

# Lezione R2

## Modello di riferimento per i sistemi real-time

Modello di  
riferimento per i  
sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task  
periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

Sistemi embedded e real-time

1 ottobre 2020

Marco Cesati

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

SERT'20

R2.1

### Di cosa parliamo in questa lezione?

Non voglio riscrivere  
→ sempre da zero!

In questa lezione definiamo il **modello di riferimento** che ci permetterà di descrivere con esattezza i sistemi real-time

Modello di  
riferimento per i  
sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task  
periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

- 1 Il modello di riferimento dei sistemi real-time
- 2 I vincoli temporali dei job
- 3 Il modello a task periodici
- 4 I vincoli di precedenza dei job
- 5 Il grafo dei task
- 6 I parametri funzionali dei job
- 7 La schedulazione dei job

SERT'20

R2.2

## Vincoli temporali hard real-time

- Un principio fondamentale è che non è importante minimizzare il tempo di risposta di un job hard real-time: è solo necessario assicurarsi che il job non violi il vincolo temporale imposto (ne scaduta è 5s, non sono interessato a mettercene 2!)
- Talvolta è controproducente cercare di minimizzare il tempo di risposta dei job hard real-time, in quanto questo può far crescere la loro varianza; in genere è meglio avere tempi di risposta il più possibile costanti (sono "INVERSI")
- La "definizione operativa" di hard real-time (necessità di validazione) consente di avere vincoli temporali hard formulati: ho 3 "variazioni"
  - in modo deterministico ("questo vincolo deve essere sempre soddisfatto")
  - in modo probabilistico ("la probabilità che il tempo di risposta ecceda 50 msec deve essere minore di  $10^{-5}$ )
  - in termini di alcune funzioni d'utilità ("l'utilità del calcolo di ogni legge di controllo deve essere pari a 0.8 o maggiore")

In pratica è raro trovare vincoli hard real-time formulati in modo non deterministico

- VOGLIO che siano PREDICIBILI e COSTANTI (varianze →)

## Vincoli temporali soft real-time

- Nei sistemi soft real-time, oltre al rispetto dei vincoli temporali dei job, è spesso importante cercare di ottenere anche altri obiettivi, ad esempio quello di minimizzare il tempo di risposta dei job o di massimizzare il throughput del sistema (QUESTO E' IMPORTANTE!)
- Spesso i vincoli temporali soft sono indicati in termini probabilistici; ad esempio, una centrale telefonica realizza una chiamata dell'utente come un task che deve essere completato in non più di 10 secondi nel 95% dei casi, ed in non più di 20 secondi nel 99.5% dei casi (es: centrolimo)
- L'utilità del job tardivo dipende dalla specifica applicazione:

- Sono ritardo, consegno lo stesso?*
- In una centrale telefonica, il risultato del job che attiva una chiamata ha una utilità che decresce in modo graduale in funzione del ritardo
  - In un sistema di controllo delle quotazioni del mercato azionario, il risultato di un job che aggiorna una quotazione in ritardo rispetto alla deadline ha una utilità che decresce rapidamente in funzione del ritardo



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.3

Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.4

PROVO A DEFINIRE un MODELLO! Perché Vorrei teoria generale!

## Modello di riferimento per sistemi real-time

Ogni sistema real-time può essere caratterizzato da un modello costituito da tre elementi:

- un *modello di carico*, che descrive le applicazioni supportate dal sistema (*da eseguire*)
- un *modello delle risorse*, che descrive le risorse di sistema a disposizione delle applicazioni (*risorse attive e passive*)
- *algoritmi* che definiscono come il sistema distribuisce le risorse alle applicazioni nel tempo ( $\approx$  *scheduling*)

Un buon modello dovrebbe:

- **includere tutti i dettagli essenziali** per la comprensione del sistema e per la descrizione del suo comportamento
- **omettere tutti i dettagli di basso livello o irrilevanti** che rendono la comprensione del livello inutilmente difficile

Modello di riferimento per i sistemi real-time  
Marco Cesati



Schema della lezione  
Hard e soft real-time  
**Modello di riferimento**  
Vincoli temporali  
Modello a task periodici  
Vincoli di precedenza  
Parametri funzionali  
Schedulazione

SERT'20 R2.5

## Le risorse di sistema

(rappresenta l'ambiente di esecuzione, per noi CPU, Computer...)

