Sistemi Embedded e Real-time (M. Cesati)

Compito scritto del 1 luglio 2009

Esercizio 1. Il seguente sistema di task periodici è schedulato su un processore con un algoritmo "cyclic schedule": $T_1 = (6, 3)$, $T_2 = (4, 1)$, $T_3 = (8, 1, 2)$.

(a) Determinare una dimensione appropriata per il frame. I job sono interrompibili, ma il numero di interruzioni deve essere il più piccolo possibile.

(b) Determinare una schedulazione dei task periodici nell'iperperiodo, utilizzando se possibile la dimensione del frame individuata in (a).

Esercizio 2. Si consideri un sistema di task periodici, indipendenti e interrompibili schedulati su un processore: $T_1 = (4, 1, 3)$, $T_2 = (6, 2)$, $T_3 = (10, 3)$.

(a) Determinare analiticamente se il sistema è schedulabile con EDF.

(b) Determinare analiticamente se il sistema è schedulabile con RM.

Si consideri al posto del task T_3 il task $T_3' = (8, 3, 10)$.

(c) Determinare se il nuovo sistema è schedulabile con EDF.

(d) Determinare se il nuovo sistema è schedulabile con RM.

Esercizio 3. Un sistema deve eseguire cinque job $J_1, \ldots, J_5$, con J_i di priorità maggiore di J_k se $i < k$. I job utilizzano tre risorse condivise R_1, R_2 e R_3, e le relative sezioni critiche sono: $J_1 : [R_1; 2]$, J_2 : nessuna, $J_3 : [R_2; 1]$, $J_4 : [R_1; 3]\,[R_3; 1]$, $J_5 : [R_2; 4]\,[R_3; 2]$.

(a) Quali sono i tempi massimi di blocco dei job per conflitto di risorse utilizzando il procotollo NPCS?

(b) Quali sono i tempi massimi di blocco dei job per conflitto di risorse utilizzando il procotollo priority-inheritance?

Sistemi Embedded e Real-time (M. Cesati)

Soluzioni del compito scritto del 1 luglio 2009

Esercizio 1. *Il seguente sistema di task periodici è schedulato su un processore con un algoritmo "cyclic schedule":* $T_1 = (6, 3)$, $T_2 = (4, 1)$, $T_3 = (8, 1, 2)$.
(a) *Determinare una dimensione appropriata per il frame. I job sono interrompibili, ma il numero di interruzioni deve essere il più piccolo possibile.*

- Vincolo sui tempi d'esecuzione dei job:

$$f \geq \max\{3, 1, 1\} = 3$$

- Vincolo sulla divisibilità della lunghezza dell'iperperiodo ($\text{mcm}\{6, 4, 8\} = 24$):

$$f \in \{3, 4, 6, 8, 12, 24\}$$

- Vincolo sul task T_1 ($2f - \gcd\{6, f\} \leq 6$):

$$\begin{array}{ll} 2 \cdot 3 - \gcd\{6, 3\} = 3 \leq 6 & \\ 2 \cdot 4 - \gcd\{6, 4\} = 6 \leq 6 & \\ 2 \cdot 6 - \gcd\{6, 6\} = 6 \leq 6 & \\ 2 \cdot x - \gcd\{6, x\} > 6 \text{ if } x > 6 & \Rightarrow f \leq 6 \end{array}$$

- Vincolo sul task T_2 ($2f - \gcd\{4, f\} \leq 4$):

$$\begin{array}{ll} 2 \cdot 3 - \gcd\{4, 3\} = 5 > 4 & \Rightarrow f \neq 3 \\ 2 \cdot 4 - \gcd\{4, 4\} = 4 \leq 4 & \\ 2 \cdot x - \gcd\{4, x\} > 4 \text{ if } x > 4 & \Rightarrow f \leq 4 \end{array}$$

- Vincolo sul task T_3 ($2f - \gcd\{8, f\} \leq 2$):

$$2 \cdot 4 - \gcd\{8, 4\} = 4 > 2 \qquad \Rightarrow f \neq 4$$

Non esiste alcun valore di $f \geq 3$ che soddisfa tutte le condizioni. È necessario dunque frammentare un job per ammettere valori di f inferiori; in particolare, l'unico task con job di durata maggiore di 3 è T_1. Scegliamo di spezzare ciascun job di T_1 in due frammenti con tempi di esecuzione rispettivamente uguali a 1 e 2.

