Sistemi Embedded e Real-time (M. Cesati)

Compito scritto del 22 febbraio 2012

Esercizio 1. Si consideri il seguente sistema di task periodici con job non interrompibili: $T_1 = (0, 2, 1, 5)$, $T_2 = (6, 3, 1, 4)$, $T_3 = (3, 9, 1, 6)$.

(a) Determinare una schedulazione ciclica strutturata per il sistema di task che minimizza l'overhead dello scheduler.

(b) Al tempo $t = 10$ viene generato un job aperiodico di durata $e = 2$ e scadenza assoluta $t = 40$. È possibile accettare il job assumendo che non esistano altri job aperiodici da eseguire? 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	2	4	8	(periodo)
e_i	0.5	1	2	(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 "priority-inheritance" per regolare gli accessi alla risorsa condivisa.

(a) Supponendo che venga utilizzato uno scheduler RM 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 RM 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. In fase di progettazione di un sistema real-time multiprocessore si prevede di dover eseguire al massimo 30 task periodici $T_i = (2, 1/5)$ e 40 task periodici $T'_i = (3, 1/5)$.

(a) Supponendo di utilizzare uno scheduler partizionato RM-FF, determinare un numero minimo di processori che garantisca il rispetto di tutte le scadenze dei vari task.

(b) Ripetere l'analisi del punto (a) supponendo di utilizzare uno scheduler partizionato EDF-FF.

Sistemi Embedded e Real-time (M. Cesati)

Soluzioni del compito scritto del 22 febbraio 2012

Esercizio 1. *Si consideri il seguente sistema di task periodici con job non interrompibili:* $T_1 = (0, 2, 1, 5)$, $T_2 = (6, 3, 1, 4)$, $T_3 = (3, 9, 1, 6)$.

(a) *Determinare una schedulazione ciclica strutturata per il sistema di task che minimizza l'overhead dello scheduler.*

Determiniamo la dimensione del frame più opportuna:

- Vincolo sulle fasi dei task: f deve dividere esattamente la fase dei task T_2 e T_3, perciò $f \in \{1, 3\}$.
- Vincolo sui tempi d'esecuzione dei job:

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

- Vincolo sulla divisibilità della lunghezza dell'iperperiodo ($\text{mcm}\{2, 3, 9\} = 18$):

$$f \in \{1, 2, 3, 6, 9, 18\} \cap \{1, 3\} = \{1, 3\}$$

- Vincolo sul task T_1 ($2f - \gcd\{2, f\} \leq 5$):

$$2 \cdot 1 - \gcd\{2, 1\} = 1 \leq 5$$
$$2 \cdot 3 - \gcd\{2, 3\} = 5 \leq 5$$

- Vincolo sul task T_2 ($2f - \gcd\{3, f\} \leq 4$):

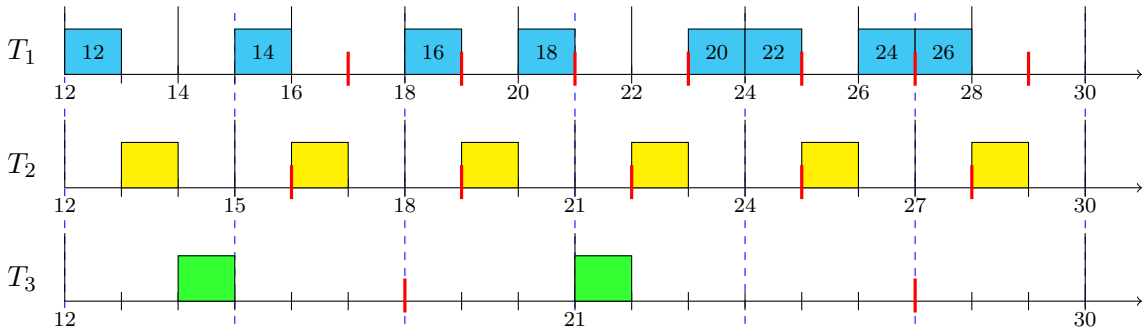
$$2 \cdot 1 - \gcd\{3, 1\} = 1 \leq 4$$
$$2 \cdot 3 - \gcd\{3, 3\} = 3 \leq 4$$

