Sistemi Embedded e Real-time (M. Cesati)

Compito scritto del 19 settembre 2012

Esercizio 1. Si consideri il seguente sistema di task periodici : $T_1 = (3, 2, \frac{1}{2}, 1)$, $T_2 = (2, 2, \frac{1}{2}, 1)$, $T_3 = (1, 2, \frac{1}{2}, 2)$, $T_4 = (0, 3, \frac{1}{2}, 1)$. 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_3 passi da 2 a 1, ossia che $T_3 = (1, 2, \frac{1}{2}, 1)$: esiste una schedulazione ciclica strutturata con la dimensione del frame considerata per il punto (a)? Giustificare la risposta.

Esercizio 2. Si consideri un sistema di tre task periodici con scadenze uguali al periodo così caratterizzati:

	T_1	T_2	T_3	
p_i	3	5	7	(periodo)
e_i	1	1	1	(tempo d'esecuzione)
K_i	1	0	1	(numero di auto-sospensioni)
x_i	0.1	0	0.1	(tempo max auto-sospensione)
θ_i	0.1	0.2	0.1	(tempo max non interrompibilità)

I task T_1 e T_3 accedono alla stessa risorsa condivisa R, per un periodo di tempo massimo rispettivamente pari a 0.1 e 0.2. Il task T_2 non fa uso di risorse condivise. Viene utilizzato un algoritmo "ceiling-priority" per regolare gli accessi alla risorsa condivisa.

(a) Supponendo che venga utilizzato uno scheduler EDF con cambio di contesto di costo $CS = 0.05$, determinare analiticamente se il sistema è schedulabile su un singolo processore.

(b) Ripetere l'analisi precedente supponendo che lo scheduler EDF sia invocato periodicamente con periodo $p_0 = 0.16$, che il suo overhead sia pari a $e_0 = 0.01$ per ogni invocazione, e che per rendere eseguibile un job si impieghi un tempo massimo pari a $CS_0 = 0.02$.

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 priority inheritance?

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

Sistemi Embedded e Real-time (M. Cesati)

Soluzioni del compito scritto del 19 settembre 2012

Esercizio 1. *Si consideri il seguente sistema di task periodici :* $T_1 = (3, 2, \frac{1}{2}, 1)$, $T_2 = (2, 2, \frac{1}{2}, 1)$, $T_3 = (1, 2, \frac{1}{2}, 2)$, $T_4 = (0, 3, \frac{1}{2}, 1)$. *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:

- Vincolo sulle fasi dei task: f deve dividere 3, 2 e 1, quindi $f = 1$
- Vincolo sui tempi d'esecuzione dei job:

$$f \geq \max\{1/2, 1/2, 1/2, 1/2\} = 1/2 \Rightarrow f \geq 1$$

- Vincolo sulla divisibilità della lunghezza dell'iperperiodo ($\text{mcm}\{2, 2, 2, 3\} = 6$):

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

  Quindi combinando con le condizioni precedenti l'unico valore ammissibile è $f = 1$.

- Vincolo sul task T_1 ($2f - \gcd\{2, f\} \leq 1$):

$$2 \cdot 1 - \gcd\{2, 1\} = 1 \quad \leq 1 \quad \text{ok}$$

- Vincolo sul task T_2 ($2f - \gcd\{2, f\} \leq 1$): identico al task T_1, ossia

$$2 \cdot 1 - \gcd\{2, 1\} = 1 \quad \leq 1 \quad \text{ok}$$

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

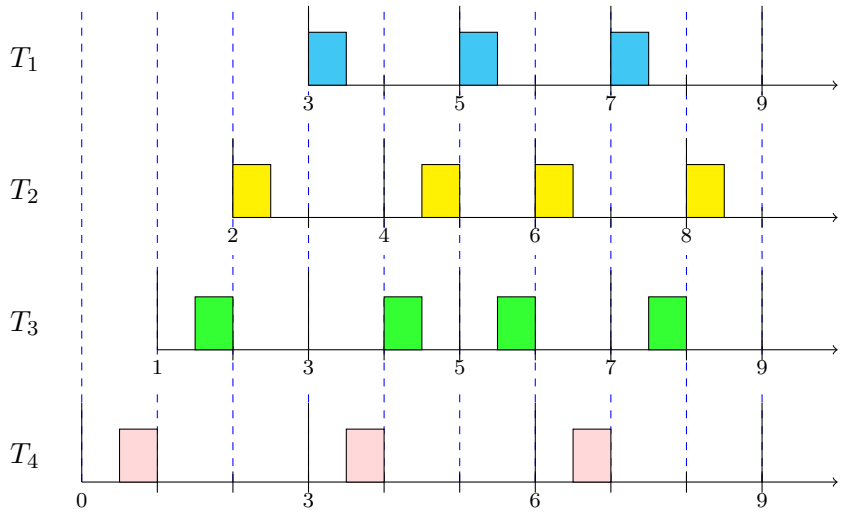
$$2 \cdot 1 - \gcd\{2, 1\} = 1 \quad \leq 2 \quad \text{ok}$$

