

8/11/22

Schedulabilità per
RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità
generale

Condizioni di
schedulabilità

Lezione R6

Schedulabilità per RM e DM

Sistemi embedded e real-time

15 ottobre 2020

Marco Cesati

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

SERT'20

R6.1

Di cosa parliamo in questa lezione?

Schedulabilità per
RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità
generale

Condizioni di
schedulabilità

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

- ➊ Test di schedulabilità per algoritmi a priorità fissa
- ➋ Test di schedulabilità generale per scadenze arbitrarie
- ➌ Condizioni di schedulabilità per algoritmi a priorità fissata

SERT'20

R6.2

Schedulabilità con algoritmi a priorità fissa

Schedulabilità per RM e DM

Marco Cesati

- Alcuni algoritmi di schedulazione con priorità **dinamica**, ad es. **EDF**, sono ottimali: $U_{EDF} = 1$ (carico al 100% con garanzia sulle scadenze!)
- Nessun algoritmo **X con priorità fissa** per i task può essere ottimale in senso assoluto: $U_X < 1$, ma uguale a 1
- L'algoritmo **RM** (Rate Monotonic) è ottimale (in senso assoluto) per i sistemi di task armonici con scadenze implicite → scad = periodo da completare prima del nuovo periodo → periodi "multipli" tra loro
- L'algoritmo **DM** (Deadline Monotonic) è ottimale tra gli algoritmi a priorità **fissa**, ma non in senso assoluto (3 alcuni task non schedulabili)
- L'algoritmo **RM** è ottimale tra gli algoritmi a priorità **fissa** per sistemi di task con scadenza proporzionale al periodo (perché equivalente a RM)

generale, non armonici! Se lo fossero, basterebbe vedere se U_1 per alg. priorità fissa.

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?

SERT'20

R6.3

Serve saperlo perché in alcuni contesti sono più adatti

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**, ogni Job "si comporta bene" ≠ rispettare scadenze

[io dico solo che finisce PRIMA del periodo, ma nel periodo non lo so]

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 (fallimento scadenza.)

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 . Rilasci in fase ↗ istanti critici

Se ho lo task, n° 9 task superiori rilasciano, e io rilascio ⇒ **ISTANTE CRITICO**

