1. State whether each of the following statements are true or false. Justify your answers.

(a) A process scheduling discipline is preemptive if the CPU cannot be forcibly removed from a process.

(b) When a new process is admitted in the system, the short-term scheduler must execute in order to keep the CPU busy.

(c) For a process, response time = turnaround time – waiting time.

(d) Partitioned multi-processor scheduling suffers from migration overheads due to data in private core-specific caches.

1a) A process scheduling discipline is preemptive if the CPU cannot be forcibly removed from a process.

➔False.

Justification: Preemptive means the CPU can be removed from the process <u>at any time</u>.

1b) When a new process is admitted in the system, the short-term scheduler must execute in order to keep the CPU busy.

➔False.

Justification: To keep the CPU busy, the short-term scheduler **must** execute only when a running process terminates or blocks. When a new process is admitted, it **may** be required to execute to keep the CPU busy only if there is no other running process in the system.

1c) For a process, response time = turnaround time – waiting time.

➔False.

Justification: "turnaround time – waiting time" is the time spent by a process in the "waiting" and "running" states combined. Whereas "response time" is the time until the first response is produced. So they are not necessarily related.

1d) Partitioned multi-processor scheduling suffers from migration overheads due to data in private core-specific caches.

➔False.

Justification: Migration overheads occur in global scheduling when a process partially executes on one core and then migrates to another. In partitioned scheduling processes don’t migrate between cores. However, partitioned scheduling has the problem of unbalanced loading of the cores depending on the process-core mapping.

2. Consider the following set of processes, with the CPU burst time given in milliseconds:

Process	CPU Burst Time	Priority	Arrival Time (Order)
P_1	10	3	0 (1)
P_2	1	1	0 (2)
P_3	2	3	2 (1)
P_4	1	4	2 (2)
P_5	5	2	4 (1)

(a) Draw six Gantt charts illustrating the execution of these processes using

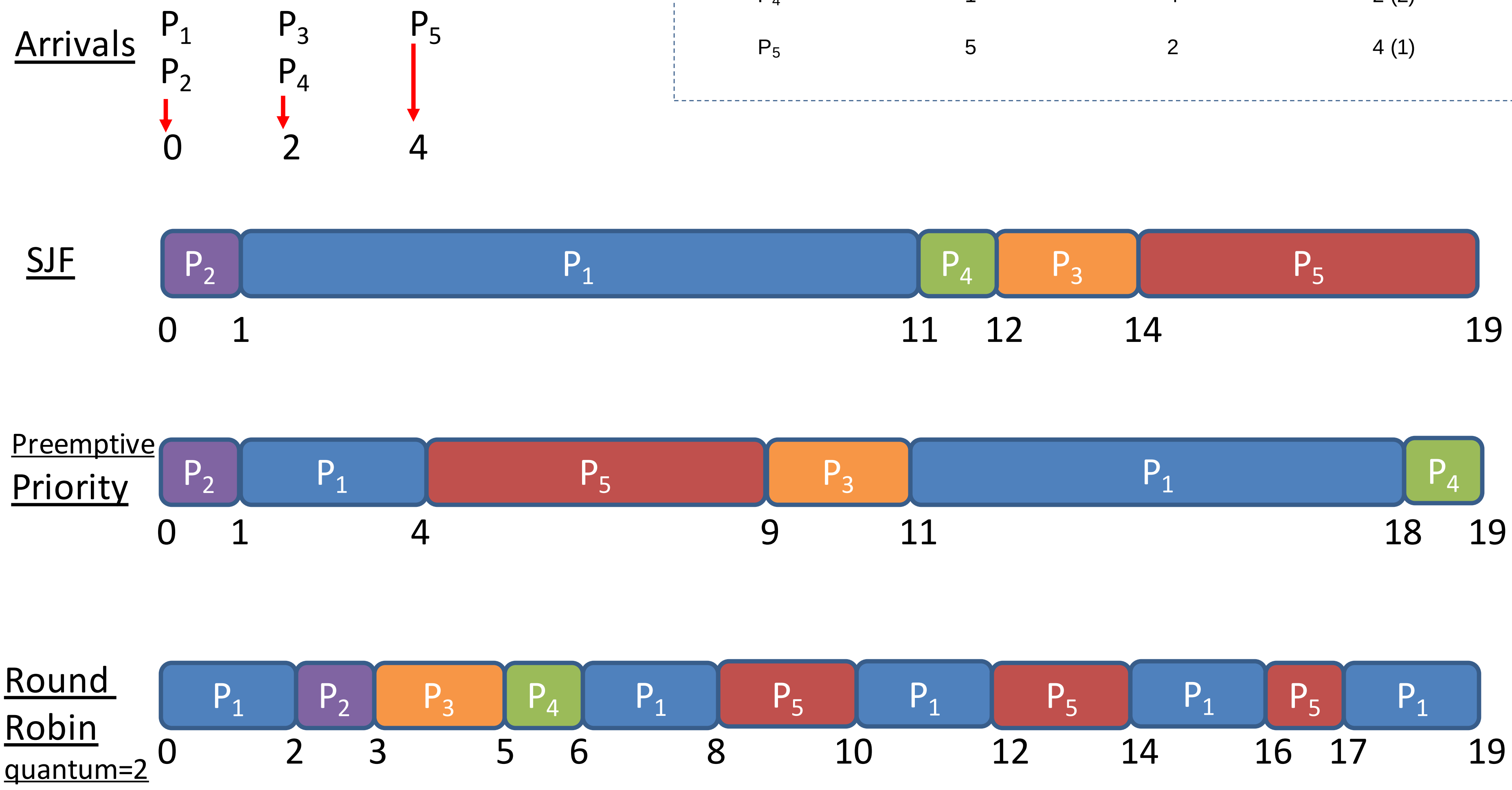
i. Shortest Job First (SJF), Preemptive Priority based (smaller priority number implies higher priority) and Round-Robin (quantum=2) uni-processor scheduling.

