SISTEMI OPERATIVI

I processi

Sommario

- Il concetto di processo
- Schedulazione dei processi

Il concetto di processo (1)

- Inizialmente i sistemi di calcolo eseguivano un programma alla volta
  - Completo controllo del sistema
  - e dell'accesso alle risorse
- Oggi più programmi vengono eseguiti contemporaneamente
  - Condivisione risorse
  - Concorrenza
- È necessario un controllo più ferreo sull'accesso alle risorse
- Da questa esigenza nasce il concetto **di processo (di elaborazione)**

Il concetto di processo (2)

- Queste attività sono dette **processi**
- I termini **job** e **processo** spesso usati come sinonimi
- **Processo** – “un programma in esecuzione”
- È solo una prima approssimazione,
- Un processo è molto di più ….

Programma ≠ Processo

- **Programma**
  - Entità passiva
  - Lista istruzioni
- **Processo**
  - Entità attiva
  - Contatore di programma
  - Valori delle variabili
  - Risorse in uso

- Anche se due processi possono essere associati allo stesso programma, essi sono due differenti *istanze di esecuzione* dello stesso codice!

Evoluzione della computazione di un processo

- Il processo è una **funzione** che trasforma informazioni eseguendo le istruzioni del programma
  - partendo dai valori iniziali
    - eventualmente acquisiti durante l'esecuzione stessa attraverso le periferiche
  - e producendo i risultati finali
    - emessi attraverso le periferiche

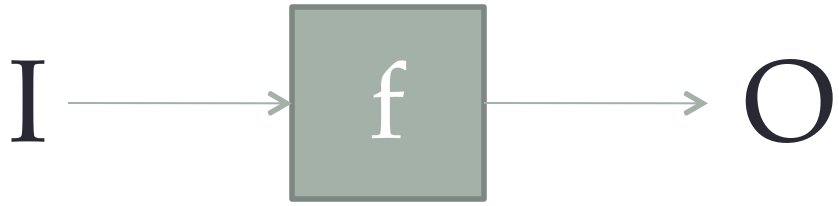
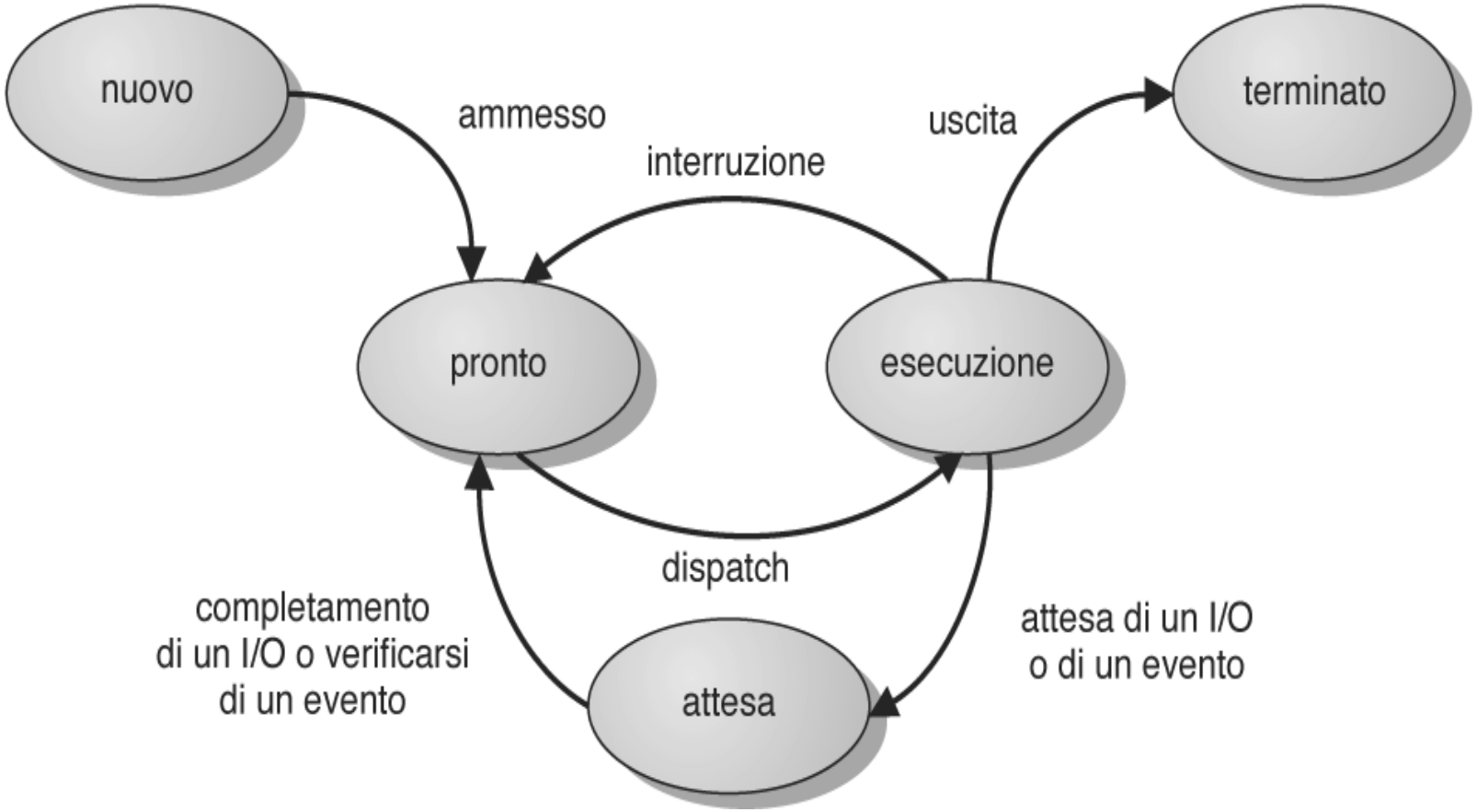