- Vincolo sul task T_3 ($2f - \gcd\{9, f\} \leq 6$):

$$2 \cdot 1 - \gcd\{9, 1\} = 1 \leq 6$$
$$2 \cdot 3 - \gcd\{9, 3\} = 3 \leq 6$$

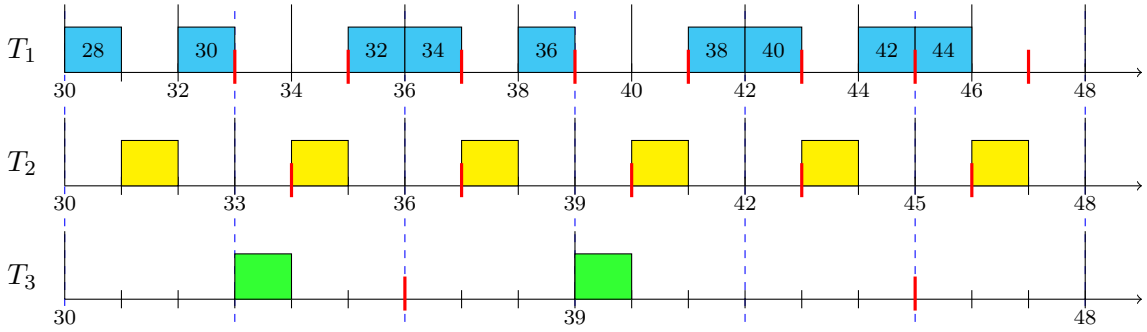
Le dimensioni intere ammissibili per il frame sono dunque $f = 1$ e $f = 3$. La dimensione maggiore minimizza l'overhead globale dello scheduler, quindi scegliamo come dimensione del frame il valore tre.

Verifichiamo l'esistenza di una schedulazione ciclica strutturata per l'insieme di task con $f = 3$. Notiamo che, poiché T_2 ha fase 6 e T_3 ha fase 3, non possiamo limitarci a considerare un intervallo lungo un iperperiodo iniziante dall'istante 0. È facile verificare che il primo istante in cui tutti i task sono attivi ed in fase è $t = 12$, perciò cerchiamo di determinare una schedulazione ciclica nell'intervallo tra 12 e 30 (poiché l'iperperiodo è lungo 18).



(Per maggior chiarezza è stato indicato in ciascun job di T_1 il corrispondente istante di rilascio.)

Si noti come nell'iperperiodo non sia stato possibile schedulare il job di T_1 rilasciato a $t = 28$ e avente scadenza assoluta a $t = 28+5 = 33$: deve essere schedulato nell'iperperiodo successivo:



All'istante $t = 48$ si è nella situazione identica a $t = 30$: tutti i job rilasciati sono stati eseguiti, tranne un job di T_1 che ha scadenza assoluta a $t = 51$. La schedulazione tra 30 e 48 può perciò essere ripetuta all'infinito.

(b) *Al tempo $t = 10$ viene generato un job aperiodico di durata $e = 2$ e scadenza assoluta $t = 40$. È possibile accettare il job assumendo che non esistano altri job aperiodici da eseguire? Giustificare la risposta.*

In generale la schedulazione ciclica strutturata lascia libera una sola unità di tempo di CPU in un intero iperperiodo. Ciò risulta evidente anche considerando il fattore di utilizzazione del sistema di task:

$$U_T = \frac{1}{2} + \frac{1}{3} + \frac{1}{9} = \frac{17}{18}.$$

Il job aperiodico arriva all'istante 10 e l'evento viene riconosciuto dallo scheduler all'istante $t = 12$. Nella schedulazione ciclica costruita al punto (a) la CPU sarà disponibile in slot da 1 unità di tempo agli istanti $t = 18 \cdot k + 11$, per k intero positivo, ossia $t = 11, 29, 47, \ldots$ Si può concludere pertanto che lo scheduler (che interviene all'istante 12) rifiuta il job aperiodico in quanto esso non può completare le sue due unità di tempo di esecuzione entro la scadenza posta a $t = 40$.

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	2	4	8	(periodo)
e_i	0.5	1	2	(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 "priority-inheritance" per regolare gli accessi alla risorsa condivisa.

(a) *Supponendo che venga utilizzato uno scheduler RM 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 il task T_3 può bloccare direttamente T_1 per 0.2 unità di tempo; inoltre T_3 può bloccare indirettamente anche T_2 per 0.2 unità di tempo a causa del protocollo "priority inheritance".

