Lezione R6

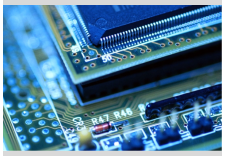
Schedulabilità per RM e DM

Sistemi embedded e real-time

13 novembre 2012

Marco Cesati

Dipartimento di Ingegneria Civile e Ingegneria Informatica
Università degli Studi di Roma Tor Vergata

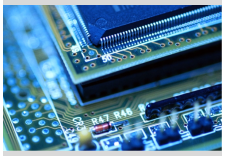


Di cosa parliamo in questa lezione?

In questa lezione continuiamo la discussione di algoritmi e criteri di schedulabilità per il modello a task periodici

1. Test di schedulabilità per algoritmi a priorità fissa
2. Test di schedulabilità generale per scadenze arbitrarie
3. Condizioni di schedulabilità per algoritmi a priorità fissata

Schedulabilità per RM e DM
Marco Cesati

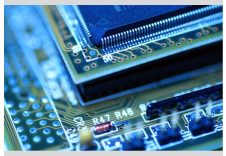


Schedulabilità con algoritmi a priorità fissa

- Alcuni algoritmi di schedulazione con priorità dinamica, ad es. EDF, sono ottimali: $U_{EDF} = 1$
- Nessun algoritmo X con priorità fissa per i task può essere ottimale in senso assoluto: $U_X < 1$
- L'algoritmo RM (Rate Monotonic) è ottimale (in senso assoluto) per i sistemi di task armonici con scadenze implicite
- L'algoritmo DM (Deadline Monotonic) è ottimale tra gli algoritmi a priorità fissa, ma non in senso assoluto
- L'algoritmo RM è ottimale tra gli algoritmi a priorità fissa per sistemi di task con scadenza proporzionale al periodo

Problema generale

Fissato un sistema di task ed un algoritmo di schedulazione a priorità fissa (tipicamente RM), come verificare se l'algoritmo determinerà **sempre** una schedulazione fattibile?



Istanti critici

Supponiamo che in un insieme di task in cui le fasi iniziali **non** sono pre-determinate i tempi di risposta siano *piccoli*: ogni job deve terminare prima che un altro job dello stesso task sia rilasciato

Definizione di istante critico di un task

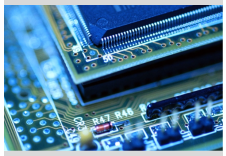
Se tutti i job di un task T_i rispettano la scadenza relativa, l'*istante critico* è un momento in cui il rilascio di un job comporta il massimo tempo di risposta possibile per quel job

Se almeno un job di T_i non rispetta la scadenza relativa, l'*istante critico* è un momento in cui il rilascio di un job provoca il mancato rispetto della scadenza di quel job

Teorema (Liu, Layland 1973)

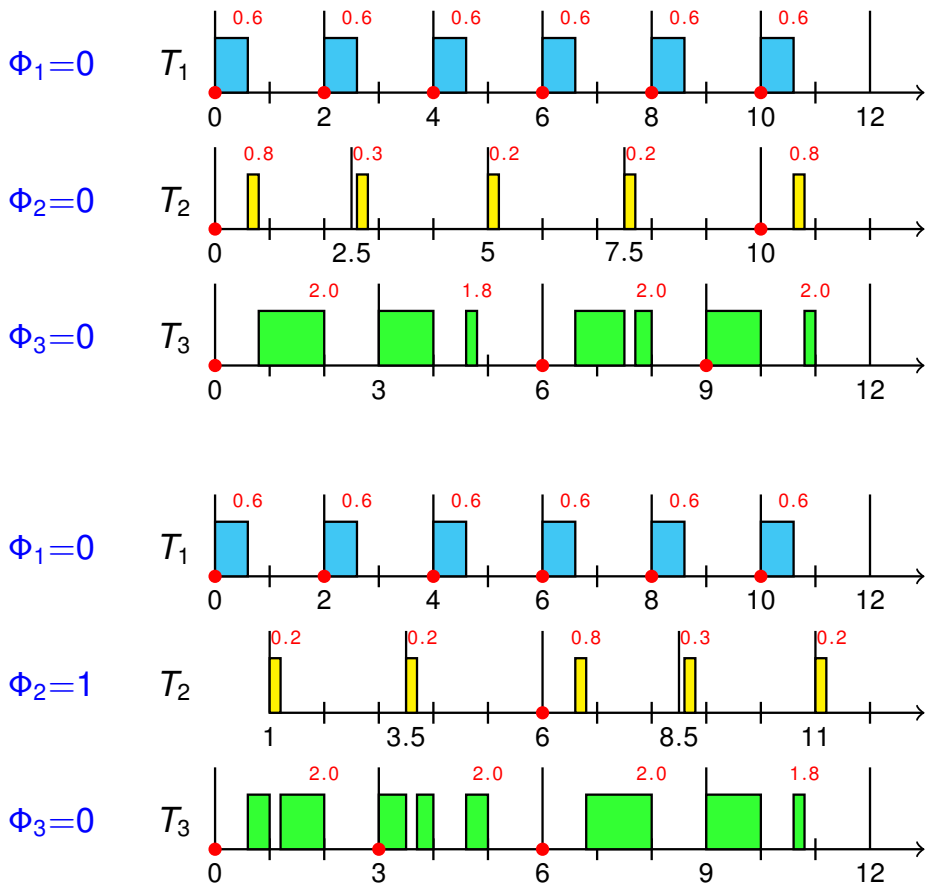
In una sistema con task a priorità fissa e tempi di risposta piccoli, l'istante in cui uno dei job di T_i viene rilasciato contemporaneamente ai job di *tutti* i task con priorità maggiore di T_i è un istante critico di T_i