ii. First-Come First-Served (FCFS) partitioned with P1, P5 on core 1 and P2, P3 and P4 on core 2; Shortest Remaining Time First (SRTF) and Round-Robin (quantum=2) global scheduling on 2 cores.

Any volunteer?

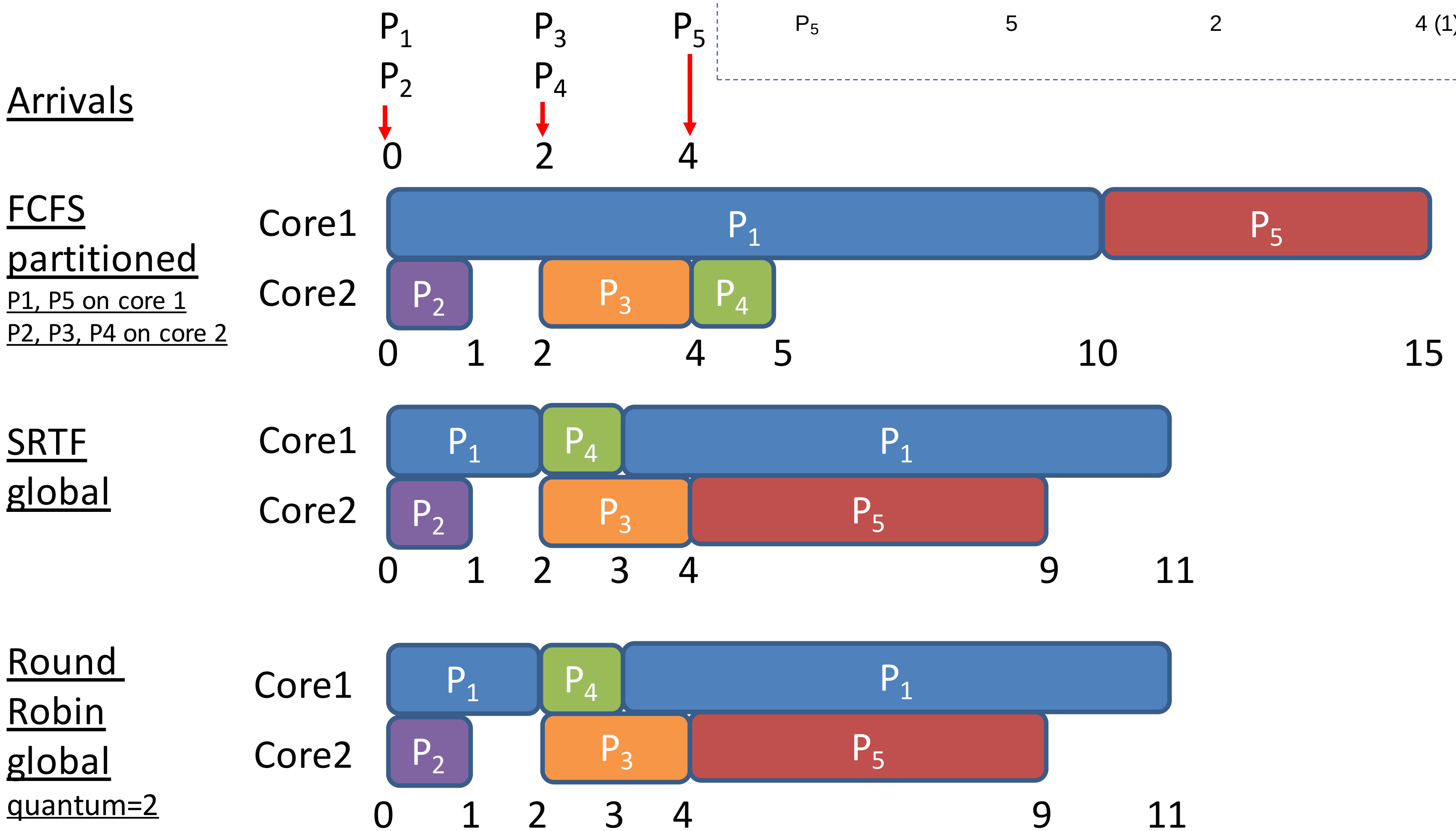
Gantt Charts (uni-processor)

