Scheduling di processi “hard real-time”

Eugenio Faldella
Dipartimento di Informatica – Scienza e Ingegneria
Scuola di Ingegneria e Architettura, Università di Bologna

eugenio.faldella@unibo.it
http://www.ing.unibo.it

UN PRIMO MODELLO DI RIFERIMENTO PER LA SCHEDULAZIONE DI PROCESSI PERIODICI

- N processi P_1, P_2, …, P_N indipendenti
  - senza vincoli di precedenza
  - senza risorse condivise
- ogni processo P_j (j = 1, 2, …,N)
  - è periodico, con periodo T_j prefissato
  - è caratterizzato da un tempo massimo di esecuzione C_j prefissato, con $C_j < T_j$
  - è caratterizzato da una deadline $D_j = T_j$
- l'esecuzione dei processi è affidata ad un sistema di elaborazione monoprocessore
- il tempo impiegato dal processore per operare una commutazione di contesto tra processi è trascurabile

UN TEOREMA SULLA SCHEDULABILITÀ

I requisiti temporali sono coerenti e consistenti ?

Condizione necessaria (ma in generale non sufficiente) affinché un insieme di N processi periodici sia schedulabile è che il risultante fattore di utilizzazione del processore sia non superiore a 1:

$$U = \sum_{j=1}^{N} U_j = \sum_{j=1}^{N} \frac{C_j}{T_j} \leq 1$$

Il j^{mo} termine della sommatoria $C_j / T_j = (C_j (H / T_j)) / H$ rappresenta la frazione dell'iperperiodo $H = mcm(T_1, T_2, \ldots, T_N)$ richiesta per l'esecuzione di P_j.

SCHEDULAZIONE "CLOCK-DRIVEN" …

- Schedulazione di tipo
  - off-line,
  - guaranteed,
  - non-preemptive,

  semplice da realizzare ed efficiente (l'overhead associato al cambio di contesto e alla comunicazione tra processi è trascurabile), ma non idonea in contesti che implicano dinamicità e flessibilità.
- I parametri temporali dei processi si intendono noti a priori e non soggetti a variazioni significative a run-time.
- Tutti i vincoli temporali vengono soddisfatti a priori in sede di costruzione di un "feasible schedule".

  Problema NP-hard
- Allo scopo possono essere usati algoritmi anche molto complessi, senza incorrere in una penalizzazione delle prestazioni conseguibili a run-time.

… SCHEDULAZIONE "CLOCK-DRIVEN"

- Lo schedule viene costruito con riferimento al generico iperperiodo, assumendo che qualunque decisione riguardante la schedulazione dei processi venga presa in corrispondenza di predefiniti "istanti decisionali". Tali istanti possono essere o meno equidistanziati.
- Lo schedule risultante è di norma esplicitato in termini di una tabella, la cui generica k-esima entry è del tipo:

  $(\Delta t_k, P(t_k))$ nel caso di istanti decisionali non equidistanziati, con $\Delta t_k = t_{k+1} - t_k$,

  $(\underline{P}(t_k))$ nel caso di istanti decisionali equidistanziati,

  dove $P(t_k)$ e $\underline{P}(t_k)$ identificano rispettivamente il processo o l'insieme di processi da schedulare all'istante t_k.
- Lo scheduler a run-time, avvalendosi di un timer hardware che, opportunamente programmato, genera un interrupt in corrispondenza di ogni istante decisionale, ciclicamente si limita a interpretare il contenuto di tale tabella, procedendo a riprogrammare il timer se necessario (ovvero nel caso di istanti decisionali non equidistanziati), a selezionare il processo o i processi da schedulare, a porsi quindi in attesa del successivo interrupt.

Ipotesi per un corretto funzionamento: "no job overrun"

TIMER-DRIVEN SCHEDULER

1° esempio

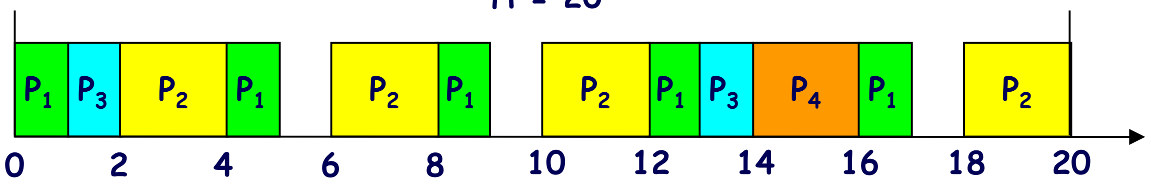
	C	T
P_1	1	4
P_2	2	5
P_3	1	10
P_4	2	20

U = 0.85

?

Istanti decisionali non equidistanziati

H = 20



!

Δt_k, $P(t_k)$

1, P_1
1, P_3
2, P_2
2, P_1
2, P_2
2, P_1
2, P_2
1, P_1
1, P_3
2, P_4
2, P_1
2, P_2

2° esempio

	C	T
P_1	1	6
P_2	5	8
P_3	2	12

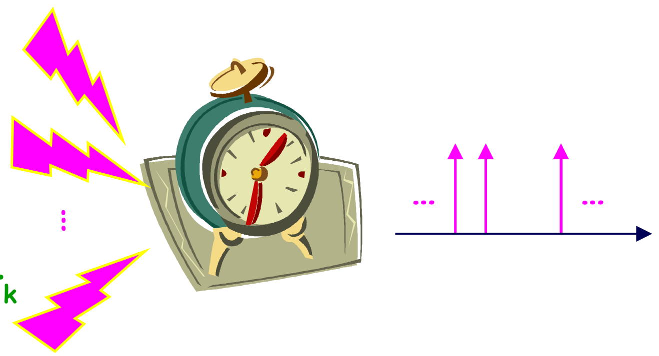
U = 0.96

$\Delta t_1 = t_2 - t_1$

$\Delta t_2 = t_3 - t_2$

⋮

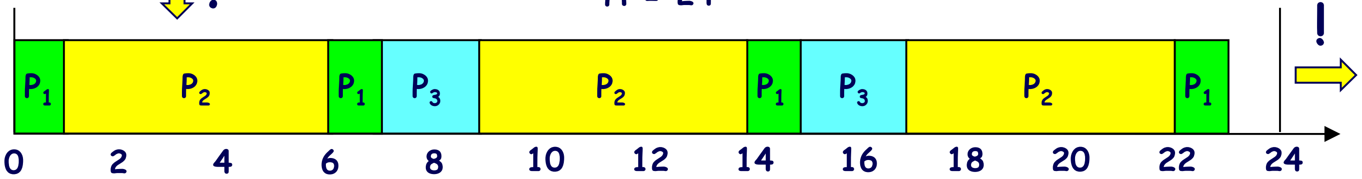
$\Delta t_k = t_{k+1} - t_k$



1, P_1
5, P_2
1, P_1
2, P_3
5, P_2
1, P_1
2, P_3
5, P_2
2, P_1

?