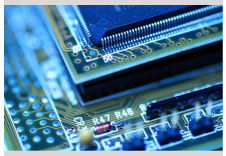
Schedulabilità per RM e DM
Marco Cesati



Schema della lezione
Test di schedulabilità
Test di schedulabilità generale
Condizioni di schedulabilità

Istanti critici per T_1=(2, 0.6), T_2=(2.5, 0.2), T_3=(3, 1.2)





Schedulabilità per priorità fissa e tempi di risposta piccoli

Supponiamo che in un sistema con task a priorità fissa i tempi di risposta siano piccoli

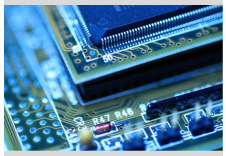
Definizione della funzione di tempo necessario

Siano dati i task $T_1, \ldots, T_i$ in fase al tempo t_0 con priorità decrescenti. Il tempo necessario per eseguire tutti i job dei task $T_1, \ldots T_i$ nell'intervallo $[t_0, t_0 + t]$ $(t \leq p_i)$ è

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

Test di schedulabilità (Lehoczky, Sha, Ding 1989)

Siano dati i task $T_1, \ldots, T_i$ in fase al tempo t_0 con priorità decrescenti, con $T_1, \ldots, T_{i-1}$ effettivamente schedulabili. Il task T_i può essere schedulato nell'intervallo di tempo $[t_0, t_0 + D_i]$ se esiste $t \leq D_i$ tale che $w_i(t) \leq t$

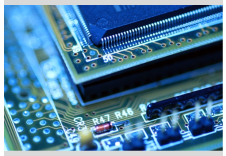


Applicazione del test di schedulabilità

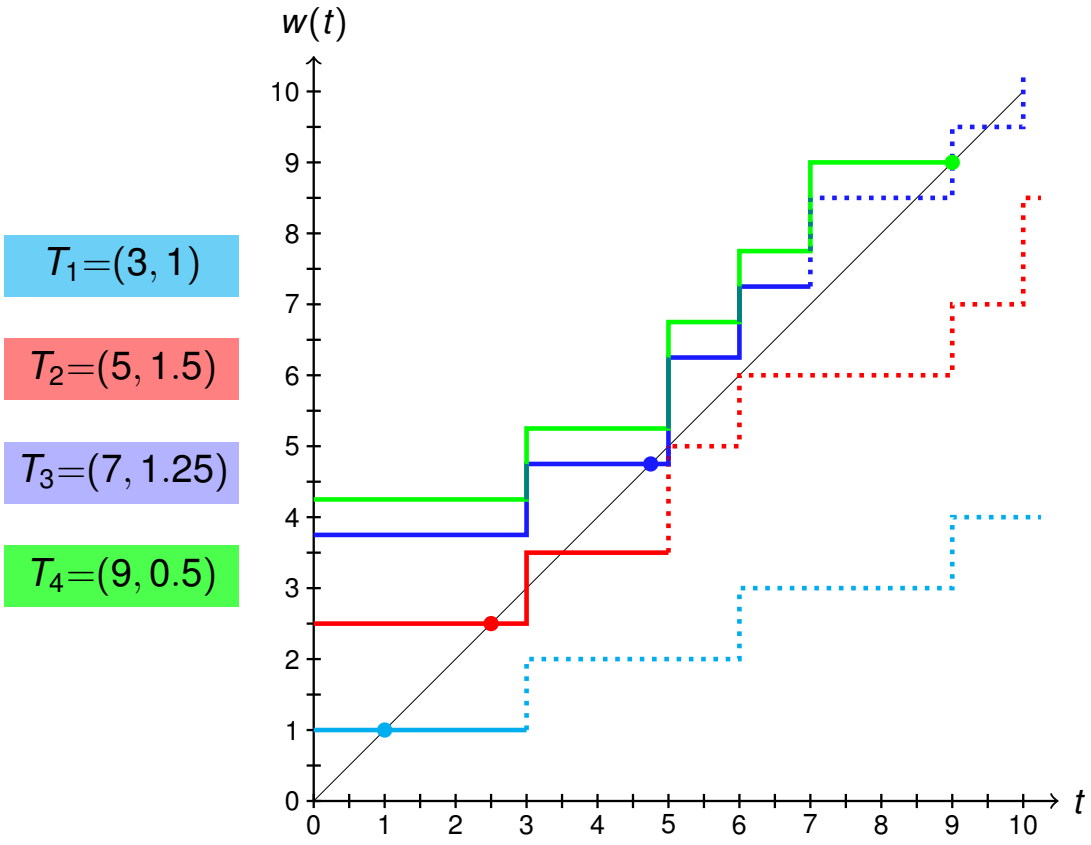
- Siano dati i task $T_1, T_2, \ldots, T_n$ con priorità decrescenti
- Si considera un task T_i alla volta, cominciando da quello con priorità maggiore (T_1)
- Si calcola il valore della funzione di tempo necessario $w_i(t)$ per tutti i valori di $t \leq D_i$ tali che t è un multiplo intero di p_k, per $k \in \{1, 2, \ldots i\}$
- Se per almeno uno di questi valori di t vale $w_i(t) \leq t$ allora T_i è effettivamente schedulabile
- Altrimenti il test fallisce: un job di T_i *potrebbe* mancare la propria scadenza

