Sistemi Embedded e Real-time (M. Cesati)

Compito scritto del 11 settembre 2012

Esercizio 1. Si consideri il seguente sistema di task periodici : $T_1 = (3, 1)$, $T_2 = (4, 1)$, $T_3 = (4, 1)$, $T_4 = (6, 1)$. I job sono non interrompibili.

(a) Determinare una schedulazione ciclica strutturata per la dimensione del frame che minimizza l'overhead dello scheduler.

(b) Si supponga che la scadenza relativa del task T_1 passi da 3 a 7, ossia che $T_1 = (3, 1, 7)$: esiste una schedulazione ciclica strutturata con un frame di dimensione 4? Giustificare la risposta.

Esercizio 2. Si consideri un sistema di task periodici indipendenti, sempre interrompibili e che non si auto-sospendono: $T_1 = (4, 1)$, $T_2 = (5, 1)$, $T_3 = (6, 1)$, $T_4 = (7, 1)$.

(a) Supponendo che venga utilizzato uno scheduler RM e che l'overhead dello scheduler e del cambio di contesto siano trascurabili, verificare analiticamente che il sistema è schedulabile su un singolo processore.

(b) Supponendo che l'overhead del cambio di contesto sia pari a $CS = 1/50$, verificare analiticamente che il sistema non è più schedulabile su un singolo processore.

(c) Con riferimento al punto (b): si supponga che, mentre i periodi dei task T_1, T_3 e T_4 non possano essere variati, sia possibile invece aumentare leggermente il periodo del task T_2, fino al valore limite 6. Determinare analiticamente un valore per il periodo di T_2 che garantisca la schedulabilità del sistema.

Esercizio 3. Un sistema deve eseguire quattro job $J_1, \ldots, J_4$, con J_i di priorità maggiore di J_k se $i < k$. I job utilizzano quattro risorse condivise $R_1, \ldots R_4$, e le relative sezioni critiche sono: $J_1 : [R_2, 4][R_3, 5]$, $J_2 : [R_3, 3][R_1, 2]$, $J_3 : [R_1, 2\,[R_4, 1]\,]$, $J_4 : [R_2, 1][R_4, 1]$.

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

(b) Quali sono i tempi massimi di blocco per conflitto di risorse utilizzando il protocollo priority ceiling?

Sistemi Embedded e Real-time (M. Cesati)

Soluzioni del compito scritto del 11 settembre 2012

Esercizio 1. *Si consideri il seguente sistema di task periodici :* $T_1 = (3, 1)$, $T_2 = (4, 1)$, $T_3 = (4, 1)$, $T_4 = (6, 1)$. *I job sono non interrompibili.*

(a) *Determinare una schedulazione ciclica strutturata per la dimensione del frame che minimizza l'overhead dello scheduler.*

Determiniamo le possibili dimensioni f del frame dello scheduler ciclico:

- Nessun vincolo dovuto alla fase dei task, poiché essi sono tutti in fase.
- Vincolo sui tempi d'esecuzione dei job:

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

- Vincolo sulla divisibilità della lunghezza dell'iperperiodo ($\text{mcm}\{3, 4, 4, 6\} = 12$):

$$f \in \{1, 2, 3, 4, 6, 12\}.$$

- Vincolo sul task T_1 ($2f - \gcd\{3, f\} \leq 3$):

$$\begin{array}{lll} 2 \cdot 1 - \gcd\{3, 1\} = 1 & \leq 3 & \text{ok} \\ 2 \cdot 2 - \gcd\{3, 2\} = 3 & \leq 3 & \text{ok} \\ 2 \cdot 3 - \gcd\{3, 3\} = 3 & \leq 3 & \text{ok} \\ 2 \cdot 4 - \gcd\{3, 4\} = 7 & > 3 & \text{no} \\ 2 \cdot 6 - \gcd\{3, 6\} = 9 & > 3 & \text{no} \\ 2 \cdot 12 - \gcd\{3, 12\} = 21 & > 3 & \text{no} \end{array}$$