Diagramma degli stati di un processo



Lo stato di un processo

- E' lo **stato di uso del processore** da parte di un processo
- Possibili stati:
  - Nuovo (new): Il processo è stato creato
  - In esecuzione (running): le istruzioni vengono eseguite
  - In attesa (waiting): il processo sta aspettando il verificarsi di qualche evento
  - Pronto all'esecuzione (ready): il processo è in attesa di essere assegnato ad un processore
  - Terminato (terminated): il processo ha terminato l'esecuzione
- **1 solo** processo **per unità** di elaborazione può essere in esecuzione

Algoritmi di schedulazione

- Schedulazione FCFS
  (First Come First Served, primo arrivato primo servito)
- Schedulazione SJF
  (Shortest Job First, il lavoro più corto per primo)
- A Priorità
  (il più urgente è il primo servito)
- Schedulazione RR Round Robin
  (a ‘’quanti’’ di tempo)

Algoritmi di schedulazione

- Schedulazione FCFS (First Come First Served)

  - L'algoritmo FCFS esegue i processi nello stesso ordine in cui essi vengono sottomessi al sistema.
  - Il primo processo ad essere eseguito è esattamente quello che per primo richiede l'uso della CPU.
  - Quelli successivi vengono serviti non appena questo ha terminato la propria esecuzione, e così avviene successivamente per tutti gli altri posti in coda.

Algoritmi di schedulazione

- Schedulazione FCFS (First Come First Served)

Questo tipo di algoritmo è molto semplice da implementare ma solitamente è anche poco efficiente: se un processo ha un lungo periodo di elaborazione senza poter essere interrotto, tutti gli altri devono aspettare la sua naturale terminazione.

Prendiamo ad esempio la sottomissione nell'ordine (FCFS) dei seguenti processi con la seguente durata espressa in millisecondi:

p1: 10
p2: 4
p3: 2

Verranno eseguiti nello stesso ordine: p1 → p2 → p3

Algoritmi di schedulazione

- Tempo medio di attesa (ta)

Prendiamo ad esempio la sottomissione nell'ordine (FCFS) dei seguenti processi con la seguente durata espressa in millisecondi:

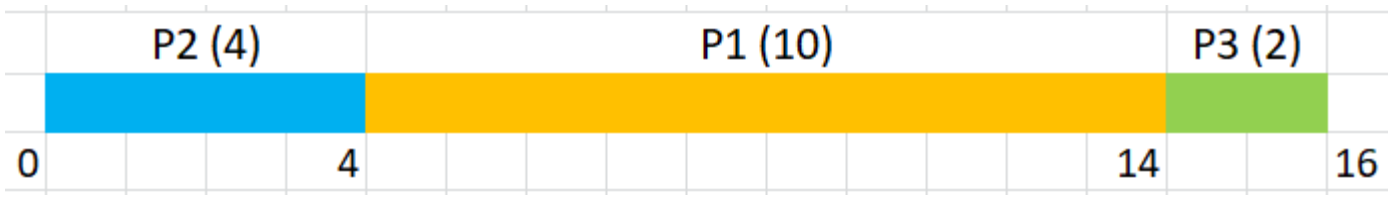
Il processo p1 non attende nulla, perché entra immediatamente in esecuzione. Il processo p2 attende 10 millisecondi, e p3 14.
Il tempo medio d'attesa quindi è