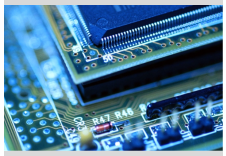
Esempio: T_1=(3, 1), T_2=(5, 1.5), T_3=(7, 1.25), T_4=(9, 0.5)

t	3	5	6	7	9
$w_1(t)$	1.0				
$w_2(t)$	2.5	3.5			
$w_3(t)$	3.75	4.75	6.25	7.25	
$w_4(t)$	4.25	5.25	6.75	7.75	9.0



Applicazione del test di schedulabilità (2)





Massimo tempo di risposta di un task

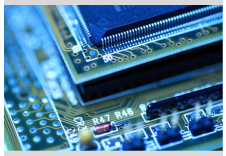
Il massimo tempo di risposta W_i di un task T_i in un sistema a priorità fissa con tempi di risposta piccoli è:

$$W_i = \min\{t \leq D_i : t = w_i(t)\}$$

Se l'equazione $w_i(t) = t$ non ha soluzioni minori o uguali a D_i, allora qualche job in T_i mancherà la propria scadenza relativa

Algoritmo per il calcolo di W_i (Audsley et al., 1993):

- $t^{(1)} = e_i$ (prima approssimazione)
- $t^{(k+1)} = w_i(t^{(k)})$
- continuare a iterare finché
  - $t^{(k+1)} = t^{(k)}$ e $t^{(k)} \leq D_i \quad \Rightarrow \quad W_i = t^{(k)}$
  - $t^{(k+1)} > D_i \quad \Rightarrow$ non schedulabile



Alternative al test di schedulabilità

Il test di schedulabilità assume il caso peggiore e analizza le richieste di tempo dei task

Non sarebbe molto più semplice provare a simulare la schedulazione? ***Non sempre***

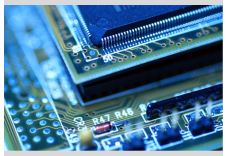
Per i sistemi più semplici simulare la schedulazione è concettualmente più facile

D'altra parte per i sistemi più complicati non è sempre possibile simulare, ad esempio perché

- non è possibile determinare facilmente il caso peggiore
- il caso peggiore cambia da task a task
- è difficile integrare nella simulazione altri fattori (come job parzialmente non interrompibili) che invece possono essere considerati estendendo il test di schedulabilità

In ogni caso, sia la simulazione che il test di schedulabilità hanno complessità asintotica pari a $O(n \cdot \frac{p_n}{p_1})$

Schedulabilità per RM e DM
Marco Cesati



Schema della lezione
Test di schedulabilità
Test di schedulabilità generale
Condizioni di schedulabilità

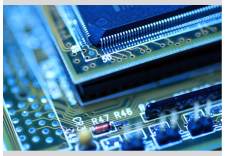
Task periodici con tempi di risposta arbitrari

Finora abbiamo considerato task a priorità fissa con tempi di risposta piccoli

Analizzare un sistema di task con tempi di risposta *arbitrari* è più difficile:

- un job non deve completare necessariamente prima che il successivo job dello stesso task sia rilasciato
- è ammissibile che $D_i > p_i$
- vi possono essere nello stesso istante più job di uno stesso task in attesa di essere eseguiti
- un job rilasciato contemporaneamente a tutti i job dei task con priorità maggiore non ha necessariamente il massimo tempo di risposta possibile

Come sempre assumeremo che i job di uno stesso task pronti per l'esecuzione verranno eseguiti in modalità FIFO



Intervalli totalmente occupati

Come analizzare i sistemi di task con tempi di risposta grandi?

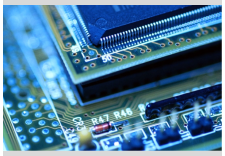
Sia dato un insieme di task $\mathcal{T}_i = \{T_1, \ldots, T_i\}$ con valori di priorità $\pi_1 < \pi_2 < \ldots < \pi_i$ (valore minore = priorità maggiore)

Si definisce un *intervallo totalmente occupato* (*busy interval*) di livello π_i un intervallo di tempo $(t_0, t_1]$ tale che

1. all'istante t_0 tutti i job di $\mathcal{T}_i$ rilasciati prima di t_0 sono stati completati
2. all'istante t_0 un job di $\mathcal{T}_i$ è rilasciato
3. l'istante t_1 è il primo istante in cui tutti i job di $\mathcal{T}_i$ rilasciati a partire da t_0 sono stati completati

