Sistemi Embedded e Real-time (M. Cesati)

Compito scritto del 17 luglio 2009

Esercizio 1. Il seguente sistema di task periodici è schedulato su un processore con un algoritmo "cyclic schedule": $T_1 = (4, 1, 6)$, $T_2 = (6, 1)$, $T_3 = (8, 1)$, $T_4 = (8, 3, 4)$.

(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).

(c) Assumendo che la schedulazione determinata in (b) si ripeta all'infinito, determinare il tempo di risposta di un job aperiodico rilasciato a $t = 2$ con tempo d'esecuzione $e = 4$ sia nel caso in cui si utilizzi *slack stealing* che nel caso in cui non lo si utilizzi.

Esercizio 2. Un server procastinabile con periodo $p_s = 4$, budget $e_s = 1$ e fase indeterminata è schedulato su un singolo processore insieme a due task periodici, indipendenti e interrompibili: $T_1 = (5, 1)$ e $T_2 = (10, 3)$.

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

(b) Indicare un esempio di schedulazione RM in cui il task T_2 presenta il peggiore tempo di risposta possibile. In quali condizioni si verifica?

Esercizio 3. Un sistema è costituito da tre task periodici con fasi indeterminate $T_1 = (6, 2)$, $T_2 = (19, 1)$ e $T_3 = (20, 2)$. Ciascun job dei tre task è sempre interrompibile e si auto-sospende al massimo una volta; il periodo di sospensione dura al massimo 2 unità di tempo. I task utilizzano due risorse condivise R_1 e R_2 come segue: $T_1 : [R_1, 1]$, $T_2 : [R_2, 1]$, $T_3 : [R_2, 2\,[R_1, 1]\,]$.

Determinare se l'insieme di task è schedulabile su un singolo processore con RM assumendo che lo scheduler abbia overhead trascurabile e che l'accesso alle risorse condivise sia controllato dal protocollo *priority ceiling*.

Sistemi Embedded e Real-time (M. Cesati)

Soluzioni del compito scritto del 17 luglio 2009

Esercizio 1. *Il seguente sistema di task periodici è schedulato su un processore con un algoritmo "cyclic schedule":* $T_1 = (4, 1, 6)$, $T_2 = (6, 1)$, $T_3 = (8, 1)$, $T_4 = (8, 3, 4)$.

(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:

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

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

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

- Vincolo sul task T_1 ($2f - \gcd\{4, f\} \leq 6$):

$$\begin{array}{ll} 2 \cdot 3 - \gcd\{4, 3\} = 5 \leq 6 & \\ 2 \cdot 4 - \gcd\{4, 4\} = 4 \leq 6 & \\ 2 \cdot 6 - \gcd\{4, 6\} = 10 > 6 & \Rightarrow f \neq 6 \\ 2 \cdot x - \gcd\{4, x\} > 6 \text{ if } x > 6 & \Rightarrow f \leq 6 \end{array}$$

- Vincolo sul task T_2 ($2f - \gcd\{6, f\} \leq 6$):

$$\begin{array}{l} 2 \cdot 3 - \gcd\{6, 3\} = 3 \leq 6 \\ 2 \cdot 4 - \gcd\{6, 4\} = 6 \leq 6 \end{array}$$

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

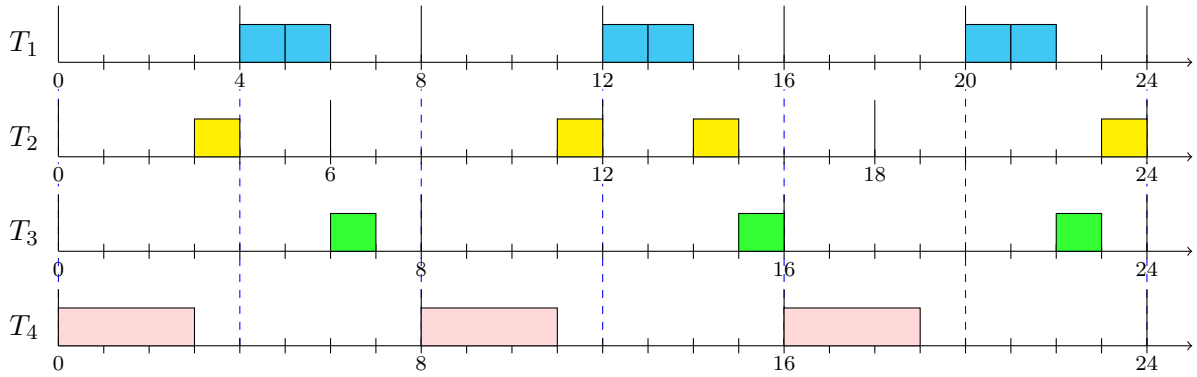
$$\begin{array}{l} 2 \cdot 3 - \gcd\{8, 3\} = 5 \leq 8 \\ 2 \cdot 4 - \gcd\{8, 4\} = 4 \leq 8 \end{array}$$

- Vincolo sul task T_4 ($2f - \gcd\{8, f\} \leq 3$):

$$\begin{array}{ll} 2 \cdot 3 - \gcd\{8, 3\} = 5 > 4 & \Rightarrow f \neq 3 \\ 2 \cdot 4 - \gcd\{8, 4\} = 4 \leq 4 & \end{array}$$

Una dimensione opportuna per il frame potrebbe dunque essere $f = 4$.