H = 24

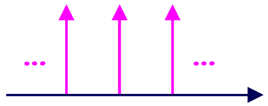


!

CYCLIC EXECUTIVE

Istanti decisionali equidistanziati

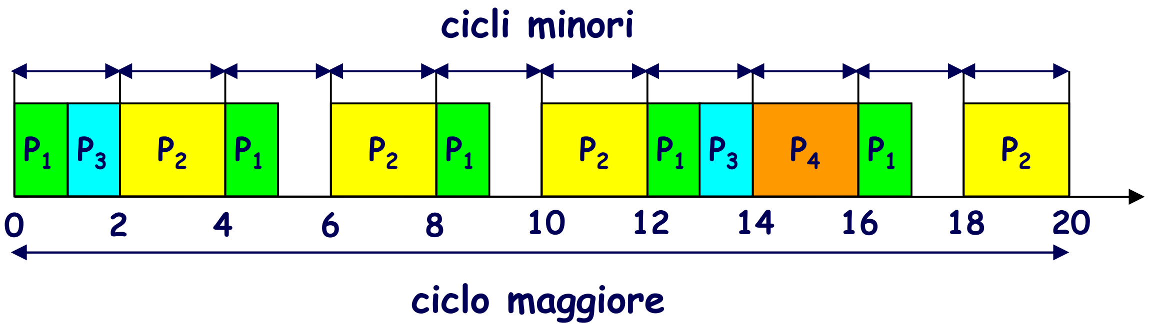
$\Delta t=2$



1° esempio

	C	T
P_1	1	4
P_2	2	5
P_3	1	10
P_4	2	20

?



!

$\underline{P}(t_k)$
P_1, P_3
P_2
P_1
P_2
P_1
P_2
P_1, P_3
P_4
P_1
P_2

2° esempio

	C	T
P_1	1	6
P_2	5	8
P_3	2	12

?

non esiste un feasible schedule, a meno che …

UN TIPICO AMBIENTE DI ESECUZIONE

PREELABORAZIONI INTERNE

↓

LETTURA INGRESSI

↓

ESECUZIONE DEL PROGRAMMA

↓

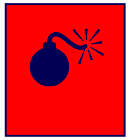
AGGIORNAMENTO USCITE

Programmable Logic Controller (PLC)

una differenza sostanziale rispetto ad altre piattaforme computazionali:
il sistema operativo

programmazione molto semplice
<u>non esiste a run-time</u>
<u>la nozione esplicita di processo</u>
(non si pongono problemi di concorrenza e quindi di condivisione di risorse)
a meno che ...

il tempo necessario per eseguire un'intera scansione del programma risulti incompatibile con i vincoli real-time imposti dall'applicazione



AMBIENTI DI ESECUZIONE SEQUENZIALE …

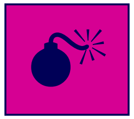
<u>aspetti temporali</u>

un approccio empirico alquanto diffuso

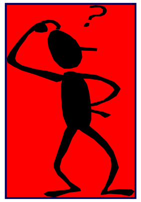
gestione:	A_1	C[ms]	T[ms]	C/T
allarmi	P_1	15	25	.6
motorizzazioni	P_2	5	50	.1
“user interface”	P_3	7.5	100	.075

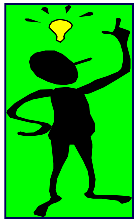
$U(A_1) = 0.775$

$P_1\ P_2\ P_3$ ☹ P_1



$T_{ciclo} = C_1 + C_2 + C_3 = 27.5 > T_1 = 25$





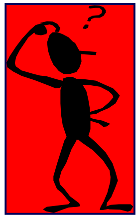
$P_1\ P_2\ P_1\ P_3$ ☺

\+ “I/O refresh”

… AMBIENTI DI ESECUZIONE SEQUENZIALE

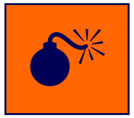
gestione:	A_1'	C[ms]	T[ms]	C/T
allarmi	P_1	15	25	.6
motorizzazioni	P_2	5	50	.1
"user interface"	P_3	7.5	100	.075
termoregolatori	P_4	10	100	.1

$U(A_1') = 0.875$

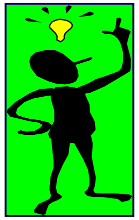


$P_1\ P_2\ P_1\ P_3\ P_1\ P_4$ ☺ P_1

☹ P_2



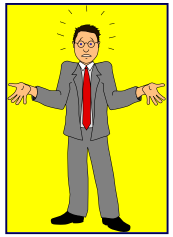
$T_{ciclo} = 3 * C_1 + C_2 + C_3 + C_4 = 67.5 > T_2 = 50$



$P_1\ P_2\ P_1\ P_3\ P_1\ P_2\ P_1\ P_4$ ☺ (😐)

gestione:	A_1''	C[ms]	T[ms]	C/T
allarmi	P_1	15	25	.6
motorizzazioni	P_2	5	50	.1
"user interface"	P_3	17.5	100	.175

$U(A_1'') = 0.875$



L'APPROCCIO CYCLIC EXECUTIVE

Scenario più elementare:
costruzione di un feasible schedule per un insieme di N processi "semplicemente periodici" (ovvero con periodi in relazione armonica).

