OPERATING SYSTEM

Process/CPU Scheduling Techniques

Scheduling examples

Scheduling Algorithms/Schemes/Policies

Basic categories

- Non-preemptive Mode: A process continues to execute until it terminates, blocks itself to wait for I/O or by requesting some operating system service.
- Preemptive Mode: The currently running process may be interrupted and moved to the Ready state by the operating system.

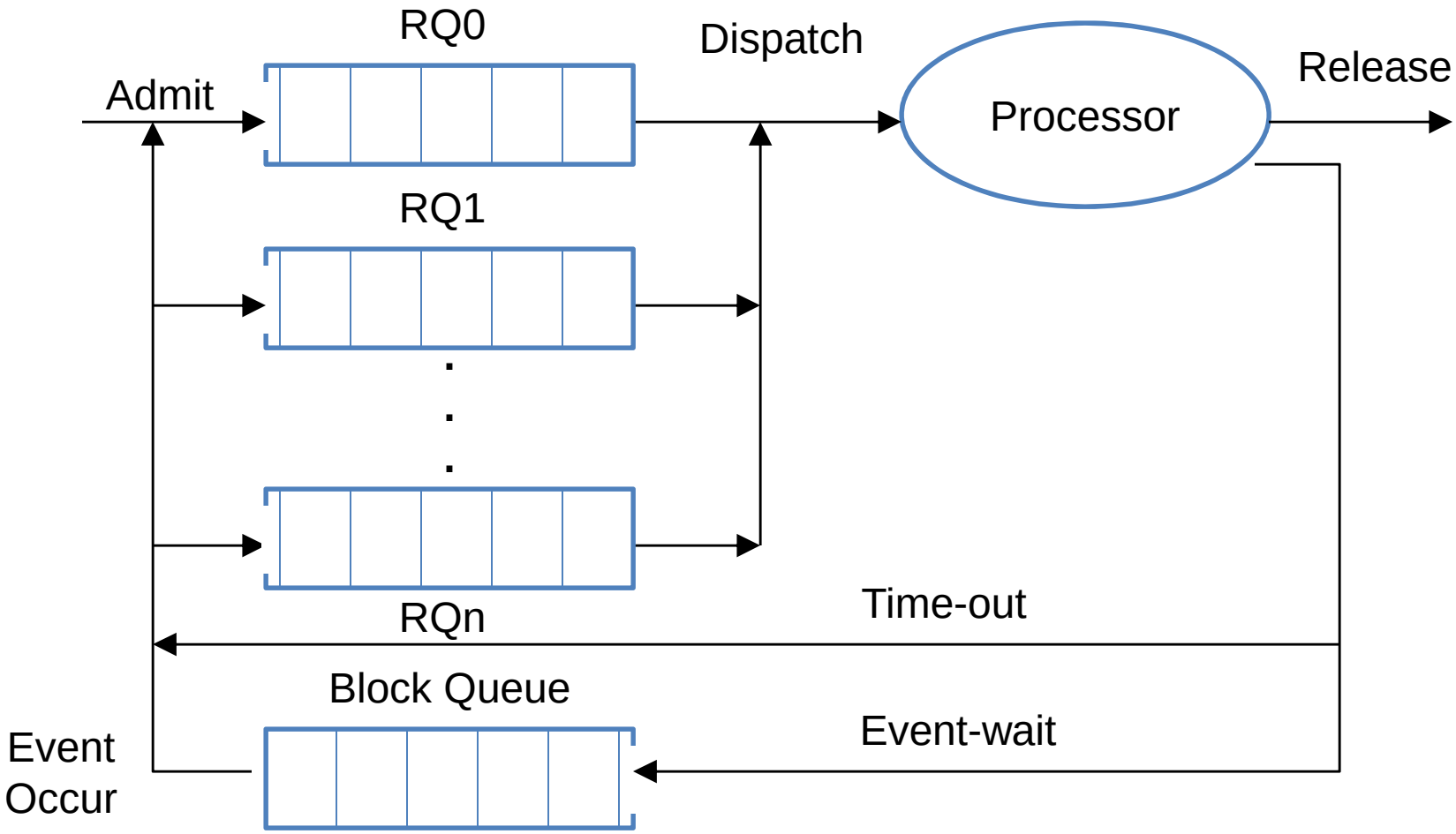
  The decision to preempt may be due to:
  - a new process arrives
  - an interrupt occurs that places a Blocked process in the Ready state
  - periodically on the basis of a clock interrupt.