Le **risorse di sistema** rappresentano sostanzialmente l'ambiente d'esecuzione delle applicazioni

Esistono due tipi di risorse di sistema:

- Le **risorse attive**, spesso chiamate anche *server* o *processori*: computer, canali di comunicazione, dischi rigidi, database server, ...
- Le **risorse passive**, indicate semplicemente come *risorse*: memoria, ogni genere di primitiva di sincronizzazione, numeri di sequenza, ... (Semaforo è verde/rosso, non veloce/lento)

Modello di riferimento per i sistemi real-time  
Marco Cesati



Schema della lezione  
Hard e soft real-time  
**Modello di riferimento**  
Vincoli temporali  
Modello a task periodici  
Vincoli di precedenza  
Parametri funzionali  
Schedulazione

A differenza delle risorse (passive), la velocità o capacità intrinseca dei **processori** influenza il tempo d'esecuzione dei job (e quindi delle applicazioni)

*Ma la disponibilità di memoria o di un semaforo non influenza il tempo di esecuzione di un job?*

Certamente, ma una volta che la risorsa è assegnata al job le sue caratteristiche non influenzano il tempo d'esecuzione!

(Job vuole Proc. e Ram, quando li ha solo il primo influenza sui tempi)

## Le risorse di sistema (2)

- Nel modellare un sistema, spesso si omette l'indicazione del tipo specifico di **processori**, e si elencano semplicemente i processori come  $P_1, P_2, \dots, P_m$
- Analogamente, spesso si omette di specificare il tipo di **risorse**, e si elencano le risorse come  $R_1, R_2, \dots, R_p$
- Se una **risorsa** è così **abbondante** da non costituire mai un vincolo od uno ostacolo per l'esecuzione dei job, essa viene **omessa** e non citata nel modello

Ad esempio, una risorsa che viene spesso omessa è la **memoria**: si assume che ve ne sia sempre a sufficienza per l'esecuzione di tutti i job

Nei casi in cui questo non sia giustificato è necessario complicare il modello per descrivere la quantità di memoria totale del sistema e l'occupazione di memoria dei vari job



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.7

## Il modello del carico

↪ devo "caratterizzarlo"



Le applicazioni real-time sono descritte in termini di **task**, ciascuno dei quali è costituito da un insieme di entità schedulabili chiamate **job**

Caratteristiche di un **job**:

- vincoli temporali, scadenze
- parametri funzionali (proprietà intrinseche del job)
- uso delle risorse ( se lavora con  $R_i$ , deve usorlo )
- parametri d'interconnessione (dipendenza da altri job e come altri job dipendono da questo)



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.8

## Vincoli temporali di un job

**IMPORTANTE**

→ da qui "inizio" a contare

- **Istante di rilascio** (od anche *release time*): istante di tempo  $r_i$  in cui il job  $J_i$  diventa disponibile per l'esecuzione
- **Scadenza** (od anche *deadline*): istante di tempo  $d_i$  in cui il job  $J_i$  deve aver completato l'esecuzione <sup>ASSOLUTA</sup>
- **Scadenza relativa**: il massimo intervallo di tempo  $D_i$  che può intercorrere tra  $r_i$  e  $d_i$  per il job  $J_i$  ( $D_i = d_i - r_i$ ), e' intervallo
- **Intervallo di fattibilità**: l'intervallo di tempo  $(r_i, d_i]$

I vincoli temporali di tutti i job sono in genere inclusi nelle specifiche del sistema real-time

Spesso però le specifiche non indicano con esattezza gli **istanti di rilascio**, ma solo un intervallo in cui si verificherà il rilascio (*jitter*):

$$r_i \in [r_i^-, r_i^+]$$

di solito sono "forniti", poiché allo base  
devo gestire anche eventi esterni non predefinibili!

