

Sistemi Embedded e Real-time (M. Cesati)

Compito scritto del 15 luglio 2011

Esercizio 1. Si consideri il seguente sistema di task periodici : $T_1 = (4, 1)$, $T_2 = (8, 4, 1, 4)$, $T_3 = (5, 1, 7)$, $T_4 = (4, 10, 5/2, 7)$. I job sono non interrompibili. È possibile determinare una schedulazione ciclica strutturata per il sistema di task utilizzando un singolo processore? 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. Si consideri il sistema di task $T_1 = (4, 1)$, $T_2 = (5, 1/2)$, $T_3 = (8, 3)$ e $T_4 = (10, 1)$ sempre interrompibili ed indipendenti.

(a) Determinare analiticamente se il sistema di task è schedulabile su un singolo processore con l'algoritmo RM.

(b) Determinare analiticamente se il sistema di task è schedulabile su un singolo processore con l'algoritmo DM (Deadline Monotonic), supponendo anche che la scadenza relativa di T_2 sia $11/2$ e la scadenza relativa di T_3 sia 5.

(c) Si consideri ancora l'insieme di task utilizzato nel punto (a), tranne che per il task T_1 il cui periodo e la cui scadenza relativa salgono da 4 a $9/2$. Cosa è possibile concludere sulla schedulabilità del sistema con l'algoritmo RM?

Sistemi Embedded e Real-time (M. Cesati)

Soluzioni del compito scritto del 15 luglio 2011

Esercizio 1. Si consideri il seguente sistema di task periodici : $T_1 = (4, 1)$, $T_2 = (8, 4, 1, 4)$, $T_3 = (5, 1, 7)$, $T_4 = (4, 10, 5/2, 7)$. I job sono non interrompibili. È possibile determinare una schedulazione ciclica strutturata per il sistema di task utilizzando un singolo processore? Giustificare la risposta.

Determiniamo le possibili dimensioni f del frame dello scheduler ciclico:

- Vincolo sulle fasi dei task:
 - il task T_2 ha fase 8, quindi f deve dividere 8: $f \in \{1, 2, 4, 8\}$
 - il task T_3 ha fase 4, quindi f deve dividere 4: $f \in \{1, 2, 4\}$
- Vincolo sui tempi d'esecuzione dei job:

$$f \geq \max\{1, 1, 1, 5/2\} = 5/2 \Rightarrow f \geq 3$$

- Vincolo sulla divisibilità della lunghezza dell'iperperiodo ($\text{mcm}\{4, 4, 5, 10\} = 20$):

$$f \in \{1, 2, 4, 5, 10, 20\}.$$

Quindi combinando con le condizioni precedenti l'unico valore ammissibile è $f = 4$.

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

$$2 \cdot 4 - \text{gcd}\{4, 4\} = 4 \leq 4 \quad \text{ok}$$

- Vincolo sul task T_2 ($2f - \text{gcd}\{4, f\} \leq 4$): identico al task T_1 , ossia

$$2 \cdot 4 - \text{gcd}\{4, 4\} = 4 \leq 4 \quad \text{ok}$$

- Vincolo sul task T_3 ($2f - \text{gcd}\{5, f\} \leq 7$):

$$2 \cdot 4 - \text{gcd}\{5, 4\} = 7 \leq 7 \quad \text{ok}$$

- Vincolo sul task T_4 ($2f - \text{gcd}\{10, f\} \leq 7$):

