Lezione R12

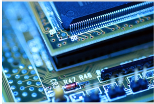
Real-time su multiprocessore II

Sistemi operativi open-source, embedded e real-time

6 dicembre 2017

Marco Cesati

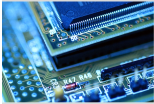
Dipartimento di Ingegneria Civile e Ingegneria Informatica
Università degli Studi di Roma Tor Vergata

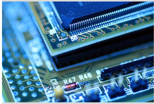


Di cosa parliamo in questa lezione?

In questa lezione continuiamo a parlare del problema della schedulazione real-time in sistemi multiprocessore, con particolare enfasi sugli scheduler globali

1. EDF-FFDD
2. Scheduler globali
3. EDF globale
4. EDF–US[ζ]
5. EDF(k)
6. RM globale
7. RM–US[ζ]





Algoritmo EDF-FF

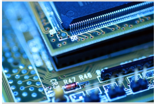
L'euristica "first fit" accoppiata all'algoritmo di schedulazione EDF dà luogo all'algoritmo di allocazione "on-line" EDF-FF:

1. ordina arbitrariamente i processori: $P_1, P_2, \ldots$
2. assegna ciascun task T_i al primo processore P_j tale che l'insieme dei task già assegnati a P_j insieme a T_i risulta ancora schedulabile tramite EDF

- $U_{\text{EDF-FF}} = \dfrac{\beta \cdot m + 1}{\beta + 1}, \quad \beta = \left\lfloor 1 / \max_k \dfrac{e_k}{p_k} \right\rfloor$ (Lopez & al., 2000)
- Fattore di approssimazione: 1.7 (Garey & Johnson, 1979)

EDF-FF è ottimale tra tutti gli algoritmi partizionati:

$$\beta = 1 \quad \Longrightarrow \quad U_{\text{EDF-FF}} = (m+1)/2$$



Algoritmo EDF-FFDD

È possibile estendere gli algoritmi partizionati basati su “first fit” anche a task sporadici con scadenze arbitrarie

Ad esempio, EDF-FFDD (Baruah & Fisher 2005):

1. ordina arbitrariamente i processori: $P_1, P_2, \ldots$
2. ordina i task sporadici per densità $\frac{e_i}{\min\{p_i, d_i\}}$ decrescenti
3. assegna ciascun task T_i al primo processore P_j tale che l'insieme dei task già assegnati a P_j insieme a T_i risulta ancora schedulabile tramite EDF

Condizione di schedulabilità:

$$\Delta_T \le \begin{cases} m - (m-1)\Delta_{\max} & \text{se } \Delta_{\max} \ge 1/2 \\ m/2 + \Delta_{\max} & \text{se } \Delta_{\max} \le 1/2 \end{cases}$$

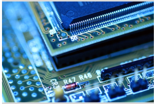
Scheduler multiprocessore globali

Ha senso considerare algoritmi di schedulazioni multiprocessore globali (non partizionati)?

Assumiamo in questa lezione che tutti i job siano indipendenti, interrompibili, migrabili e senza auto-sospensioni

Problematiche da affrontare:

- anomalie di schedulazione
- istanti critici
- utilità rispetto a EDF partizionato



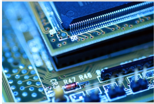
Anomalie di schedulazione

I sistemi multiprocessori presentano **anomalie di schedulazione** (cfr. R11.15)

Teorema (Ha & Liu, 1994)

Sistemi di task periodici interrompibili e migrabili su multiprocessore schedulati con algoritmi a priorità fissa (a livello di task o di job) sono predicibili e quindi non presentano anomalie di schedulazione dipendenti dal tempo di esecuzione dei job

Purtroppo esistono altre anomalie legate alla variazione di altri parametri temporali

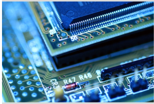


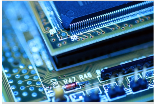
Utilità rispetto a EDF partizionato

Algoritmi a priorità fissa a livello di job come **EDF** sono ottimali nel caso uniprocessore e negli scheduler multiprocessore partizionati.

Quale vantaggio avrebbero gli scheduler globali?

Tra gli scheduler partizionati EDF è ottimale, ma non è ottimale in assoluto. Esistono algoritmi globali a priorità dinamica a livello di job che sono ottimali (cfr. algoritmo Pfair)





Scheduler globali a priorità dinamica a livello di job

Teorema (Baruah 2008)

Un sistema di task sporadici con scadenze arbitrarie è schedulabile con m processori con un algoritmo globale a priorità dinamica a livello di job se

$$\Delta_T = \sum_i \frac{e_i}{\min_i\{d_i, p_i\}} \leq m \quad \text{e} \quad \Delta_{\max} = \max_i \frac{e_i}{\min\{p_i, d_i\}} \leq 1$$

Corollario

Un sistema di task sporadici con scadenze implicite è schedulabile con m processori con un algoritmo globale a priorità dinamica a livello di job se e solo se

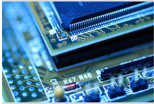
$$U_T = \sum_i \frac{e_i}{p_i} \leq m \quad \text{e} \quad U_{\max} = \max_i \frac{e_i}{p_i} \leq 1$$

Istanti critici

Il rilascio in fase di tutti i job non corrisponde necessariamente ad un istante critico (cfr. R11.18)

Non è possibile o pratico analizzare il sistema tramite la funzione di tempo necessario: come dimostrare la schedulabilità di un sistema?