- Il limite inferiore alla dimensione del frame diventa: $f \geq 2$.
- È possibile considerare un nuovo divisore di 24: $f = 2$.

- Vincolo sul task T_1 ($2f - \gcd\{6, f\} \leq 6$):

$$2 \cdot 2 - \gcd\{6, 2\} = 2 \leq 6$$

- Vincolo sul task T_2 ($2f - \gcd\{4, f\} \leq 4$):

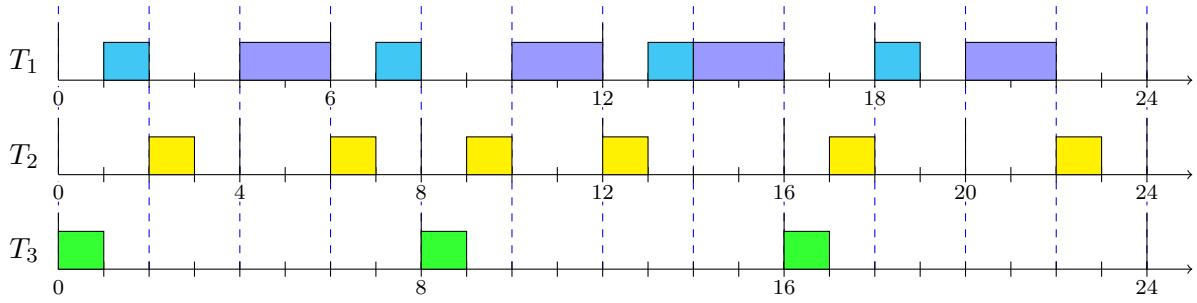
$$2 \cdot 2 - \gcd\{4, 2\} = 2 \leq 4$$

- Vincolo sul task T_3 ($2f - \gcd\{8, f\} \leq 2$):

$$2 \cdot 2 - \gcd\{8, 2\} = 2 \leq 2$$

Una dimensione del frame ammissibile è $f = 2$; essa comporta una interruzione per ciascuna esecuzione del task T_1.

(b) *Determinare una schedulazione dei task periodici nell'iperperiodo, utilizzando se possibile la dimensione del frame individuata in (a).*



Equivalentemente, la schedulazione è descrivibile elencando i blocchi di schedulazione:

$L(0) = \langle T_3, T_{1,1} \rangle$, $L(1) = \langle T_2 \rangle$, $L(2) = \langle T_{1,2} \rangle$, $L(3) = \langle T_2, T_{1,1} \rangle$, $L(4) = \langle T_3, T_2 \rangle$, $L(5) = \langle T_{1,2} \rangle$, $L(6) = \langle T_2, T_{1,1} \rangle$, $L(7) = \langle T_{1,2} \rangle$, $L(8) = \langle T_3, T_2 \rangle$, $L(9) = \langle T_{1,1} \rangle$, $L(10) = \langle T_{1,2} \rangle$, $L(11) = \langle T_2 \rangle$.

Esercizio 2. *Si consideri un sistema di task periodici, indipendenti e interrompibili schedulati su un processore:* $T_1 = (4, 1, 3)$, $T_2 = (6, 2)$, $T_3 = (10, 3)$.

(a) *Determinare analiticamente se il sistema è schedulabile con EDF.*

Poiché T_1 ha scadenza relativa inferiore al periodo, è necessario considerare la densità dei task:

$$\Delta = \frac{1}{3} + \frac{2}{6} + \frac{3}{10} = \frac{29}{30} < 1$$

Di conseguenza, il sistema di task è schedulabile con EDF.

(b) *Determinare analiticamente se il sistema è schedulabile con RM.*

Poiché la scadenza relativa di T_1 è inferiore al periodo non è possibile utilizzare le condizioni di schedulabilità.

Applichiamo il test di schedulabilità calcolando la funzione di tempo necessario

$$w_i(t) = e_i + \sum_k \left\lceil \frac{t}{p_k} \right\rceil \cdot e_k$$

ove la sommatoria è estesa a tutti i task con priorità maggiore a quella del task in esame.