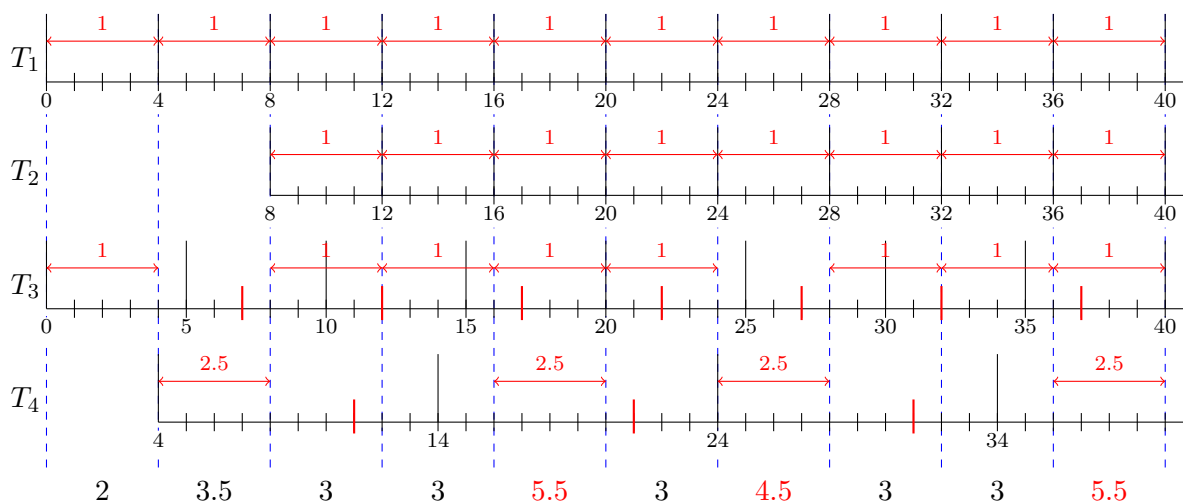
$$2 \cdot 4 - \text{gcd}\{10, 4\} = 6 \leq 7 \quad \text{ok}$$

L'unica dimensione ammissibile per il frame dello scheduler ciclico strutturato è $f = 4$.

Dobbiamo ora verificare se esiste effettivamente una schedulazione valida di tutti i job nei frame di dimensione 4. A tale scopo, si consideri che

- Un job può essere schedulato solo se il suo istante di rilascio cade in un frame precedente oppure coincide con l'inizio stesso del frame.
- Lo scheduler deve controllare l'avvenuto rispetto delle scadenze dei vari job; per questo motivo, un job deve essere completato (e dunque deve essere completamente eseguito) in un frame precedente a quello contenente la sua scadenza oppure in un frame che termina esattamente nell'istante di scadenza del job.

Quindi per ciascun job è possibile determinare in quali frame esso deve essere eseguito. Si ottiene così lo schema seguente, in cui il lavoro da effettuare per ciascun task è rappresentato da frecce che si estendono tra uno o più frame e valori numerici che indicano la somma dei tempi d'esecuzione dei job in quei frame.



Considerando il lavoro totale che il processore deve svolgere in ciascun frame, risulta evidente che in alcuni frame (ad esempio quelli inizianti agli istanti 16, 24 e 36) non sarà possibile eseguire tutti i job, in quanto il tempo di esecuzione totale supera la dimensione del frame (4).

Si conclude perciò che il sistema di task in questione non ammette alcuna schedulazione ciclica strutturata.

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

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 :

$$U''_T + \frac{b'_1}{p_1} = \frac{36833}{42000} + \frac{73}{150} = \frac{57273}{42000} > 1 \quad \implies \text{NO}$$

- Schedulabilità di T_2 :

$$U''_T + \frac{b'_2}{p_2} = \frac{36833}{42000} + \frac{31}{250} = \frac{42041}{42000} > 1 \quad \implies \text{NO}$$

- Schedulabilità di T_3 :

$$U''_T + \frac{b'_3}{p_3} = \frac{36833}{42000} + \frac{13}{175} = \frac{39953}{42000} < 1 \quad \implies \text{OK}$$

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

Esercizio 3. Si consideri il sistema di task $T_1 = (4, 1)$, $T_2 = (5, 1/2)$, $T_3 = (8, 3)$ e $T_4 = (10, 1)$ sempre interrompibili ed indipendenti.