- Vincolo sul task T_4 ($2f - \gcd\{3, f\} \leq 1$):

$$2 \cdot 1 - \gcd\{3, 1\} = 1 \quad \leq 1 \quad \text{ok}$$

L'unica dimensione ammissibile per il frame dello scheduler ciciclo strutturato è $f = 1$.

Non è difficile determinare una schedulazione ciclica strutturata per $f = 1$. Ad esempio:



La schedulazione tra gli istanti 3 e 9 può essere ripetuta all'infinito. Infatti la situazione agli istanti 3 e 9 è identica: tutti i job precedentemente rilasciati sono stati eseguiti; sono rilasciati job dei task T_1, T_3 e T_4; il successivo job di T_2 verrà rilasciato tra una unità di tempo.

(b) *Si supponga che la scadenza relativa del task T_3 passi da 2 a 1, ossia che $T_3 = (1, 2, \frac{1}{2}, 1)$: esiste una schedulazione ciclica strutturata con la dimensione del frame considerata per il punto (a)? Giustificare la risposta.*

La non esistenza di una schedulazione ciclica strutturata può essere facilmente desunta considerando che all'istante 3 vengono rilasciati i job dei task T_1, T_3 e T_4, tutti con scadenza relativa 1. La somma dei tempi di esecuzione di questi job è 3/2, che è superiore alla scadenza relativa dei job.

Esercizio 2. *Si consideri un sistema di tre task periodici con scadenze uguali al periodo così caratterizzati:*

	T_1	T_2	T_3	
p_i	3	5	7	*(periodo)*
e_i	1	1	1	*(tempo d'esecuzione)*
K_i	1	0	1	*(numero di auto-sospensioni)*
x_i	0.1	0	0.1	*(tempo max auto-sospensione)*
θ_i	0.1	0.2	0.1	*(tempo max non interrompibilità)*

I task T_1 e T_3 accedono alla stessa risorsa condivisa R, per un periodo di tempo massimo rispettivamente pari a 0.1 *e* 0.2. *Il task T_2 non fa uso di risorse condivise. Viene utilizzato un algoritmo "ceiling-priority" per regolare gli accessi alla risorsa condivisa.*

(a) *Supponendo che venga utilizzato uno scheduler EDF con cambio di contesto di costo $CS = 0.05$, determinare analiticamente se 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.

Si osserva che un job del task T_3 può bloccare direttamente un job del task T_1 per 0.2 unità di tempo; a causa del teorema di Baker, invece, un job di T_1 non può bloccare un job di T_3 in quanto la scadenza relativa di T_1 è inferiore a quella di T_3. Il teorema di Baker continua a valere anche in presenza di auto-sospensione se si utilizza il protocollo "ceiling-priority", in quanto se un job si auto-sospende avendo acquisito una risorsa condivisa allora nessun job di priorità inferiore a quello auto-sospeso può essere posto in esecuzione.

Inoltre, un job di T_2 può essere bloccato indirettamente da un job di T_3 per un tempo massimo di 0.2 unità di tempo a causa del protocollo "ceiling-priority".

Per tenere in considerazione l'overhead dello scheduler e dei cambi di contesto, aumentiamo i tempi di esecuzione dei job. Si osserva che, utilizzando il protocollo "ceiling-priority", quando un job richiede una risorsa questa è sempre libera, quindi il job non viene mai bloccato. In altri termini, il job non può essere sospeso a causa di un blocco su una risorsa. D'altra parte, nel protocollo "ceiling-priority" un job può essere ritardato al suo avvio, ma questo meccanismo è realizzato semplicemente confrontando la sua priorità con la priorità corrente del job in esecuzione. In conclusione, ciascun job di ogni task avrà un overhead dovuto soltanto al numero di auto-sospensioni più il rilascio iniziale:

- $e'_1 = e_1 + 2\,(K_1 + 1)\,CS = 1.2$
- $e'_2 = e_2 + 2\,(K_2 + 1)\,CS = 1.1$

- $e'_3 = e_3 + 2\,(K_3 + 1)\,CS = 1.2$

Siamo dunque in grado di determinare i tempi massimi di blocco di ciascun task:

	T_1	T_2	T_3
$b_i(ss) = x_i + \sum_{k=1}^{i-1} \min\{e'_k, x_k\}$	0.1	0.1	0.2
$b_i(np) = \max_{i<k\leq 3}\{\theta_k\}$	0.2	0.1	0.0
$b_i(rc) = \max_{1\leq k\leq 3}\{B_d(i,k), B_i(i,k)\}$	0.2	0.2	0.0
$b_i = b_i(ss) + (K_i + 1)\,(b_i(np) + b_i(rc))$	0.9	0.4	0.2

Verifichiamo la condizione di schedulabilità per i tre job. Il fattore di utilizzo del sistema è

$$U'_T = \frac{e'_1}{p_1} + \frac{e'_2}{p_2} + \frac{e'_3}{p_3} = \frac{6}{15} + \frac{11}{50} + \frac{12}{70} = \frac{277}{350}.$$

- Schedulabilità di T_1: $\quad U'_T + \dfrac{b_1}{p_1} = \dfrac{277}{350} + \dfrac{3}{10} = \dfrac{191}{175} > 1 \quad \Longrightarrow$ NO
- Schedulabilità di T_2: $\quad U'_T + \dfrac{b_2}{p_2} = \dfrac{277}{350} + \dfrac{2}{25} = \dfrac{61}{70} < 1 \quad \Longrightarrow$ OK
- Schedulabilità di T_3: $\quad U'_T + \dfrac{b_3}{p_3} = \dfrac{277}{350} + \dfrac{1}{35} = \dfrac{41}{50} < 1 \quad \Longrightarrow$ OK

Il sistema di task non è pertanto schedulabile con EDF.

(b) *Ripetere l'analisi precedente supponendo che lo scheduler EDF sia invocato periodicamente con periodo $p_0 = 0.16$, che il suo overhead sia pari a $e_0 = 0.01$ per ogni invocazione, e che per rendere eseguibile un job si impieghi un tempo massimo pari a $CS_0 = 0.02$.*

I tempi di esecuzione e di blocco dei task devono essere aumentati per tenere in considerazione il fatto che lo scheduler è eseguito periodicamente. In particolare:

	T_1	T_2	T_3
$e_i'' = e_i' + (K_i + 1)\, CS_0$	1.24	1.12	1.24
$b_i(np)' = \left(\left\lceil \max_{i<k\leq 3} \theta_k/p_0 \right\rceil + 1 \right) p_0$	0.48	0.32	0.16
$b_i' = b_i(ss) + (K_i + 1)\,(b_i(np)' + b_i(rc))$	1.46	0.62	0.52

Verifichiamo la condizione di schedulabilità per i tre job. Il fattore di utilizzo del sistema è

$$U_T'' = \frac{e_0}{p_0} + \frac{e_1''}{p_1} + \frac{e_2''}{p_2} + \frac{e_3''}{p_3} = \frac{1}{16} + \frac{277}{350} + \frac{1}{75} + \frac{1}{250} + \frac{1}{175} = \frac{36833}{42000}.$$

- Schedulabilità di T_1:

$$U_T'' + \frac{b_1'}{p_1} = \frac{36833}{42000} + \frac{73}{150} = \frac{57273}{42000} > 1 \qquad \Longrightarrow \text{NO}$$

- Schedulabilità di T_2:

$$U_T'' + \frac{b_2'}{p_2} = \frac{36833}{42000} + \frac{31}{250} = \frac{42041}{42000} > 1 \qquad \Longrightarrow \text{NO}$$

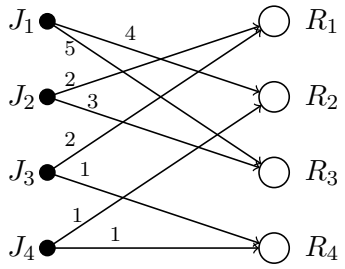
- Schedulabilità di T_3:

$$U_T'' + \frac{b_3'}{p_3} = \frac{36833}{42000} + \frac{13}{175} = \frac{39953}{42000} < 1 \qquad \Longrightarrow \text{OK}$$

Come era lecito aspettarsi fin dall'inizio, il sistema di task non è schedulabile.

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 priority inheritance?*



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

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:

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

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

Nel caso peggiore, la durata dei tempi di blocco per il protocollo stack-based priority-ceiling coincide con quella del protocollo priority-ceiling. Poiché non esistono job di pari priorità, i tempi di blocco per il protocollo priority-ceiling sono uguali a quelli determinati al punto (a).