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

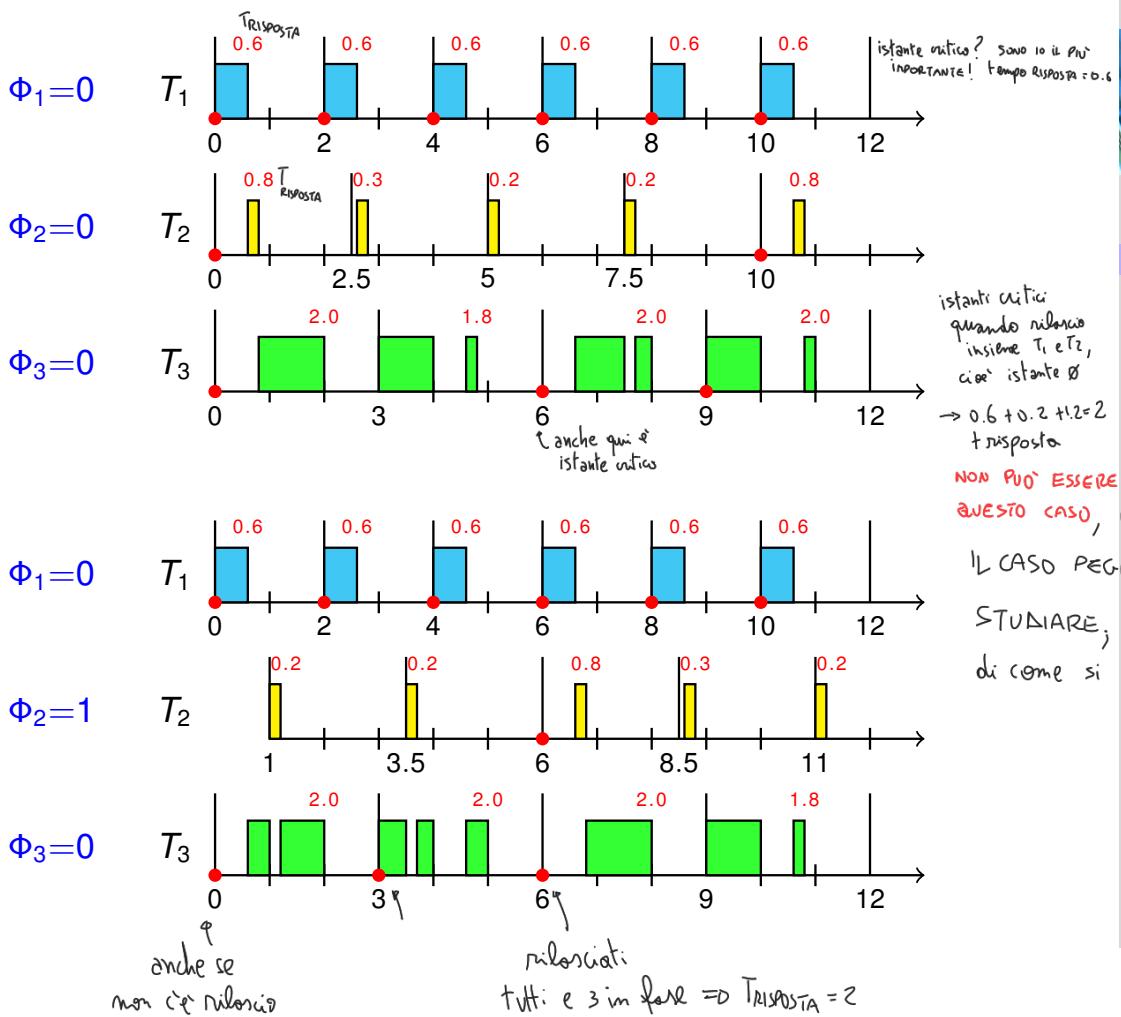
Condizioni di schedulabilità

SERT'20

R6.4

in RM T_1 è prioritario,

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



Schedulabilità per RM e DM
Marco Cesati

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

SERT'20 R6.5

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, \dots, T_i in fase al tempo t_0 con priorità decrescenti. Il tempo necessario per eseguire tutti i job dei task

T_1, \dots, T_i nell'intervallo $[t_0, t_0 + t]$ ($t \leq p_i$) è perché assumo si concludano entro il loro periodo.

ordinati per priorità decrescenti (T_1 più alto) dove sono im fase

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

tempo exec. n° rilasci nel periodo, c'è $\lceil \cdot \rceil$, perchè job rilasciato ha priorità superiore

task priorità impennare

Schedulabilità per RM e DM
Marco Cesati

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

Test di schedulabilità (Lehoczky, Sha, Ding 1989)

Siano dati i task T_1, \dots, T_i in fase al tempo t_0 con priorità decrescenti, con T_1, \dots, T_{i-1} effettivamente schedulabili. (rispettano scadenze)

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 $w_i(t) \leq t$ (eseguo Job + Job prioritari rispetto me)

Scad.
relativa

Applicazione del test di schedulabilità (esempio)

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

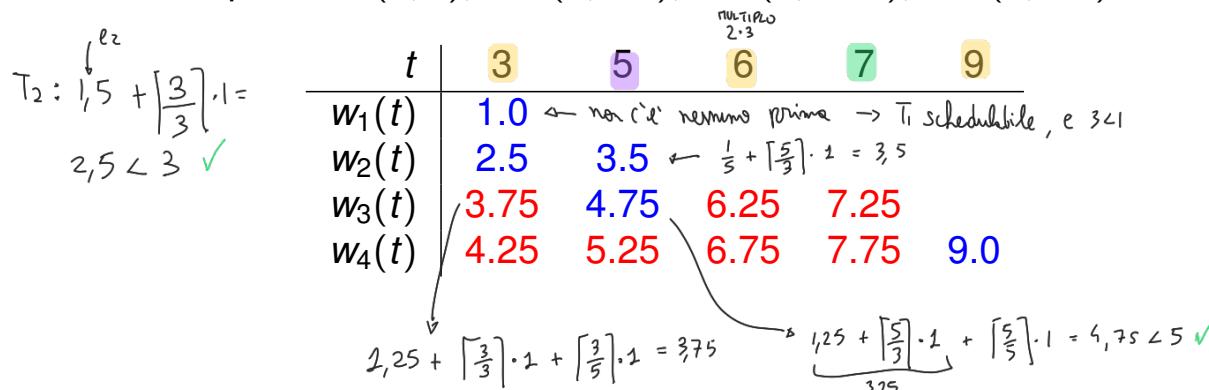
Test di schedulabilità generale

Condizioni di schedulabilità

- Siano dati i task T_1, T_2, \dots, 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 $k \in \{1, 2, \dots, i\}$. Perché w_{ii} è a gradini, perché c'è ponte interno!
- 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

↳ test basato su worst case!