Per tenere in considerazione l'overhead dello scheduler e dei cambi di contesto, aumentiamo i tempi di esecuzione dei job:

- $e'_1 = e_1 + 4\,(K_1 + 1)\,CS = 9/10$ (accede a risorse condivise)
- $e'_2 = e_2 + 2\,(K_2 + 1)\,CS = 1 + 1/10$ (non accede a risorse condivise)
- $e'_3 = e_3 + 4\,(K_3 + 1)\,CS = 2 + 2/5$ (accede a risorse condivise)

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

	T_1	T_2	T_3
$b_i(ss) = x_i + \sum_{k=1}^{i-1} \min\{e'_k, x_k\}$	1/10	1/10	1/5
$b_i(np) = \max_{i<k\leq 3}\{\theta_k\}$	1/5	1/10	0
$b_i(rc) = \max_{1\leq k\leq 3}\{B_d(i,k), B_i(i,k)\}$	1/5	1/5	0
$b_i = b_i(ss) + (K_i + 1)\,(b_i(np) + b_i(rc))$	9/10	2/5	1/5

Verifichiamo la condizione di schedulabilità per i tre job. Poiché i task sono armonici (ciascun periodo è un multiplo intero di un periodo inferiore) la soglia di schedulabilità è pari a 1.

- Schedulabilità di T_1: $\dfrac{e'_1 + b_1}{p_1} = \dfrac{9}{10} \leq 1 \quad \Longrightarrow$ OK
- Schedulabilità di T_2: $\dfrac{e'_1}{p_1} + \dfrac{e'_2 + b_2}{p_2} = \dfrac{33}{40} \leq 1 \quad \Longrightarrow$ OK
- Schedulabilità di T_3: $\dfrac{e'_1}{p_1} + \dfrac{e'_2}{p_2} + \dfrac{e'_3 + b_3}{p_3} = 1 + \dfrac{1}{20} > 1 \quad \Longrightarrow$ no

Consideriamo il task T_3: poiché la somma del fattore di utilizzazione e del tempo di blocco rapportato al periodo supera l'unità, si può già concludere che è impossibile garantire il rispetto della scadenza di T_3. Applichiamo comunque il test di schedulabilità al task T_3.

La funzione di tempo necessario per T_3 è:

$$w_3(t) = e'_3 + b_3 + \lceil t/p_1 \rceil e'_1 + \lceil t/p_2 \rceil e'_2 = 13/5 + \lceil t/2 \rceil \cdot 9/10 + \lceil t/4 \rceil \cdot 11/10$$

Cerchiamo il punto fisso dell'equazione iterativa $w_3(t) = t$:

$$\begin{aligned} w_3(12/5) = &\quad 13/5 + \lceil 6/5 \rceil\, 9/10 + \lceil 3/5 \rceil\, 11/10 &&= 11/2 \\ w_3(11/2) = &\quad 13/5 + \lceil 11/4 \rceil\, 9/10 + \lceil 11/8 \rceil\, 11/10 &&= 15/2 \\ w_3(15/2) = &\quad 13/5 + \lceil 15/4 \rceil\, 9/10 + \lceil 15/8 \rceil\, 11/10 &&= 8 + 2/5 \end{aligned}$$

Poiché l'istante 8.4 è oltre la scadenza assoluta del job (8), il task T_3 non è schedulabile.

(b) *Ripetere l'analisi precedente supponendo che lo scheduler RM 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
$e''_i = e'_i + (K_i + 1)\, CS_0$	47/50	28/25	61/25
$b_i(np)' = \left(\left\lceil \max_{i<k\leq 3} \theta_k/p_0 \right\rceil + 1 \right) p_0$	12/25	8/25	4/25
$b'_i = b_i(ss) + (K_i + 1)\,(b_i(np)' + b_i(rc))$	73/50	31/50	13/25

Si noti che le quantità $b_i(ss)$ non possono variare rispetto al punto (a) in quanto per ogni task valeva $e'_i > x_i$.