## Task aperiodici e task sporadici

Molti sistemi real-time devono essere in grado di reagire ad eventi esterni che occorrono ad istanti di tempo casuali

In risposta a tali eventi il sistema esegue un insieme di job particolari, i cui istanti di rilascio non sono conosciuti se non nel momento in cui l'evento esterno si verifica

Si definisce **task aperiodico** un task in cui gli istanti di rilascio dei job sono casuali e non predefiniti, <sup>NON POSSO</sup> <sub>DIRE NULLA!</sub>

Si definisce **task sporadico** un task in cui gli istanti di rilascio dei job sono casuali e non predefiniti, ma esiste un intervallo di tempo minimo tra il rilascio di due job (es: tra  $J_1$  e  $J_2$  passano ALMENO x secondi)

**Attenzione:** il testo di Liu utilizza definizioni differenti: un job con istanti di rilascio casuali è *aperiodico* se è soft real-time, mentre è *sporadico* se è hard real-time

Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.9

Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.10

## Tempo d'esecuzione

Il **tempo d'esecuzione**  $e_i$  di un job  $J_i$  è l'ammontare di tempo richiesto per completare l'esecuzione di  $J_i$  eseguendolo da solo ed avendo a disposizione tutte le risorse necessarie

Il **tempo d'esecuzione** non dipende dalla schedulazione, ma esclusivamente dalla funzione realizzata dal job e dalla velocità del processore utilizzato per eseguirlo, scheduling NON interviene

*Qual è la principale difficoltà nel definire il tempo d'esecuzione di un job?*

Tutti i processori moderni (CPU, reti, ecc.) hanno un comportamento non deterministico: • HW molto complesso!

- la loro complessità è così elevata che è impossibile prevedere a priori il tempo d'esecuzione di un job
- il tempo effettivo varia da esecuzione a esecuzione

Si può determinare a priori solo l'**intervallo** in cui cade  $e_i$ :

$$e_i \in [e_i^-, e_i^+]$$

NON deterministico, messo su bene come funziona all'interno!  
non esiste più info su temporizzazione

SERT'20

R2.11

Altrimenti, come siamo a lavorare con gli intervalli?  
**WCET** (Worst case!)

Se il nostro scopo è quello di determinare se un certo job all'interno di un sistema potrà essere completato entro la sua scadenza, possiamo sperare che:

- non si abbia realmente necessità di conoscere il suo tempo d'esecuzione  $e_i$
- sia necessario unicamente conoscere il suo **massimo tempo d'esecuzione**  $e_i^+$

In letteratura il **massimo tempo d'esecuzione** è spesso indicato con l'acronimo **WCET** (Worst Case Execution Time)

In molti casi con il termine **tempo d'esecuzione**  $e_i$  si indica il **WCET**  $e_i^+$ , e si tralascia di indicare che in realtà il tempo d'esecuzione del job potrebbe essere inferiore

Worst Case

Anche stimare con ragionevole approssimazione il **WCET** di un job potrebbe essere difficile od impossibile!

Si consideri ad esempio il tempo d'esecuzione di un processo su una CPU con diversi livelli di memoria cache

Senza cache, prima da 2s a 2min (100 volte più lento), testata dal produttore!

Modello di riferimento per i sistemi real-time  
Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

Modello di riferimento per i sistemi real-time  
Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

deve essere ben stimato!

SERT'20

R2.12

# Nell' Hard Real Time voglio: Sistemi predicibili e deterministici

In un sistema hard real-time è necessario validare che tutti i job "hard" rispettano i loro vincoli temporali

spesso troppo variabile!

Per ottenere ciò è necessario conoscere i WCET dei job "hard".  
come so che quel tempo è il peggiore? ↗ Però conta solo lui!

Di conseguenza, la proprietà più importante di un sistema hard real-time è la **predicibilità** del suo comportamento

Spesso tale **predicibilità** è ottenuta a scapito delle prestazioni:

- Si utilizzano processori meno sofisticati ↗ PORTA VARIANZA
- Si scrivono programmi senza strutture di dati dinamiche
- Si evitano interazioni troppo complesse tra i job (osservare 2 interazioni vs osservare 100 interazioni)
- Si minimizza il numero di primitive di sincronizzazione delle risorse condivise

Un sistema hard real-time così concepito è, auspicabilmente, quasi **deterministico**: le esecuzioni di ogni job hanno durata pressoché costante

Come si caratterizza il modello del carico / applicazione?