Scheduling Algorithms/Schemes/Policies

(1) Pure Priority Scheduling Scheme

- Each process is assigned a priority.
- The scheduler always chooses a process of higher priority over one of the lower priority.
- Multiple “ready queues” with priorities RQ0 > RQ1 > … > RQn.
- Process is selected from the highest priority ready queue.
- Problem: Lower priority processes may suffer starvation, when there is a steady supply of higher-priority ready processes.
- Solution: Priority of a process can change with its age or execution history.

Scheduling Algorithms/Schemes/Policies



Priority Scheduling (Simplified Queuing Diagram)

Scheduling Algorithms/Schemes/Policies

(2) First-Come, First-Served Scheme

- First-Come, First-Served (FCFS) scheme is also called First-In, First-Out (FIFO) scheme.
- The dispatcher runs the processes at head of single ready queue, new processes come in at the end of the queue.
- Non-Preemptive technique
- Shorter processes have to wait a lot while a long process is executed by processor.
- FCFS tends to favor CPU-Bound processes over I/O-Bound processes.
- FCFS is easier and faster but may result inefficient use of processor and the I/O devices.

Scheduling Algorithms/Schemes/Policies

(2) First-Come, First-Served Scheme (con..)

- I/O-Bound Processes: Processes that perform lots of I/O operations. Each I/O is followed by a short CPU burst.
- CPU-Bound Processes: Processes that perform lots of computation and do little I/O. They tend to have long few CPU bursts.

Scheduling Algorithms/Schemes/Policies

(3) Round-Robin Scheme

- Like FCFS but preemptive
- A clock-interrupt is generated at periodic intervals.
- When the interrupt occurs, the currently running process is placed in the Ready queue
- Next process is selected on first-come, first-served basis.
- Also known as time-slicing or time-quantization, because each process is given a slice / quantum of time

Scheduling Algorithms/Schemes/Policies

(3) Round-Robin Scheme (con…)

Length or time quantum/slice: Short vs. Long

- If the time quantum is very short, then short processes will move through the system relatively quickly.
- However, there is processing overhead involved in handling the clock interrupt and performing the dispatching function.
- Standard time quantum is slightly greater than the time required for a typical interaction.

CPU-Bound vs. I/O-Bound Processes

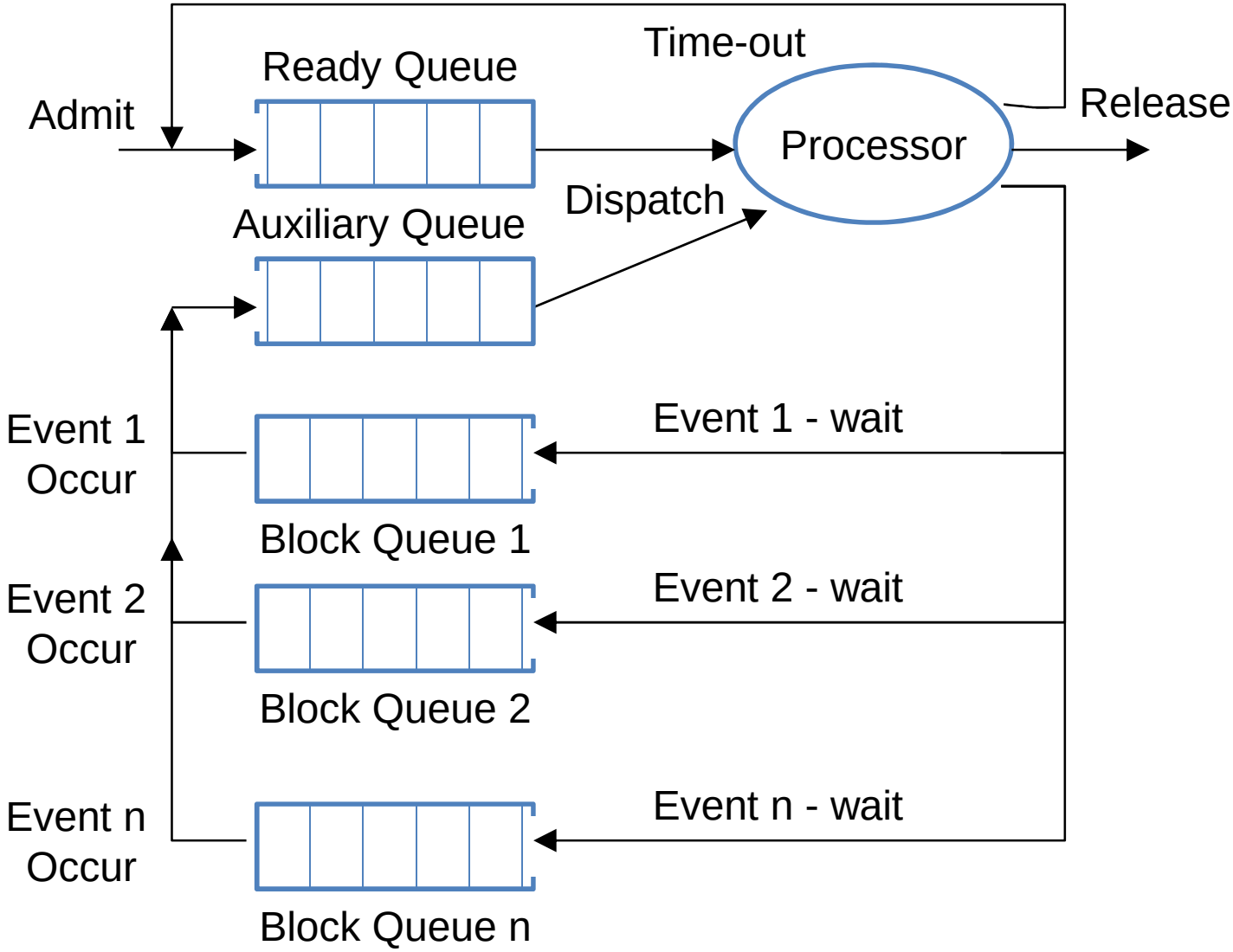
- CPU-bound processes tend to receive an unfair portion of processor time, which results in poor performance of I/O-bound processes.