A_1	C	T	C/T
P_1	15	25	.6
P_2	5	50	.1
P_3	7.5	100	.075

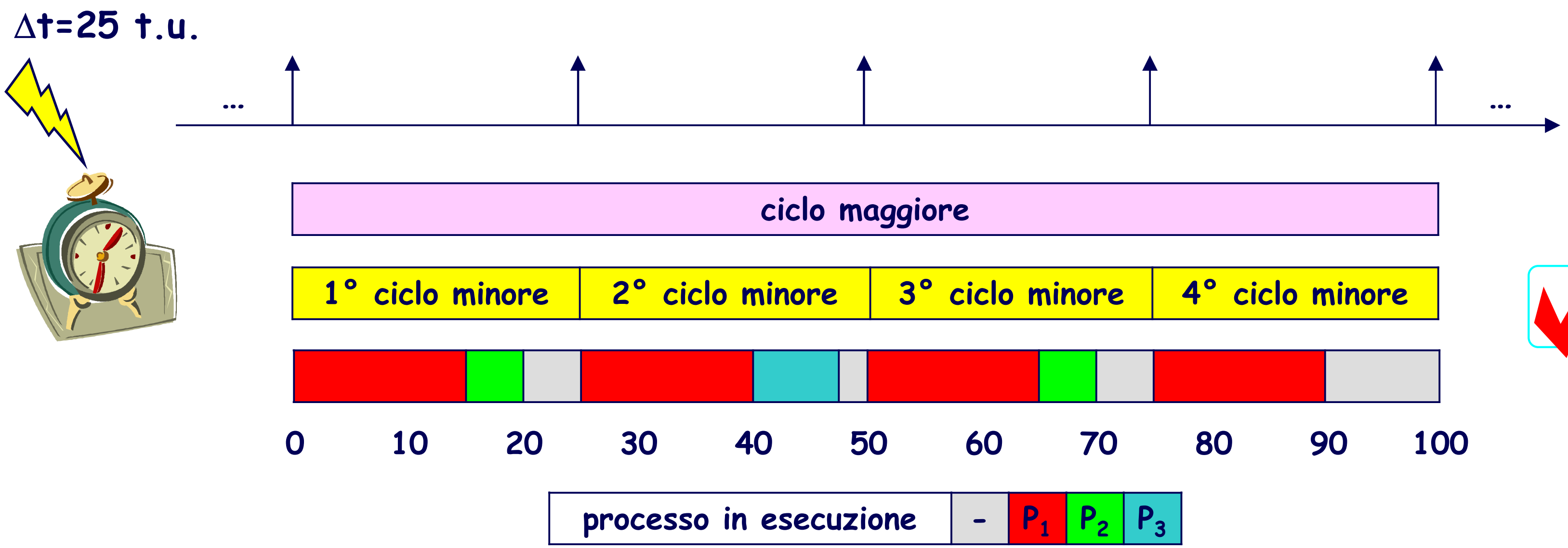
$U(A_1) = 0.775$

estensione del ciclo maggiore:

$M = \max(T_1, T_2, \ldots, T_N) = 100$

estensione del ciclo minore:

$m = \min(T_1, T_2, \ldots, T_N) = 25$



CONSIDERAZIONI REALIZZATIVE

Linguaggio standard per PLC: IEC 61131-3 “structured text”

```
(* loop *)
    if (next_minor_cycle_start_time) (* from rtc interrupt *)
    then
        next_minor_cycle_start_time := false;
        case (minor_cycle) of
            1: call P1_handler; call P2_handler;
            2: call P1_handler; call P3_handler;
            3: call P1_handler; call P2_handler;
            4: call P1_handler;
        end_case;
        if (minor_cycle < last_minor_cycle)
        then minor_cycle := minor_cycle + 1;    (*    next minor cycle *)
        else minor_cycle := first_minor_cycle;  (* repeat major cycle *)
        end_if;
        if (next_minor_cycle_start_time)
        then … ; (* job overrun *)
        end_if;
    end_if;
(* end_loop *)
```

PECULIARITÀ DELL'APPROCCIO …

- semplice realizzabilità ☺
  - non esiste a run time la nozione esplicita di processo
  - non si pongono problemi di concorrenza e quindi di condivisione di risorse

- macchinosità 😐
  - il procedimento può risultare laborioso, specialmente se $\exists$ P_k per cui $T_k >> T_j$, $\forall$ $j \neq k$ (ad es., con riferimento a A_1, se T_3 fosse di 10.000 anziché 100 t.u.)

- limitata diretta applicabilità ☹
  - esistono dei vincoli per quanto riguarda i valori dei parametri T_j e C_j (sempre nel rispetto della condizione $U \leq 1$). Ad es., con riferimento a A_1, se C_3 fosse di 17.5 anziché 7.5 t.u., l'insieme dei processi risulterebbe non schedulabile pur risultando $U = 0.875$, a meno di …

… e LIMITI

… a meno di ricorrere alla tecnica del "job slicing", ovvero ad un partizionamento del (la procedura corrispondente al) processo P_3

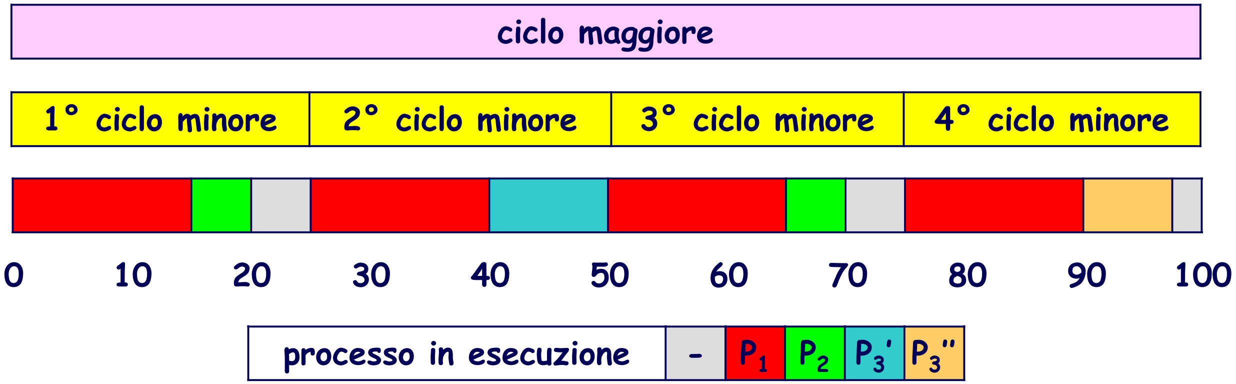
A_2	C	T	C/T
P_1	15	25	.6
P_2	5	50	.1
P_3	17.5	100	.175

$U(A_2) = 0.875$

P_3'	10	100	.1
P_3''	7.5	100	.075

con il vincolo di precedenza

$P_3' \prec P_3''$



STRUTTURA GENERALE [BAKER-SHAW (88)]...

Dimensionamento del ciclo maggiore:

(1) M = iperperiodo = mcm $(T_1, \ldots, T_N)$

Dimensionamento del ciclo minore (o "frame"):

vincolo	motivazione
(2) $M \bmod m = 0$	onde garantire che un ciclo maggiore consista di un numero intero di cicli minori: $n_{cm} = M / m$
(3) $m \geq C_i, \ \forall i$	onde garantire che ogni job di qualunque processo possa iniziare e completare la propria esecuzione all'interno di un ciclo minore ("no job preemption")
(4) $m \leq T_i, \ \forall i$	onde garantire l'esecuzione in ogni ciclo maggiore del numero di job associati a ciascun processo: $n_{Ji} = M / T_i, \ \forall i$

se maggiore, non tutti i periodi possono essere garantiti

in generale il vincolo è più restrittivo

… STRUTTURA GENERALE

Dimensionamento del ciclo minore (o “frame”):