Process	CPU Burst Time	Priority	Arrival Time (Order)
P_1	10	3	0 (1)
P_2	1	1	0 (2)
P_3	2	3	2 (1)
P_4	1	4	2 (2)
P_5	5	2	4 (1)



Gantt Charts (multi-processor)

Process	CPU Burst Time	Priority	Arrival Time (Order)
P_1	10	3	0 (1)
P_2	1	1	0 (2)
P_3	2	3	2 (1)
P_4	1	4	2 (2)
P_5	5	2	4 (1)

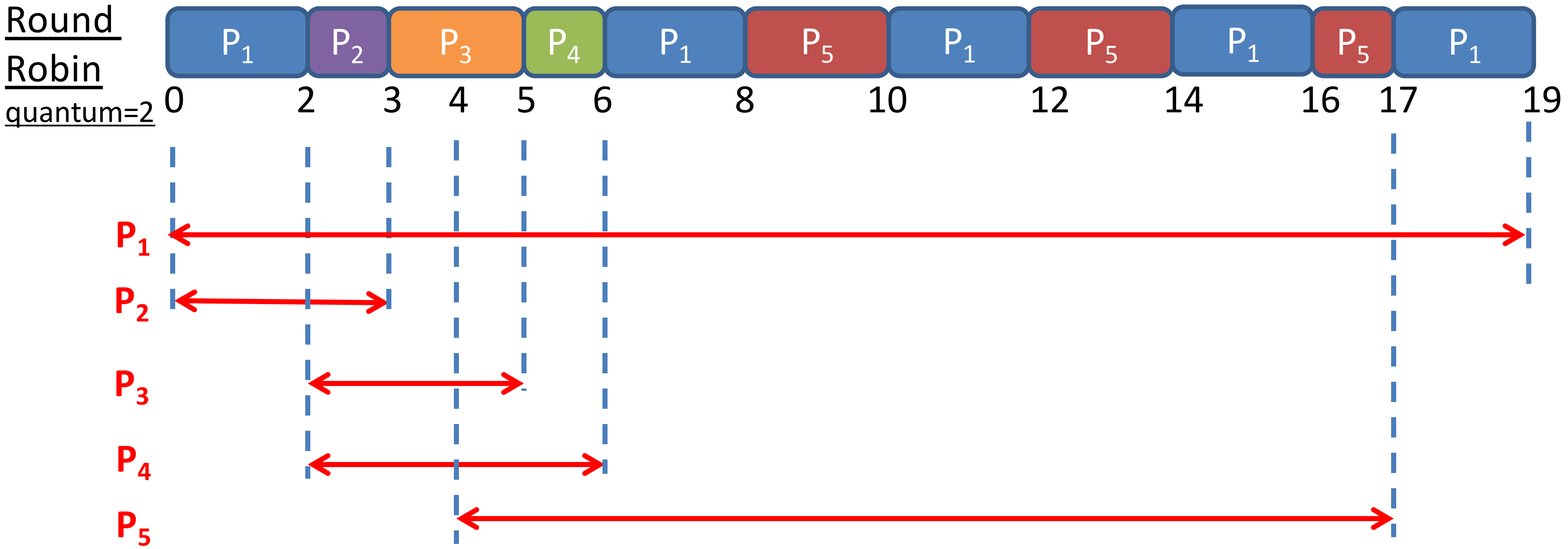


Turnaround Time

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

Process	Uni-processor				Multi-processor		
	SJF	Priority-Preemptive	RR (q=2)		FCFS	SRTF	RR(q=2)
P_1	11	18	19		10	11	11
P_2	1	1	3		1	1	1
P_3	12	9	3		2	2	2
P_4	10	17	4		3	1	1
P_5	15	5	13		11	5	5