## Modello a task periodici

Il **modello a task periodici** è un modello di carico deterministico molto conosciuto introdotto da Liu e Layland nel 1973

- È in grado di rappresentare adeguatamente la maggior parte delle applicazioni real-time
- Molti algoritmi di schedulazione basati su questo modello sono facili da implementare, efficienti, ed hanno un comportamento facile da prevedere
- Esistono metodi e strumenti per progettare, analizzare e validare sistemi real-time basati su questo modello.
- Non rappresenta bene sistemi i cui job sono caratterizzati da elevati valori di jitter per gli istanti di rilascio oppure alta varianza nei tempi d'esecuzione



## Modello a task periodici (2)

Principio di base: ogni **computazione**, **trasmissione di dati** od altra attività che è eseguita ripetutamente ad intervalli di tempo regolari è modellata come un **task periodico**:

- Un **task periodico**  $T_i$  è una sequenza di **job**
- Il **periodo**  $p_i$  di  $T_i$  è l'esatto intervallo temporale tra gli istanti di rilascio dei job di  $T_i$  (task con  $p_i=1s$ , ogni 1 sec. rilascio job)
- Il **tempo d'esecuzione**  $e_i$  di  $T_i$  è il massimo tra tutti i **tempi d'esecuzione** dei job di  $T_i$ , e' del task, cioè di tutti i job  
*Premendo sempre tempo maggiore*

$T_{22}$  ha  
 $J_1 = 2s$ ,  
 $J_2 = 4s$   
↳  $e_{12} = 4s$   
(come se assunnessi  $J_1 = J_2 = 4s$ )

**Attenzione:** Nel testo di Liu il termine **task periodico** indica in genere un **task sporadico**, ossia il periodo del task corrisponde all'intervalllo minimo tra gli istanti di rilascio dei job

•  $J_{i+1}$  parte solo dopo  $J_i$ , entrambi nel task. Nello stesso task i job DEVONO AVERE stesso tempo di rilascio!

Il testo di Liu tende a confondere i task sporadici con i task periodici in quanto la maggior parte dei risultati teorici relativi ai task periodici continuano ad essere validi per i task sporadici *normalmente*  $p_i \geq e_i$ , altrimenti ho problemi!

Modello di riferimento per i sistemi real-time  
Marco Cesati



Schema della lezione  
Hard e soft real-time  
Modello di riferimento  
Vincoli temporali  
Modello a task periodici  
Vincoli di precedenza  
Parametri funzionali  
Schedulazione

SERT'20 R2.15

## Modello a task periodici (3)

Alcune notazioni e definizioni:

- I task sono  $T_1, T_2, \dots, T_n$
- Gli job del task  $T_i$  sono  $J_{i,1}, J_{i,2}, \dots$
- Cumulativamente, tutti i job del sistema sono  $J_1, J_2, \dots$
- L'istante di rilascio  $r_{i,1}$  del primo job  $J_{i,1}$  di  $T_i$  è chiamato **fase** di  $T_i$  ( $\Phi_i = r_{i,1}$ )
- Task con la stessa fase sono detti **in fase** (periodo può essere diverso)
- $H$  indica il minimo comune multiplo di tutti i periodi  $p_i$  dei task del sistema
- Un intervallo temporale di lunghezza  $H$  è chiamato **iperperiodo** dei task periodici
- Il (massimo) numero  $N$  di job in ogni iperperiodo è

*n° job rilasciati  
in ogni iperperiodo*  $N = \sum_{i=1}^n H/p_i$  sempre intero!