(a) Determinare analiticamente se il sistema di task è schedulabile su un singolo processore con l'algoritmo RM.

Osserviamo preliminarmente che i quattro task del sistema sono a due a due armonici: T_3 ha una frequenza multipla di T_1 , e T_4 ha una frequenza multipla di T_2 . È perciò possibile applicare il teorema di Kuo-Mok, e dunque confronteremo il fattore di utilizzazione totale del sistema U_T con $U_{RM}(2)$ invece che con $U_{RM}(4)$:

$$U_T = \frac{1}{4} + \frac{1}{10} + \frac{3}{8} + \frac{1}{10} = \frac{33}{40} = 0.825 < 0.828 < 2\sqrt{2} - 2 = U_{RM}(2)$$

Possiamo dunque concludere che il sistema di task è schedulabile con RM.

In alternativa, è possibile controllare la schedulabilità utilizzando il test iperbolico. In questo caso si devono considerare separatamente i fattori di utilizzazione dei due sottoinsiemi di task armonici. Per il primo $(\{T_1, T_3\})$ si ha:

$$U_{T'} = \frac{1}{4} + \frac{3}{8} = \frac{5}{8},$$

mentre per il secondo $(\{T_2, T_4\})$ si ha:

$$U_{T''} = \frac{1}{10} + \frac{1}{10} = \frac{1}{5}.$$

Quindi applicando il test iperbolico:

$$(1 + U_{T'}) \cdot (1 + U_{T''}) = 1 + \frac{19}{20} < 2 \Rightarrow \text{OK}$$

(b) Determinare analiticamente se il sistema di task è schedulabile su un singolo processore con l'algoritmo DM (Deadline Monotonic), supponendo anche che la scadenza relativa di T_2 sia $11/2$ e la scadenza relativa di T_3 sia 5 .

Riordinando i task in accordo alle priorità assegnate dall'algoritmo DM, il sistema di task da considerare è il seguente:

$$T_1 = (4, 1), \quad T_2' = (8, 3, 5), \quad T_3' = (5, 1/2, 11/2), \quad T_4 = (10, 1).$$

Poiché alcuni task hanno scadenze esplicite è necessario avvalersi del test di schedulabilità.

- Il task T_1 , poiché ha priorità massima e fattore di utilizzazione inferiore ad uno, è senz'altro schedulabile.
- La funzione di tempo necessario del task T'_2 è: $w'_2(t) = 3 + \lceil t/4 \rceil$.

Ricerca del punto fisso dell'equazione $w'_2(t) = t$:

$$w'_2(3) = 3 + \lceil 3/4 \rceil = 3 + 1 = 4; \quad w'_2(4) = 3 + \lceil 4/4 \rceil = 3 + 1 = 4.$$

Poiché il tempo di risposta di T'_2 è pari a 4 ed è inferiore alla scadenza relativa 5, T'_2 è schedulabile.

- La funzione di tempo necessario del task T'_3 è: $w'_3(t) = 1/2 + \lceil t/4 \rceil + \lceil t/8 \rceil \cdot 3$.

Ricerca del punto fisso dell'equazione $w'_3(t) = t$:

$$\begin{aligned} w'_3(1/2) &= 1/2 + \lceil 1/8 \rceil + \lceil 1/16 \rceil \cdot 3 = 9/2; \\ w'_3(9/2) &= 1/2 + \lceil 9/8 \rceil + \lceil 9/16 \rceil \cdot 3 = 11/2; \\ w'_3(11/2) &= 1/2 + \lceil 11/8 \rceil + \lceil 11/16 \rceil \cdot 3 = 11/2. \end{aligned}$$

Poiché il tempo di risposta di T'_3 è pari a 11/2 ed è uguale alla scadenza relativa, il primo job di T'_3 è schedulabile. Dato però che la scadenza è oltre il periodo di T'_3 è necessario verificare la schedulabilità dei successivi job di T'_3 con il test generale.