- Schedulabilità di T_1:

$$\frac{e_0}{p_0} + \frac{e''_1 + b'_1}{p_1} = \frac{1}{16} + \frac{6}{5} = 1 + \frac{21}{80} > 1 \qquad \Longrightarrow \text{Non fattibile}$$

- Schedulabilità di T_2:

$$\frac{e_0}{p_0} + \frac{e_1''}{p_1} + \frac{e_2'' + b_2'}{p_2} = \frac{1}{16} + \frac{47}{100} + \frac{87}{200} = \frac{378}{400} \leq 1 \qquad \Longrightarrow \text{OK}$$

- Schedulabilità di T_3:

$$\frac{e_0}{p_0} + \frac{e_1''}{p_1} + \frac{e_2''}{p_2} + \frac{e_3'' + b_3'}{p_3} = \frac{1}{16} + \frac{47}{100} + \frac{7}{25} + \frac{37}{100} = 1 + \frac{73}{400} > 1 \qquad \Longrightarrow \text{Non fattibile}$$

Consideriamo il task T_1: poiché la somma del fattore di utilizzazione e del tempo di blocco rapportato al periodo supera l'unità, si può già concludere che è impossibile garantire il rispetto della scadenza di T_1.

Verifichiamo comunque la non schedulabilità di T_1 studiando la sua funzione di tempo necessario. Benché T_1 abbia priorità massima tra i task del sistema dato, per tenere conto dell'overhead dello scheduler periodico dobbiamo aggiungere al sistema tre task di priorità maggiore di T_1: $T_0 = (p_0, e_0)$, $T_{0,2} = (p_2, CS_0)$ e $T_{0,3} = (p_3, CS_0)$. La funzione di tempo necessario di T_1 diventa perciò:

$$\begin{aligned} w_1(t) &= e_1'' + b_1' + \lceil t/p_0 \rceil \; e_0 + \lceil t/p_2 \rceil \; CS_0 + \lceil t/p_3 \rceil \; CS_0 \\ &= 12/5 + \lceil 25\, t/4 \rceil \, /100 + \lceil t/4 \rceil \, /50 + \lceil t/8 \rceil \, /50 \end{aligned}$$

Cerchiamo il punto fisso dell'equazione iterativa $w_1(t) = t$:

$$w_1(47/50) = \quad 12/5 + \lceil 47/8 \rceil \, /100 + \lceil 47/200 \rceil \, /50 + \lceil 47/400 \rceil \, /50 = \quad 2 + 1/2$$

Nel caso peggiore un job del task T_1 si conclude 0.5 unità di tempo oltre la propria scadenza assoluta, quindi il sistema non è fattibile.

Esercizio 3*. In fase di progettazione di un sistema real-time multiprocessore si prevede di dover eseguire al massimo 30 task periodici $T_i = (2, 1/5)$ e 40 task periodici $T_i' = (3, 1/5)$.*

(a) *Supponendo di utilizzare uno scheduler partizionato RM-FF, determinare un numero minimo di processori che garantisca il rispetto di tutte le scadenze dei vari task.*

Il fattore di utilizzazione dell'algoritmo di schedulazione partizionato RM-FF è

$$U_{\text{RM-FF}}(m) = m \cdot \left(\sqrt{2} - 1\right)$$

ove m è il numero di processori nel sistema. I task periodici da eseguire hanno un fattore di utilizzazione pari a

$$U_T = 30 \cdot \frac{1/5}{2} + 40 \cdot \frac{1/5}{3} = 3 + \frac{8}{3} = \frac{17}{3}$$

Se $U_T \leq U_{RM\text{-}FF}(m)$, tutte le scadenze dei task sono rispettate, il che significa che m processori sono sufficienti. Si ottiene perciò:

$$m \geq \frac{17/3}{\sqrt{2}-1} \approx 13.68$$

È quindi necessario includere nel sistema almeno 14 processori.

(b) *Ripetere l'analisi del punto (a) supponendo di utilizzare uno scheduler partizionato EDF-FF.*

Il fattore di utilizzazione dell'algoritmo di schedulazione partizionato EDF-FF è

$$U_{EDF\text{-}FF}(m, \beta) = \frac{\beta \cdot m + 1}{\beta + 1}$$

ove m è il numero di processori e $\beta = \lfloor 1/\max_k\{e_k/p_k\} \rfloor$.

Nel caso in esame, il massimo fattore di utilizzazione dei task è $1/10$, quindi $\beta = 10$. Di conseguenza, un numero sufficiente di processori per garantire la fattibilità del sistema di task si ottiene risolvendo $U_T \leq U_{EDF\text{-}FF}(m, 10)$, ossia:

$$\frac{17}{3} \leq \frac{10\,m + 1}{10 + 1}$$

che equivale a

$$m \geq \frac{11 \cdot 17/3 - 1}{10} = 6 + \frac{2}{15}.$$

È quindi necessario includere nel sistema almeno 7 processori.