Modello di riferimento per i sistemi real-time  
Marco Cesati



Schema della lezione  
Hard e soft real-time  
Modello di riferimento  
Vincoli temporali  
Modello a task periodici  
Vincoli di precedenza  
Parametri funzionali  
Schedulazione

SERT'20 R2.16

## Modello a task periodici (4)

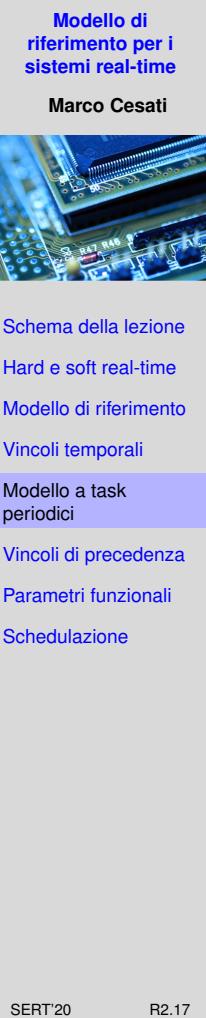
TASK produce lavoro del vi'.  
per il processore!

- Il rapporto  $U_i = e_i/p_i$  è l'*utilizzazione* del task  $T_i$ , se  $> 1$  ha problemi
- L'*utilizzazione totale*  $U$  del sistema è la somma delle utilizzazioni di tutti i task:

$$U = \sum_{i=1}^n (U_i \text{ di } T_i)$$

- Tutti i job di uno stesso task  $T_i$  hanno la stessa *scadenza relativa*  $D_i$ : se un job è rilasciato all'istante  $t$ , deve essere completato entro  $t + D_i$   $\doteq$  scadenza assoluta

In molti casi (ma non tutti!) un sistema real-time può essere modellato assumendo che tutti i job hanno istante di rilascio all'inizio di ciascun periodo e *scadenze implicite*, ossia scadenze relative pari alla lunghezza del periodo ( $D_i = p_i$ )



## Esempio di modello a task periodici

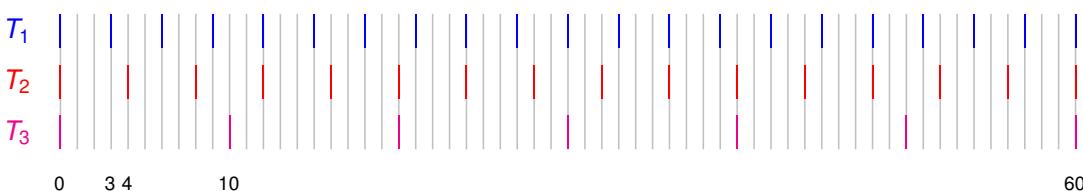
Consideriamo tre task periodici in fase con periodi  $p_1 = 3$ ,  $p_2 = 4$  e  $p_3 = 10$ , e tempi d'esecuzione  $e_1 = 1$ ,  $e_2 = 1$  e  $e_3 = 3$

Quanti sono i differenti job nel sistema reale modellato dai task?

Non possiamo saperlo, ma non è importante!

Quanti sono i job eseguiti in un iperperiodo?

- $H = \text{lcm}\{3, 4, 10\} = 60$
- $N = \sum_{i=1}^3 H/p_i = 60/3 + 60/4 + 60/10 = 41 \text{ Job}$



## Esempio di modello a task periodici (2)

Qual è l'utilizzazione totale del sistema?

- Le utilizzazioni dei tre task periodici sono  $e_1/p_1 = 1/3$ ,  $e_2/p_2 = 1/4$  e  $e_3/p_3 = 3/10$

- L'utilizzazione totale è

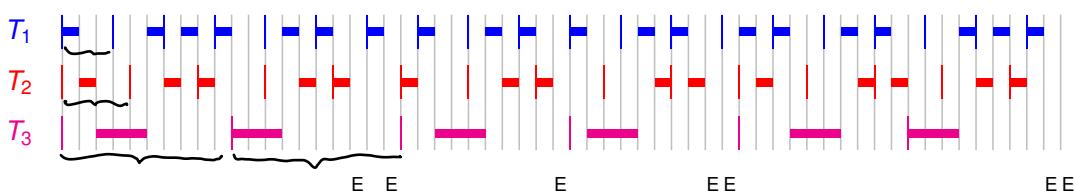
$$U = \frac{1}{3} + \frac{1}{4} + \frac{3}{10} = \frac{53}{60} \approx 88.3\%$$

