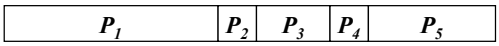
5.Scheduling della CPU. Esercizi

• (5.1)

Processo	Durata	priorità
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2

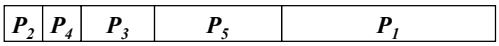


FCFS

5.Scheduling della CPU. Esercizi

• (5.1)

Processo	Durata	priorità
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2



SJF

5.Scheduling della CPU. Esercizi

• (5.1)

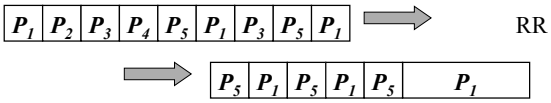
Processo	Durata	priorità
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2

Prior.

5.Scheduling della CPU. Esercizi

• (5.1)

Processo	Durata	priorità
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2



RR

5.Scheduling della CPU. Esercizi

- (5.1) Calcolare il tempo di turnaround per ciascun processo e per ciascun algoritmo di scheduling indicato (notate che basta fare la differenza tra il tempo in cui P lascia la CPU e quando è arrivato)
-

	FCFS	RR	SJF	Prior.
P_1	10	19	19	16
P_2	11	2	1	1
P_3	13	7	4	18
P_4	14	4	2	19
P_5	19	14	9	6

5.Scheduling della CPU. Esercizi

- (5.1) Calcolare il tempo di attesa per processo e per ciascun algoritmo di scheduling indicato. (notate che questo valore si può calcolare come il tempo di turnaround *meno* il burst time)
-

	FCFS	RR	SJF	Prior.
P_1	0	9	9	6
P_2	10	1	0	0
P_3	11	5	2	16
P_4	13	3	1	18
P_5	14	9	4	1
	9.6	5.4	**3.2**	8.2

5.Scheduling della CPU. Esercizi

- (5.2) Si supponga che i seguenti processi arrivino in esecuzione al tempo indicato e che consumeranno la quantità di tempo indicata. Si supponga uno scheduling non pre-emptive e di decidere sulla base delle informazioni disponibili al momento in cui le decisioni vanno prese.

Processo	t. di arrivo	burst time
P_1	0.0	8
P_2	0.4	4
P_3	1.0	1

5.Scheduling della CPU. Esercizi

1. (5.2) Calcolare il turnaround medio dei processi usando gli algoritmi di scheduling FCFS e SJF:
   - FCFS: 10.53 (attenzione, 11 è sbagliato...)
   - SJF: 9.53 (occhio! P_3 passa prima di P_2)
2. quale è il turnaround medio se si lascia inattiva la CPU per la prima unità di tempo e poi si usa SJF? (l'idea è di non assegnare la CPU fino a che non sono presenti tutti i processi per prendere la decisione migliore: Future Knowledge Scheduling)
   - FKS: 6.86

5.Scheduling della CPU. Esercizi

- (5.3) Dire se esiste, e quale è, la relazione fra le seguenti coppie di algoritmi di scheduling:
  - priorità e SJF
  - code multiple con retroazione e FCFS
  - priorità e FCFS
  - RR e SJF

5.Scheduling della CPU. Esercizi

- (5.3) *priorità e SJF:*
  - Il job più corto ha la priorità più alta
- (5.3) *code multiple con retroazione e FCFS:*
  - di solito, la coda con priorità peggiore è gestita FCFS
- (5.3) *priorità e FCFS:*
  - FCFS assegna la priorità più alta al job che esiste da più tempo (e quindi è arrivato per primo in coda...)
- (5.3) *RR e SJF:*
  - nessuna relazione

5.Scheduling della CPU. Esercizi

- (5.4) Si consideri un algoritmo di scheduling a breve termine che favorisce i processi che hanno usato poco la CPU di recente. Perché questo algoritmo favorisce i processi I/O bound, ma non provoca starvation nei processi CPU bound?
- l'algoritmo favorirà i processi I/O bound perchè questi richiedono relativamente poca CPU. Per questa ragione, la CPU sarà spesso libera per essere utilizzata dai processi CPU bound.

5.Scheduling della CPU. Esercizi

- (5.5) Spiegate le differenze tra i seguenti algoritmi di scheduling rispetto al livello di discriminazione in favore (o a sfavore) dei processi brevi:
  - FCFS
  - RR
  - code multiple con retroazione

5.Scheduling della CPU. Esercizi

- (5.5) *FCFS:*
  - è sfavorevole ai job corti perchè qualsiasi job corto che arrivi dopo job lunghi dovrà aspettare molto tempo per avere la CPU
- (5.5) *RR:*
  - tratta tutti i job allo stesso modo. Quindi i job corti termineranno prima di quelli lunghi, perchè richiedono meno CPU
- (5.5) *code multiple con retroazione:*
  - funzionano in modo simile al RR, favorendo di fatto i job corti