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

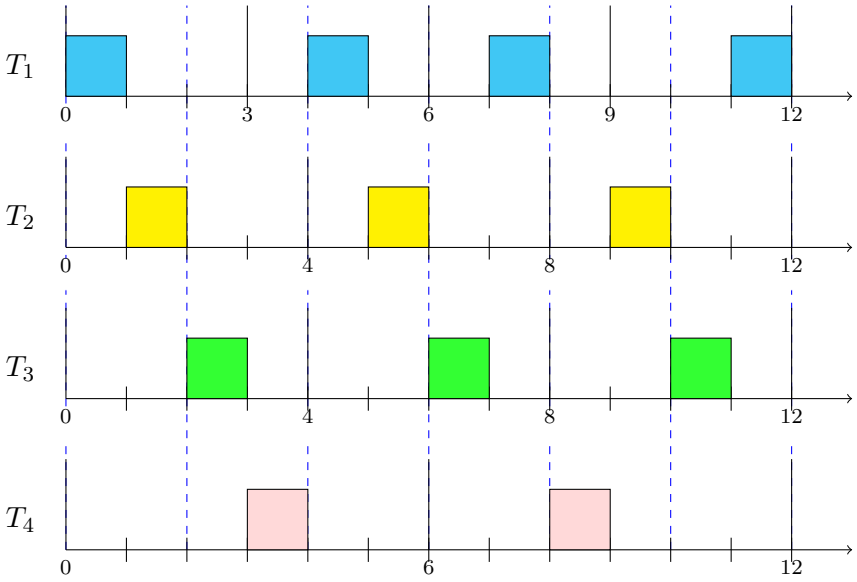
$$\begin{array}{lll} 2 \cdot 1 - \gcd\{4, 1\} = 1 & \leq 4 & \text{ok} \\ 2 \cdot 2 - \gcd\{4, 2\} = 2 & \leq 4 & \text{ok} \\ 2 \cdot 3 - \gcd\{4, 3\} = 5 & > 4 & \text{no} \end{array}$$

- Vincolo sul task T_3 ($2f - \gcd\{4, f\} \leq 4$): identico al task T_2.
- Vincolo sul task T_4 ($2f - \gcd\{6, f\} \leq 6$):

$$\begin{array}{lll} 2 \cdot 1 - \gcd\{6, 1\} = 1 & \leq 6 & \text{ok} \\ 2 \cdot 2 - \gcd\{6, 2\} = 2 & \leq 6 & \text{ok} \end{array}$$

Le uniche dimensioni ammissibili per il frame dello scheduler ciciclo strutturato sono $f = 1$ e $f = 2$. La dimensione $f = 2$ è quella che minimizza l'overhead dello scheduler.

Non è difficile determinare una schedulazione valida nell'intero iperperiodo utilizzando $f = 2$. Ad esempio:



(b) *Si supponga che la scadenza relativa del task T_1 passi da 3 a 7, ossia che $T_1 = (3, 1, 7)$: esiste una schedulazione ciclica strutturata con un frame di dimensione 4? Giustificare la risposta.*

Per prima cosa controlliamo che la dimensione $f = 4$ soddisfa le condizioni analitiche:

- Vincolo sul task T_1 ($2f - \gcd\{3, f\} \leq 7$):

$$2 \cdot 4 - \gcd\{3, 4\} = 7 \quad \leq 7 \quad \text{ok}$$

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

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

- Vincolo sul task T_3 ($2f - \gcd\{4, f\} \leq 4$): identico al task T_2.
- Vincolo sul task T_4 ($2f - \gcd\{6, f\} \leq 6$):

$$2 \cdot 4 - \gcd\{6, 4\} = 6 \quad \leq 6 \quad \text{ok}$$

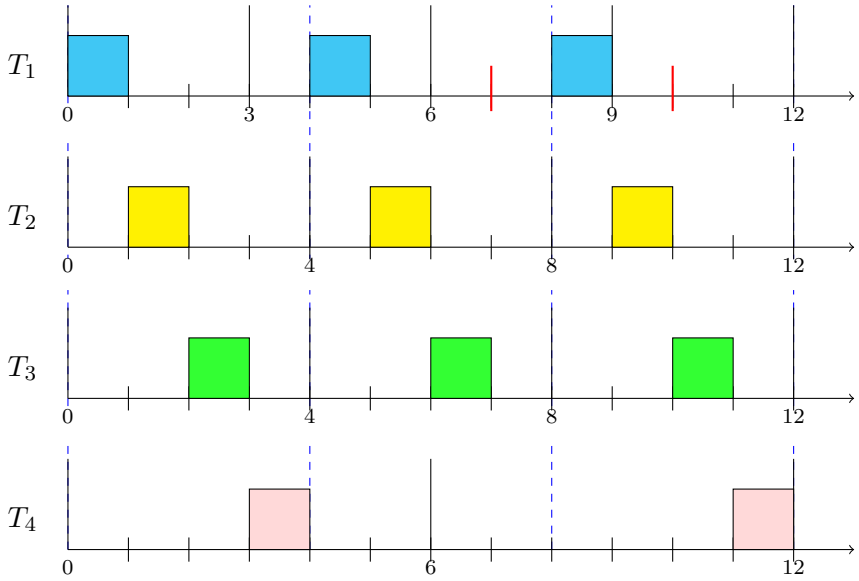
Perciò la dimensione del frame $f = 4$ è ammissibile. Ciò non significa, comunque, che esista una schedulazione ciclica strutturata per questa dimensione del frame.

