Sistemi Embedded e Real-time (M. Cesati)

Compito scritto del 9 settembre 2013

Esercizio 1. Si consideri il seguente sistema di task periodici : $T_1 = (6, 6, 2, 6)$, $T_2 = (0, 15, 3, 15)$, $T_3 = (24, 5, 2, 4)$.

(a) Nell'ipotesi che i job siano non interrompibili, verificare che non esiste alcuna dimensione ammissibile per il frame di una schedulazione ciclica strutturata.

(b) Nell'ipotesi che i job siano interrompibili, determinare una schedulazione ciclica strutturata che minimizza l'overhead dello scheduler ed il numero di frammenti dei job.

Esercizio 2. Si consideri un sistema di task periodici sempre interrompibili e che non si auto-sospendono: $T_1 = (3, \frac{7}{5})$, $T_2 = (6, 1)$, $T_3 = (12, \frac{7}{5})$.

(a) Supponendo che i task sono indipendenti, che venga utilizzato uno scheduler RM, e che l'overhead dello scheduler e del cambio di contesto sia pari a $\mathrm{CS} = \frac{1}{10}$, determinare analiticamente se il sistema è schedulabile su un singolo processore.

(b) Ripetere l'analisi precedente supponendo anche che i job dei task T_1 e T_3 utilizzano una risorsa condivisa ciascuno per un tempo massimo di $\frac{7}{5}$ unità di tempo, e che la risorsa sia controllata tramite il protocollo *priority ceiling*.

Esercizio 3. Un sistema multiprocessore è dotato di $m = 32$ processori identici. Su di esso deve essere eseguito in ogni unità di tempo un lavoro totale stimato in 26 unità di tempo. Il carico è rappresentato da un certo numero di task periodici di frequenza minima pari a 1 schedulati per mezzo dell'algoritmo EDF-FF.

È possibile garantire la schedulabilità del sistema suddividendo in modo appropriato il carico tra 100 task periodici (non necessariamente identici)? Giustificare la risposta.

Sistemi Embedded e Real-time (M. Cesati)

Soluzioni del compito scritto del 9 settembre 2013

Esercizio 1. *Si consideri il seguente sistema di task periodici :* $T_1 = (6, 6, 2, 6)$, $T_2 = (0, 15, 3, 15)$, $T_3 = (24, 5, 2, 4)$.

(a) *Nell'ipotesi che i job siano non interrompibili, verificare che non esiste alcuna dimensione ammissibile per il frame di una schedulazione ciclica strutturata.*

Determiniamo le possibili dimensioni f del frame dello scheduler ciclico:

- I task hanno fasi 6, 0 e 24. Poiché f deve essere un divisore esatto di tutte le fasi si ha:
$$f \in \{1, 2, 3, 6\}\,.$$
- Vincolo sui tempi d'esecuzione dei job:
$$f \geq \max\{2, 3, 2\} = 3 \Rightarrow f \geq 3$$
- Vincolo sulla divisibilità della lunghezza dell'iperperiodo ($\text{mcm}\{6, 15, 5\} = 30$):
$$f \in \{1, 2, 3, 5, 6, 10, 15, 30\}\,.$$
Quindi considerando i vincoli precedenti: $f \in \{3, 6\}$.
- Vincolo sul task T_1 ($2f - \gcd\{6, f\} \leq 6$):
$$\begin{aligned} 2 \cdot 3 - \gcd\{6, 3\} = 3 &\leq 6 \Rightarrow f = 3 \text{ ok} \\ 2 \cdot 6 - \gcd\{6, 6\} = 6 &\leq 6 \Rightarrow f = 6 \text{ ok} \end{aligned}$$
- Vincolo sul task T_2 ($2f - \gcd\{15, f\} \leq 15$):
$$\begin{aligned} 2 \cdot 3 - \gcd\{15, 3\} = 3 &\leq 15 \Rightarrow f = 3 \text{ ok} \\ 2 \cdot 6 - \gcd\{15, 6\} = 9 &\leq 15 \Rightarrow f = 6 \text{ ok} \end{aligned}$$
- Vincolo sul task T_3 ($2f - \gcd\{5, f\} \leq 4$):
$$\begin{aligned} 2 \cdot 3 - \gcd\{5, 3\} = 5 &> 4 \Rightarrow f = 3 \text{ NO} \\ 2 \cdot 6 - \gcd\{5, 6\} = 11 &> 4 \Rightarrow f = 6 \text{ NO} \end{aligned}$$

Si conclude che non esiste alcuna dimensione ammissibile per il frame.