*processore  
occupato  
iperperiodo*

*60 - 53 = 7  
3 7 intervalli  
con processore  
Libero*

È possibile schedulare questi task con un solo processore se hanno scadenze implicite, ossia  $D_j = p_j$  con  $j \in \{1, 2, 3\}$ ? **Sì!**

Come? "PROVO!" (a mano)



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.19

## Vincoli di precedenza tra job

Un **vincolo di precedenza** tra due job  $J$  e  $J'$  è una condizione che impone al sistema di schedulare l'esecuzione dei due job in un preciso ordine



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

Ad esempio, in un database un job autentica l'utente, mentre un altro job esegue l'interrogazione della base di dati; un vincolo di precedenza impedisce al secondo job di iniziare l'esecuzione finché il primo non ha terminato l'autenticazione

I **vincoli di precedenza** possono essere modellati tramite una relazione d'ordine parziale tra i job del sistema

$J \prec J'$  significa che  $J'$  non può iniziare l'esecuzione prima che  $J$  sia stato completato

I **vincoli di precedenza** semplificano in generale il compito di trovare una schedulazione per i job? **No!** Sono considerazioni in più da fare

La semplificazione si avrebbe solo con un ordinamento totale!

SERT'20

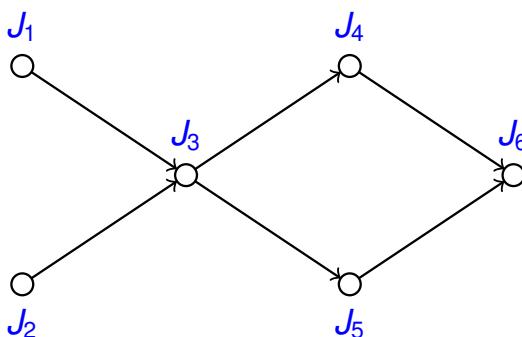
R2.20

## Grafo di precedenza

In generale i vincoli di precedenza tra i job possono essere espressi in forma compatta tramite un grafo diretto aciclico (DAG), in cui:

- I nodi rappresentano job
- Un arco diretto dal job  $J$  al job  $J'$  indica che  $J \prec J'$  e che non esiste alcun job  $J''$  tale che  $J \prec J'' \prec J'$
- Indicheremo la precedenza immediata con  $J \rightarrow J'$

$$\begin{aligned} J_1 &\rightarrow J_3, & J_2 &\rightarrow J_3, \\ J_3 &\rightarrow J_4, & J_3 &\rightarrow J_5, \\ J_4 &\rightarrow J_6, & J_5 &\rightarrow J_6 \end{aligned}$$



$J_3 \prec J_6$ ? **Sì, per la proprietà transitiva dell'ordinamento**

Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.21

## Parametri funzionali dei job

I **parametri funzionali** dei job contribuiscono a caratterizzare il modello di carico del sistema real-time

In generale ogni job può essere caratterizzato da un grande numero di attributi, ma come **parametri funzionali** consideriamo solo quelli che incidono sulla schedulazione dei job

I **parametri funzionali** più importanti sono:

- Interrompibilità (o “preemptivity”)
- Criticalità (o importanza)
- Funzione d'utilità (o “laxity function”)

Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.22

## Interrompibilità dei job

Un job si definisce **interrompibile** (o **preemptable**) se la sua esecuzione può essere sospesa in qualunque istante per permettere l'esecuzione di altri job e, più tardi, ripresa dal punto di sospensione

Un job **non interrompibile**, viceversa, deve essere eseguito dall'inizio alla fine senza interruzioni

Esempi:

- I processi in Linux sono job **interrompibili**: lo scheduler del kernel può interrompere la loro esecuzione nel caso in cui un processo a priorità maggiore diventi eseguibile
- La spedizione di un frame Ethernet è un job **non interrompibile**: non è possibile interrompere la trasmissione per spedire un altro frame e poi riprendere la trasmissione dei dati non ancora inviati

Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.23

## Interrompibilità dei job (2)

Anche per i job in generale **interrompibili** possono esistere condizioni che impediscono di fatto la loro sospensione

*Quali sono tipici casi in cui non è possibile interrompere un job?*

- Il job sta eseguendo una operazione che deve essere conclusa in tempi rapidi (ad esempio, la programmazione di un dispositivo hardware)
- Il job sta modificando una struttura di dati condivisa con altri job: se esso fosse interrotto in mezzo alla procedura di aggiornamento, ed un altro job cominciasse a propria volta a modificare la struttura di dati, alla fine questa avrebbe uno stato non consistente
- Il job sta effettuando il salvataggio delle informazioni che consentiranno il recupero dell'esecuzione

Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.24

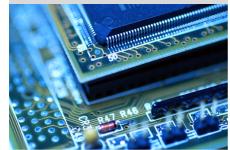
## Contesto di un job interrompibile

Si definisce **contesto** l'insieme delle informazioni necessarie per riprendere l'esecuzione di un job interrotto



Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

Ad esempio, nel caso di processi eseguiti da una CPU il **contesto** include il valore dei registri della CPU, lo stack del processo, le tabelle di paginazione, ...

Si definisce **cambio di contesto** (o **context switch**) la procedura di salvataggio del contesto del job interrotto e di recupero del contesto del job che andrà in esecuzione

Si definisce **context-switch time** il tempo necessario per operare un cambio di contesto

SERT'20

R2.25

## Criticalità dei job

In qualunque sistema esistono job di diversa importanza

→ Soggettivo, allora si può sbagliare,  
allora può fallire. Se oggettivo, pone  
dimostrare che non fallisce!

La **criticalità** di un job è un parametro numerico che indica l'importanza relativa del job rispetto agli altri job nel sistema

Consideriamo ad esempio un sistema avionico; in ordine decrescente di criticalità potremmo avere i job che controllano:

- l'assetto di volo
- la posizione attuale del velivolo
- la velocità e direzione per ottimizzare i consumi di carburante
- la temperatura, umidità e pressione nelle cabine
- il video proiettato sui televisori dei passeggeri

La **criticalità** di un job non coincide con la sua **priorità** od il suo **peso**: questi ultimi sono attributi che, pur incidendo sulla modalità di schedulazione del job, non sono necessariamente correlati alla **criticalità**

Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

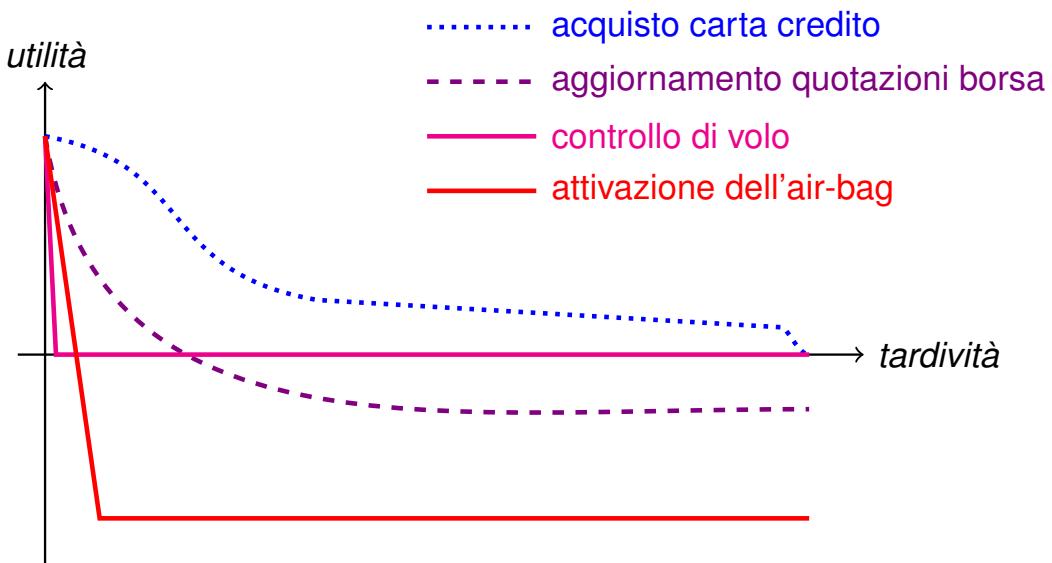
SERT'20

R2.26

## Funzione d'utilità dei job

Un parametro funzionale essenziale per i job di un sistema real-time è quello che indica se i suoi vincoli temporali sono **hard** oppure **soft**