Scheduling Algorithms/Schemes/Policies

(4) Virtual Round-Robin Scheme

- Similar to Round Robin
- When a running process times out, it is returned to the Ready queue.
- When a process is blocked for I/O, it joins an I/O queue.
- Auxiliary Queue based on FCFS is used to which processes are moved after being released from an I/O wait.
- Processes in the auxiliary queue get preference over those in the main Ready queue.
- Superior to the Round-Robin in terms of fairness.

Scheduling Algorithms/Schemes/Policies



Queuing diagram for Virtual Round Robin Scheduler

Scheduling Algorithms/Schemes/Policies

(5) Shortest Job First Scheme

- Shortest Job First (SJF) is a non-preemptive policy
- The process with the shortest expected processing time is selected next.
- The difficulty with SJF policy is the “need to know the estimate time required for processing each process”.
- Short processes will jump to the head of the queue past longer processes

Scheduling Algorithms/Schemes/Policies

(6) Highest Response Ratio Next (HRRN)

- The response ratio is given by

$$RR = (w+s)/s$$

where

w = time spent waiting for the processor

s = expected service time

- Non-Preemptive scheme
- When the current process completes or is blocked, choose the ready process with the greatest value of RR.
- This approach is attractive, because it accounts for the age of the process.
- Shorter jobs are favored (a small denominator yields a larger ratio), aging without service increases the ratio.
- As with the shortest process next, the expected time must be estimated before using the technique.