Al contrario, è possibile dimostrare che tale schedulazione non può esistere. Infatti osserviamo preliminarmente che il fattore di utilizzazione del sistema di task è pari a

$$\frac{1}{3} + \frac{1}{4} + \frac{1}{4} + \frac{1}{6} = 1.$$

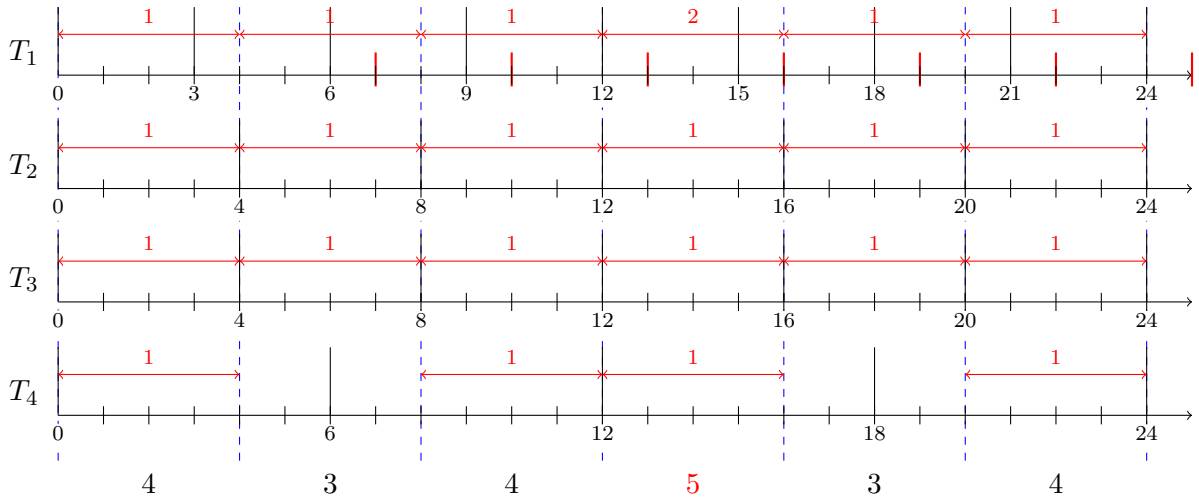
Di conseguenza, una eventuale schedulazione ciclica dovrebbe "riempire" completamente tutti i frame, senza mai lasciare il processore inutilizzato.

Consideriamo ora i primi tre frame (ossia il primo iperperiodo) della eventuale schedulazione:



All'istante $t = 7$ non è possibile schedulare alcun job. Infatti, il terzo job di T_1 non può essere eseguito, in quanto è stato rilasciato all'istante 6 che cade in mezzo al secondo frame: deve essere eseguito nel terzo frame. Analogamente, il secondo job di T_4 non può essere eseguito, in quanto anch'esso rilasciato all'istante 6. Infine, i job già rilasciati dei task T_2 e T_3 sono stati già completati. Poiché esiste un intervallo di tempo in cui il processore è forzatamente inutilizzato, si può concludere immediatamente che non esiste una schedulazione ciclica strutturata per $f = 4$.

In alternativa, una dimostrazione che non può esistere la schedulazione ciclica può essere ottenuta determinando per ciascun job in quali frame esso deve essere eseguito. Si ottiene così lo schema seguente, in cui il lavoro da effettuare per ciascun task è rappresentato da frecce che si estendono tra uno o più frame e valori numerici che indicano la somma dei tempi d'esecuzione dei job in quei frame.



Considerando il lavoro totale che il processore deve svolgere in ciascun frame, risulta evidente che nel frame iniziante all'istante 12 non sarà possibile eseguire tutti i job, in quanto il tempo di esecuzione totale supera la dimensione del frame (4).