vincolo

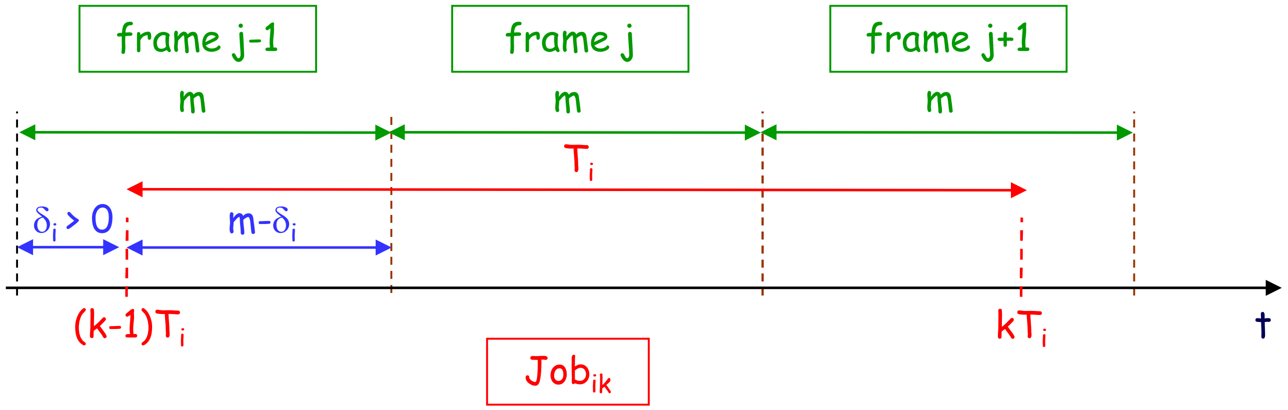
(5) da applicare per tutti i task con periodo non multiplo di m

$(5)\ 2\,m - mcd\,(m,\ T_i) \leq T_i,\ \forall\ i \mid (T_i \bmod m) > 0$

N.B.: $(5) \equiv (4)$ se $(T_i \bmod m) = 0$

motivazione

onde garantire che tra release time e deadline di un qualunque job sia comunque compreso un ciclo minore completo, nell’ambito del quale il job possa essere interamente eseguito (“no job early execution”) ed al termine del quale sia verificabile la condizione “no job overrun”



$$\max\,(m - \delta_i) + m = m - \min\,(\delta_i) + m = m - mcd\,(m,\ T_i) + m \leq T_i,\ \forall\ i$$

spiegazione del punto (5)

ipotesi: $\phi_i = n * m,\ \forall\ i,\ n = 0\ (, 1, 2, \ldots)$

IDENTIFICAZIONE DEL "FRAME-SIZE"

1° esempio

	C	T	C/T
P_1	1	4	0.25
P_2	2	5	0.40
P_3	1	10	0.10
P_4	2	20	0.10

(1) $M = \text{mcm}\ (T_1, \ldots, T_4)$ ⇒ $M = 20$

(3) $m \geq C_i,\ \forall\ i$ ⇒
(4) $m \leq T_i,\ \forall\ i$ ⇒ $m = 2\ ..\ 4$

(2) $M \bmod m = 0$ ⇒ $m = 2, 4$

(5) $2\ m - \text{mcd}\ (m, T_i) \leq T_i$

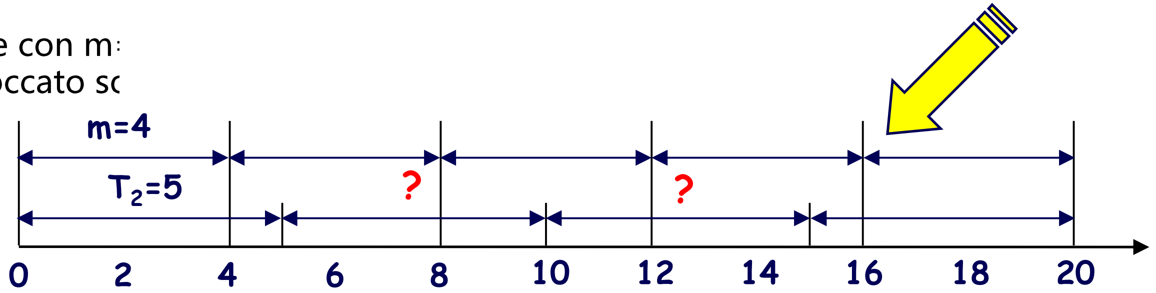
$\forall\ i\ |\ (T_i \bmod m) > 0$

m	i	mcd (m, T_i)	2 m – mcd (m, T_i)	T_i	
4	2 3	1 2	7 6	5 10	NO
2	2	1	3	5	OK

⇓

$m = 2$

qui viene mostrato come con m:
il processore rimane bloccato sc
col processo t2



PIANIFICAZIONE

Procedimento iterativo basato su criteri euristici

L'esecuzione del k-esimo ($k=1,\ldots,n_{Ji}$) job J_{ik} del processo P_i ($i=1,\ldots,N$) può essere pianificata nel j-esimo ($j=1,\ldots,n_{cm}$) ciclo minore c_j se e soltanto se:

$$(k-1)\ T_i \leq (j-1)\ m,$$

$$j\ m \leq k\ T_i,$$

$$C_i \leq m - \sum_s C_s,\ \forall\ s \mid P_s \in S_j,$$

dove S_j denota l'insieme dei processi la cui esecuzione è già stata pianificata in c_j.

Criteri di allocazione "job - ciclo minore" (in caso di più alternative ammissibili):

1. precedenza ai job di processi con frequenza di esecuzione più elevata;
2. precedenza ai job con tempo di esecuzione più elevato;
3. precedenza ai cicli minori con minor tempo residuo libero.

COSTRUZIONE DI UN FEASIBLE SCHEDULE …

	C	T
P_1	1	4
P_2	2	5
P_3	1	10
P_4	2	20

$M = 20,\ m = 2 \Rightarrow n_{cm} = 10 \quad n_{J1} = 5 \quad n_{J2} = 4 \quad n_{J3} = 2 \quad n_{J4} = 1$

identificazione del ciclo minore (o dei cicli minori) in cui ciascun job può essere eseguito:

j11 deve essere eseguito tra 0...4 quindi solo c1 e c2 sono candidati per osp

j41 avendo solo 1 necessita' per iperperiodo puo' partire in ogni momento (essendo c_i lunghi m=2)