(0 + 10 + 14) / 3 = 8 millisecondi

Algoritmi di schedulazione

- Tempo medio di attesa (ta)

Ta = (0 + 10 + 14) / 3 = 8 millisecondi

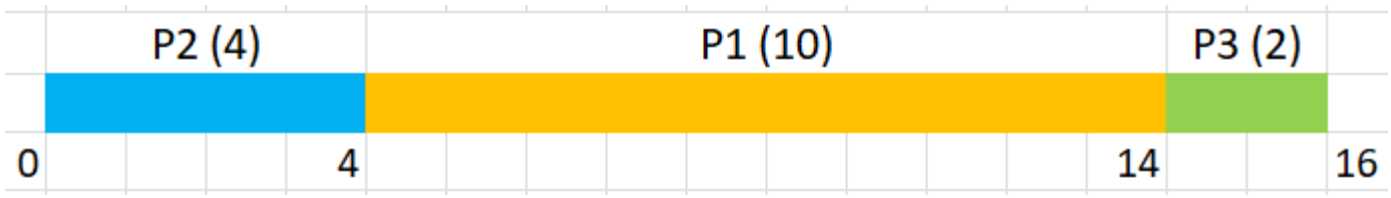


Ta = (0 + 4 + 14) / 3 = 6 millisecondi

Algoritmi di schedulazione

- Tempo medio di terminazione (Tt)

Tt = (10 + 14 + 16) / 3 = 13,3 millisecondi



Tt = (4 + 14 + 16) / 3 = 11,3 millisecondi

Algoritmi di schedulazione

- Schedulazione SJF (Shortest Job First)

differisce dal FCFS perchè la coda dei processi ready è ordinata per valori crescenti di durata.

p1: 10
p2: 4
p3: 2

Verranno eseguiti nell' ordine: p3 → p2 → p1

Ta = (0 + 2 + 6) / 3 = 2,6 millisecondi

Tt = (2 + 6 + 16) / 3 = 8 millisecondi

Algoritmi di schedulazione

- A Priorità (il più urgente è il primo servito)

La priorità è un parametro che determina l'importanza o l'urgenza di un processo o, quanto un utente è disposto a pagare perchè il proprio job sia eseguito rapidamente.

In una sistema basato sulle priorità, il processo in esecuzione è sempre quello con la priorità più elevata tra i processi pronti per l'esecuzione.

Algoritmi di schedulazione

- Schedulazione RR Round Robin (a ‘’quanti’’ di tempo)

Intervalli di tempo (conosciuti anche come "quanti di tempo") sono assegnati a ciascun processo in porzioni uguali e in modo circolare.
I processi vengono gestiti in base all’ordine di arrivo
Esempio:

‘’quanto’’ = 2

p1: 10	2 + 2+ 2 + 2 + 2
p2: 4	2 + 2
p3: 2	2

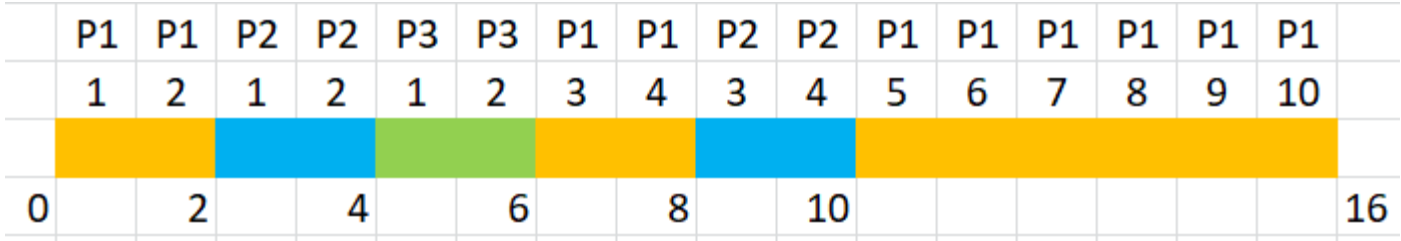
Esecuzione: p1 → p2 → p3 → p1 → p2 → p1 → p1 → p1

Algoritmi di schedulazione

- Schedulazione RR Round Robin (a ‘’quanti’’ di tempo)

