



ESERCITAZIONE 7 - Soluzioni

Algoritmi di schedulazione

Fattibilità di sistemi real-time

Gestione della memoria e del disco

Algoritmi di schedulazione

Algoritmi di schedulazione (1)

1) Supponendo di utilizzare l'algoritmo first-come first-served per lo scheduling in un sistema batch, se arrivano 4 job (in ordine A, poi B dopo 2ms, C dopo 4ms e D dopo 5ms) con i seguenti tempi di esecuzione $A=8$, $B=4$, $C=2$ e $D=4$ quali sono i tempi di turnaround? Qual è il tempo medio di turnaround e di attesa?

Algoritmi di schedulazione (1)

Soluzioni (1)

- ▶ First come – First served: Ordine in cui chi arriva prima viene eseguito per primo.
- ▶ Tempo turnaround: tempo che intercorre dal momento in cui un processo entra in coda per essere eseguito fino al termine della sua esecuzione.
- ▶ Tempo di attesa: tempo che un processo trascorre in coda prima di essere eseguito.
- ▶ TA = Time of arrival = tempo assoluto di ingresso in coda di un processo
- ▶ TE = Time of execution = tempo necessario per l'esecuzione di un processo

Algoritmi di schedulazione (1)

Soluzioni (2)

► Tabella con TA e TE di ogni processo dell'esercizio.

PROCESSO	TA	TE
A	0	8
B	2	4
C	4	2
D	5	4

Algoritmi di schedulazione (1)

Soluzioni (3)

- Strategia FCFS (FIFO): ordine A-B-C-D
- TS = Tempo assoluto di inizio di esecuzione di ogni processo
- TF = tempo assoluto di fine dell'esecuzione di un processo

Processo	TA	TE	TS	TF
A	0	8	0	8
B	2	4	8	12
C	4	2	12	14
D	5	4	14	18

Algoritmi di schedulazione (1)

Soluzioni (4)

➤ $TT = \text{Tempo Turnaround} = TF - TA$

➤ $TW = \text{Tempo di attesa} = TS - TA$

Processo	TA	TE	TS	TF	TW	TT
A	0	8	0	8	0	8
B	2	4	8	12	6	10
C	4	2	12	14	8	10
D	5	4	14	18	9	13

Algoritmi di schedulazione (1)

Soluzioni (5)

- Tempo medio (entambi i casi): somma tempi e divisione per numero processi.

$$TW(\text{medio}) = \frac{0+6+8+9}{4} = 5.75$$

$$TT(\text{medio}) = \frac{(8+10+10+13)}{4} = 10.25$$

Fattibilità di sistemi real-time

Fattibilità di sistemi real-time (1)

1) Supponendo di dover valutare la fattibilità di un sistema soft real-time con eventi periodici $P_0=300\mu s$, $P_1=900\mu s$, $P_2=45ms$, $P_3=150ms$ e rispettivi tempi di elaborazione $C_0=25\mu s$, $C_1=300\mu s$, $C_2=9ms$, $C_3=50\mu s$ il sistema è sostenibile? Se si aggiunge un nuovo evento periodico $P_4=110ms$, quanto è il tempo massimo di elaborazione affinché il sistema rimanga sostenibile?

Fattibilità di sistemi real-time (1)

Soluzioni (1)

- ▶ **Deadline:** tempo entro il quale le operazioni devono necessariamente essere completate
- ▶ Sistemi non obbligatoriamente **veloci** ma piuttosto **affidabili**.
- ▶ Eventi **periodici**: eseguiti a ciclo continuo con cadenza ben definita.
- ▶ Eventi periodici hanno:
 - ▶ **Periodo**: tempo entro il quale l'evento deve essere ripetuto (P_i)
 - ▶ **Tempo d'esecuzione** (C_i)
- ▶ Sistema RT sostenibile (con m eventi periodici):
 - ▶ Vale $\sum_{i=1}^m \left(\frac{C_i}{P_i} \right) \leq 1$

Fattibilità di sistemi real-time (1)

Soluzioni (2)

- Sostituzione valori all'interno della formula:

$$\frac{C_0}{P_0} + \frac{C_1}{P_1} + \frac{C_2}{P_2} + \frac{C_3}{P_3} = \frac{25}{300} + \frac{300}{900} + \frac{9}{45} + \frac{50}{150} = \frac{57}{60} \leq 1$$

Ref testo: Moderni sistemi operativi, paragrafo 2.5.4. *Schedulazione nei sistemi real time*, pag 135 e successive

Fattibilità di sistemi real-time (1)

Soluzioni (3)

- Massimo tempo di esecuzione per il nuovo processo P4.

$$\frac{C_0}{P_0} + \frac{C_1}{P_1} + \frac{C_2}{P_2} + \frac{C_3}{P_3} + \frac{C_4}{P_4} = \frac{57}{60} + \frac{C_4}{110} \leq 1$$

$$\frac{C_4}{110} \leq 1 - 57/60$$

$$C_4 \leq \left(\frac{3}{60}\right) * 110 \Rightarrow C_4 \leq 5.5$$

Massimo tempo di esecuzione: 5.5ms

Gestione della memoria e del disco

Gestione della memoria e del disco (1)

1) Nell'ambito della gestione della memoria con liste concatenate, considerando una lista parzialmente piena di questo tipo (per ogni tripla, il primo elemento è un flag per capire se si tratta di un buco o un processo, il secondo è l'indirizzo di partenza e il terzo è la lunghezza dell'elemento):

$P, 0, 6, \rightarrow H, 6, 3, \rightarrow P, 9, 8 \rightarrow P, 17, 4, \rightarrow H, 21, 2, \rightarrow P, 23, 6, \rightarrow H, 29, 4, X$

Dove viene posizionato il nuovo processo P che occupa 4 blocchi? Dove verrebbe posizionato se invece ne occupasse solo 2, secondo gli algoritmi first fit e best fit?