Esercizio 2. *Si consideri un sistema di task periodici indipendenti, sempre interrompibili e che non si auto-sospendono:* $T_1 = (4, 1)$, $T_2 = (5, 1)$, $T_3 = (6, 1)$, $T_4 = (7, 1)$.

(a) *Supponendo che venga utilizzato uno scheduler RM e che l'overhead dello scheduler e del cambio di contesto siano trascurabili, verificare analiticamente che il sistema è schedulabile su un singolo processore.*

Osserviamo preliminarmente che le scadenze relative dei task coincidono con i rispettivi periodi. Possiamo dunque cercare di applicare una condizione di schedulabilità basata sul fattore di utilizzazione del sistema di task.

Il fattore di utilizzazione totale del sistema è pari a

$$\frac{1}{4} + \frac{1}{5} + \frac{1}{6} + \frac{1}{7} = \frac{319}{420}.$$

Poiché $319/420 > 0{,}759 > 0{,}757 > U_{\mathrm{RM}}(4)$, la condizione di Liu-Layland non permette di stabilire la schedulabilità.

Applicando il test iperbolico si ottiene invece:

$$\left(1 + \frac{1}{4}\right) \cdot \left(1 + \frac{1}{5}\right) \cdot \left(1 + \frac{1}{6}\right) \cdot \left(1 + \frac{1}{7}\right) = 2,$$

dunque possiamo concludere immediatamente che il sistema di task è schedulabile con RM.

(b) *Supponendo che l'overhead del cambio di contesto sia pari a $CS = 1/50$, verificare analiticamente che il sistema non è più schedulabile su un singolo processore.*

Poiché i job non si auto-sospendono, è possibile modellare il costo del cambio di contesto aumentando i tempi di esecuzione dei job per un tempo pari al doppio di *CS*. Pertanto, il sistema di task equivalente è

$$T_1' = (4, 26/25), T_2' = (5, 26/25), T_3' = (6, 26/25), T_4' = (7, 26/25).$$

Per verificare che il sistema di task non è più schedulabile il test iperbolico, che certamente otterrà un valore della produttoria maggiore di due, non è di alcun aiuto: infatti tale test esprime una condizione non necessaria.

Dobbiamo dunque svolgere l'analisi utilizzando le funzioni di tempo necessario.

- Task T_1': $w_1(t) = 26/25 \leq 4 \quad \Rightarrow$ ok.
- Task T_2': $w_2(t) = 26/25 \cdot (1 + \lceil t/4 \rceil)$:
  $w_2(26/25) = 52/25, \quad w_2(52/25) = 52/25 < 5 \Rightarrow$ ok.
- Task T_3': $w_3(t) = 26/25 \cdot (1 + \lceil t/4 \rceil + \lceil t/5 \rceil)$:
  $w_3(26/25) = 78/25, \quad w_3(78/25) = 78/25 < 6 \Rightarrow$ ok.
- Task T_4': $w_4(t) = 26/25 \cdot (1 + \lceil t/4 \rceil + \lceil t/5 \rceil + \lceil t/6 \rceil)$:
  $w_4(26/25) = 104/25, \quad w_4(104/25) = 26/5, \quad w_4(26/5) = 156/25,$
  $w_4(156/25) = 182/25 = 7 \cdot 26/25 > 7 \Rightarrow$ NO!

Poiché T_4 può mancare la propria scadenza, il sistema di task non è più schedulabile.

(c) *Con riferimento al punto (b): si supponga che, mentre i periodi dei task T_1, T_3 e T_4 non possano essere variati, sia possibile invece aumentare leggermente il periodo del task T_2, fino al valore limite 6. Determinare analiticamente un valore per il periodo di T_2 che garantisca la schedulabilità del sistema.*

Un metodo per determinare un valore x per il periodo di T_2' che certamente garantisce la schedulabilità del sistema consiste nell'imporre la condizione data dal test iperbolico:

$$\left(1 + \frac{26}{25 \cdot 4}\right) \cdot \left(1 + \frac{26}{25 \cdot x}\right) \cdot \left(1 + \frac{26}{25 \cdot 6}\right) \cdot \left(1 + \frac{26}{25 \cdot 7}\right) \leq 2.$$

Risolvendo rispetto ad x si trova:

$$x \geq \frac{344916}{58975} \approx 5{,}848.$$

Si osservi che, poiché il test iperbolico è una condizione solo necessaria e non sufficiente, il valore ottenuto potrebbe non essere il minimo periodo che garantisce la schedulabilità.