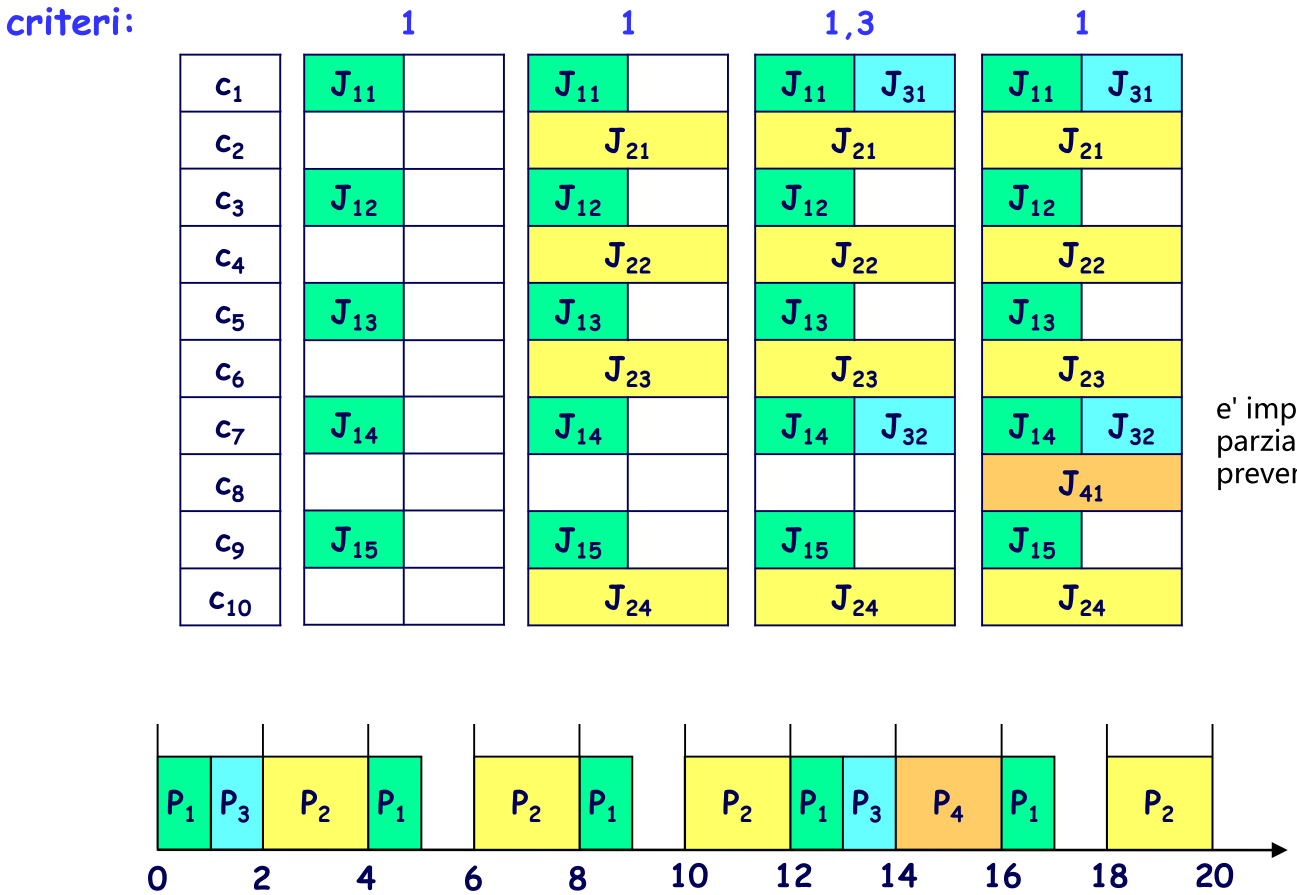
j21 ha alcuni blocchi nor utilizzabili perche' iniziano e finiscono in dı momenti incompatibili c le specifiche richieste

	J_{11}	J_{12}	J_{13}	J_{14}	J_{15}	J_{21}	J_{22}	J_{23}	J_{24}	J_{31}	J_{32}	J_{41}
c_1	×					×				×		×
c_2	×					×				×		×
c_3		×								×		×
c_4		×					×			×		×
c_5			×				×			×		×
c_6			×					×			×	×
c_7				×				×			×	×
c_8				×							×	×
c_9					×				×		×	×
c_{10}					×				×		×	×

0
2
4
6
8
10
12
14
16
18
20
t

… COSTRUZIONE DI UN FEASIBLE SCHEDULE

pianificazione dell'esecuzione di ciascun job:



e' importante cercare di occupare cicli gia' parzialmente occupati (es con j32) per prevenire problemi in futuro: vedi j41

PARTIZIONAMENTO DEI PROCESSI …

Non sempre esiste un feasible schedule tutte le condizioni devono essere rispettate

2° esempio

	C	T	C/T
P_1	2	5	0.40
P_2	2	8	0.25
P_3	5	20	0.25

(1) $M = \text{mcm}\ (T_1, \ldots, T_3)$ ⇨ $M = 40$

(3) $m \geq C_i, \ \forall\ i$ ⇨

(4) $m \leq T_i, \ \forall\ i$ ⇨ $\left.\right\}$ $m = 5$

(2) $M \bmod m = 0$ ⇨ $m = 5$

(5) $2\ m - \text{mcd}\ (m,\ T_i) \leq T_i$

$\forall\ i \mid (T_i \bmod m) > 0$

m	i	mcd (m, T_i)	2 m − mcd (m, T_i)	T_i
5	2	1	9	8

NO

In tali casi occorre partizionare il processo (o i processi) con maggiore tempo di esecuzione in sottoprocessi ("job slicing"), rilassando così il vincolo (3):

$$P_i\ (C_i, T_i) \Rightarrow P_i'\ (C_i', T_i),\ P_i''\ (C_i'', T_i),\ \ldots$$

con $C_i' + C_i'' + \ldots = C_i$ ed i vincoli di precedenza $P_i' \prec P_i'' \prec \ldots$

… PARTIZIONAMENTO DEI PROCESSI …

	C	T	C/T
P_1	2	5	0.40
P_2	2	8	0.25
P_3	5	20	0.25

per risolvere il problema, si partiziona il processo

P_3 (5,20) $\Rightarrow$ $P_{3'}$ (3,20), $P_{3''}$ (2,20)

	C	T	C/T
P_1	2	5	0.40
P_2	2	8	0.25
$P_{3'}$	3	20	0.15
$P_{3''}$	2	20	0.10

$\boxed{P_{3'} \prec P_{3''}}$

(1) $M = \text{mcm}\ (T_1, \ldots, T_3)$ $\Rightarrow$ $M = 40$

(3) $m \geq C_i, \ \forall\ i$ $\Rightarrow$

(4) $m \leq T_i, \ \forall\ i$ $\Rightarrow$

$\}$ $m = 3\ ..\ 5$

(2) $M \bmod m = 0$ $\Rightarrow$ $m = 4,\ 5$

NO

(5) $2\ m - \text{mcd}\ (m,\ T_i) \leq T_i$

$\forall\ i \mid (T_i \bmod m) > 0$

m	i	mcd (m, T_i)	2 m - mcd (m, T_i)	T_i
4	1	1	7	5

NO

Ulteriore rilassamento del vincolo (3):

$P_{3'}$ (3,20) $\Rightarrow$ P_{3A} (2,20), P_{3B} (1,20) con $P_{3A} \prec P_{3B}$