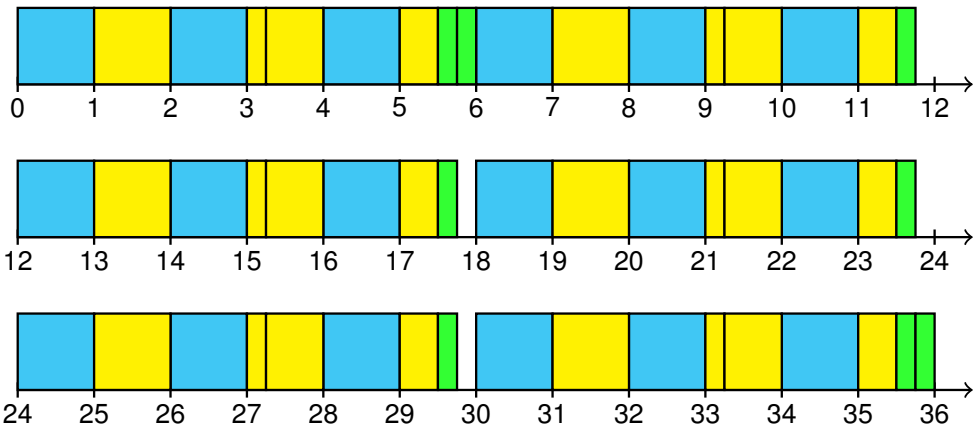
È possibile che in un intervallo totalmente occupato il processore sia idle od esegua task non in $\mathcal{T}_i$? ***No!***

- Se il processore fosse idle, l'intervallo terminerebbe prima
- $\mathcal{T}_i$ contiene tutti gli i task di priorità maggiore, quindi nessun task al di fuori di $\mathcal{T}_i$ può essere eseguito



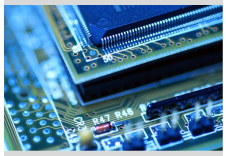
Esempio di intervalli totalmente occupati

Task: T_1=(2, 1), T_2=(3, 1.25), T_3=(5, 0.25)



- Intervalli di livello π_1=1: $(0, 1]$, $(2, 3]$, $(4, 5]$, ...
- Intervalli di livello π_2=2: $(0, 5.5]$, $(6, 11.5]$, $(12, 17.5]$, ...
- Intervalli di livello π_3=3: $(0, 6]$, $(6, 11.75]$, $(12, 17.75]$,...

Tutti gli intervalli di livello 1 e 2 sono *in fase*: i primi job di tutti i task nell'intervallo sono rilasciati all'inizio



Test di schedulabilità generale

Il test di schedulabilità generale per tempi di risposta arbitrari (Lehoczky 1990) è ancora basato sull'analisi del caso peggiore (task in fase)

La differenza rispetto al test per tempi di risposta piccoli è che il primo job rilasciato contemporaneamente agli altri potrebbe non avere il massimo tempo di risposta

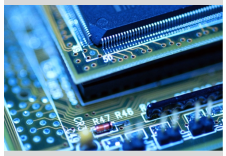
Idea: per ogni task T_i, analizzare *tutti* i job di T_i eseguiti nel primo intervallo totalmente occupato di livello π_i

Come determinare l'intervallo totalmente occupato?

- Inizio determinato dal rilascio dei primi job (in fase!) dei task in $\mathcal{T}_i = \{T_1, \ldots, T_i\}$
- Lunghezza calcolata risolvendo iterativamente

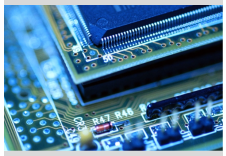
$$t = \sum_{k=1}^{i} \left\lceil \frac{t}{p_k} \right\rceil e_k$$

(la prima approssimazione può essere $\sum_{k=1}^{i} e_k$)



Test di schedulabilità generale (2)

- Siano dati i task in $\{T_1, \ldots, T_n\}$ con priorità $\pi_1 < \ldots < \pi_n$; si considera un task T_i alla volta, cominciando dal task di massima priorità T_1
- Caso peggiore per la schedulabilità di T_i: assumere che i task in $\mathcal{T}_i = \{T_1, \ldots, T_i\}$ sono in fase
- Se il primo job di *tutti* i task in $\mathcal{T}_i$ termina entro il primo periodo del task: decidere se T_i è schedulabile controllando se $J_{i,1}$ termina entro la scadenza tramite la funzione di tempo richiesto $w_{i,1}(t) := w_i(t)$
- Altrimenti almeno un primo job di $\mathcal{T}_i$ termina dopo il periodo del task; calcolare la lunghezza t^L dell'intervallo totalmente occupato di livello π_i che inizia da $t = 0$
- Calcolare i tempi di risposta massimi di tutti i $\lceil t^L/p_i \rceil$ job di T_i nell'intervallo
- Decidere se T_i è schedulabile controllando se tutti i tempi di risposta rispettano la scadenza



Test di schedulabilità generale (3)

Come calcolare i tempi di risposta di tutti i job di T_i nell'intervallo totalmente occupato di livello π_i?

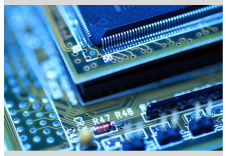
Lemma

Il tempo di risposta massimo $W_{i,j}$ del j-esimo job di T_i in un intervallo totalmente occupato di livello π_i in fase è uguale al minimo valore di t che soddisfa l'equazione:

$$t = w_{i,j}(\, t + (j-1)\, p_i\,) - (j-1)\, p_i$$

$$\text{con} \quad w_{i,j}(t) = j\, e_i + \sum_{k=1}^{i-1} \left\lceil \frac{t}{p_k} \right\rceil e_k$$

In pratica per controllare se il j-esimo job di T_i rispetta la scadenza relativa è sufficiente verificare se la diseguaglianza $w_{i,j}(t) \leq t$ è soddisfatta per qualche $t \in [(j-1)p_i, (j-1)p_i + D_i]$



Esempio di test di schedulabilità generale

Task: T_1=($\Phi_1, 2, 1, 1$), T_2=($\Phi_2, 3, 1.25, 4$), T_3=($\Phi_3, 5, 0.25, 7$)
(le fasi iniziali non sono note)

Verifica di T_1

$w_1(t) = w_{1,1}(t) = e_1 = 1 = D_1 \quad \Rightarrow T_1$ OK!