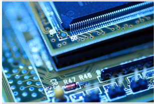
Possiamo utilizzare le condizioni di schedulabilità legate al fattore di utilizzazione o alla densità dei task!



Effetto Dhall

Vale comunque l'**effetto Dhall**: dato un qualunque numero $m > 1$ di processori, esistono sempre insiemi di task periodici schedulabili con utilizzazione totale costante (rispetto a m) che non possono essere schedulati con un algoritmo RM, DM o EDF (cfr. R11.12,13)

L'effetto Dhall è condizionato alla presenza di un task periodico con utilizzazione vicina all'unità. Possiamo quindi correlare l'utilizzazione massima dei vari task con la schedulabilità del sistema



Condizione di schedulabilità per EDF globale

Teorema **(Srinivasan & Baruah 2002; Goossens, Funk & Baruah 2003)**

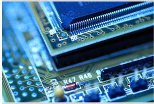
Un sistema T di task sporadici con scadenze implicite, utilizzazione totale U_T, e utilizzazione massima dei task $U_{\max}$, è schedulabile con EDF globale su un sistema con m processori se

$$U_T \leq m - (m-1) \cdot U_{\max}$$

Casi limite:

- $U_{\max} = 1 \quad \Rightarrow \quad U_T \leq 1 \quad \Rightarrow$ effetto Dhall
- $U_{\max} = 0 \quad \Rightarrow \quad U_T \leq m \quad \Rightarrow$ EDF globale è ottimale

Il teorema vale per scadenze non implicite?



Condizione di schedulabilità per EDF globale (2)

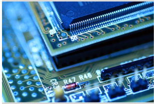
Teorema (Bertogna, Cirenei & Lipari 2005)

Un sistema T di task sporadici con scadenze arbitrarie, densità totale Δ_T, e densità massima dei task $\Delta_{\max}$, è schedulabile con EDF globale su un sistema con m processori se

$$\Delta_T \leq m - (m-1) \cdot \Delta_{\max}$$

Esistono diverse altre condizioni di schedulabilità per EDF globale basate sulla densità dei singoli task e/o sul calcolo del tempo di processore richiesto dai task

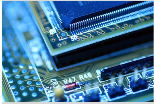
È possibile avere algoritmi di schedulazione a priorità fissa migliori di EDF globale per insiemi di task generici?



Algoritmo EDF–US[ζ]

EDF–US[ζ] è un algoritmo di schedulazione “ibrido”

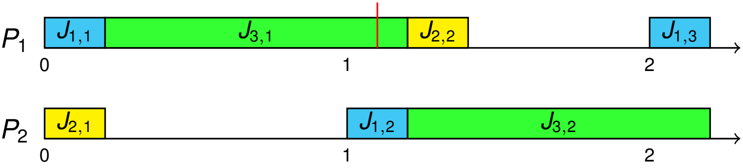
- Proposto da Srinivasan e Baruah nel 2002
- ζ è un parametro dell’algoritmo, $\zeta \leq 1$
- Una parte dei task ha priorità fissa
  - Sono i task T_i tali che $e_i/p_i > \zeta$
  - Tutti questi hanno identica priorità
- L’altra parte dei task ha priorità inferiore
  - Schedulati con EDF
- Idea fondamentale:
  - ai task “grandi” sono assegnati per primi i processori disponibili
  - i task “piccoli” vengono schedulati con il tempo di processore rimanente



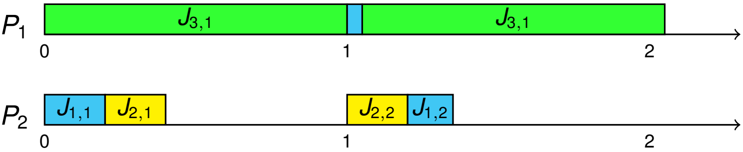
Esempio di schedulazione EDF–US[ζ]

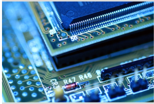
Consideriamo $T_1 = (1, 1/10)$, $T_2 = (1, 1/10)$, $T_3 = (21/20, 1)$ ($U_T = 121/105$)

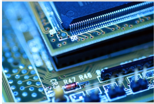
Schedulazione con EDF globale:



Schedulazione con EDF–US[1/2]:







Prestazioni di EDF–US[ζ]

Teorema (Srinivasan & Baruah, 2002)

Un sistema di task sporadici con scadenze implicite è schedulabile con EDF–US[$m/(2m-1)$] su m processori se

$$U_T \leq m^2/(2m-1)$$

Corollario (Baker, 2005)

