Sistemi Embedded e Real-time (M. Cesati)

Compito scritto del 15 luglio 2013

Esercizio 1. Il seguente sistema di task periodici con job non interrompibili è schedulato su un processore con un algoritmo "cyclic schedule": $T_1 = (1, 2, \frac{1}{2}, 2)$, $T_2 = (2, 1)$, $T_3 = (4, 1)$.

(a) Esiste una schedulazione ciclica strutturata? Giustificare la risposta.

(b) Esiste una schedulazione ciclica strutturata se la scadenza relativa di T_1 è pari a 3? Giustificare la risposta.

Esercizio 2. Un server procastinabile con periodo $p_s = 4$, budget $e_s = 2$ e fase indeterminata è schedulato su un singolo processore insieme a due task periodici, indipendenti e interrompibili: $T_1 = (5, 1)$ e $T_2 = (13/2, 1/2)$.

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

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

(c) Il server procastinabile è sostituito da un server sporadico semplice con gli stessi parametri $p_s = 4$ e $e_s = 2$: cosa si può concludere sulla schedulabilità con EDF e con RM?

Esercizio 3. Un sistema è costituito da tre task periodici con fasi indeterminate $T_1 = (10, 2)$, $T_2 = (20, 3)$ e $T_3 = (21, 4)$. Ciascun job di T_1 e T_2 è sempre interrompibile e si auto-sospende al massimo una volta; nessun job rimane sospeso per più di tre unità di tempo. I job di T_3 sono sempre interrompibili e non si auto-sospendono. I task utilizzano due risorse condivise R_1 e R_2 come segue: $T_1 : [R_1, 1]$, $T_2 : [R_2, 1]\,[R_1, 2]$, $T_3 : [R_2, 2\,[R_1, 1]\,]$.

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 15 luglio 2013

Esercizio 1. *Il seguente sistema di task periodici con job non interrompibili è schedulato su un processore con un algoritmo "cyclic schedule":* $T_1 = (1, 2, \frac{1}{2}, 2)$, $T_2 = (2, 1)$, $T_3 = (4, 1)$.

(a) *Esiste una schedulazione ciclica strutturata? Giustificare la risposta.*

Determiniamo la dimensione del frame f più appropriata:

- Vincolo sulle fasi dei task: tutti i task sono in fase, tranne il task T_1 che ha fase 1. Poiché la dimensione del frame deve dividere 1, l'unica dimensione ammissibile per il frame è $f = 1$.
- Vincolo sui tempi d'esecuzione dei job:

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

- Banalmente $f = 1$ divide l'iperperiodo ($\text{mcm}\{2, 2, 4\} = 4$).
- Poiché $f = 1$, per ogni task vale $2 \cdot 1 - \gcd\{p_i, 1\} = 1$. Considerato che le scadenze relative dei task sono comunque maggiori di 1, tutti i vincoli dei task sono rispettati.

In conclusione, l'unica dimensione ammissibile per la dimensione del frame è $f = 1$.

È facile verificare però che non può esistere alcuna schedulazione ciclica strutturata. Infatti:

- Il fattore di utilizzazione totale del sistema è

$$U_{\text{T}} = \frac{1/2}{2} + \frac{1}{2} + \frac{1}{4} = 1\,.$$

  - Di conseguenza, qualunque schedulazione ammissibile non può lasciare mai il processo "idle", a regime.

- L'esecuzione di un job di T_1 "consuma" mezzo frame. Nel restante mezzo frame non può essere schedulato un altro job in quanto:

  - Job di T_2 e T_3 sono lunghi un intero frame e non sono interrompibili.