Per determinare il minimo valore del periodo che consente la schedulabilità ragioniamo sulla analisi svolta utilizzando le funzioni di tempo necessario. Poiché il periodo del task T_2' non può diventare maggiore del periodo del task T_3', le priorità relative dei task non cambiano. In particolare, se x è il valore del periodo di T_2', con $5 < x \leq 6$, possiamo concludere immediatamente che i task T_1', T_2' e T_3' continuano ad essere schedulabili.

Per garantire la schedulabilità del task T_4' è necessario che l'equazione $w_4(t) = t$, con $w_4(t) = 26/25 \cdot (1 + \lceil t/4 \rceil + \lceil t/x \rceil + \lceil t/6 \rceil)$ possieda una soluzione $t \leq 7$. Poiché il valore di $w_4(t)$ tende a crescere al diminuire di x, e poiché stiamo cercando il valore minimo per x, cominciamo assumendo che il valore del punto fisso sia vicino al limite superiore 7. In particolare supponiamo che $6 < t \leq 7$. In questo caso si ha:

$$w_4(t) = \frac{26}{25} \cdot \left(1 + 2 + \left\lceil \frac{t}{x} \right\rceil + 2\right) = \frac{26}{5} + \frac{26}{25} \left\lceil \frac{t}{x} \right\rceil .$$

Poiché vale $5 < x \leq 6$, $\lceil t/x \rceil = 2$, ed il punto fisso è $t = 26/5 + 52/25 = 182/25 > 7$. Perciò nessun valore di x garantisce l'esistenza di un punto fisso nell'intervallo tra 6 e 7.

Assumiamo quindi che il punto fisso cada nell'intervallo tra 5 e 6, ossia che $5 < t \leq 6$. In questo caso si ha:

$$w_4(t) = \frac{26}{25} \cdot \left(1 + 2 + \left\lceil \frac{t}{x} \right\rceil + 1\right) = \frac{104}{25} + \frac{26}{25} \left\lceil \frac{t}{x} \right\rceil .$$

Se ora fosse $t > x$, $\lceil t/x \rceil = 2$ ed il punto fisso si ha per $t = 104/25 + 52/25 = 156/25 > 6$. Dunque dobbiamo imporre che $t \leq x$, ossia $\lceil t/x \rceil = 1$. Allora il punto fisso si ha per $t = 104/25 + 26/25 = 26/5$, che in effetti è compreso tra 5 e 6.

Concludendo, ogni valore del periodo di T_2' compreso nell'intervallo $(26/5, 6]$ rende il sistema di task schedulabile con RM. Il minimo valore del periodo che consente la schedulabilità è $x = 26/5 = 5{,}2$.

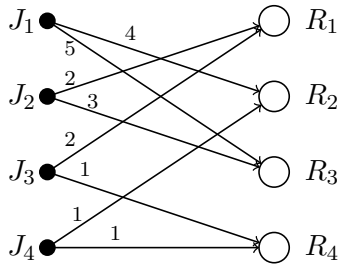
Esercizio 3. *Un sistema deve eseguire quattro job $J_1, \ldots, J_4$, con J_i di priorità maggiore di J_k se $i < k$. I job utilizzano quattro risorse condivise $R_1, \ldots R_4$, e le relative sezioni critiche sono: $J_1 : [R_2, 4][R_3, 5]$, $J_2 : [R_3, 3][R_1, 2]$, $J_3 : [R_1, 2\,[R_4, 1]\,]$, $J_4 : [R_2, 1][R_4, 1]$.*

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

Il massimo tempo di blocco per conflitto di risorse $b_i(\mathrm{rc})$ 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:

$$b_1(\mathrm{rc}) = 3; \qquad b_2(\mathrm{rc}) = 2; \qquad b_3(\mathrm{rc}) = 1; \qquad b_4(\mathrm{rc}) = 0.$$

(b) *Quali sono i tempi massimi di blocco per conflitto di risorse utilizzando il protocollo priority ceiling?*



B_d	J_2	J_3	J_4
J_1	3		1
J_2	$*$	2	
J_3		$*$	1

B_i	J_2	J_3	J_4
J_1			
J_2	$*$		1
J_3		$*$	1

$B_c \equiv B_i$

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:

$$b_1(\text{rc}) = 3; \quad b_2(\text{rc}) = 2; \quad b_3(\text{rc}) = 1; \quad b_4(\text{rc}) = 0.$$