LABORATORIO DI PROGRAMMAZIONE 2
Corso di laurea in matematica

scheduling di processi e threads

Marco Lapegna
Dipartimento di Matematica e Applicazioni
Universita' degli Studi di Napoli Federico II

wpage.unina.it/lapegna

Concetti fondamentali

Il massimo impiego della CPU è ottenuto con la multiprogrammazione.

Presenza in memoria di numerosi processi in attesa di essere eseguiti

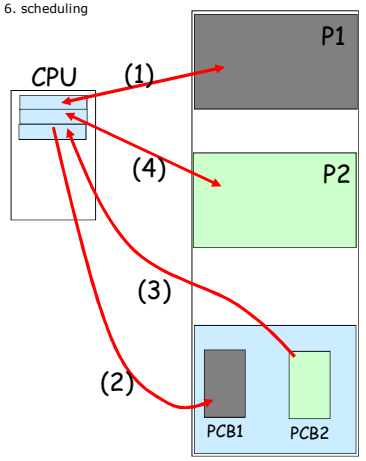
Chi decide quale processo deve accedere alla CPU?

SCHEDULER

dispatcher

- Il modulo *dispatcher* passa il controllo della CPU al processo selezionato dallo scheduler a breve termine;
- Il dispatcher effettua il cambio di contesto (context switch)
- *Latenza di dispatch* — è il tempo impiegato dal dispatcher per sospendere un processo e avviare l' esecuzione di un nuovo processo

Context switch



1 la cpu lavora in mod. utente. I registri della CPU contengono informazioni relativi a P1 (program counter, puntatori allo stack, PID, stato,…)

2 la cpu passa in modalita' sistema e salva il contenuto dei registri nel PCB1

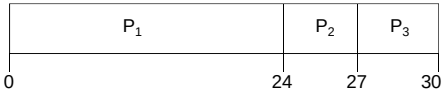
3 la cpu carica nei registri il contenuto del PCB2 precedentemente salvato e passa in mod. utente

4 la cpu ha il controllo di P2

Scheduling First–Come–First–Served (FCFS)

Processo	Tempo di burst
P_1	24
P_2	3
P_3	3

- I processi arrivano al sistema nell'ordine: P_1 , P_2 , P_3.
  Il diagramma di Gantt per lo scheduling **FCFS** è:



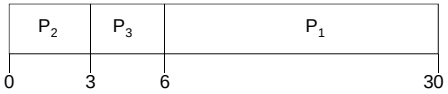
- Tempi di attesa: $P_1 = 0$; $P_2 = 24$; $P_3 = 27$.
- Tempi di completamento: $P_1 = 24$; $P_2 = 27$; $P_3 = 30$.
- Tempo medio di attesa = (0 + 24 + 27)/3 = 17.
- Tempo medio di completamento = (24 + 27 + 30)/3 = 27.

Scheduling FCFS

Se l'ordine di arrivo è

P_2 , P_3 , P_1,

il diagramma di Gantt risulta…



- Tempi di attesa: $P_1 = 6$; $P_2 = 0$; $P_3 = 3$.
- Tempi di completamento: $P_1 = 30$; $P_2 = 3$; $P_3 = 6$.
- Tempo medio di attesa = (6 + 0 + 3)/3 = 3.
- Tempo medio di completamento = (30 + 3 + 6)/3 = 13.

- Non si verifica l'effetto, per cui processi di breve durata devono attendere che un processo molto lungo liberi la CPU.

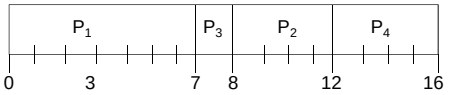
Scheduling Shortest–Job–First (SJF)

- Si associa a ciascun processo la lunghezza del suo burst di CPU successivo. Si opera lo scheduling in base al tempo rimanente dei processi.
- Due schemi:
  - **non–preemptive** — una volta che la CPU è stata allocata al processo, non gli può essere prelazionata fino al termine del CPU burst corrente;
  - **preemptive** — se arriva un nuovo processo con burst di CPU minore del tempo rimasto per il processo corrente, il nuovo processo prelaziona la CPU. Questo schema è noto come ***Shortest–Remaining–Time–First*** (**SRTF**).
- **SJF** è *ottimale* — rende minimo il tempo medio di attesa per un dato insieme di processi.