- Test di schedulabilità per T_1: $w_1(t) = 1 \leq 3 \quad \Rightarrow$ schedulabile.
- Test di schedulabilità per T_2: $w_2(t) = 2 + \lceil t/4 \rceil$;
  $w_2(4) = 3 \leq 4 \quad \Rightarrow$ schedulabile
- Test di schedulabilità per T_3: $w_3(t) = 3 + \lceil t/4 \rceil + \lceil t/6 \rceil \cdot 2$;
  $w_3(4) = 6 > 4$
  $w_3(6) = 7 > 6$
  $w_3(8) = 9 > 8$
  $w_3(10) = 10 \leq 10 \quad \Rightarrow$ schedulabile

Il sistema di task è perciò schedulabile con RM.

Si consideri al posto del task T_3 il task $T_3' = (8, 3, 10)$.

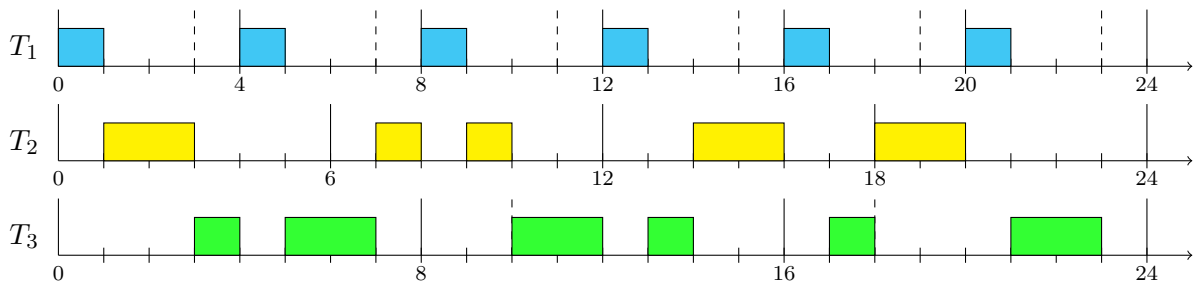
(c) *Determinare se il nuovo sistema è schedulabile con EDF.*

Poichè la densità del nuovo sistema è:

$$\Delta = \frac{1}{3} + \frac{2}{6} + \frac{3}{8} = \frac{25}{24} > 1$$

non è possibile dedurne alcuna conclusione.

Simuliamo la schedulazione EDF per un iperperiodo ($H = \text{mcm}\{4, 6, 8\} = 24$).



Poiché tutti i job rilasciati nell'iperperiodo terminano entro l'iperperiodo stesso, l'esistenza di una schedulazione valida entro l'iperperiodo permette di concludere che il sistema è schedulabile con EDF.

(d) *Determinare se il nuovo sistema è schedulabile con RM.*

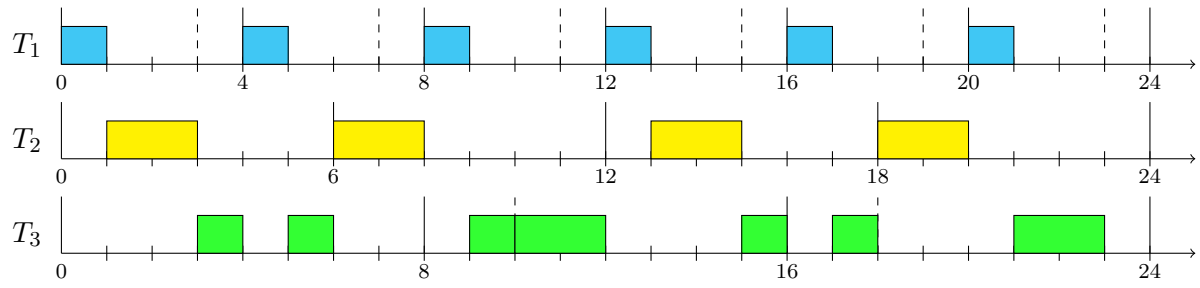
L'analisi svolta al punto (b) dell'esercizio permette di concludere che i task T_1 e T_2 nonché il primo job del task T_3 sono schedulabili. Poiché però il primo job di T_3 termina dopo il rilascio del successivo job, è necessario controllare anche la schedulabilità di tutti gli altri job di T_3 in un intervallo totalmente occupato.