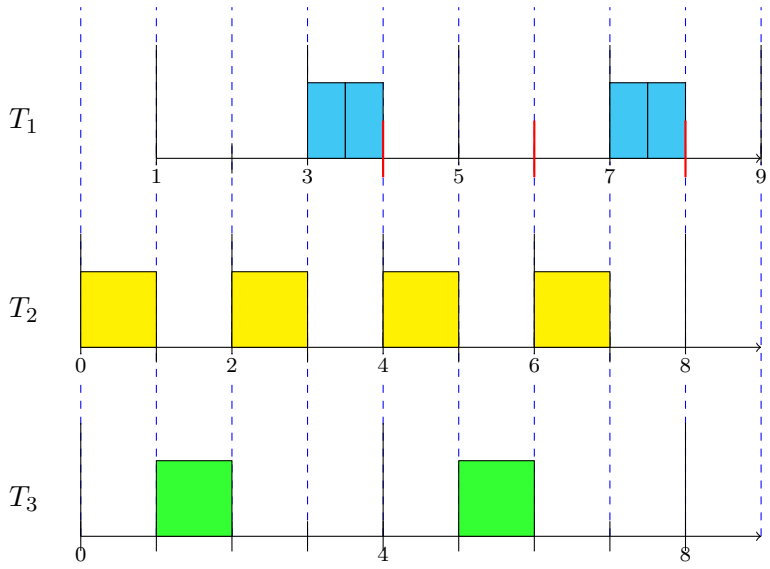
– La scadenza relativa dei job di T_1 è pari al periodo, pertanto nel mezzo frame occupato da un job di T_1 non può essere validamente schedulato alcun altro job di T_1.

La conclusione è che non esiste alcuna schedulazione ciclica strutturata per questo sistema di task.

(b) *Esiste una schedulazione ciclica strutturata se la scadenza relativa di T_1 è pari a 3? Giustificare la risposta.*

Valgono quasi tutte le considerazioni fatte per il punto precedente: l'unica dimensione ammissibile per il frame è $f = 1$. In questo caso tuttavia potrebbe essere possibile schedulare all'interno dello stesso frame due job differenti del task T_1, perché la scadenza relativa di T_1 è maggiore del suo periodo.

È facile determinare una schedulazione ciclica strutturata. Ad esempio:



All'istante $t = 8$ si ripetono tutte le condizioni esistenti all'istante $t = 0$: tutti i job precedentemente rilasciati sono stati eseguiti, i task T_2 e T_3 rilasciano un job, ed il task T_1 rilascerà un job all'inizio del frame successivo. Pertanto la schedulazione determinata in $[0, 8]$ può essere ripetuta all'infinito.

Esercizio 2. *Un server procastinabile con periodo $p_s = 4$, budget $e_s = 2$ e fase indeterminata è schedulato su un singolo processore insieme a due task periodici, indipendenti e interrompibili:* $T_1 = (5, 1)$ *e* $T_2 = (13/2, 1/2)$.

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

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

$$\sum_{k=1}^{n} \frac{e_k}{\min(D_k, p_k)} + \frac{e_s}{p_s}\left(1 + \frac{p_s - e_s}{D_i}\right) \leq 1$$

- Server procastinabile: $1/5+(1/2)/(13/2)+2/4 = 101/130 \leq 1 \quad \Rightarrow$ schedulabile
- Task T_1: $1/5 + 1/13 + (2/4)(1 + (4-2)/5) = 127/130 \leq 1 \quad \Rightarrow$ schedulabile
- Task T_2: $1/5 + 1/13 + (2/4)(1 + (4-2)/(13/2)) = 121/130 \leq 1 \quad \Rightarrow$ schedulabile

Il sistema di task è dunque schedulabile con EDF.

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

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

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

(a condizione che per ogni task $T_1, \ldots, T_i$ la scadenza relativa coincida con il periodo).

- Server procastinabile: $e_s/p_s = 2/4 \leq 1 \quad \Rightarrow$ schedulabile
- Task T_1: $1/5 + 2/4 + 2/5 = 11/10 > 1 \quad \Rightarrow$ non schedulabile
- Task T_2: $1/5+(1/2)/(13/2)+2/4+2/(13/2) = 141/130 > 1 \quad \Rightarrow$ non schedulabile

Poiché la somma dei fattori di utilizzo e dei tempi di blocco relativi supera l'unità, possiamo concludere che non è possibile garantire la schedulabilità del sistema con RM.