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



SERT'20

R6.7

Applicazione del test di schedulabilità (2)

Schedulabilità per RM e DM

Marco Cesati

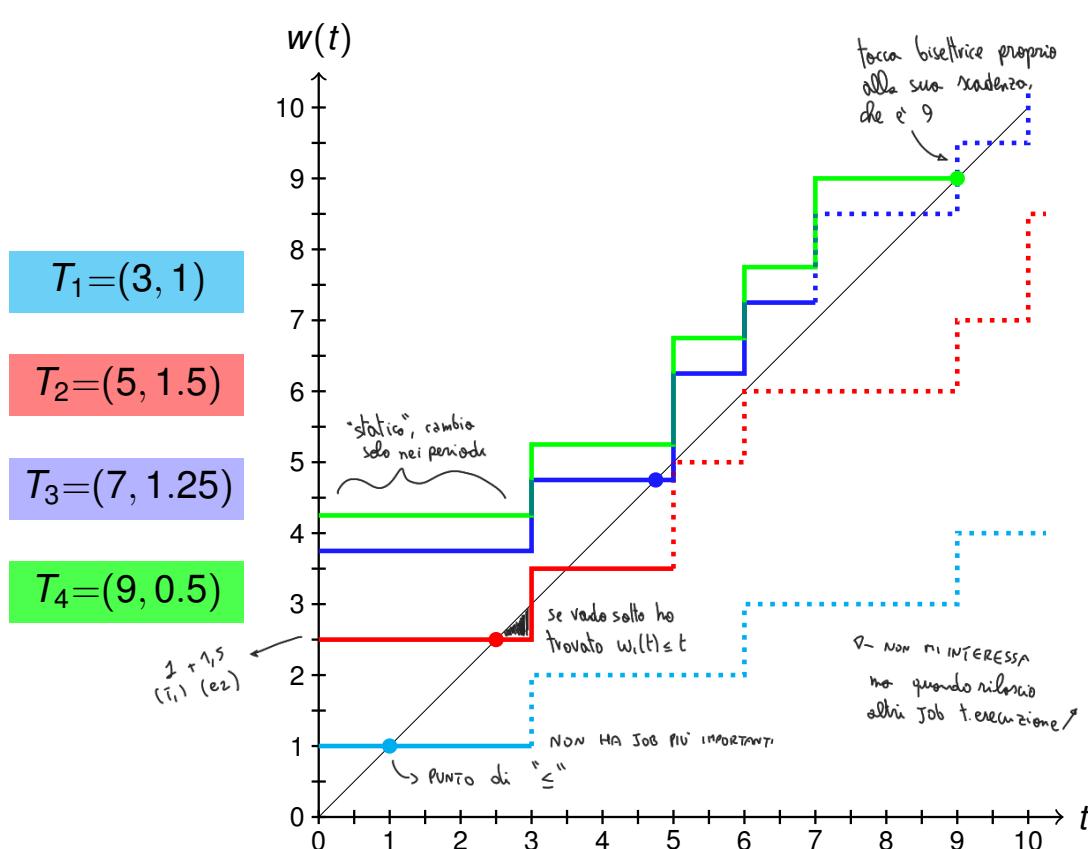


Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità



Le intersezioni con lo bisettrice sono i Tempi di risposta

SERT'20

R6.8

Massimo tempo di risposta di un task

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità

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

$$W_i = \min\{t \leq D_i : t = w_i(t)\} \quad \left(\begin{array}{l} \text{"modo} \\ \text{"alternativo"} \end{array} \right)$$

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 *oltre fallire!*
 - $t^{(k+1)} = t^{(k)}$ e $t^{(k)} \leq D_i \Rightarrow W_i = t^{(k)}$
 - $t^{(k+1)} > D_i \Rightarrow$ non schedulabile

SERT'20

R6.9

Alternative al test di schedulabilità

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni 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 *(Noi ci siamo messi in condizioni semplici)*
- 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 $O(n \cdot \frac{p_n}{p_1})$ Pseudo polinomiale

SERT'20

R6.10

Task periodici con tempi di risposta arbitrari $\left(\begin{smallmatrix} \text{NON PIÙ} \\ \text{PICCOLI} \end{smallmatrix} \right)$

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: (Posso andare oltre il periodo con X-danza relativa)

- un job non deve completare necessariamente prima che il successivo job dello stesso task sia rilasciato
$$\left(\begin{array}{l} \text{Non posso eseguire} \\ J_2 \text{ di } T_i \text{ se } J_1 \text{ di } T_i \\ \text{ancora non concluso} \end{array} \right) \quad \begin{array}{l} \text{Si} \\ \text{accumulano} \end{array}$$
- 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 \rightarrow thm. di prima invalido \rightarrow non ho caso peggiore dato dal thm.