Scheduling SJF senza prelazione

Processo	Tempo di arrivo	Tempo di burst
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (senza prelazione):



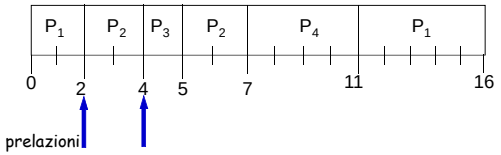
- Tempo medio di attesa = (0 + 6 + 3 + 7)/4 = 4.
- Tempo medio di completamento = (7 + 10 + 4 + 11)/4 = 8.

Scheduling SJF con prelazione

(shortest remaining time first)

Processo	Tempo di arrivo	Tempo di burst
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (con prelazione):



- Tempo medio di attesa = (9 + 1 + 0 +2)/4 = 3.
- Tempo medio di completamento = (16 + 5 + 1 +6)/4 = 7.

Scheduling Shortest–Job–First (SJF)

Lo scheduling SJF e' ottimale,
in quanto minimizza i tempi medi di attesa

Tale stima e' calcolata sulla previsione dei tempi di utilizzo della CPU che non sono informazioni presenti nel PCB

Come stimare i tempi di uso della CPU

Previsione sull'uso della CPU

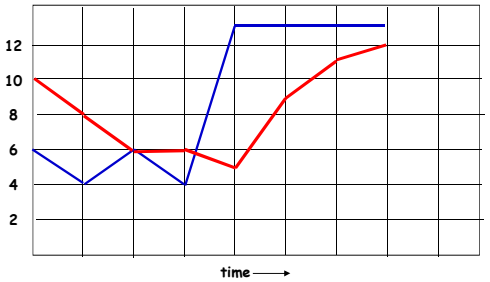
- Può essere stimato utilizzando la lunghezza dei burst di CPU precedenti, impiegando una media esponenziale.

1. t_n = lunghezza dell'$n-$esimo CPU burst
2. τ_{n+1} = valore stimato del prossimo CPU burst
3. α, $0 \le \alpha \le 1$
4. Si definisca :

$$\tau_{n+1} = \alpha t_n + (1-\alpha)\tau_n.$$

esempio

$$\tau_{n+1} = \alpha t_n + (1-\alpha)\tau_n \qquad \alpha = 1/2$$



Effettivi	6	4	6	4	13	13	13	13
Stimati	10	8	6	6	5	9	11	12

Esempi di media esponenziale

- $\alpha = 0$
  - $\tau_{n+1} = \tau_n$
  - La storia recente non è presa in considerazione.
- $\alpha = 1$
  - $\tau_{n+1} = t_n$
  - Viene considerato soltanto l'ultimo CPU burst.
- Espandendo la formula si ottiene:

$$\tau_{n+1} = \alpha\, t_n + (1-\alpha)\,\alpha\, t_{n-1} + \ldots + (1-\alpha)^j \alpha\, t_{n-j} + \ldots + (1-\alpha)^{n+1} \tau_0$$

- Poiché α e $(1-\alpha)$ sono entrambi minori o uguali ad 1, ciascun termine ha minor peso del suo predecessore.

Scheduling a priorità

- Un valore di priorità (intero) è associato a ciascun processo.
- La CPU viene allocata al processo con la priorità più alta
  - con prelazione
  - senza prelazione

- ESEMPIO
  - **SJF** è uno scheduling a priorità dove la priorità è rappresentata dal successivo tempo di burst.