(b) *Nell'ipotesi che i job siano interrompibili, determinare una schedulazione ciclica strutturata che minimizza l'overhead dello scheduler ed il numero di frammenti dei job.*

Avendo la possibilità di spezzare i job, il vincolo sulla dimensione minima del frame viene rilassato, e dunque è possibile prendere in considerazione anche i valori $f \in \{1, 2\}$ precedentemente scartati.

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

$$\begin{aligned} 2 \cdot 1 - \gcd\{6, 1\} = 1 &\leq 6 \Rightarrow f = 1 \text{ ok} \\ 2 \cdot 2 - \gcd\{6, 2\} = 2 &\leq 6 \Rightarrow f = 2 \text{ ok} \end{aligned}$$

- Vincolo sul task T_2 ($2f - \gcd\{15, f\} \leq 15$):

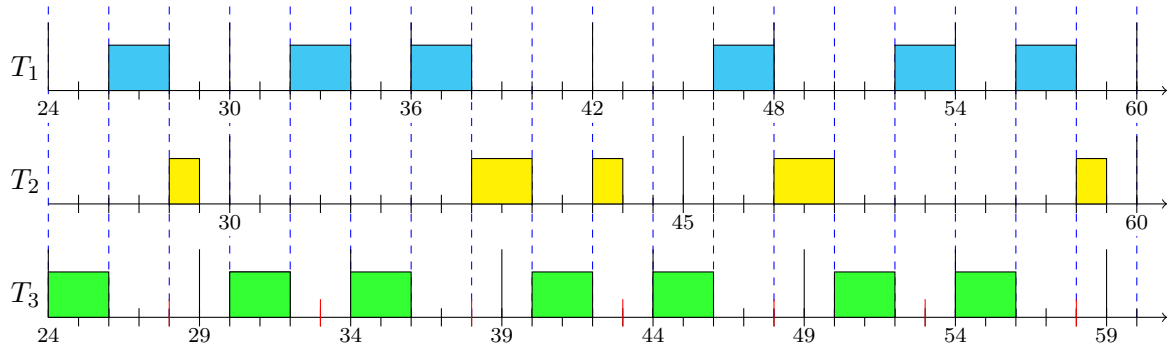
$$\begin{aligned} 2 \cdot 1 - \gcd\{15, 1\} = 1 &\leq 15 \Rightarrow f = 1 \text{ ok} \\ 2 \cdot 2 - \gcd\{15, 2\} = 3 &\leq 15 \Rightarrow f = 2 \text{ ok} \end{aligned}$$

- Vincolo sul task T_3 ($2f - \gcd\{5, f\} \leq 4$):

$$\begin{aligned} 2 \cdot 1 - \gcd\{5, 1\} = 1 &\leq 4 \Rightarrow f = 1 \text{ ok} \\ 2 \cdot 2 - \gcd\{5, 2\} = 3 &\leq 4 \Rightarrow f = 2 \text{ ok} \end{aligned}$$

Il valore $f = 2$ minimizza l'overhead dello scheduler, dunque consideriamo frame di dimensione 2. Per poter schedulare il job del task T_2 all'interno di un frame è necessario spezzarlo. Per avere un numero minimo di frammenti, imponiamo che ciascun job di T_2 sia suddiviso in un frammento di dimensione uguale a due ed in un frammento di dimensione uno.

L'esistenza di una schedulazione ciclica strutturata per $f = 2$ è dimostrata dal seguente diagramma che mostra i blocchi di schedulazione tra gli istanti 24 e 60:



Si noti che l'istante 24 corrisponde al rilascio del primo job di T_3: nei blocchi di schedulazione dei frame precedenti a $t = 24$ sono assenti i job di T_3. Analogamente, nei blocchi di schedulazione dei frame precedenti a $t = 6$ sono assenti anche i job di T_1.