Un sistema di task sporadici con scadenze implicite è schedulabile con EDF–US[$1/2$] su m processori se

$$U_T \leq (m+1)/2$$

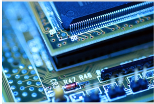
Nessun algoritmo di schedulazione globale con priorità fissa a livello di task o job può avere fattore di utilizzazione maggiore di $(m+1)/2$ (Andersson et. al. 2001, cfr. R11.19)

Possono esistere algoritmi di schedulazione globale a priorità fissa migliori di EDF–US[1/2]? ***Sì!***

Algoritmo EDF(k)

EDF(k) è un algoritmo di schedulazione “ibrido”

- Proposto da Goossens, Funk e Baruah nel 2003
- k è un parametro dell’algoritmo, $k < m$
- Una parte dei task ha priorità fissa
  - Sono i $k - 1$ task T_i che hanno fattore di utilizzazione e_i/p_i più alto
  - Tutti questi hanno identica priorità
- L’altra parte dei task ha priorità inferiore
  - Schedulati con EDF
- Medesima idea fondamentale:
  - ai task “grandi” sono assegnati per primi i processori disponibili
  - i task “piccoli” vengono schedulati con il tempo di processore rimanente



Prestazioni di EDF(k)

Teorema (Goossens, Funk & Baruah, 2003)

Un sistema di task sporadici con scadenze implicite è schedulabile con EDF(k) su m processori se

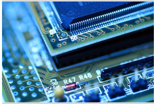
$$(k-1) + \left\lceil \frac{U_T - u_k}{1 - u_k} \right\rceil \le m$$

ove u_k è il fattore di utilizzazione del k-esimo task (ordinando i task per fattore di utilizzazione non decrescente)

Algoritmo EDF(k_{min}) (Baker 2005)

Sia fissato m (il numero di processori), e sia k_{min} il minimo valore di k che soddisfa la condizione del teorema precedente. Un sistema di task sporadici con scadenze implicite è schedulabile con EDF(k_{min}) se $U_T \le (m+1)/2$

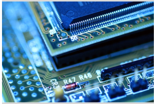
L'insieme di sistemi di task schedulabili da EDF–US[$1/2$] è un sottoinsieme proprio dell'insieme di EDF(k_{min})!



Scheduler globali basati su priorità fissa a livello di task

Scheduler globali o ibridi che utilizzano protocolli a priorità fissa a livello di task

- non possono avere prestazioni superiori agli scheduler globali o ibridi basati su priorità dinamica a livello di task
- in pratica sono molto considerati, perché
  - semplici da implementare
  - presenti in ogni RTOS
  - più robusti in caso di job che superano il WCET calcolato in fase di design



Scheduler RM globale

Teorema (Andersson, Baruah & Jonsson 2001)

Un sistema di task periodici con scadenze implicite tale che $U_{\max} \le m/(3m-2)$ è schedulabile globalmente con RM su m processori se

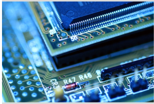
$$U_T \le m^2/(3m-1)$$

Teorema (Baker 2003; Bertogna, Cirenei & Lipari 2005)

Un sistema di task sporadici con scadenze implicite è schedulabile globalmente con RM su m processori se

$$U_T \le \frac{m}{2}(1 - U_{\max}) + U_{\max}$$

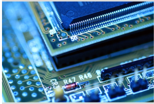
Esistono anche condizioni di schedulabilità per sistemi di task con scadenze arbitrarie ma hanno in genere formulazioni più complicate



Algoritmo RM–US[ζ]

RM–US[ζ] è un algoritmo di schedulazione “ibrido”, analogo a EDF–US[ζ]

- Proposto da Andersson, Baruah e Jonsson nel 2001
- ζ è un parametro dell’algoritmo, $\zeta \leq 1$
- Una parte dei task ha priorità fissa
  - Sono i task T_i tali che $e_i/p_i > \zeta$
  - Tutti questi hanno identica priorità
- L’altra parte dei task ha priorità inferiore
  - Schedulati con RM
- Medesima idea fondamentale di EDF–US[ζ]:
  - ai task “grandi” sono assegnati per primi i processori disponibili
  - i task “piccoli” vengono schedulati con il tempo di processore rimanente



Prestazioni di RM–US[ζ]

Teorema (Andersson, Baruah & Jonsson 2001)

Un sistema di task periodici con scadenze implicite è schedulabile con RM–US[$m/(3m-2)$] su m processori se

$$U_T \leq m^2/(3m-1)$$

Teorema (Baker 2003; Bertogna, Cirenei & Lipari 2005)

Un sistema di task sporadici con scadenze implicite è schedulabile con RM–US[$1/3$] su m processori se

$$U_T \leq (m+1)/3$$

Il miglior valore possibile per ζ è $\zeta = 0{,}37482$, con utilizzazione massima pari a 0,37482 (Lundberg 2002)

Sono stati studiati anche gli algoritmi RM(k) e RM($k_{\min}$), analoghi a quelli basati su EDF