Priorita'

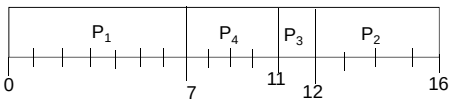
La priorita' e' un numero intero in un intervallo fissato
Es
- $0 \le p \le 20$
- $0 \le p \le 4096$

Ogni sistema ha il proprio modo di misurare la priorita'
➔ p=0 priorita' alta oppure
➔ p=0 priorita' bassa (es. Windows e Linux)

Esempio : priorita' senza prelazione

p=0 massima priorita'

Processo	priorita'	Tempo di burst
P_1	1	7
P_2	3	4
P_3	2	1
P_4	1	4



- Tempo medio di attesa = (0 + 12 + 11 +7)/4 = 7.5.
- Tempo medio di completamento = (7 + 16 + 12 +11)/4 = 11.5

Problemi:

- SJF:
  - Forte disparita' tra processi corti e processi lunghi
- Priorita':
  - **Possibilita' di Starvation** (posticipazione indefinito) — i processi a bassa priorità potrebbero non venir mai eseguiti.

Soluzione ≡ ***Aging*** (invecchiamento)
aumento graduale della priorità dei processi che si trovano in attesa nel sistema da lungo tempo.

Highest Response Ratio Next scheduling

- Scheduling senza prelazione a priorita' variabile
- E' un modo per realizzare l'aging
- La priorita' e' funzione di
  - Tempo di attesa
  - Tempo di burst
- Esempio

$$Priorita = \frac{T_{attesa} + T_{burst}}{T_{burst}}$$

Favorisce i processi che hanno gia' atteso molto

Favorisce i processi corti

p=1 minima priorita'

Esempio HRRN vs SJF

p=1 minima priorita'

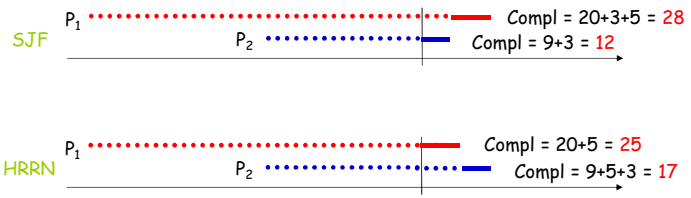
Processo	Tempo attesa	Tempo di burst
P_1	20	5
P_2	9	3

Processo	Priorita' HRRN
P_1	(20+5)/5=5
P_2	(9+3)/3 = 4

- SJF prima P_2 e poi P_1
- HRRN prima P_1 e poi P_2

Esempio HRRN vs SJF

Processo	Tempo attesa	Tempo di burst
P_1	20	5
P_2	9	3



Tempi medi di completamento
SJF = (28+12)/2 = 20 HRRN = (25+17)/2=21

problema

HRRN e' stato introdotto per superare l'eccessiva disparita' di trattamento tra processi corti e processi lunghi operata da SJF

Come misurare questa qualita' di HRRN?

La deviazione standard (o scarto quadratico medio) è

una misura della dispersione di un insieme di numeri $x_1, \ldots, x_n$

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2} \quad con \quad \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

media

σ piccolo ➔ dati "vicini" tra loro
σ grande ➔ dati "lontani" tra loro

Esempio HRRN vs SJF

- SJF

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2} = \sqrt{\frac{1}{2}(64+64)} = 8$$

- HRRN

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i - \bar{x}\right)^2} = \sqrt{\frac{1}{2}(16+16)} = 4$$

La politica HRRN e' piu' equa
(i tempi di completamento sono piu' omogenei)

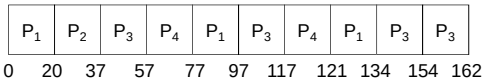
Scheduling Round Robin (RR)

- Scheduling con prelazione progettato appositamente per sistemi time sharing
- A ciascun processo viene allocata una piccola unità di tempo di CPU (*quanto di tempo* o *timeslice*), generalmente 10–100 millisecondi. Trascorso il quanto di tempo, il processo è forzato a rilasciare la CPU e accodato alla ready queue.
- Se ci sono n processi nella ready queue ed il quanto di tempo è q, ciascun processo occupa $1/n$ del tempo di CPU in frazioni di, al più, q unità di tempo.
- Nessun processo attende per più di $(n-1)\times q$ unità di tempo.