Scheduling Algorithms/Schemes/Policies

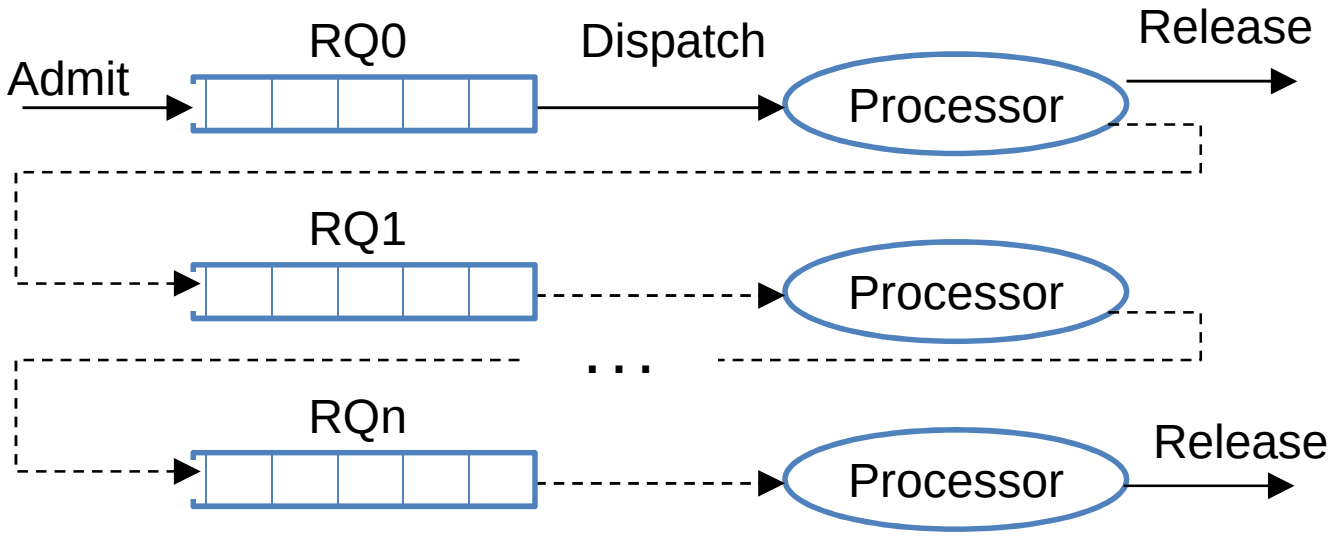
(7) Feedback Scheme

- This scheme is based on the time spent by a process in execution.
- Scheduling is done on a preemptive basis, and a dynamic priority mechanism is used.
- When a process first enters the system, it is placed in RQ0 (high priority queue).
- When it returns to the Ready state after its first execution, it is placed in RQ1 (lower priority queue).
- After each subsequent execution, it is demoted to the lower priority queue.
- A shorter process will complete quickly, without migrating very far down the hierarchy of Ready queues.
- A longer process will gradually drift downwards.

Scheduling Algorithms/Schemes/Policies

(7) Feedback Scheme

- Thus newer, shorter processes are favored over older longer processes.
- Last queue is treated with Round Robin technique, while other are treated as FCFS



* Dotted lines shows a time sequence rather than a static transitions.

Scheduling Algorithms/Schemes/Policies

(8) Fair-Share Scheduling (FFS) Scheme

- Preemptive technique
- The processes are divided into groups
- Each group is managed by certain scheme, mostly Round Robin Scheme is used
- The turn take into account:
  - The CPU usage of the group to which the process belongs.
  - The technique used to schedule a process of a group (e.g. Round Robin, HRRN etc.)
- Used when there is a multiuser system

Scheduling Algorithms Examples

Consider the following set of processes that arrive at time 0, with the length of CPU-burst time given in milliseconds:

Process	Burst-Time
P1	24
P2	3
P3	3

If we use a time quantum of 4 milliseconds, then find out the average waiting time with Round Robin (RR).

Solution:

	0 to 3	4 to 6	7 to 9	10 to 29
Round Robin (q = 4)	P1			P1
		P2		
			P4	

Average waiting time (RR, q = 4) = (6+4+7)/3 = 5.66

Scheduling Algorithms Examples

Consider the following set of processes, with the length of the CPU burst time given in milliseconds (excluding the transition time):

Process	Burst Time	Priority
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

The processes are assumed to have arrived in the order P1, P2, P3, P3, P4, P5, all at time 0.

(a) Draw four Gantt charts illustrating the execution of these processes using FCFS, SNF, a non-preemptive priority (a smaller priority number implies a higher priority) and RR (quantum =1) scheduling.