A volte, a tale indicazione viene aggiunto un altro parametro funzionale: la **funzione d'utilità**, che lega la **tardività** del job con l'**utilità** del risultato prodotto



Modello di riferimento per i sistemi real-time
Marco Cesati
Schema della lezione
Hard e soft real-time
Modello di riferimento
Vincoli temporali
Modello a task periodici
Vincoli di precedenza
Parametri funzionali
Schedulazione

SERT'20 R2.27

## Schedulazione dei job

Lo **scheduler** (o **schedulatore**) è il **modulo del sistema real-time** che implementa gli algoritmi utilizzati per ordinare l'esecuzione dei job e controllare l'accesso alle risorse

Modello di riferimento per i sistemi real-time
Marco Cesati
Schema della lezione
Hard e soft real-time
Modello di riferimento
Vincoli temporali
Modello a task periodici
Vincoli di precedenza
Parametri funzionali
Schedulazione

SERT'20 R2.28

Si definisce **schedule** (o **schedulazione**) una assegnazione dei job del sistema ai processori disponibili

Una schedulazione è **valida** se:



- 1 In ogni istante un processore è assegnato ad al più un job
- 2 In ogni istante un job è assegnato ad al più un processore
- 3 Nessun job è schedulato prima del suo istante di rilascio
- 4 L'ammontare totale di tempo del processore assegnato a ciascun job è pari al suo tempo d'esecuzione (massimo o attuale a seconda dell'algoritmo di schedulazione)
- 5 Tutti i vincoli di precedenza e uso delle risorse tra i job sono soddisfatti

Schema della lezione
Hard e soft real-time
Modello di riferimento
Vincoli temporali
Modello a task periodici
Vincoli di precedenza
Parametri funzionali
Schedulazione

SERT'20 R2.28

## Schedulazione dei job (2)

Una schedulazione valida non garantisce di per se che le scadenze dei job siano rispettate!



Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

A modo di  
costruire  
un  
insieme  
di job

Una schedulazione valida è *fattibile (feasible)* se ogni job è completato entro la sua scadenza

Un insieme di job è *schedulabile* con un certo algoritmo se tale algoritmo è in grado di produrre sempre una *schedulazione fattibile*

Un algoritmo di schedulazione per applicazioni hard real-time è detto *ottimale* se l'algoritmo produce sempre una *schedulazione fattibile* quando l'insieme di job è di per sé *schedulabile*

SERT'20

R2.29

## Misure di prestazioni

I principali parametri per misurare le prestazioni dei sistemi real-time sono i valori massimi e/o medi dei seguenti attributi:

- **Tardività:** zero se la scadenza è rispettata, oppure la differenza tra l'istante di completamento e la scadenza (la venne nulla)
- **Lateness:** la differenza tra l'istante di completamento e la scadenza (può essere negativa se la scadenza è rispettata) No BUONO, Lo voglio PREDICIBILE
- **Tempo di risposta:** differenza tra l'istante di completamento e l'istante di rilascio
- **Miss rate:** percentuale di job (soft) che terminano oltre la loro scadenza
- **Loss rate:** percentuale di job (soft) non eseguiti o comunque non terminati
- **Invalid rate:** è la somma del **miss rate** e del **loss rate**, ossia la percentuale di job che non producono un risultato utile

Modello di riferimento per i sistemi real-time

Marco Cesati



Schema della lezione

Hard e soft real-time

Modello di riferimento

Vincoli temporali

Modello a task periodici

Vincoli di precedenza

Parametri funzionali

Schedulazione

SERT'20

R2.30