(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:

$L(0) = \langle T_4, T_2 \rangle$, $L(1) = \langle T_1, T_1, T_3 \rangle$, $L(2) = \langle T_4, T_2 \rangle$, $L(3) = \langle T_1, T_1, T_2, T_3 \rangle$, $L(4) = \langle T_4 \rangle$, $L(5) = \langle T_1, T_1, T_3, T_2 \rangle$.

Osservazione (9.09.2009): Si noti che questa schedulazione, pur essendo valida in assoluto, non rispetta i vincoli imposti da uno scheduler ciclico strutturato. Si consideri ad esempio il primo job di T_1: dato che la sua scadenza cade in mezzo al secondo frame, esso deve essere eseguito obbligatoriamente nel primo frame, in modo da consentire allo scheduler invocato all'istante $t = 4$ di controllare l'avvenuto rispetto della scadenza.

In generale, per la dimensione $f = 4$ non esiste alcuna schedulazione ciclica valida. Infatti nel primo frame devono essere eseguiti obbligatoriamente il primo job di T_1, il primo job di T_2 (la cui scadenza cade in mezzo al secondo frame) ed il primo job di T_4 (la cui scadenza coincide con la fine del primo frame). Il fattore di utilizzazione è perciò $5/4 > 1$.

D'altra parte, dall'analisi svolta nel punto (a) sappiamo che $f = 4$ era l'unico valore ammissibile. Perciò, a meno di non spezzare i job di qualche task periodico, non esiste alcuna schedulazione ciclica strutturata per il sistema.

(c) *Assumendo che la schedulazione determinata in (b) si ripeta all'infinito, determinare il tempo di risposta di un job aperiodico rilasciato a $t = 2$ con tempo d'esecuzione $e = 4$ sia nel caso in cui si utilizzi* slack stealing *che nel caso in cui non lo si utilizzi.*

Tra i sei cicli minori (frame) all'interno del ciclo maggiore, solo il secondo ed il quinto contengono una unità di *slack* ciascuno. Poiché il job aperiodico è rilasciato a $t = 2$, quindi all'interno del primo frame, lo scheduler assegnerà il processore al job aperiodico nei frame

inizianti agli istanti $t = 4, 16, 28, 40$. Il job aperiodico si conclude dunque nell'undicesimo frame iniziante a $t = 40$.

Il tempo di risposta del job aperiodico dipende esclusivamente dalla schedulazione all'interno dell'undicesimo frame. In presenza di *slack stealing* lo scheduler anticipa l'esecuzione del job aperiodico rispetto a quella dei task periodici; perciò il job termina a $t = 41$. Invece in assenza di *slack stealing* il job termina alla fine del frame, ossia a $t = 44$. I tempi di risposta sono pertanto:

- Con *slack stealing*: $41 - 2 = 39$ unità di tempo
- Senza *slack stealing*: $44 - 2 = 42$ unità di tempo

Esercizio 2. *Un server procastinabile con periodo $p_s = 4$, budget $e_s = 1$ e fase indeterminata è schedulato su un singolo processore insieme a due task periodici, indipendenti e interrompibili:* $T_1 = (5, 1)$ *e* $T_2 = (10, 3)$.

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

Poichè i periodi dei task periodici e del server non rispettano le condizioni del teorema di Lehoczky, Sha e Strosnider, è necessario controllare la schedulabilità di ciascun task separatamente.

Utilizzando il fattore di utilizzazione, condizione sufficiente per la schedulabilità di T_i è:

$$\sum_{k=1}^{i} \frac{e_k}{p_k} + u_s + \frac{e_s}{p_i} \leq U_{\text{RM}}(i+1)$$

- Server procastinabile: $u_s = 1/4 \leq 1$ $\Rightarrow$ schedulabile
- Task T_1: $1/5 + 1/4 + 1/5 = 13/20 = 0{,}65 < 0{,}828 < U_{\text{RM}}(2)$ $\Rightarrow$ schedulabile
- Task T_2: $1/5 + 3/10 + 1/4 + 1/10 = 17/20 = 0{,}85 > 0{,}780 > U_{\text{RM}}(3)$

Il task T_2 non è necessariamente schedulabile; è necessario dunque studiare la sua funzione di tempo necessario, che in presenza di server procastinabile diventa:

$$w_i(t) = e_i + e_s + \left\lceil \frac{t - e_s}{p_s} \right\rceil e_s + \sum_{k=1}^{i-1} \left\lceil \frac{t}{p_k} \right\rceil e_k$$

Perciò $w_2(t) = 4 + \lceil (t-1)/4 \rceil + \lceil t/5 \rceil$:

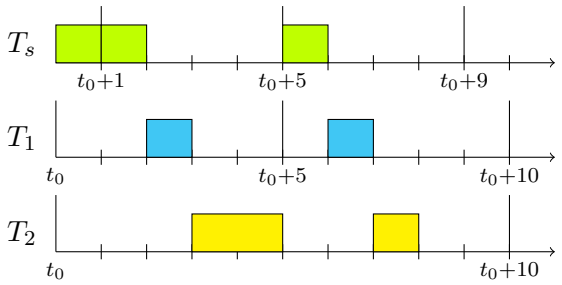
$w_2(4) = 6 > 4$
$w_2(5) = 6 > 5$
$w_2(8) = 8 \leq 8 \qquad \Rightarrow T_2$ è schedulabile

Il sistema è quindi schedulabile con RM.

(b) *Indicare un esempio di schedulazione RM in cui il task T_2 presenta il peggiore tempo di risposta possibile. In quali condizioni ciò si verifica?*

La funzione di tempo necessario $w_2(t)$ del task T_2 utilizzata nel punto (b) indica che il massimo tempo richiesto da un job di T_2 per terminare è 8 unità di tempo.

Un esempio di schedulazione RM in cui ciò si verifica è il seguente:



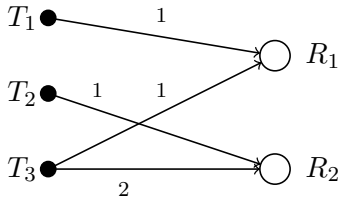
Le condizioni in cui tale circostanza si verifica sono:

- Al tempo t_0 vengono rilasciati job di T_1 e T_2
- Al tempo t_0 viene rilasciato un job aperiodico con tempo d'esecuzione $e \geq 3$
- Al tempo t_0 il budget del server procastinabile è al massimo, ossia è uguale a 1
- Il nuovo periodo del server procastinabile inizia a $t_0 + 1$, quindi a $t_0 + 1$ il suo budget è ripristinato

Esercizio 3. *Un sistema è costituito da tre task periodici con fasi indeterminate* $T_1 = (6, 2)$, $T_2 = (19, 1)$ *e* $T_3 = (20, 2)$. *Ciascun job dei tre task è sempre interrompibile e si auto-sospende al massimo una volta; il periodo di sospensione dura al massimo 2 unità di tempo. I task utilizzano due risorse condivise* R_1 *e* R_2 *come segue:* $T_1 : [R_1, 1]$, $T_2 : [R_2, 1]$, $T_3 : [R_2, 2\,[R_1, 1]\,]$.

Determinare se l'insieme di task è schedulabile su un singolo processore con RM assumendo che lo scheduler abbia overhead trascurabile e che l'accesso alle risorse condivise sia controllato dal protocollo priority ceiling.

Determiniamo i massimi tempi di blocco per conflitto di risorse $b_i(\text{rc})$ dei tre task:



B_d	T_2	T_3
T_1		1
T_2	$*$	2

B_i	T_2	T_3
T_1		
T_2	$*$	1

B_c	T_2	T_3
T_1		
T_2	$*$	1

I tempi di blocco per conflitti di risorse sono: $b_1(\text{rc}) = 1$, $b_2(\text{rc}) = 2$, $b_3(\text{rc}) = 0$.

Determiniamo i tempi di blocco dovuti all'auto-sospensione tramite la formula:

$$b_i(\text{ss}) = x_i + \sum_{k=1}^{i-1} \min(e_k, x_k)$$

$b_1(\text{ss}) = 2$; $\quad b_2(\text{ss}) = 2 + \min(2, 2) = 4$; $\quad b_3(\text{ss}) = 2 + \min(2, 2) + \min(1, 2) = 5$.

Poiché i job sono sempre interrompibili, i tempi totali di blocco sono dati dalla formula:

$$b_i = b_i(\text{ss}) + (K_i + 1) \cdot b_i(\text{rc})$$

$b_1 = 2 + 2 \cdot 1 = 4$; $\quad b_2 = 4 + 2 \cdot 2 = 8$; $\quad b_3 = 5 + 2 \cdot 0 = 5$.

Per determinare se il sistema è schedulabile, applichiamo la condizione di schedulabilità al singolo task T_i:

$$\sum_{k=1}^{i} \frac{e_k}{p_k} + \frac{b_i}{p_i} \leq U_{\text{RM}}(i)$$

- Task T_1: $\quad 2/6 + 4/6 = 1 = U_{\text{RM}}(1) \qquad \Rightarrow$ schedulabile
- Task T_2: $\quad 2/6 + 1/19 + 8/19 = 46/57 < 0.808 < 0.828 < U_{\text{RM}}(2) \qquad \Rightarrow$ schedulabile
- Task T_3: $\quad 2/6 + 1/19 + 2/20 + 5/20 < 0.736 < 0.779 < U_{\text{RM}}(3) \qquad \Rightarrow$ schedulabile

Il sistema di task è dunque schedulabile.