Verifica di T_2

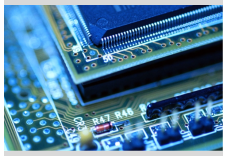
$w_{2,1}(2) = e_2 + e_1 = 2.25 > 2 \quad \Rightarrow$ no
$w_{2,1}(3) = e_2 + 2e_1 = 1.25 + 2 = 3.25 > 3 \quad \Rightarrow$ no
$w_{2,1}(4) = e_2 + 2e_1 = 1.25 + 2 = 3.25 \leq 4 \leq D_2 \quad \Rightarrow J_{2,1}$ ok

Lunghezza del busy interval di livello 2

$t^{(1)} = e_1 + e_2 = 2.25, \quad t^{(2)} = 2e_1 + e_2 = 3.25$
$t^{(3)} = 2e_1 + 2e_2 = 4.5, \quad t^{(4)} = 3e_1 + 2e_2 = 5.5$
$t^{(5)} = 3e_1 + 2e_2 = 5.5 = t^{(4)} = t^L$

job di T_2 in $(0, 5.5]$: $\lceil t^L/p_2 \rceil = 2$

$w_{2,2}(3) = 2e_2 + 2e_1 = 4.5 > 3 \quad \Rightarrow$ no
$w_{2,2}(4) = 2e_2 + 2e_1 = 4.5 > 4 \quad \Rightarrow$ no
$w_{2,2}(6) = 2e_2 + 3e_1 = 5.5 \leq 6 \leq p_2 + D_2 = 7 \quad \Rightarrow J_{2,2}$ ok
$\Rightarrow T_2$ OK!



Esempio di test di schedulabilità generale (2)

Verifica di T_3

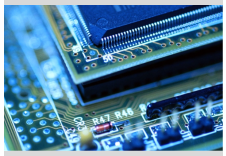
Lunghezza del busy interval di livello 3

$t^{(1)} = e_1 + e_2 + e_3 = 2.5, \quad t^{(2)} = 2e_1 + e_2 + e_3 = 3.5$
$t^{(3)} = 2e_1 + 2e_2 + e_3 = 4.75, \quad t^{(4)} = 3e_1 + 2e_2 + e_3 = 5.75$
$t^{(5)} = 3e_1 + 2e_2 + 2e_3 = 6, \quad t^{(6)} = 3e_1 + 2e_2 + 2e_3 = 6 = t^L$

job di T_3 in $(0, 6]$: $\lceil t^L/p_3 \rceil = 2$

$w_{3,1}(2) = e_3 + e_1 + e_2 = 2.5 > 2 \quad \Rightarrow$ no
$w_{3,1}(3) = e_3 + 2e_1 + e_2 = 3.5 > 3 \quad \Rightarrow$ no
$w_{3,1}(4) = e_3 + 2e_1 + 2e_2 = 4.75 > 4 \quad \Rightarrow$ no
$w_{3,1}(5) = e_3 + 3e_1 + 2e_2 = 5.75 > 5 \quad \Rightarrow$ no
$w_{3,1}(6) = e_3 + 3e_1 + 2e_2 = 5.75 \leq 6 \leq D_3 = 7 \quad \Rightarrow J_{3,1}$ ok
$w_{3,2}(5) = 2e_3 + 3e_1 + 2e_2 = 6 > 5 \quad \Rightarrow$ no
$w_{3,2}(6) = 2e_3 + 3e_1 + 2e_2 = 6 \leq 6 \leq p_3 + D_3 = 12 \quad \Rightarrow J_{3,2}$ ok
$\Rightarrow T_3$ OK!

Risultato: il sistema $\{T_1, T_2, T_3\}$ è schedulabile qualunque siano le fasi dei task



Condizioni di schedulabilità

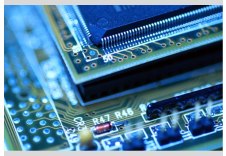
Il test di schedulabilità generale determina se un insieme di task con fasi sconosciute può essere effettivamente schedulato

Quali sono i suoi limiti?

- Non può essere applicato se non sono noti periodi, scadenze e tempi di esecuzione dei task
- Il risultato ottenuto non è più valido se un task varia periodo, scadenza o tempo d'esecuzione
- È computazionalmente costoso $\Rightarrow$ poco adatto per scheduling on-line

Una condizione di schedulabilità è una condizione sufficiente per la schedulabilità di un sistema di task calcolabile velocemente ed applicabile anche quando alcuni parametri temporali dei task non sono noti

Ad esempio, il fattore di utilizzazione dell'algoritmo EDF fornisce una condizione di schedulabilità: $U_{\mathcal{T}} \leq U_{\text{EDF}} = 1$