- La lunghezza dell'intervallo totalmente occupato si ottiene risolvendo iterativamente $t = \lceil t/4 \rceil + \lceil t/6 \rceil \cdot 2 + \lceil t/8 \rceil \cdot 3$; il valore iniziale è la somma dei tempi d'esecuzione: $t_0 = 6$; $t_1 = 2 + 2 + 3 = 7$; $t_2 = 2 + 4 + 3 = 9$; $t_3 = 3 + 4 + 6 = 13$; $t_4 = 4 + 6 + 6 = 16$; $t_5 = 4 + 6 + 6 = 16 = t_4$.
- Numero di job di T_3 nell'intervallo totalmente occupato: $\lceil 16/8 \rceil = 2$.
- Test per il secondo job di T_3 utilizzando $w_{3,2}(t) = 2 \cdot 3 + \lceil t/4 \rceil + \lceil t/6 \rceil \cdot 2$, per $t \in [8, 10 + 8]$:

  $w_{3,2}(8) = 12 > 8$
  $w_{3,2}(10) = 13 > 10$
  $w_{3,2}(12) = 13 > 12$
  $w_{3,2}(16) = 16 \leq 16$

Di conseguenza, il nuovo sistema è schedulabile con RM.

A titolo di esempio, non richiesto dall'esercizio, si riporta una schedulazione RM per un iperperiodo ($H = \text{mcm}\{4, 6, 8\} = 24$).



Poiché tutti i job rilasciati nell'iperperiodo terminano entro l'iperperiodo stesso, l'esistenza di una schedulazione valida entro l'iperperiodo permette di concludere che il sistema è schedulabile con RM.

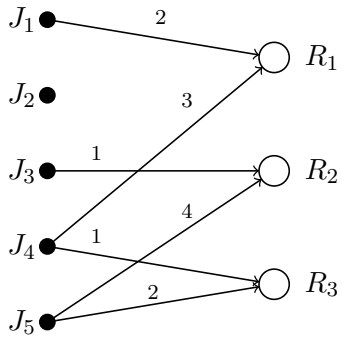
Esercizio 3*. Un sistema deve eseguire cinque job $J_1, \ldots, J_5$, con J_i di priorità maggiore di J_k se $i < k$. I job utilizzano tre risorse condivise R_1, R_2 e R_3, e le relative sezioni critiche sono: $J_1 : [R_1; 2]$, $J_2 :$ nessuna, $J_3 : [R_2; 1]$, $J_4 : [R_1; 3]\,[R_3; 1]$, $J_5 : [R_2; 4]\,[R_3; 2]$.*

(a) *Quali sono i tempi massimi di blocco dei job per conflitto di risorse utilizzando il procotollo NPCS?*

Il massimo tempo di blocco per conflitto di risorse $b_i(\text{rc})$ del job J_i è dato dalla lunghezza della più lunga sezione critica tra tutti i job di priorità inferiore. Perciò assumendo che i job non si auto-sospendano:

$$b_1(\text{rc}) = 4; \quad b_2(\text{rc}) = 4; \quad b_3(\text{rc}) = 4; \quad b_4(\text{rc}) = 4; \quad b_5(\text{rc}) = 0.$$

(b) *Quali sono i tempi massimi di blocco dei job per conflitto di risorse utilizzando il procotollo priority-inheritance?*



B_d	J_2	J_3	J_4	J_5
J_1			3	
J_2	$*$			
J_3		$*$		4
J_4			$*$	2

B_i	J_2	J_3	J_4	J_5
J_1				
J_2	$*$		3	
J_3		$*$	3	
J_4			$*$	4

Assumendo che i job non si auto-sospendano, i tempi di blocco per conflitti di risorse di ciascun job sono determinati dal valore massimo su ciascuna riga delle tabelle:

$$b_1(\text{rc}) = 3; \quad b_2(\text{rc}) = 3; \quad b_3(\text{rc}) = 4; \quad b_4(\text{rc}) = 4; \quad b_5(\text{rc}) = 0.$$