I blocchi di schedulazione dei 15 frame tra $t = 30$ e $t = 60$ possono ripetersi indefinitivamente a partire da $t = 60$. Infatti si consideri che:

- Agli istanti $t = 30$ e $t = 60$ è rilasciato un job di T_1, e tutti i job precedentemente rilasciati sono stati già completati.
- Agli istanti $t = 30$ e $t = 60$ sono rilasciati i due frammenti di T_2, e tutti i frammenti precedentemente rilasciati sono stati già completati.
- Agli istanti $t = 30$ e $t = 60$ risulta rilasciato e non eseguito un job di T_3 con scadenza pari a $t + 3$ (quindi, da eseguire immediatamente).

Pertanto, dato che le condizioni a $t = 60$ sono identiche a quelle di $t = 30$, la schedulazione tra 30 e 60 può ripetersi all'infinito.

Esercizio 2. *Si consideri un sistema di task periodici sempre interrompibili e che non si auto-sospendono:* $T_1 = (3, \frac{7}{5})$, $T_2 = (6, 1)$, $T_3 = (12, \frac{7}{5})$.

(a) *Supponendo che i task sono indipendenti, che venga utilizzato uno scheduler RM, e che l'overhead dello scheduler e del cambio di contesto sia pari a* $CS = \frac{1}{10}$, *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. Notiamo inoltre che i task sono armonici: per ciascuna coppia di task, uno dei periodi è un multiplo dell'altro. Pertanto, l'algoritmo di schedulazione *Rate Monotonic* è ottimale ed il suo fattore di utilizzazione è uguale a uno.

Dobbiamo però tenere conto dell'overhead introdotto dallo scheduler e dai cambi di contesto, valutato pari a CS = 1/10. Ciò può essere facilmente ottenuto aumentando il tempo di esecuzione dei job di ciascun task in accordo alla formula:

$$e'_i = e_i + 2 \cdot (1 + K_i) \cdot \mathrm{CS}$$

Poiché i job in esame non si autosospendono si ottiene:

$$\begin{aligned} e'_1 &= \frac{7}{5} + 2 \cdot \frac{1}{10} = \frac{8}{5} \\ e'_2 &= 1 + 2 \cdot \frac{1}{10} = \frac{6}{5} \\ e'_3 &= \frac{7}{5} + 2 \cdot \frac{1}{10} = \frac{8}{5} \end{aligned}$$

Il fattore di utilizzazione del sistema è dunque:

$$U_T = \frac{8/5}{3} + \frac{6/5}{6} + \frac{8/5}{12} = \frac{8}{15} + \frac{1}{5} + \frac{2}{15} = \frac{13}{15} \leq 1$$

Perciò il sistema di task è schedulabile con RM.

(b) *Ripetere l'analisi precedente supponendo anche che i job dei task T_1 e T_3 utilizzano una risorsa condivisa ciascuno per un tempo massimo di $\frac{7}{5}$ unità di tempo, e che la risorsa sia controllata tramite il protocollo* priority ceiling.

Determiniamo i tempi di blocco dei vari job dovuti all'accesso alla risorsa condivisa.

Poiché T_1 e T_3 accedono alla stessa risorsa, T_3 può bloccare direttamente l'esecuzione di T_1 per un tempo massimo pari a $\frac{7}{5}$ unità di tempo. Inoltre T_3 può bloccare anche T_2 a causa del *priority inheritance* (se T_1 è bloccato da T_3, la priorità di T_3 supera quella di T_2, che quindi è bloccato da T_1), ancora per un tempo massimo pari a $\frac{7}{5}$ unità di tempo. Di conseguenza, per i tempi di blocco totali $b_i = (1 + K_i) \cdot b_i(rc)$ si ottiene:

$$\begin{aligned} b_1 = (1 + K_1) \cdot b_1(rc) &= \frac{7}{5} \\ b_2 = (1 + K_2) \cdot b_2(rc) &= \frac{7}{5} \\ b_3 &= 0 \end{aligned}$$