Per determinare l'intervallo totalmente occupato di T'_3 cerchiamo il punto fisso dell'equazione $t = \sum_{k=1}^3 \lceil t/p_k \rceil \cdot e_k = \lceil t/4 \rceil + \lceil t/8 \rceil \cdot 3 + \lceil t/5 \rceil / 2$:

$$\begin{aligned} t^{(1)} &= e_1 + e_2 + e_3 = 9/2 \\ t^{(2)} &= \lceil 9/8 \rceil + \lceil 9/16 \rceil \cdot 3 + \lceil 9/10 \rceil / 2 = 11/2 \\ t^{(3)} &= \lceil 11/8 \rceil + \lceil 11/16 \rceil \cdot 3 + \lceil 11/10 \rceil / 2 = 6 \\ t^{(4)} &= \lceil 3/2 \rceil + \lceil 3/4 \rceil \cdot 3 + \lceil 6/5 \rceil / 2 = 6 \end{aligned}$$

Nell'intervallo totalmente occupato di lunghezza 6 vengono rilasciati $\lceil 6/5 \rceil = 2$ job di T'_3 . La funzione di tempo necessario del secondo job di T'_3 è $w'_{3,2}(t) = 1/2 + w'_3(t) = 1 + \lceil t/4 \rceil + \lceil t/8 \rceil \cdot 3$. Cerchiamo il punto fisso dell'equazione $t = w'_{3,2}(t)$ con $t \in [5, 21/2]$:

$$\begin{aligned} w'_{3,2}(5) &= 1 + \lceil 5/4 \rceil + \lceil 5/8 \rceil \cdot 3 = 6 \\ w'_{3,2}(6) &= 1 + \lceil 6/4 \rceil + \lceil 6/8 \rceil \cdot 3 = 6 \end{aligned}$$

Poiché il secondo job si completa all'istante 6 che precede la scadenza assoluta 21/2, concludiamo che il task T'_3 è schedulabile.

- L'intervallo totalmente occupato di T_4 si ottiene risolvendo iterativamente l'equazione $t = \lceil t/4 \rceil + \lceil t/8 \rceil \cdot 3 + \lceil t/5 \rceil / 2 + \lceil t/10 \rceil$:

$$\begin{aligned} t^{(1)} &= e_1 + e_2 + e_3 + e_4 = 11/2 \\ t^{(2)} &= \lceil 11/8 \rceil + \lceil 11/16 \rceil \cdot 3 + \lceil 11/10 \rceil / 2 + \lceil 11/20 \rceil = 7 \\ t^{(3)} &= \lceil 7/4 \rceil + \lceil 7/8 \rceil \cdot 3 + \lceil 7/5 \rceil / 2 + \lceil 7/10 \rceil = 7 \end{aligned}$$

Nell'intervallo totalmente occupato viene rilasciato $\lceil 7/10 \rceil = 1$ job di T_4 .

La funzione di tempo necessario (del primo job) di T_4 è:

$$w_4(t) = 1 + \lceil t/4 \rceil + \lceil t/8 \rceil \cdot 3 + \lceil t/5 \rceil / 2.$$

Ricerca del punto fisso dell'equazione $w_4(t) = t$:

$$\begin{aligned} w_4(1) &= 1 + \lceil 1/4 \rceil + \lceil 1/8 \rceil \cdot 3 + \lceil 1/5 \rceil / 2 = 11/2; \\ w_4(11/2) &= 1 + \lceil 11/8 \rceil + \lceil 11/16 \rceil \cdot 3 + \lceil 11/10 \rceil / 2 = 7; \\ w_4(7) &= 1 + \lceil 7/4 \rceil + \lceil 7/8 \rceil \cdot 3 + \lceil 7/5 \rceil / 2 = 7. \end{aligned}$$

Poiché il tempo di risposta di T_4 è pari a 7 ed è inferiore alla scadenza relativa 10, T_4 è schedulabile.