‘’quanto’’ = 2

p1: 10	2 + 2 + 2 + 2 + 2
p2: 4	2 + 2
p3: 2	2



Ta = (0 + 2 + 4) / 3 = 2 millisecondi

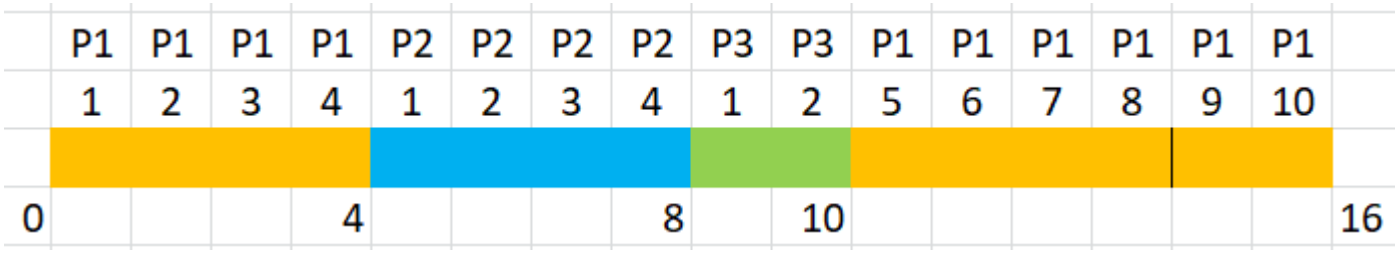
Tt = (16 + 10 + 6) / 3 = 10,6 millisecondi

Algoritmi di schedulazione

- Schedulazione RR Round Robin (a ‘’quanti’’ di tempo)

‘’quanto’’ = 4

p1: 10	4 + 4 + 2
p2: 4	4
p3: 2	2



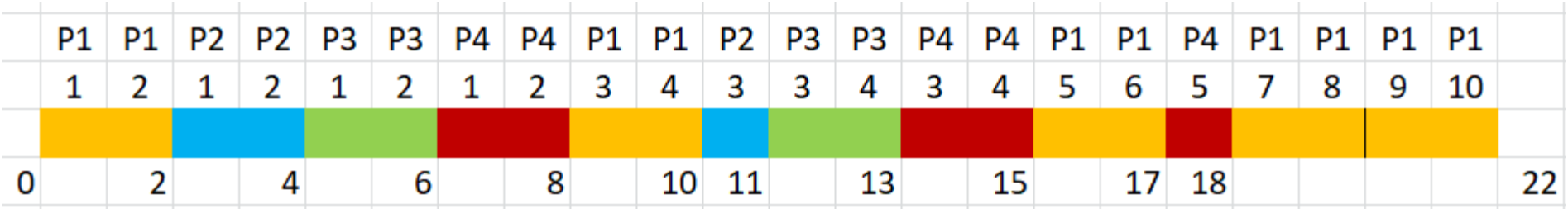
Ta = (0 + 4 + 8) / 3 = 4 millisecondi

Tt = (16 + 10 + 8) / 3 = 11,3 millisecondi

Algoritmi di schedulazione

- Schedulazione RR Round Robin (a ‘’quanti’’ di tempo)

‘’quanto’’ = 2	p1: 10	2 + 2 + 2 + 2 + 2
	p2: 3	2 + 1
	p3: 4	2 + 2
	p4: 5	2 + 2 + 1



Ta = (0 + 2 + 4 + 6) / 4 = 3 millisecondi

Tt = (22 + 11 + 13 + 18) / 4 = 16 millisecondi

Algoritmi di schedulazione

- Schedulazione RR Round Robin (a ‘’quanti’’ di tempo)

Ad esempio nell'ipotesi che vi siano i seguenti processi in coda con relativa durata in millisecondi, e la quantità di tempo stabilita di 20 ms:

p1: 30	20 + 10
p2: 15	15
p3: 60	20 + 20 + 20
p4: 45	20 + 20 + 5

Verranno eseguiti nel seguente ordine:
p1 (interrotto dopo 20 ms, ne rimangono altri 10)
p2 (termina la propria esecuzione perché dura meno di 20 ms)
p3 (interrotto dopo 20 ms, ne rimangono altri 40)
p4 (interrotto dopo 20 ms, ne rimangono altri 25)
p1 (termina la propria esecuzione perché necessitava di meno di 20 ms)
p3 (interrotto dopo 20 ms, ne rimangono altri 20)
p4 (interrotto dopo 20 ms, ne rimangono altri 5)
p3 (termina la propria esecuzione perché necessitava di esattamente 20 ms)
p4 (termina la propria esecuzione)

Algoritmi di schedulazione

- Schedulazione RR Round Robin (a ‘’quanti’’ di tempo)

Quale ‘’quanto’’ di tempo scegliere?

Il più piccolo possibile?

Nel calcolo di Ta e Tt non vengono presi in considerazione i tempi del ‘*context switch*’, ovvero i tempi morti necessari per fermare un processo e avviarne un altro