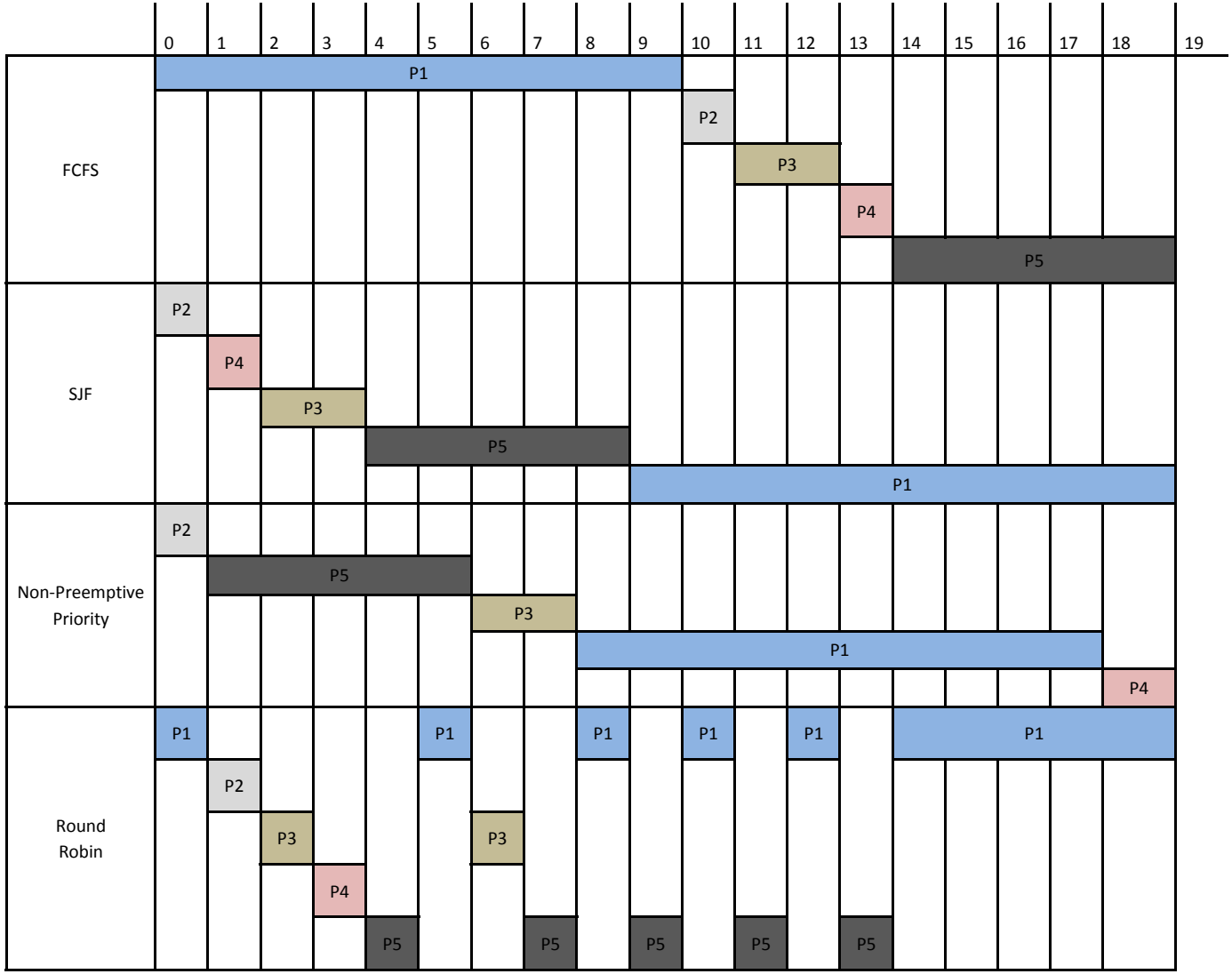
(b) What is the turnaround time of each process for each of the scheduling algorithm in part a?

(c) What is the waiting time of each process for each of the scheduling algorithm in part a?

(d) Which of the schedules in part a, results in the minimal average waiting time (over all processes)?

Scheduling Algorithms Examples

(a) Gantt Chart



Scheduling Algorithms Examples

(b) Turnaround of each process & (c) Waiting time of each process

		P1	P2	P3	P4	P5
FCFS	Turnaround	10	11	13	14	19
	Waiting	0	10	11	13	14
SJF	Turnaround	19	1	4	2	9
	Waiting	9	0	2	1	4
Non-preemptive Priority	Turnaround	18	1	9	19	6
	Waiting	9	0	6	18	1
Round Robin	Turnaround	19	2	7	4	14
	Waiting	9	1	5	3	9

(d) Minimal average waiting time

According to above example, SJF technique results minimal average time of overall processes.

Scheduling Algorithms Examples

Suppose that the following processes arrive for execution at the times indicated. Each process will run the listed amount of time. In answering the questions, use non-preemptive scheduling and base all decisions on the information you have at the time of the decision must be made.

Process	Arrival Time	Burst Time
P1	0.0	8
P2	0.4	4
P3	1.0	1

(a) What is the average turnaround time for these processes with the FCFS scheduling algorithm?

(b) What is the average turnaround time of these processes for SJF scheduling algorithm?

(c) The SJF algorithm is supposed to improve performance, but notice that we chose to run process P1 at time 0 because we did not know that two shorter processes would arrive soon. Compute what the average turnaround time will be, if the CPU is left idle for the first 1 unit and then SJF scheduling is used. Remember that processes P1 and P2 are waiting during this idle time, so their waiting time may increase. This algorithm could be known as future-knowledge scheduling.

Solution:

Average turnaround time (Non-preemptive FCFS) = (8+12+13)/3 = 11

Average turnaround time (Non-preemptive SJF) = (8+9+13)/3 = 10

Average turnaround time (Non-preemptive+1 idle unit) = (1+6+14)/3 = 7

Questions