Si conclude pertanto che il sistema di task è schedulabile con DM.

(c) Si consideri ancora l'insieme di task utilizzato nel punto (a), tranne che per il task T_1 il cui periodo e la cui scadenza relativa salgono da 4 a 9/2. Cosa è possibile concludere sulla schedulabilità del sistema con l'algoritmo RM?

Riordinando in accordo alle priorità assegnate dall'algoritmo RM, il sistema di task da considerare è il seguente:

$$T'_1 = (9/2, 1), \quad T_2 = (5, 1/2), \quad T_3 = (8, 3), \quad T_4 = (10, 1).$$

L'unica differenza rispetto al sistema di task schedulabile analizzato nel punto (a) è che il periodo del task di priorità massima è aumentato, ovvero è diminuita la sua frequenza. Poiché i task sono supposti essere interrompibili, indipendenti e schedulati su di un unico processore, la schedulazione è *predicibile*, e dunque ci aspettiamo che il nuovo sistema di task rimanga schedulabile. Verifichiamo analiticamente questa conclusione.

Il test iperbolico, applicato a tre sottoinsiemi di task armonici, non è conclusivo:

$$\left(1 + \frac{2}{9}\right) \cdot \left(1 + \frac{3}{8}\right) \cdot \left(1 + \frac{1}{5}\right) = 2 + \frac{1}{60} > 2.$$

È necessario perciò avvalersi del test di schedulabilità:

- Il task T'_1 , poiché ha priorità massima e fattore di utilizzazione inferiore ad uno, è senz'altro schedulabile.
- La funzione di tempo necessario del task T_2 è: $w_2(t) = 1/2 + \lceil 2t/9 \rceil$.

Ricerca del punto fisso dell'equazione $w_2(t) = t$:

$$w_2(1/2) = 1/2 + \lceil 1/9 \rceil = 3/2; \quad w_2(3/2) = 1/2 + \lceil 3/9 \rceil = 3/2.$$

Poiché il tempo di risposta di T_2 è pari a $3/2$ ed è inferiore alla scadenza relativa 5, T_2 è schedulabile.

- La funzione di tempo necessario del task T_3 è: $w_3(t) = 3 + \lceil 2t/9 \rceil + \lceil t/5 \rceil / 2$.

Ricerca del punto fisso dell'equazione $w_3(t) = t$:

$$\begin{aligned} w_3(3) &= 3 + \lceil 6/9 \rceil + \lceil 3/5 \rceil / 2 = 9/2; \\ w_3(9/2) &= 3 + \lceil 9/9 \rceil + \lceil 9/10 \rceil / 2 = 9/2. \end{aligned}$$

Poiché il tempo di risposta di T_3 è pari a $9/2$ ed è inferiore alla scadenza relativa 8, T_3 è schedulabile.

- La funzione di tempo necessario del task T_4 è: $w_4(t) = 1 + \lceil 2t/9 \rceil + \lceil t/5 \rceil / 2 + \lceil t/8 \rceil \cdot 3$.

Ricerca del punto fisso dell'equazione $w_4(t) = t$:

$$\begin{aligned} w_4(1) &= 1 + \lceil 2/9 \rceil + \lceil 1/5 \rceil / 2 + \lceil 1/8 \rceil \cdot 3 = 11/2; \\ w_4(11/2) &= 1 + \lceil 11/9 \rceil + \lceil 11/10 \rceil / 2 + \lceil 11/16 \rceil \cdot 3 = 7; \\ w_4(7) &= 1 + \lceil 14/9 \rceil + \lceil 7/5 \rceil / 2 + \lceil 7/8 \rceil \cdot 3 = 7. \end{aligned}$$

Poiché il tempo di risposta di T_4 è pari a 7 ed è inferiore alla scadenza relativa 10, T_4 è schedulabile.

Come ci aspettava, il sistema di task risulta schedulabile con RM.