… PARTIZIONAMENTO DEI PROCESSI …

	C	T	C/T
P_1	2	5	0.40
P_2	2	8	0.25
$P_{3'}$	3	20	0.15
$P_{3''}$	2	20	0.10

$P_{3'}$ (3,20) $\Rightarrow$ P_{3A} (2,20), P_{3B} (1,20)

necessario partizionamento ulteriore per completamento corretto

	C	T	C/T
P_1	2	5	0.40
P_2	2	8	0.25
P_{3A}	2	20	0.10
P_{3B}	1	20	0.05
$P_{3''}$	2	20	0.10

$\boxed{P_{3A} \prec P_{3B} \prec P_{3''}}$

(1) $M = mcm\ (T_1, \ldots, T_3)$ $\Rightarrow$ $M = 40$

(3) $m \geq C_i, \ \forall\ i$ $\Rightarrow$

(4) $m \leq T_i, \ \forall\ i$ $\Rightarrow$ $\left.\right\}$ $m = 2 \ldots 5$

(2) $M \bmod m = 0$ $\Rightarrow$ $m = 2, 4, 5$

NO

(5) $2\ m - mcd\ (m, T_i) \leq T_i$

$\forall\ i \mid (T_i \bmod m) > 0$

m	i	mcd (m, T_i)	2 m – mcd (m, T_i)	T_i
2	1	1	3	5

OK

… PARTIZIONAMENTO DEI PROCESSI …

	C	T
P_1	2	5
P_2	2	8
P_{3A}	2	20
P_{3B}	1	20
$P_{3''}$	2	20

$P_{3A} \prec P_{3B} \prec P_{3''}$

$M = 40,\ m = 2$

$\Downarrow$

$n_{cm} = 20$

$n_{J1} = 8$

$n_{J2} = 5$

$n_{J3A} = n_{J3B} = n_{J3''} = 2$

	J_{11}	J_{12}	J_{13}	J_{14}	J_{15}	J_{16}	J_{17}	J_{18}	J_{21}	J_{22}	J_{23}	J_{24}	J_{25}	$J_{3.1}$	$J_{3.2}$
c_1	×								×					×	
c_2	×								×					×	
c_3									×					×	
c_4		×							×					×	
c_5		×								×				×	
c_6			×							×				×	
c_7			×							×				×	
c_8										×				×	
c_9				×							×			×	
c_{10}				×							×			×	
c_{11}					×						×				×
c_{12}					×						×				×
c_{13}												×			×
c_{14}						×						×			×
c_{15}						×						×			×
c_{16}							×					×			×
c_{17}							×						×		×
c_{18}													×		×
c_{19}								×					×		×
c_{20}								×					×		×

0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 t

… PARTIZIONAMENTO DEI PROCESSI

	C	T
P_1	2	5
P_2	2	8
P_{3A}	2	20
P_{3B}	1	20
$P_{3''}$	2	20

$P_{3A} \prec P_{3B} \prec P_{3''}$

$M = 40,\ m = 2$

$\Downarrow$

$n_{cm} = 20$

$n_{J1} = 8$

$n_{J2} = 5$

$n_{J3A} = n_{J3B} = n_{J3''} = 2$

c_1	J_{11}	J_{11}	J_{11}	J_{11}	J_{11}
c_2		J_{21}	J_{21}	J_{21}	J_{21}
c_3			J_{3A1}	J_{3A1}	J_{3A1}
c_4	J_{12}	J_{12}	J_{12}	J_{12}	J_{12}
c_5		J_{22}	J_{22}	J_{22}	J_{22}
c_6	J_{13}	J_{13}	J_{13}	J_{13}	J_{13}
c_7				J_{3B1}	J_{3B1}
c_8					$J_{3''1}$
c_9	J_{14}	J_{14}	J_{14}	J_{14}	J_{14}
c_{10}		J_{23}	J_{23}	J_{23}	J_{23}
c_{11}	J_{15}	J_{15}	J_{15}	J_{15}	J_{15}
c_{12}			J_{3A2}	J_{3A2}	J_{3A2}
c_{13}		J_{24}	J_{24}	J_{24}	J_{24}
c_{14}	J_{16}	J_{16}	J_{16}	J_{16}	J_{16}
c_{15}				J_{3B2}	J_{3B2}
c_{16}	J_{17}	J_{17}	J_{17}	J_{17}	J_{17}
c_{17}		J_{25}	J_{25}	J_{25}	J_{25}
c_{18}					$J_{3''2}$
c_{19}	J_{18}	J_{18}	J_{18}	J_{18}	J_{18}
c_{20}					

CRITERI EURISTICI A SUPPORTO DELLA PIANIFICAZIONE …

3° esempio

	C	T	C/T
P_1	2	8	0.250
P_2	1	9	0.111
P_3	4	12	0.333
P_4	4	36	0.111
P_5	4	72	0.056

(1) $M = \text{mcm}\ (T_1, \ldots, T_5)$ ⇨ $M = 72$

(3) $m \geq C_i,\ \forall\ i$ ⇨

(4) $m \leq T_i,\ \forall\ i$ ⇨

$\Big\}$ $m = 4\ ..\ 8$

(2) $M \bmod m = 0$ ⇨ $m = 4,\ 6,\ 8$

(5) $2\ m - \text{mcd}\ (m,\ T_i) \leq T_i$

$\forall\ i \mid (T_i \bmod m) > 0$

m	i	mcd (m, T_i)	2 m - mcd (m, T_i)	T_i	
8	2 3 4	1 4 4	15 12 12	9 12 36	NO
6	1 2	2 3	10 9	8 9	NO
4	2	1	7	9	OK