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

Schedulabilità per RM e DM
Marco Cesati



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

SERT'20 R6.11

Intervalli totalmente occupati

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

Sia dato un insieme di task $T_i = \{T_1, \dots, T_i\}$ con valori di priorità $\pi_1 < \pi_2 < \dots < \pi_i$ (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 T_i rilasciati prima di t_0 sono stati completati (ma lavoro smettuto)
- ② all'istante t_0 un job di T_i è rilasciato (rilascio nuovo job)
- ③ l'istante t_1 è il primo istante in cui tutti i job di T_i rilasciati a partire da t_0 sono stati completati (è come fare "ciccare")

È possibile che in un intervallo totalmente occupato il processore sia idle od esegua task non in T_i ? **No!**

- Se il processore fosse idle, l'intervallo terminerebbe prima
- T_i contiene tutti gli i task di priorità maggiore, quindi nessun task al di fuori di T_i può essere eseguito

Schedulabilità per RM e DM
Marco Cesati



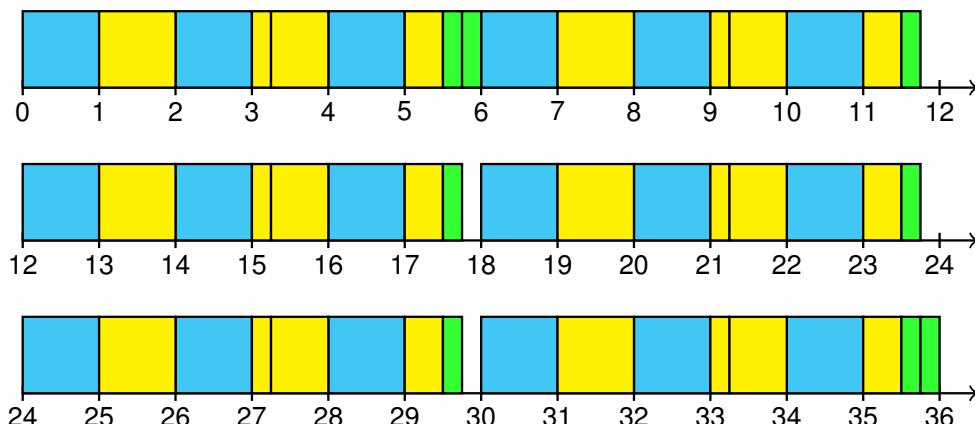
Schema della lezione
Test di schedulabilità
Test di schedulabilità generale
Condizioni di schedulabilità

t_0 è quando diventa idle, se fosse prima, t_0 sarebbe più piccolo)

SERT'20 R6.12

Esempio di intervalli totalmente occupati

Task: $T_1 = (2, 1)$, $T_2 = (3, 1.25)$, $T_3 = (5, 0.25)$ in fase!



- Intervalli di livello $\pi_1=1$: $(0, 1]$, $(2, 3]$, $(4, 5]$, ...
- Intervalli di livello $\pi_2=2$: $(0, 5.5]$, $(6, 11.5]$, $(12, 17.5]$, ...
- Intervalli di livello $\pi_3=3$: $(0, 6]$, $(6, 11.75]$, $(12, 17.75]$, ...

perché a 5.5 c'è T_3 di priorità inferiore che non mi interessa

e 6, non 5.5 perché devo considerare tutti i job di T_3

Tutti gli intervalli di livello 1 e 2 hanno lunghezza massima perché **in fase**: i primi job di tutti i task sono rilasciati all'inizio

quindi davo sfioro o meno? NO

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? (Lunghezza max?)

- Inizio determinato dal rilascio dei primi job (**in fase!**) dei task in $T_i = \{T_1, \dots, T_i\}$
- Lunghezza massima calcolata risolvendo iterativamente

$$t \stackrel{?}{=} \sum_{k=1}^i \left\lceil \frac{t}{p_k} \right\rceil e_k \quad \begin{array}{l} \text{sostituisco fino} \\ \text{a trovare "t".} \end{array}$$

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

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità

SERT'20

R6.13

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità

SERT'20

R6.14

Test di schedulabilità generale (2)

- Siano dati i task in $\{T_1, \dots, T_n\}$ con priorità $\pi_1 < \dots < \pi_n$; 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 $\mathcal{T}_i = \{T_1, \dots, T_i\}$ 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 $w_{i,1}(t) := w_i(t)$
- (scad oltre il periodo) ↗ parto da test per tempi di risposta piccoli se $D_i < p_i$
- Altrimenti almeno un primo job di T_i termina dopo il periodo del task; calcolare la lunghezza t^L dell'intervallo totalmente occupato di livello π_i che inizia da $t = 0$ (con alg. iterativo)
- Calcolare i tempi di risposta massimi di tutti i $\lceil t^L/p_i \rceil$ job di T_i nell'intervallo totalmente occupato
- Decidere se T_i è schedulabile controllando se tutti i tempi di risposta rispettano la scadenza

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità

parto da test
per tempi di risposta piccoli
se $D_i < p_i$

SERT'20

R6.15

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 ?