Scheduling RR con quanto = 20

Processo	Tempo di burst
P_1	53
P_2	17
P_3	68
P_4	24

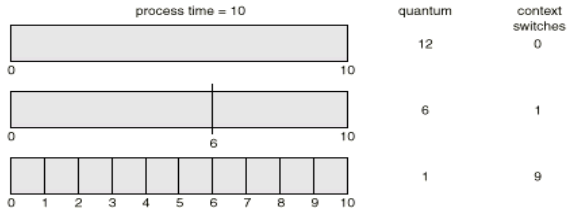
- Il diagramma di Gantt è:



- In genere si ha un tempo medio di attesa maggiore rispetto a **SJF**, tuttavia si ha un miglior tempo di completamento per i processi lunghi.

Quanto di tempo

- La grandezza del quanto di tempo determina il tempo di risposta delle richieste interattive
- Quanto di tempo grande
  - Processi in esecuzione per molto tempo
  - Di fatto diventa FCFS
- Quanto di tempo piccolo
  - Il sistema impiega piu' tempo nel cambio di contesto che nell'esecuzione dei programmi



riepilogo

- Algoritmo First Come First Served (FCFS) : i processi vengono eseguiti nell'ordine di arrivo
- Algoritmo Shortest Job First (SJF) : i processi vengono eseguiti da quello con il minor al maggior tempo di esecuzione rimanente
- Algoritmo con priorita' : i processi vengono eseguiti secodo un ordine definito da una priorita'
- Algoritmo Round Robin (RR) : viene assegnato un quanto di tempo ad ogni processo, scaduto il quale la cpu viene assegnata al successivo processo

Esercizio 1:

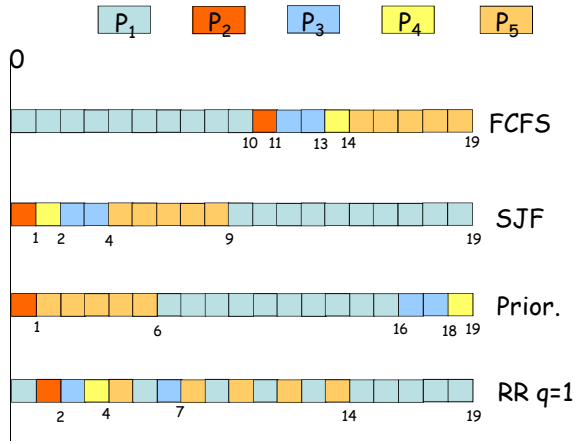
Siano i seguenti processi nella ready queue, arrivati nell'ordine specificato e tutti presenti ad un dato istante 0

Proc.	Esecuz.	prior.
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2

si considerino i seguenti algoritmi di scheduling:
- FCFS
- SJF senza prelazione
- priorita' senza prelazione (p=0 pr. Max)
- Round robin con q=1

1. Si disegnino i 4 diagrammi di Gantt
2. Calcolare i tempi di completamento nei 4 casi
3. Calcolare il tempo medio di completamento nei 4 casi
4. Calcolare la deviazione standard nei casi SJF e RR