(c) *Il server procastinabile è sostituito da un server sporadico semplice con gli stessi parametri $p_s = 4$ e $e_s = 2$: cosa si può concludere sulla schedulabilità con EDF e con RM?*

Poiché è stato dimostrato analiticamente che il sistema di task con un server procastinabile è schedulabile con EDF, sostituendo il server procastinabile con un server sporadico semplice avente le stesse dimensioni si continuerà ad avere la garanzia di schedulabilità. Infatti, il server sporadico semplice si comporta nel caso peggiore come un qualunque task periodico, e quindi a differenza del server procastinabile non richiede di considerare nell'analisi di schedulabilità tempi di blocco aggiuntivi. Perciò le conclusioni dell'analisi svolta precedentemente sono valide, a maggior ragione, anche per un sistema con un server sporadico semplice.

Per quanto riguarda l'algoritmo RM, è necessario ripetere l'analisi applicando la condizione di schedulabilità senza tempi di blocco aggiuntivi:

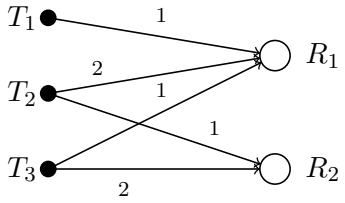
$$\sum_{k=1}^{2} \frac{e_k}{p_k} + \frac{e_s}{p_s} = \frac{1}{5} + \frac{1/2}{13/2} + \frac{2}{4} = \frac{101}{130} < 0.777 < 0.779 < U_{\mathrm{RM}}(3).$$

Si conclude perciò che il sistema con server sporadico semplice è schedulabile con RM.

Esercizio 3. *Un sistema è costituito da tre task periodici con fasi indeterminate $T_1 = (10, 2)$, $T_2 = (20, 3)$ e $T_3 = (21, 4)$. Ciascun job di T_1 e T_2 è sempre interrompibile e si auto-sospende al massimo una volta; nessun job rimane sospeso per più di tre unità di tempo. I job di T_3 sono sempre interrompibili e non si auto-sospendono. I task utilizzano due risorse condivise R_1 e R_2 come segue: $T_1 : [R_1, 1]$, $T_2 : [R_2, 1]\,[R_1, 2]$, $T_3 : [R_2, 2\,[R_1, 1]\,]$.*

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 $b_i(\mathrm{rc})$ dei tre task:



B_d	T_2	T_3
T_1	2	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: $b_1(\text{rc}) = 2$, $b_2(\text{rc}) = 2$, $b_3(\text{rc}) = 0$.

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

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

$b_1(\text{ss}) = 3$; $\quad b_2(\text{ss}) = 3 + \min(2,3) = 5$; $\quad b_3(\text{ss}) = 0 + \min(2,3) + \min(3,3) = 5$.

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

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

$b_1 = 3 + 2 \cdot 2 = 7$; $\quad b_2 = 5 + 2 \cdot 2 = 9$; $\quad b_3 = 5 + 2 \cdot 0 = 5$.

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: $\quad 2/10 + 7/10 = 9/10 < 1 = U_{\text{RM}}(1) \qquad \Rightarrow$ schedulabile
- Task T_2: $\quad 2/10 + 3/20 + 9/20 = 4/5 = 0.800 < 0.828 < U_{\text{RM}}(2) \qquad \Rightarrow$ schedulabile
- Task T_3: $\quad 2/10 + 3/20 + 4/21 + 5/21 < 0.7786 < 0.779 < U_{\text{RM}}(3) \qquad \Rightarrow$ schedulabile

Il sistema di task è dunque schedulabile.