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità

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:

$$t = w_{i,j}(t + (j-1)p_i) - (j-1)p_i \stackrel{\text{Job esegibile}}{\Leftarrow} \text{da rilascio a scadenza}$$

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

per j -esimo job conteggio $j.e_i$

Esempio: se T_1 ha 3 job, il 3° job viene completato dopo 1° e 2° job, ci metto 3*tempoEsecuzione (tempi job uguali)

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 $t \in [(j-1)p_i, (j-1)p_i + D_i]$

SERT'20

R6.16

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) il test è ok per fase non specificata

Schedulabilità per RM e DM

Marco Cesati



Verifica di T_1

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

Verifica di T_2

$$w_{2,1}(2) = e_2 + e_1 = 2.25 > 2 \Rightarrow \text{no}$$

$$w_{2,1}(3) = e_2 + 2e_1 = 1.25 + 2 = 3.25 > 3 \Rightarrow \text{no}$$

$$w_{2,1}(4) = e_2 + 2e_1 = 1.25 + 2 = 3.25 \leq 4 \leq D_2 \Rightarrow J_{2,1} \text{ ok}$$

T2 è schedulabile, ma ha completato oltre il periodo \Rightarrow non posso più considerare tempi piccoli, devo considerare gli intervalli totalmente occupati, uso l'equazione iterativa:

Lunghezza del busy interval di livello 2

$$t^{(1)} = e_1 + e_2 = 2.25, \quad t^{(2)} = 2e_1 + e_2 = 3.25$$

$$t^{(3)} = 2e_1 + 2e_2 = 4.5, \quad t^{(4)} = 3e_1 + 2e_2 = 5.5$$

$$t^{(5)} = 3e_1 + 2e_2 = 5.5 = t^L \Rightarrow \text{intervolo tot. occupato è } 5.5 \text{ (per livello 2)}$$

$$\# \text{ job di } T_2 \text{ in } (0, 5.5]: \lceil t^L/p_2 \rceil = 2 \text{ task di } T_2 \text{ rilasciati}$$

$$w_{2,2}(3) = 2e_2 + 2e_1 = 4.5 > 3 \Rightarrow \text{no}$$

$$w_{2,2}(4) = 2e_2 + 2e_1 = 4.5 > 4 \Rightarrow \text{no}$$

$$w_{2,2}(6) = 2e_2 + 3e_1 = 5.5 \leq 6 \leq p_2 + D_2 = 7 \Rightarrow J_{2,2} \text{ ok}$$

$$\begin{array}{c} \downarrow \quad \uparrow \quad \uparrow \\ \text{periodo: 3} \quad \text{di: } T_2 \end{array} \Rightarrow \text{applicato busy interval una volta, devo calcolarlo sempre perché ho ecceduto!}$$

Esempio di test di schedulabilità generale

(2)

Verifica di T_3

Lunghezza del busy interval di livello 3

$$t^{(1)} = e_1 + e_2 + e_3 = 2.5, \quad t^{(2)} = 2e_1 + e_2 + e_3 = 3.5$$

$$t^{(3)} = 2e_1 + 2e_2 + e_3 = 4.75, \quad t^{(4)} = 3e_1 + 2e_2 + e_3 = 5.75$$

$$t^{(5)} = 3e_1 + 2e_2 + 2e_3 = 6, \quad t^{(6)} = 3e_1 + 2e_2 + 2e_3 = 6 = t^L$$

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

$$w_{3,1}(2) = e_3 + e_1 + e_2 = 2.5 > 2 \Rightarrow \text{no}$$

$$w_{3,1}(3) = e_3 + 2e_1 + e_2 = 3.5 > 3 \Rightarrow \text{no}$$

$$w_{3,1}(4) = e_3 + 2e_1 + 2e_2 = 4.75 > 4 \Rightarrow \text{no}$$

$$w_{3,1}(5) = e_3 + 3e_1 + 2e_2 = 5.75 > 5 \Rightarrow \text{no}$$

$$w_{3,1}(6) = e_3 + 3e_1 + 2e_2 = 5.75 \leq 6 \leq D_3 = 7 \Rightarrow J_{3,1} \text{ ok}$$

1° job
di
T₃

2° job
di
T₃

$$w_{3,2}(5) = 2e_3 + 3e_1 + 2e_2 = 6 > 5 \Rightarrow \text{no}$$

$$w_{3,2}(6) = 2e_3 + 3e_1 + 2e_2 = 6 \leq 6 \leq p_3 + D_3 = 12 \Rightarrow J_{3,2} \text{ ok}$$

$$\Rightarrow T_3 \text{ OK!}$$

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità

Risultato: il sistema $\{T_1, T_2, T_3\}$ è schedulabile qualunque siano le fasi dei task, se no non posso garantire nulla!