Diagrammi di Gantt



Tempi di completamento e medi

- FCFS
  - $P_1 \rightarrow 10$ $P_2 \rightarrow 11$ $P_3 \rightarrow 13$ $P_4 \rightarrow 14$ $P_5 \rightarrow 19$
  - Tm = 67/5 = 13.4
- SJF
  - $P_1 \rightarrow 19$ $P_2 \rightarrow 1$ $P_3 \rightarrow 4$ $P_4 \rightarrow 2$ $P_5 \rightarrow 9$
  - Tm = 35/5 = 7 s= 6.6
- Priorita'
  - $P_1 \rightarrow 16$ $P_2 \rightarrow 1$ $P_3 \rightarrow 18$ $P_4 \rightarrow 19$ $P_5 \rightarrow 6$
  - Tm = 60/5 = 12
- RR q=1
  - $P_1 \rightarrow 19$ $P_2 \rightarrow 2$ $P_3 \rightarrow 7$ $P_4 \rightarrow 4$ $P_5 \rightarrow 14$
  - Tm = 46/5 = 9.2 s= 6.3

Esercizio 2:

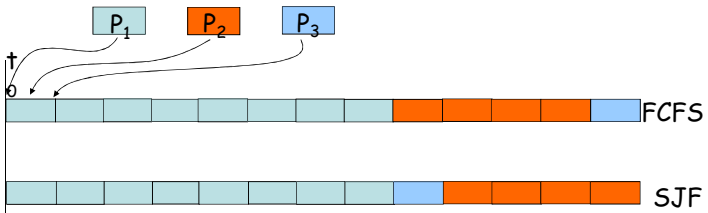
Siano i seguenti processi nella ready queue, arrivati nei tempi specificati

Proc.	arrivo	esecuz.
P_1	0.0	8
P_2	0.5	4
P_3	1.0	1

si considerino i seguenti algoritmi di scheduling:
- FCFS senza prelazione
- SJF senza prelazione

1. Si disegnino i diagrammi di Gantt
2. Calcolare i tempi di completamento nei 2 casi
3. Calcolare il tempo medio di completamento nei 2 casi

soluzione



- FCFS
  - $P_1 \rightarrow 8$ $P_2 \rightarrow 11.5$ $P_3 \rightarrow 12$
  - Tm = 31.5/3 = 10.5
- SJF
  - $P_1 \rightarrow 8$ $P_2 \rightarrow 12.5$ $P_3 \rightarrow 8$
  - Tm = 28.5/3 = 9.5

Esercizio 3:

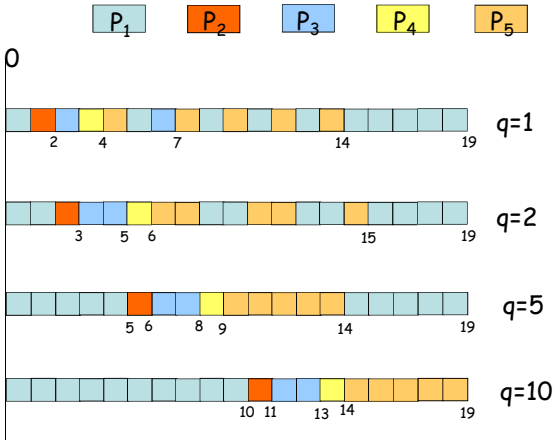
Siano i seguenti processi nella ready queue, arrivati nell'ordine specificato e tutti presenti ad un dato istante 0

Proc.	Esecuz.
P_1	10
P_2	1
P_3	2
P_4	1
P_5	5

si consideri l'algoritmo RR con i seguenti quanti di tempo:
- q=1
- q=2
- q=5
- q=10

1. Si disegnino i 4 diagrammi di Gantt
2. Calcolare i tempi di completamento nei 4 casi
3. Calcolare il tempo medio di completamento nei 4 casi