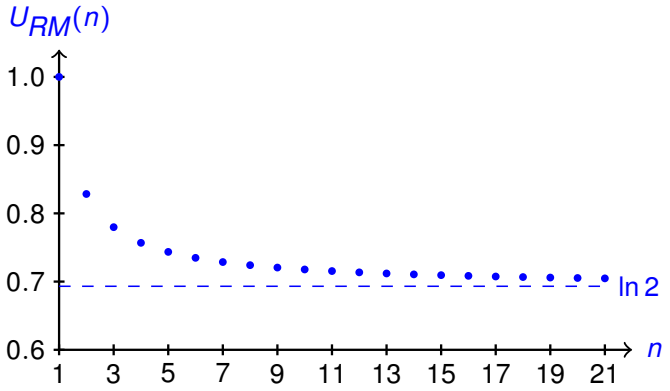
Condizione di schedulabilità di RM

Condizione di Liu-Layland (1973)

Un sistema $\mathcal{T}$ di n task indipendenti ed interrompibili con scadenze relative uguali ai rispettivi periodi ($D_i = p_i$) può essere effettivamente schedulato su un processore in accordo all'algoritmo RM se il suo fattore di utilizzazione $U_{\mathcal{T}}$ è minore od uguale a

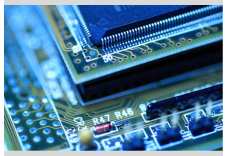
$$U_{RM}(n) = n\left(2^{1/n} - 1\right)$$

$$\lim_{n\to\infty} U_{RM}(n) = \ln 2 = 0.693$$



Nota: nelle condizioni del teorema, RM $\equiv$ DM

Schedulabilità per RM e DM
Marco Cesati



Schema della lezione
Test di schedulabilità
Test di schedulabilità generale
Condizioni di schedulabilità

Esempi di applicazione di $U_{RM}(n)$

Il sistema T_1=(1, 0.25), T_2=(1.25, 0.1), T_3=(1.5, 0.3), T_4=(1.75, 0.07), T_5=(2, 0.1) ha fattore di utilizzazione

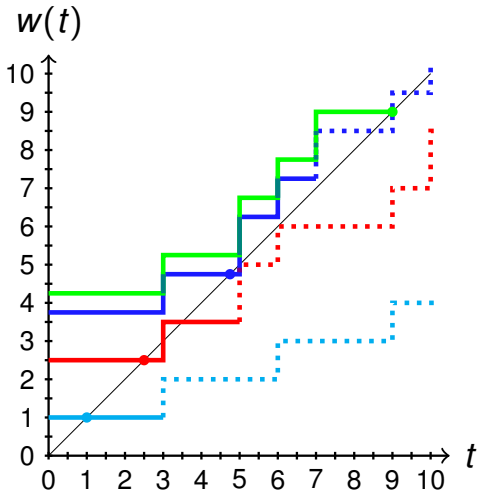
$U_{\mathcal{T}} = 0.62 \leq 0.743 = U_{RM}(5)$ $\Rightarrow$ è schedulabile con RM

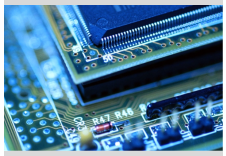
Il sistema T_1=(3, 1), T_2=(5, 1.5), T_3=(7, 1.25), T_4=(9, 0.5) ha fattore di utilizzazione

$U_{\mathcal{T}} = 0.867 > 0.757 = U_{RM}(4)$ $\Rightarrow$ *forse* non schedulabile

In realtà è schedulabile!

t	3	5	6	7	9
$w_1(t)$	1.0				
$w_2(t)$	2.5	3.5			
$w_3(t)$	3.75	4.75	6.25	7.25	
$w_4(t)$	4.25	5.25	6.75	7.75	9.0





Test iperbolico per RM

Test iperbolico (Bini, Buttazzo, Buttazzo 2001)

Un sistema $\mathcal{T}$ di n task indipendenti ed interrompibili con scadenze relative uguali ai rispettivi periodi ($D_i = p_i$) può essere effettivamente schedulato su un processore con RM se

$$\prod_{k=1}^{n} \left(1 + \frac{e_k}{p_k}\right) \leq 2$$

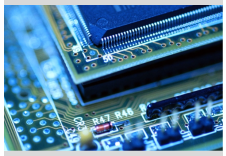
Si applica conoscendo solo il fattore di utilizzazione u_k dei task

Come sono correlati test iperbolico e condizione di Liu-Layland?

Assumendo per ogni task $u_k = U_{\mathcal{T}}/n$:

$$\prod_{k=1}^{n} \left(1 + \frac{U_{\mathcal{T}}}{n}\right) \leq 2 \iff U_{\mathcal{T}} \leq n\left(2^{1/n} - 1\right)$$

Esistono casi in cui il test iperbolico è soddisfatto ma la condizione di Liu-Layland non lo è



Test per sottoinsiemi di task armonici

Condizione di Kuo-Mok (1991)

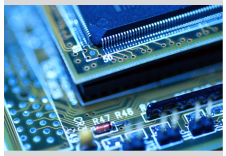
Se un sistema $\mathcal{T}$ di task periodici, indipendenti ed interrompibili con $p_i = D_i$ può essere partizionato in n_h sottoinsiemi disgiunti $\mathcal{Z}_1, \ldots \mathcal{Z}_{n_h}$, ciascuno dei quali contiene task semplicemente periodici, allora il sistema è schedulabile con RM se

$$\sum_{k=1}^{n_h} U_{\mathcal{Z}_k} \leq U_{RM}(n_h) \quad \text{oppure se} \quad \prod_{k=1}^{n_h} (1 + U_{\mathcal{Z}_k}) \leq 2$$

