QUEUE SCHEDULING



CONCLUSION & FUTURE WORK

The treatment of shortest process in SJF scheduling tends to result in increased waiting time for long processes. And the long process will never get served, though it produces minimum average waiting time and average turnaround time. It is recommended that any kind of simulation for any CPU scheduling algorithm has limited accuracy. The only way to evaluate a scheduling algorithm is to code it and to put it in the operating system, only then the proper working capability of the algorithm can be measured in real time systems.

COMPARISON & CONCLUSION

- Each of stated scheduling algorithms has its own features and characteristics. Table 1 indicates the difference and compares them.

Scheduling Algorithm	CPU Overhead	Throughput	Turnaround Time	Response Time
First In First Out	Low	Low	High	Low
Shortest Job First	Medium	High	Medium	Medium
Priority based scheduling	Medium	Low	High	High
Round-robin scheduling	High	Medium	Medium	High

REFERENCE

- http://en.wikipedia.org/wiki/Scheduling_(computing)

- http://www.just.edu.jo/~basel/os/os_slides/OS%20Chp6%20CPU%20Scheduling.pdf

- http://www.tutorialspoint.com/operating_system/os_process_scheduling_algorithms.htm

- http://www.sciencehq.com/computing-technology/multilevel-queue-scheduling.html

- http://www.tutorialspoint.com/operating_system/os_process_scheduling.htm
- http://www.gitam.edu/eresource/comp/gvr(os)/5.3.htm
- http://homepages.uel.ac.uk/u8902383/scheduling_policies.htm
- http://arxiv.org/ftp/arxiv/papers/1307/1307.4165.pdf

APPROXIMATE TIME TAKEN TO EXPLAIN THE PRESENTATION

Number of trials taken	Approx. Minutes consumed
1	20 mins
2	18 mins
3	15 mins
4	15mins
5	14mins
6	13 mins
7	13mins

Approximately total time consumed is 13 minutes