⇩

$\boxed{m = 4}$

… CRITERI EURISTICI …

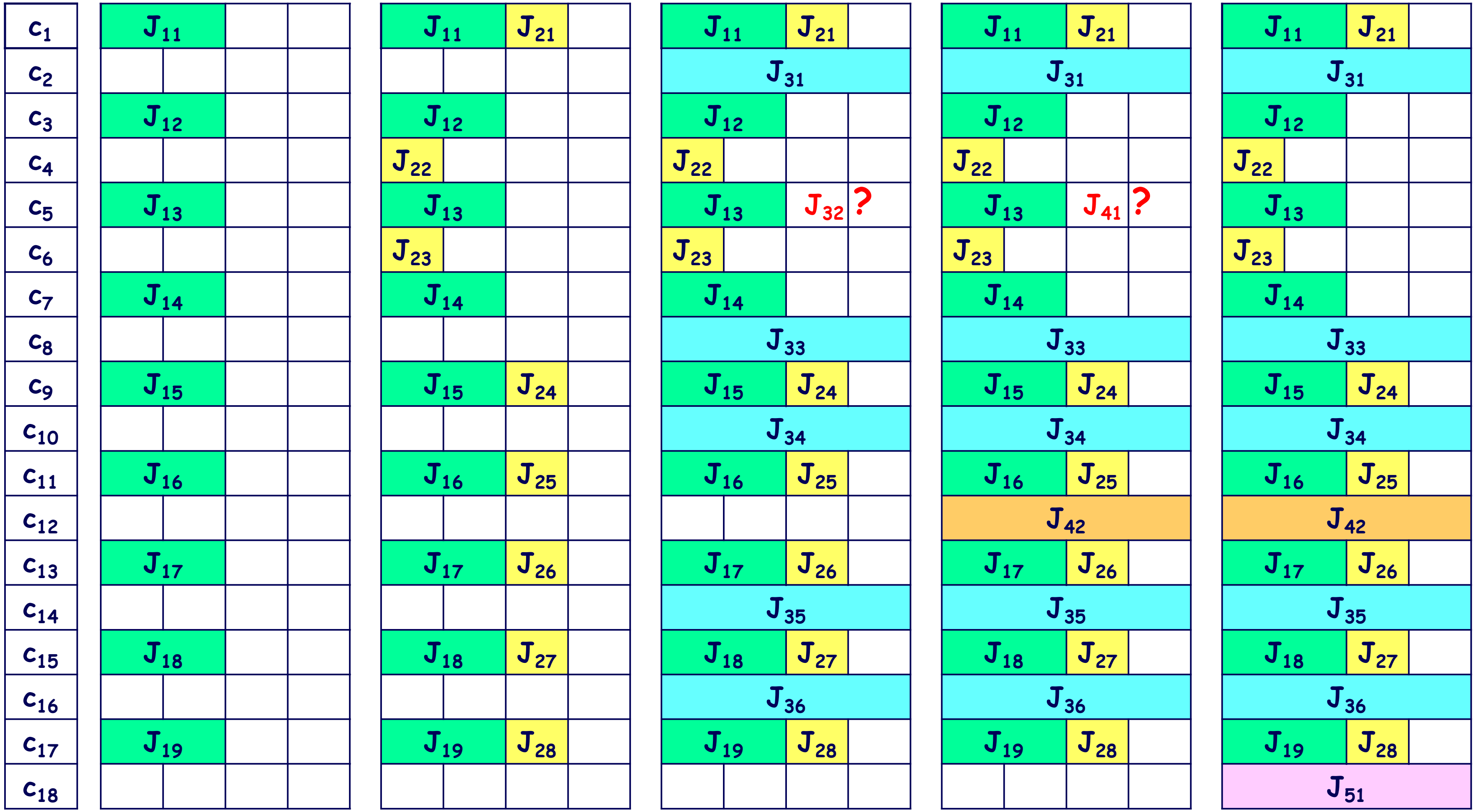
	J_{11}	J_{12}	J_{13}	J_{14}	J_{15}	J_{16}	J_{17}	J_{18}	J_{19}	J_{21}	J_{22}	J_{23}	J_{24}	J_{25}	J_{26}	J_{27}	J_{28}	J_{31}	J_{32}	J_{33}	J_{34}	J_{35}	J_{36}	J_{41}	J_{42}	J_{51}
c_1	×									×								×						×		×
c_2	×									×								×						×		×
c_3		×																×						×		×
c_4		×									×								×					×		×
c_5			×																×					×		×
c_6			×									×							×					×		×
c_7				×																×				×		×
c_8				×									×							×				×		×
c_9					×								×							×				×		×
c_{10}					×									×							×				×	×
c_{11}						×								×							×				×	×
c_{12}						×															×				×	×
c_{13}							×								×							×			×	×
c_{14}							×															×			×	×
c_{15}								×								×						×			×	×
c_{16}								×															×		×	×
c_{17}									×								×						×		×	×
c_{18}									×								×						×		×	×

0 4 8 12 16 20 24 28 32 36 40 44 48 52 56 60 64 68 72 t

$T_1 = 8$ $T_2 = 9$ $T_3 = 12$ $T_4 = 36$ $T_5 = 72$

$M = 72,\ m = 4$ ⇨ $n_{cm} = 18$ $n_{J1} = 9$ $n_{J2} = 8$ $n_{J3} = 6$ $n_{J4} = 2$ $n_{J5} = 1$

… CRITERI EURISTICI …



E' possibile identificare un feasible schedule ricorrendo alla tecnica del job slicing:

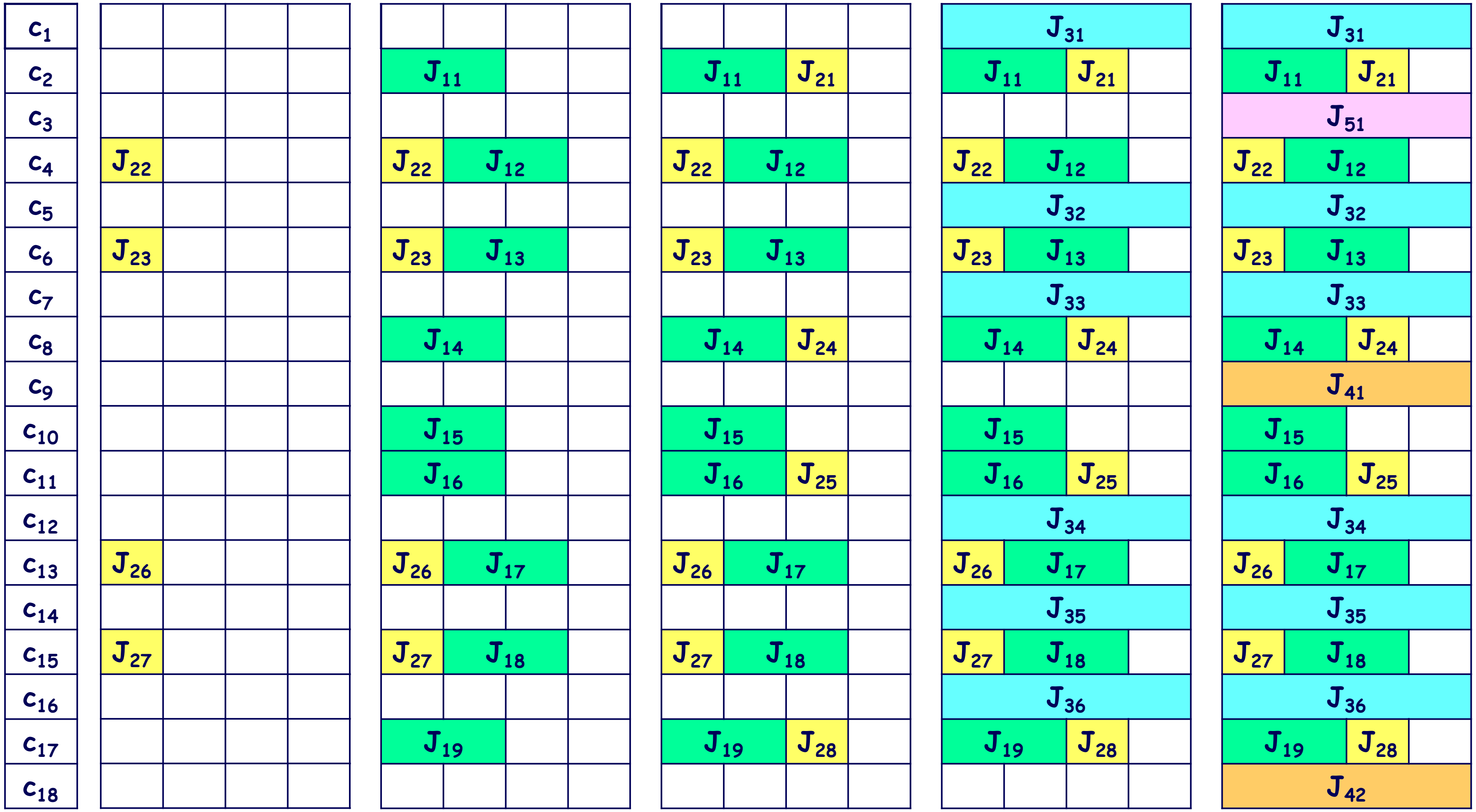
P_3 (4,12) $\Rightarrow$ $P_{3'}$ (2,12), $P_{3''}$ (2,12) con $P_{3'} \prec P_{3''}$ $\Rightarrow$ $J_{32'}$ in c_4, $J_{32''}$ in c_5