Se un sistema ha poche applicazioni molto complesse, è possibile migliorare la schedulabilità rendendo i task di ciascuna applicazione semplicemente periodici

Esempio: dati 9 task con periodi 4, 7, 8, 14, 16, 28, 32, 56, 64, il fattore di utilizzazione di Liu-Layland è $U_{RM}(9)$=0.720

Partizionando in due sottoinsiemi (potenze di 2 e multipli di 7): $U_{\mathcal{Z}_1} + U_{\mathcal{Z}_2} \leq U_{RM}(2) = 0.828$



Test per task quasi armonici

Il fattore di utilizzazione di RM è in generale $U_{RM}(n)$, ma diventa uguale a 1 per task semplicemente periodici

È possibile migliorare $U_{RM}(n)$ considerando quanto i periodi dei task sono vicini ad essere armonici?

Sia $X_i = \log_2 p_i - \lfloor \log_2 p_i \rfloor$ e $\zeta = \max_{1 \leq i \leq n} X_i - \min_{1 \leq i \leq n} X_i$

Teorema (Burchard, Liebeherr, Oh, Son 1996)

Nelle ipotesi della condizione di Liu-Layland, il fattore di utilizzazione di RM dipendente dal numero di task n e da ζ è:

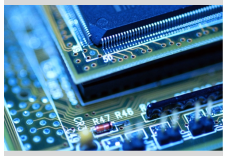
$$U_{RM}(n, \zeta) = \begin{cases} (n-1)\left(2^{\zeta/(n-1)} - 1\right) + 2^{1-\zeta} - 1 & \text{se } \zeta < 1 - 1/n \\ U_{RM}(n) & \text{se } \zeta \geq 1 - 1/n \end{cases}$$

Quando si verifica il caso $\zeta = 0$?

Quando $p_i = K \cdot 2^{x_i}$ (quindi i task sono armonici)

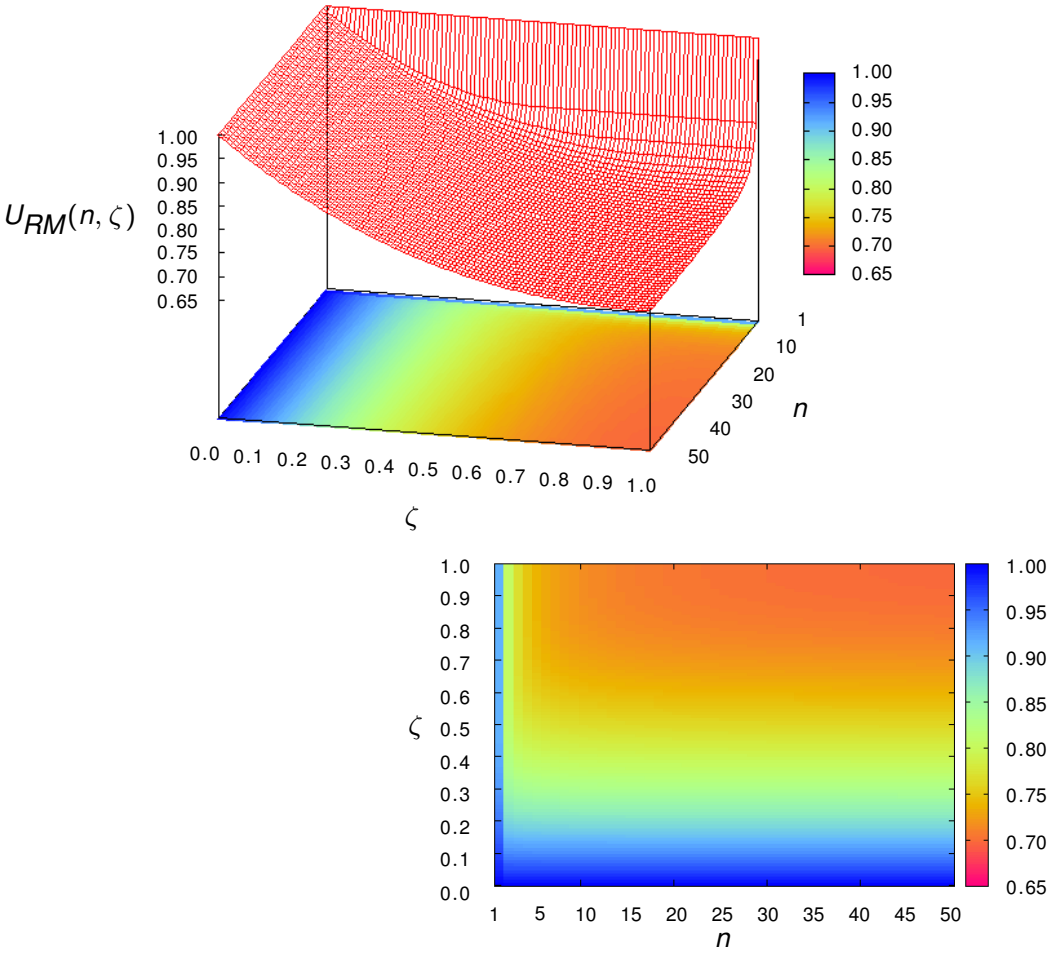
Non è vero il contrario: ad es., periodi 3, 6, 9 $\Rightarrow$ $\zeta = 0.415$