Diagrammi di Gantt



Tempi di completamento e medi

- q=1
  - $P_1 \rightarrow 19$ $P_2 \rightarrow 2$ $P_3 \rightarrow 7$ $P_4 \rightarrow 4$ $P_5 \rightarrow 14$
  - Tm = 46/5 = 9.2
- q=2
  - $P_1 \rightarrow 19$ $P_2 \rightarrow 3$ $P_3 \rightarrow 5$ $P_4 \rightarrow 6$ $P_5 \rightarrow 15$
  - Tm = 48/5 = 9.6
- q=5
  - $P_1 \rightarrow 19$ $P_2 \rightarrow 6$ $P_3 \rightarrow 8$ $P_4 \rightarrow 9$ $P_5 \rightarrow 14$
  - Tm = 56/5 = 11.2
- q=10
  - $P_1 \rightarrow 10$ $P_2 \rightarrow 11$ $P_3 \rightarrow 13$ $P_4 \rightarrow 14$ $P_5 \rightarrow 19$
  - Tm = 67/5 = 13.4

quindi

q piccolo ➔ tempi medi di esecuzione piccoli
q grande ➔ tempi medi di esecuzione grandi

MA

Non si e' considerato il tempo di cambio di contesto

Si ripeta l'esercizio considerando il tempo di cambio di contesto τ=1

Tempi medi di completamento con c.s.

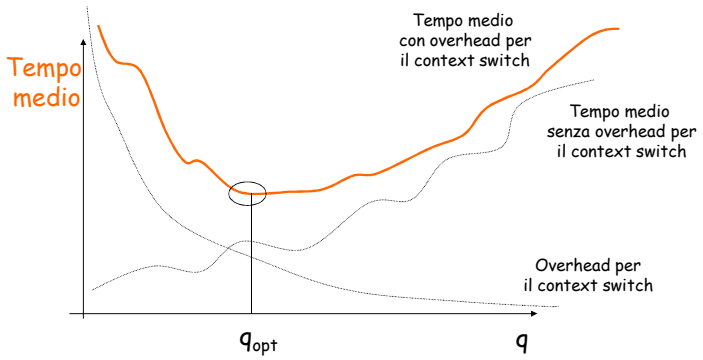
q=1
- P1 soffre di 14 cs, P2 di 1, P3 di 6, P4 di 3 e P5 di 13 cs
- Tempo medio di compl. (46+37)/5 = 83/5 = 16.6

q=2
- P1 soffre di 9 cs, P2 di 1, P3 di 2, P4 di 3 e P5 di 8 cs
- Tempo medio di compl. (48+23)/5 = 71/5 = 14.2

q=5
- P1 soffre di 5 cs, P2 di 1, P3 di 2, P4 di 3 e P5 di 4 cs
- Tempo medio di compl. (56+15)/5 = 71/5 = 14.2

q=10
- P1 soffre di 0 cs, P2 di 1, P3 di 2, P4 di 3 e P5 di 4 cs
- Tempo medio di compl. (67+10)/5 = 77/5 = 15.4

In generale



Esercizio 4:

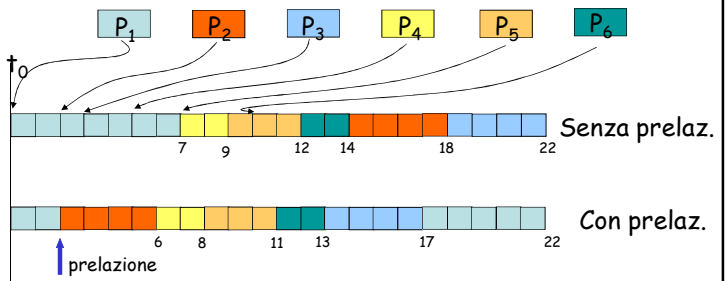
Siano i seguenti processi nella ready queue, arrivati nell'ordine specificato