Turnaround Time



Waiting Time

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

Waiting Time = Turnaround Time – CPU Burst Time

Process	Uni-processor				Multi-processor		
	SJF	Priority-Preemptive	RR (q=2)		FCFS	SRTF	RR(q=2)
P_1	1	8	9		0	1	1
P_2	0	0	2		0	0	0
P_3	10	7	1		0	0	0
P_4	9	16	3		2	0	0
P_5	10	0	8		6	0	0

Average Waiting Time

(d) Which of the schedulers in part (a) results in the minimal average waiting time (separately for uni- and multi-processors)?

→ The average waiting times are:

Uni-processor

SJF: 6, Priority: 6.2, RR(q=2): 4.6

Multi-processor

FCFS: 8/5, SRTF: 1/5, RR(q=2): 1/5

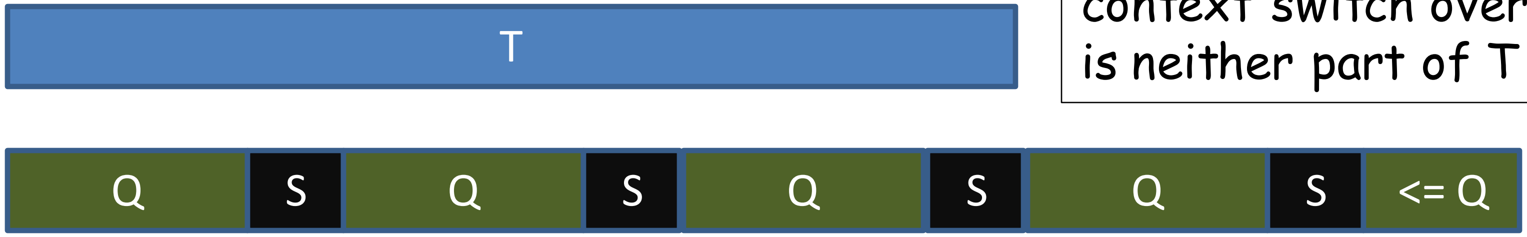
So RR(q=2) gives the minimal average waiting time under uni-processor and SRTF and RR(q=2) give the minimal average waiting time under multi-processor with 2 cores.

3. Measurements of a certain system have shown that the average process runs for time T before blocking on I/O. A process switch requires time S, which is effectively wasted (overhead). Define what is meant by CPU efficiency. For round robin scheduling with quantum Q, give a formula for the CPU efficiency for each of the following cases:

(a) $Q \rightarrow \infty$
(b) $Q > T$
(c) $S < Q < T$
(d) $Q = S$
(e) $Q \rightarrow 0$

CPU Efficiency

Sum of lengths of all the green boxes should be equal to the length of blue box. Black box is context switch overhead, which is neither part of T nor Q.



CPU efficiency= $\dfrac{\text{The useful CPU time}}{\text{Total CPU time}}$

$$= \frac{CPU\_Burst}{CPU\_Burst + Overhead\_Per\_Ctx \times \#Ctx\_Switch}$$

$$= \frac{T}{T + S \times \#Ctx\_Switch}$$

We must include the first context switch to start executing the process, because this is also overhead.

Count #Ctx_Switch

	#Ctx_Switch	**CPU Efficiency=** $\boxed{\dfrac{T}{T+S\times\#Ctx\_Switch}}$
(a) $Q \rightarrow \infty$	**1**	*T/(S + T)*
(b) $Q > T$	**1**	*T/(S + T)*
(c) $S < Q < T$	**Ceil(T/Q)**	*T/(T + S*ceil(T/Q))*
(d) $Q = S$	**Ceil(T/S)**	*T/(T + S*ceil(T/S))* *[approx. 0.5]*
(e) $Q \rightarrow 0$	**Inf** ∞	*0*