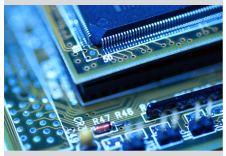
Schedulabilità per RM e DM
Marco Cesati



Schema della lezione
Test di schedulabilità
Test di schedulabilità generale
Condizioni di schedulabilità

La funzione $U_{RM}(n, \zeta)$





Schedulabilità per scadenze arbitrarie

Se per qualche task $D_i < p_i$, il limite $U_{RM}(n)$ è valido? ***No!***

La formula non è valida perché assume che la scadenza di ciascun job sia l'inizio del periodo successivo

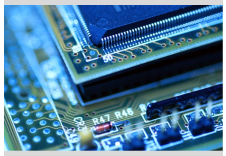
Se per qualche task $D_i > p_i$, il limite $U_{RM}(n)$ è valido? ***Sì!***

Però la formula è "pessimista": forse è possibile trovare valori di soglia superiori a $U_{RM}(n)$

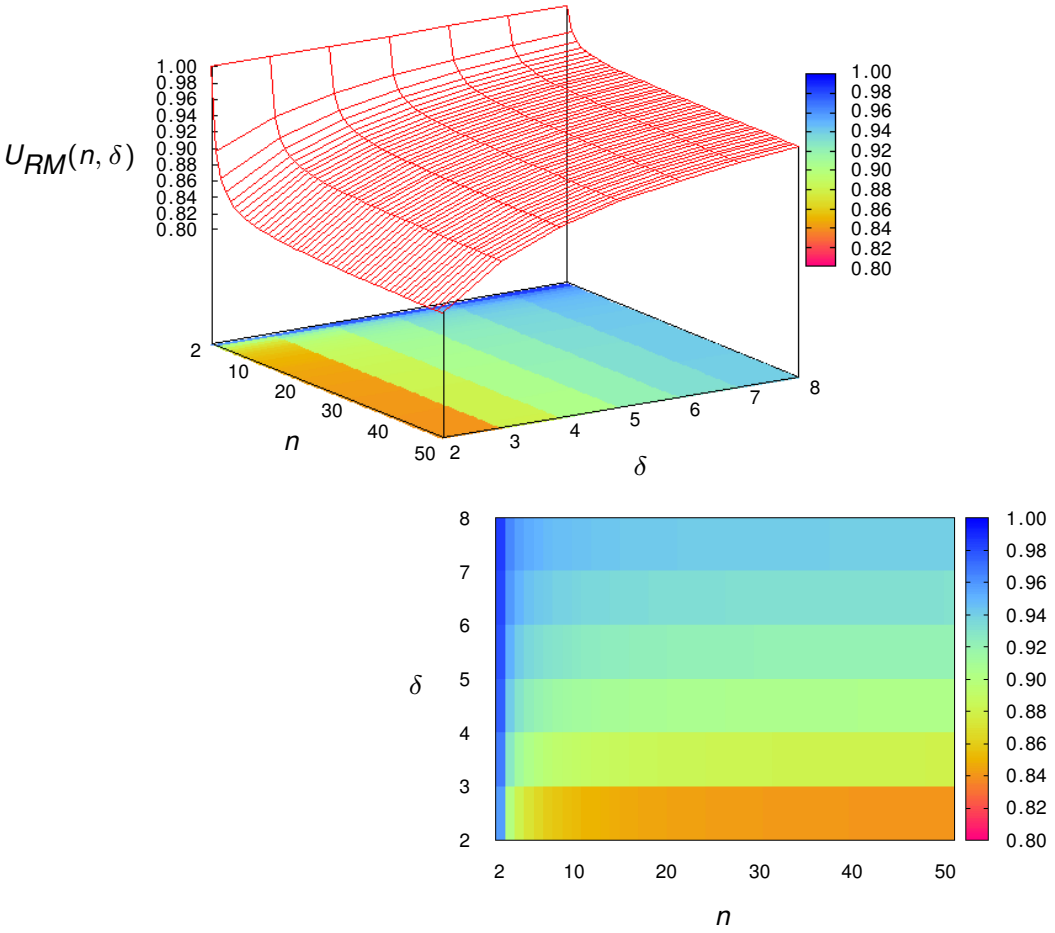
Teorema (Lehoczky, Sha, Strosnider, Tokuda 1986, 1990, 1991)

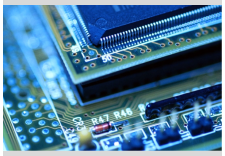
Un sistema $\mathcal{T}$ di n task indipendenti, interrompibili e con $D_i = \delta \cdot p_i$ è schedulabile con RM se $U_{\mathcal{T}}$ è minore o uguale a

$$U_{RM}(n, \delta) = \begin{cases} \delta(n-1)\left[\left(\frac{\delta+1}{\delta}\right)^{1/(n-1)} - 1\right] & \text{per } \delta = 2, 3, \ldots \\ n\left((2\delta)^{1/n} - 1\right) + 1 - \delta & \text{per } 0.5 \leq \delta \leq 1 \\ \delta & \text{per } 0 \leq \delta \leq 0.5 \end{cases}$$



La funzione $U_{RM}(n, \delta)$ per $\delta \in \{2, 3, \ldots\}$





La funzione $U_{RM}(n, \delta)$ per $\delta \in [0.5, 1]$