Per controllare la schedulabilità del sistema possiamo ancora ragionare sul fattore di utilizzo dei task. Poiché però sono presenti tempi di blocco siamo obbligati ad applicare la condizione di schedulabilità a ciascun task T_i singolarmente, in accordo alla formula:

$$\sum_{k=1}^{i} \frac{e_k}{p_k} + \frac{b_i}{p_i} \leq 1$$

Si noti però che i tempi di esecuzione debbono tenere in considerazione l'overhead dello scheduler e del cambio di contesto, e non coincidono con quelli del precedente punto dell'esercizio. Infatti l'accesso alla risorsa condivisa implica la possibilità di bloccare l'esecuzione se questa dovesse essere occupata, con conseguente doppia invocazione dello scheduler e doppio cambio di contesto aggiuntivi. Pertanto la formula per i tempi di esecuzione relativa ai soli task che accedono ad una risorsa condivisa è:

$$e''_i = e_i + 4 \cdot (1 + K_i) \cdot \mathrm{CS}$$

Verifica della schedulabilità di T_1:

Poiché $e''_1 = e_1 + 4 \cdot \mathrm{CS} = 7/5 + 4 \cdot 1/10 = 9/5$, la condizione di schedulabilità è:

$$\frac{e''_1}{p_1} + \frac{b_1}{p_1} = \frac{9/5}{3} + \frac{7/5}{3} = \frac{16}{15} > 1$$

Se ne conclude che il task T_1, e dunque l'intero sistema di task, non è più necessariamente schedulabile.

Esercizio 3. *Un sistema multiprocessore è dotato di* $m = 32$ *processori identici. Su di esso deve essere eseguito in ogni unità di tempo un lavoro totale stimato in* 26 *unità di tempo. Il carico è rappresentato da un certo numero di task periodici di frequenza minima pari a 1 schedulati per mezzo dell'algoritmo EDF-FF.*

È possibile garantire la schedulabilità del sistema suddividendo in modo appropriato il carico tra 100 *task periodici (non necessariamente identici)? Giustificare la risposta.*

Per avere la garanzia della schedulabilità del sistema si deve imporre:

$$U_T \leq U_{\text{EDF-FF}}.$$

Il fattore di utilizzazione dell'algoritmo EDF-FF è

$$U_{\text{EDF-FF}} = \frac{\beta \cdot m + 1}{\beta + 1} \qquad \text{ove} \qquad \beta = \left\lfloor \frac{1}{\max\{u_k\}} \right\rfloor.$$

D'altra parte, il carico di lavoro che il sistema dovrà sostenere è pari a $U_T = 26$. Quindi deve valere la disuguaglianza

$$26 \leq \frac{32\,\beta + 1}{\beta + 1}, \quad \text{ossia} \quad \beta \geq 4.$$

Consideriamo ora che il carico di lavoro entro un intervallo di una unità di tempo è pari a 26, e deve essere eseguito da 100 task periodici con periodo massimo pari a 1 (quindi tutti rilasciati ed eseguiti entro l'intervallo di tempo). Ciò significa che almeno uno dei task periodici deve avere un fattore di utilizzazione non minore di 26/100. Infatti in caso contrario si avrebbe:

$$\sum_{k=1}^{100} u_k < 100 \cdot 26/100 = 26, \qquad \text{cioè} \quad U_T < 26.$$

Di conseguenza si ha che:

$$\max\{u_k\} \geq 26/100 \quad \Rightarrow \quad 1/\max\{u_k\} \leq 100/26 < 3{,}9$$

e dunque

$$\beta = \left\lfloor \frac{1}{\max\{u_k\}} \right\rfloor \leq \frac{1}{\max\{u_k\}} < 3{,}9 \quad \Rightarrow \quad \beta \leq 3.$$

La conclusione è che non è possibile garantire la schedulabilità del sistema in quanto la condizione $\beta \geq 4$ non è soddisfatta.