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

test pesante, allora ↴

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: $U_T \leq U_{EDF} = 1$

Posso fare così per alg. Priorità fissa? MAGARI mettendo $U_T \leq x < 1$

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità

SERT'20

R6.19

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_T è minore od uguale a

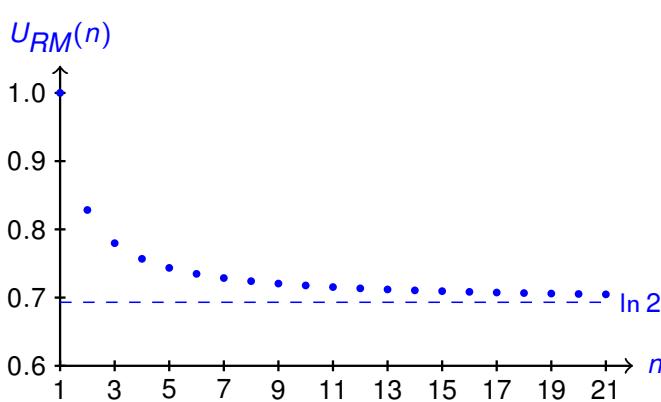
$$U_{RM}(n) = n \left(2^{1/n} - 1 \right)$$

$$\lim_{n \rightarrow \infty} U_{RM}(n) =$$

$$= \ln 2 = 0.693$$

Se proc è critico

meno del 69,3% allora ho scadenze rispettate



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

SERT'20

R6.20

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

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

$0.25/1 + 0.1/1.25 + 0.3/1.5 + 0.07/1.75 + 0.1/2$ cioè sommatoria di e(i)/

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

$$U_T = 0.867 > 0.757 = U_{RM}(4)$$

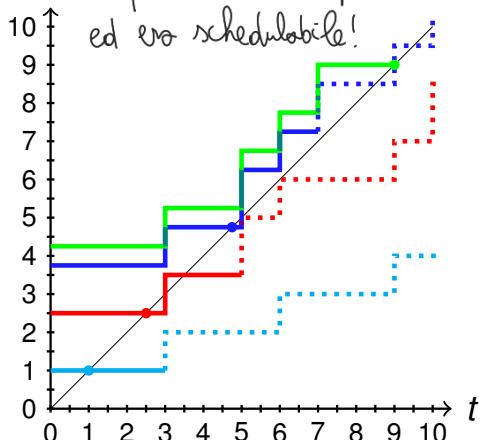
\Rightarrow forse non schedulabile

cond. sufficiente, infatti

w(t) questo è l'esempio di prima
ed era 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 |



Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità

SERT'20

R6.21

Test iperbolico per RM, meglio del test di primo!

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

$$\prod_{k=1}^n \left(1 + \frac{e_k}{p_k}\right)^{\text{fatt. utilizzazione}} \leq 2 \quad \begin{array}{l} \text{più oneroso} \\ \text{di Liu-Layland} \end{array}$$

mejor porque considera singule utilizzazioni \Rightarrow "piu' informazioni"

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

Test di schedulabilità generale

Condizioni di schedulabilità

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 $u_k = U_T/n$: (tutti stessa fattore di utilizzazione)

$$\prod_{k=1}^n \left(1 + \frac{U_T}{n}\right) \leq 2 \iff U_T \leq n(2^{1/n} - 1) \quad \begin{array}{l} \text{qui si equivalgono,} \\ \text{SOLO in questo caso} \end{array}$$

Esistono casi in cui il test iperbolico è soddisfatto ma la condizione di Liu-Layland non lo è (Buttazzo migliore/uguale a Liu-Layland)

SERT'20

R6.22

Test per sottoinsiemi di task armonici

- RM per task armonici = 1,
se quanti armonici mi ci avvicino?

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 $\mathcal{Z}_1, \dots, \mathcal{Z}_{n_h}$, ciascuno dei quali contiene task semplicemente periodici, allora il sistema è schedulabile con RM se

(partiziono in gruppi armonici)

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

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):
 $U_{\mathcal{Z}_1} + U_{\mathcal{Z}_2} \leq U_{RM}(2) = 0.828$

trovo 2
sottoinsiemi
armonici

Riesco a dimostrare
di più, non è che
faccio di più!

SERT'20

R6.23

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)

quanto sono vicini
all'armonico?