Proc.	arrivo.	esec.
P_1	0	7
P_2	2	4
P_3	3	4
P_4	5	2
P_5	7	3
P_6	10	2

si consideri l'algoritmo SJF
- senza prelazione
- con prelazione

1. Si disegnino i diagrammi di Gantt
2. Calcolare i tempi di completamento nei 2 casi
3. Calcolare il tempo medio di completamento nei 2 casi

soluzione



- Senza prelazione
  - $P_1 \rightarrow 7$ $P_2 \rightarrow 16$ $P_3 \rightarrow 19$ $P_4 \rightarrow 4$ $P_5 \rightarrow 5$ $P_6 \rightarrow 4$
  - Tm = 55/6 = 9.16
- Con prelazione
  - $P_1 \rightarrow 22$ $P_2 \rightarrow 4$ $P_3 \rightarrow 14$ $P_4 \rightarrow 3$ $P_5 \rightarrow 4$ $P_6 \rightarrow 3$
  - Tm = 50/6 = 8.33

Esercizio 6:

Siano i seguenti processi

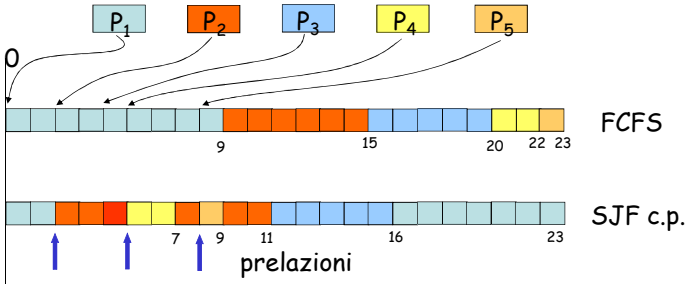
Proc.	esec.	arrivo
P_1	9	0
P_2	6	2
P_3	5	4
P_4	2	5
P_5	1	8

Si usi scheduling
- FCFS
- SJF con prelazione

Fornire i relativi diagrammi di Gantt

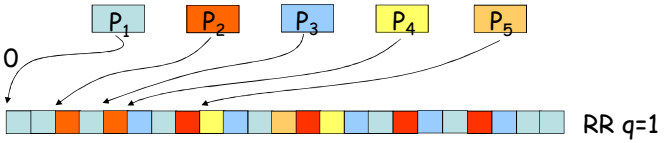
Nel caso di scheduling RR con q=1 ms, quanto deve durare il context switch per far terminare tutti i processi in 45 ms?

soluzione



- FCFS
  - $P_1 \rightarrow 9$ $P_2 \rightarrow 13$ $P_3 \rightarrow 16$ $P_4 \rightarrow 16$ $P_5 \rightarrow 15$
  - Tm = 69/5 = 13.8
- SJF Con prelazione
  - $P_1 \rightarrow 23$ $P_2 \rightarrow 9$ $P_3 \rightarrow 12$ $P_4 \rightarrow 2$ $P_5 \rightarrow 1$
  - Tm = 47/5 = 9.4

soluzione



Sia x il tempo del context switch

20 context switches ➔ tempo = 20x ms
Tempo esec dei processi = 23 ms

Tempo totale di esecuzione = 23 + 20x

23 + 20 x < 45 ➔ x < 22/20 = 1.1

note

- SJF con prelazione
  - T=2 arriva P2 con 6 u.t. mentre a P1 restano 7 u.t. ➔ Prelazione
  - T=4 arriva P3 con 5 u.t. mentre a P2 restano 4 u.t. ➔ No prelazione
  - T=5 arriva P4 con 2 u.t. mentre a P2 restano 3 u.t. ➔ Prelazione
  - T=7 finisce P4 va in esecuzione P2 con 3 u.t.
  - T=8 arriva P5 con 1 u.t. mentre a P2 restano 2 u.t. ➔ Prelazione
  - T=9 finisce P5 e vanno in esecuzione nell'ordine P2, P3 e P1 fino al loro completamento

- RR con q=10
  - T=2 supponiamo che P2 arriva un attimo prima della scadenza del quanto ed entra nella coda
  - T=4 supponiamo che P3 arriva un attimo prima della scadenza del quanto ed entra nella coda. Allo scadere del quanto, P2 va in esecuzione ed esce dalla coda mentre entra P1