Gestione della memoria e del disco (1)

Soluzioni (1)

- ▶ Lettera P rappresenta un processo
- ▶ Lettera H rappresenta un “buco” di un certo numero di blocchi
- ▶ **First-fit**: inserisce il nuovo processo nella prima posizione possibile dall’inizio della lista
- ▶ **Best-fit**: inserisce il nuovo processo nella posizione tale da contenerlo nel modo migliore e, dunque, nel buco più piccolo disponibile per contenere il processo.
- ▶ Ref: “I moderni sistemi operativi”, *Tanenbaum*, pagina 183 e seguenti (nell’edizione in mio possesso), paragrafo 4.2.2. Gestione della memoria con liste concatenate.

Gestione della memoria e del disco (1)

Soluzioni (2)

- ▶ Caso “base”, con processo di 4 blocchi
- ▶ $P,0,6, \rightarrow H,6,3, \rightarrow P,9,8 \rightarrow P,17,4, \rightarrow H,21,2, \rightarrow P,23,6, \rightarrow H,29,4, X$
- ▶ Spazio necessario per contenere un processo di dimensione 4 blocchi solo a partire dall'indirizzo 29.

Gestione della memoria e del disco (1)

Soluzioni (3)

- ▶ Caso first fit, con processo di 2 blocchi
- ▶ $P,0,6 \rightarrow H,6,3 \rightarrow P,9,8 \rightarrow P,17,4 \rightarrow H,21,2 \rightarrow P,23,6 \rightarrow H,29,4, X$
- ▶ Primo spazio sufficiente per contenere il processo inizia all'indirizzo 6.
- ▶ Si ottiene:
- ▶ $P,0,6 \rightarrow P,6,2 \rightarrow H,8,1 \rightarrow P,9,8 \rightarrow P,17,4 \rightarrow H,21,2 \rightarrow P,23,6 \rightarrow H,29,4, X$

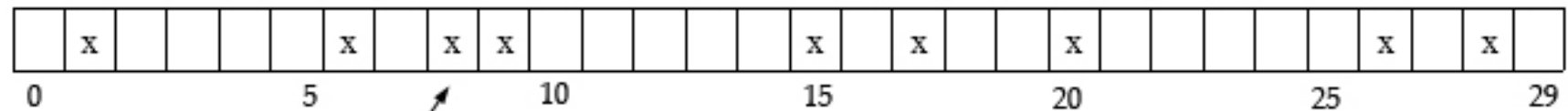
Gestione della memoria e del disco (1)

Soluzioni (4)

- ▶ Caso first fit, con processo di 2 blocchi
- ▶ $P,0,6 \rightarrow H,6,3 \rightarrow P,9,8 \rightarrow P,17,4 \rightarrow H,21,2 \rightarrow P,23,6 \rightarrow H,29,4, X$
- ▶ Miglior spazio per contenere il processo inizia all'indirizzo 21.
- ▶ Si ottiene:
- ▶ $P,0,6 \rightarrow H,6,3 \rightarrow P,9,8 \rightarrow P,17,4 \rightarrow P,21,2 \rightarrow P,23,6 \rightarrow H,29,4, X$

Gestione della memoria e del disco (2)

2) Nell'ambito della schedulazione delle richieste al disco, si consideri la situazione delle richieste pendenti e della testina illustrata in figura. Come si comporterà la testina nel caso in cui l'algoritmo di schedulazione utilizzato sarà l'SSF (Shortest Seek First)? Come si comporterà nel caso in cui sarà l'algoritmo dell'ascensore?



Posizione iniziale della testina

Le x rappresentano le richieste pendenti di accesso al disco

Gestione della memoria e del disco (2)

Soluzioni (1)

- Ref: I moderni sistemi operativi, pag 296 e seguenti, paragrafo 5.4.3. Gli algoritmi di schedulazione del braccio del disco.
- Algoritmo **SSF** (Shortest Seek First): risolve per prima le richieste più vicine alla posizione attuale della testina. La posizione successiva k è stabilita dalla formula seguente: $\underset{k}{\operatorname{argmin}} |(P - R_k)|$
- Algoritmo dell'**ascensore**: risolve tutte le richieste in una direzione (salita, ad esempio), poi risolve quelle nell'altra direzione. Garantisce la *fairness* del sistema, rendendo impossibile la *starvation*.

Gestione della memoria e del disco (2)

Soluzioni (2)

- ▶ Algoritmo **SSF**: soluzione:
 - ▶ 8, 9, 6, 1, 15, 17, 20, 26 e 28.
- ▶ Algoritmo dell'**ascensore**: soluzione:
 - ▶ 8, 9, 15, 17, 20, 26, 28, 6, 1