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

^{Storia di utilizzazioni}

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

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

Schedulabilità per RM e DM

Marco Cesati



Schema della lezione

Test di schedulabilità

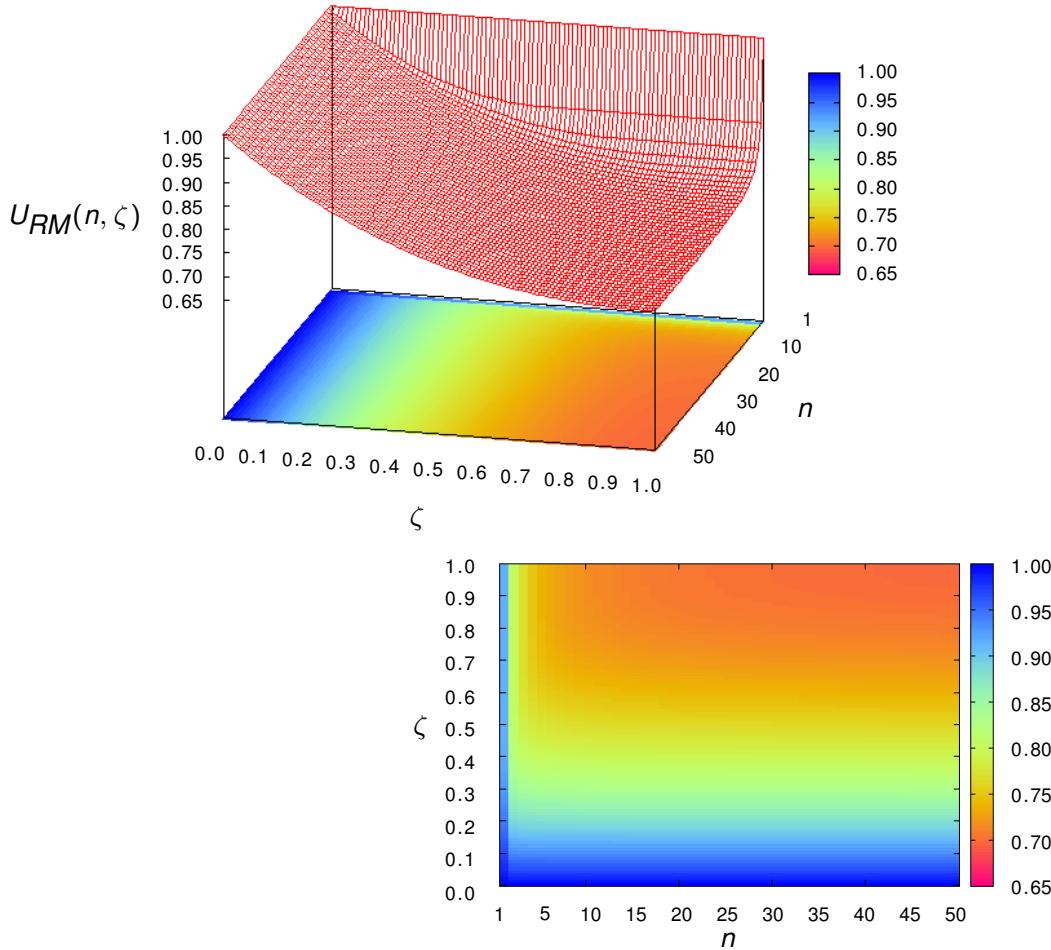
Test di schedulabilità generale

Condizioni di schedulabilità

SERT'20

R6.24

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



Schedulabilità per RM e DM
Marco Cesati

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

SERT'20 R6.25

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 $U_{RM}(n)$

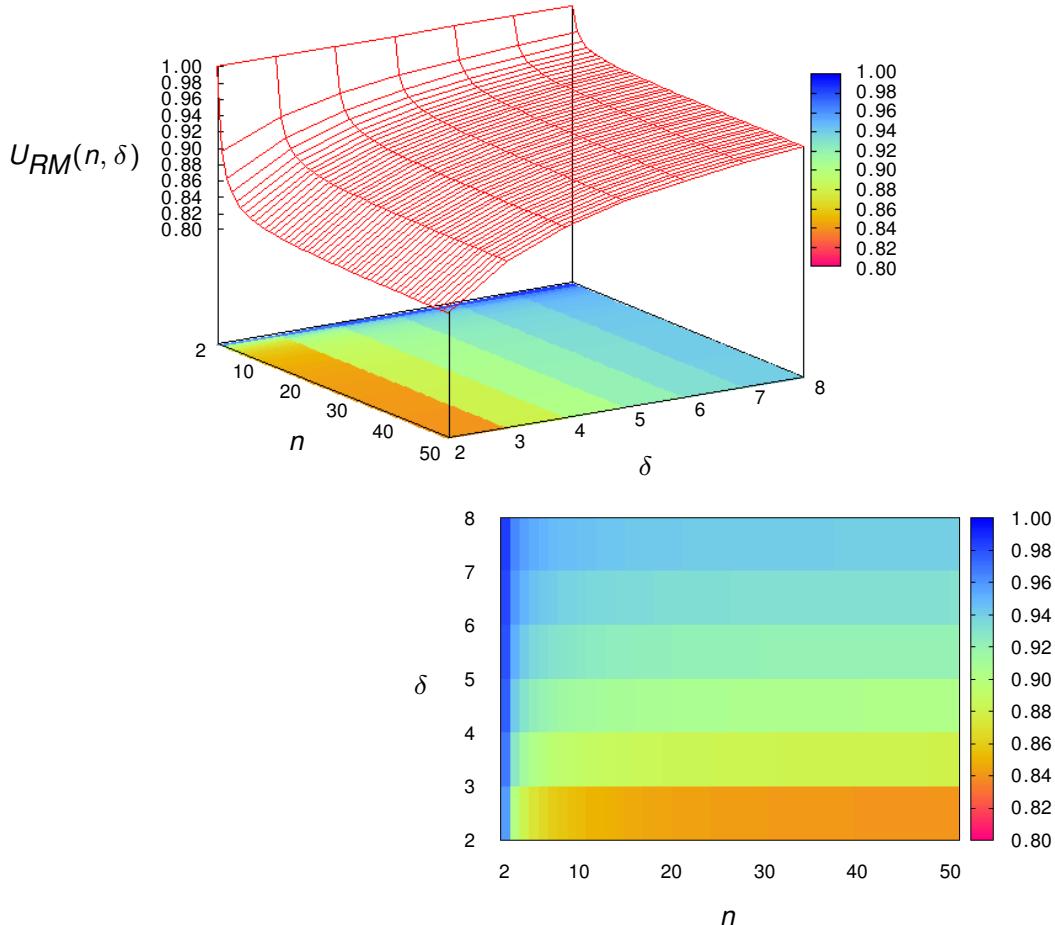
Teorema (Lehoczky, Sha, Strosnider, Tokuda 1986, 1990, 1991)

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

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

In queste condizioni $RM = DM$
difficili da implementare

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



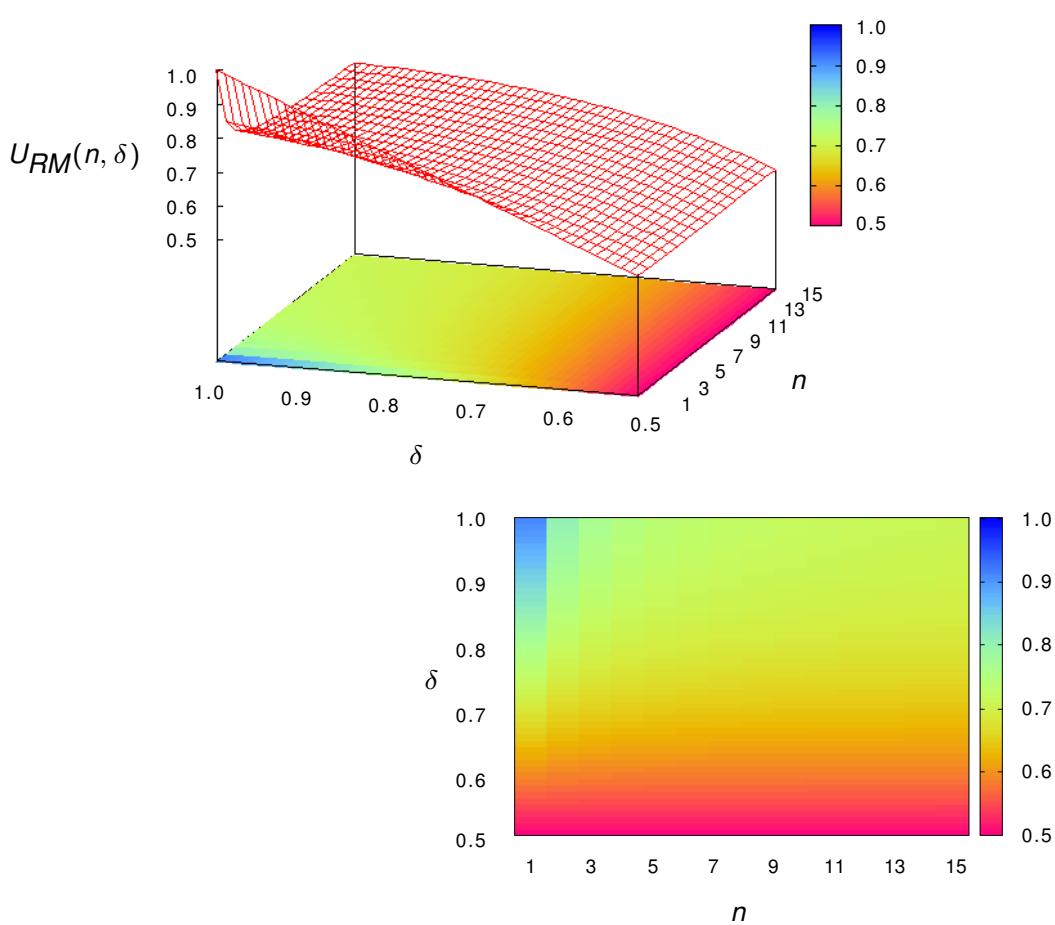
Schedulabilità per RM e DM
Marco Cesati



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

SERT'20 R6.27

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



Schedulabilità per RM e DM
Marco Cesati



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

SERT'20 R6.28