P_4 (4,36) $\Rightarrow$ $P_{4'}$ (2,36), $P_{4''}$ (2,36) con $P_{4'} \prec P_{4''}$ $\Rightarrow$ $J_{41'}$ in c_6, $J_{41''}$ in c_7

… CRITERI EURISTICI …

E' possibile identificare un feasible schedule senza ricorrere alla tecnica del job slicing privilegiando l'allocazione dei job che non ammettono alternative.

quelli con le x singole



… CRITERI EURISTICI …

4° esempio

	C	T	C/T
P_1	1	6	0.167
P_2	5	8	0.625
P_3	2	12	0.167

(1) $M = mcm\ (T_1, \ldots, T_3)$ ⇨ $M = 24$

(3) $m \geq C_i,\ \forall\ i$ ⇨

(4) $m \leq T_i,\ \forall\ i$ ⇨ $m = 5,\ 6$

(2) $M \bmod m = 0$ ⇨ $m = 6$

(5) $2\ m - mcd\ (m,\ T_i) \leq T_i$

$\forall\ i \mid (T_i \bmod m) > 0$

m	i	mcd (m, T_i)	2 m - mcd (m, T_i)	T_i
6	2	2	10	8

NO

Rilassamento del vincolo (3):
$P_2\ (5,8) \Rightarrow P_{2'}\ (3,8),\ P_{2''}\ (2,8)$
con $P_{2'} \prec P_{2''}$

… CRITERI EURISTICI …

	C	T	C/T
P_1	1	6	0.167
P_2	5	8	0.625
P_3	2	12	0.167

$P_2\ (5,8) \Rightarrow P_{2'}\ (3,8),\ P_{2''}\ (2,8)$

	C	T	C/T
P_1	1	6	0.167
$P_{2'}$	3	8	0.375
$P_{2''}$	2	8	0.250
P_3	2	12	0.167

$\boxed{P_{2'} \prec P_{2''}}$

(1) $M = \text{mcm}\ (T_1, \ldots, T_3) \Rightarrow M = 24$

(3) $m \geq C_i,\ \forall\ i \Rightarrow$

(4) $m \leq T_i,\ \forall\ i \Rightarrow$

$m = 3\ ..\ 6$

(2) $M \bmod m = 0 \Rightarrow m = 3,\ 4,\ 6$

NO

m	i	mcd (m, T_i)	2 m – mcd (m, T_i)	T_i
4	1	2	6	6
3	2	1	5	8

OK

OK

(5) $2\ m - \text{mcd}\ (m, T_i) \leq T_i,\ \forall\ i \mid (T_i \bmod m) > 0 \Rightarrow m = 3,\ 4$

… CRITERI EURISTICI …

	C	T
P_1	1	6
$P_{2'}$	3	8
$P_{2''}$	2	8
P_3	2	12

$P_{2'} \prec P_{2''}$

$\boxed{m = 4}$ $M = 24 \Rightarrow n_{cm} = 6$ $n_{J1} = 4$ $n_{J2'} = n_{J2''} = 3$ $n_{J3} = 2$

tabella 1

	J_{11}	J_{12}	J_{13}	J_{14}	$J_{2.1}$	$J_{2.2}$	$J_{2.3}$	J_{31}	J_{32}
c_1	×				×			×	
c_2					×			×	
c_3		×				×		×	
c_4			×			×			×
c_5							×		×
c_6				×			×		×

0 4 8 12 16 20 24 t

essendo p2 sliced, p2'+p2" non stanno all'interno dei vari c_i, questo significa che c'e' dissonanza tra tabella 1 e tabella 2

tabella 2

c_1	J_{11}			
c_2				
c_3	J_{12}			
c_4	J_{13}			
c_5				
c_6	J_{14}			

c_1	J_{11}	$J_{2'1}$		
c_2	$J_{2''1}$			
c_3	J_{12}	$J_{2'2}$		
c_4	J_{13}	$J_{2''2}$		
c_5	$J_{2'3}$			
c_6	J_{14}	$J_{2''3}$		

c_1	J_{11}	$J_{2'1}$		
c_2	$J_{2''1}$		J_{31}	
c_3	J_{12}	$J_{2'2}$		
c_4	J_{13}	$J_{2''2}$		
c_5	$J_{2'3}$			
c_6	J_{14}	$J_{2''3}$		

J_{32} ?

c_1	J_{11}	$J_{2'1}$		
c_2	$J_{2''1}$		$J_{3'1}$	$J_{3''1}$
c_3	J_{12}	$J_{2'2}$		
c_4	J_{13}	$J_{2''2}$		$J_{3'2}$
c_5	$J_{2'3}$			$J_{3''2}$
c_6	J_{14}	$J_{2''3}$		

E' possibile identificare un feasible schedule
solo ricorrendo nuovamente alla tecnica del job slicing.

$\Rightarrow$ $P_3\ (2,12) \Rightarrow$
$P_{3'}\ (1,12),\ P_{3''}\ (1,12)$
$P_{3'} \prec P_{3''}$

… CRITERI EURISTICI

	C	T
P_1	1	6
$P_{2'}$	3	8
$P_{2''}$	2	8
P_3	2	12

$P_{2'} \prec P_{2''}$

$m = 3$ $M = 24 \Rightarrow n_{cm} = 8$ $n_{J1} = 4$ $n_{J2'} = n_{J2''} = 3$ $n_{J3} = 2$

	J_{11}	J_{12}	J_{13}	J_{14}	$J_{2.1}$	$J_{2.2}$	$J_{2.3}$	J_{31}	J_{32}	t
c_1	×				×			×		0
c_2	×				×			×		3
c_3		×						×		6
c_4		×				×		×		9
c_5			×			×			×	12
c_6			×						×	15
c_7				×			×		×	18
c_8				×			×		×	21
										24

Anche in questo caso, applicando nell'ordine i criteri euristici delineati, è possibile identificare un feasible schedule solo ricorrendo nuovamente alla tecnica del job slicing.

E' possibile in alternativa identificare direttamente un feasible schedule privilegiando l'allocazione dei job contraddistinti da vincoli di precedenza.

c_1	$J_{2'1}$	$J_{2'1}$	$J_{2'1}$
c_2	$J_{2''1}$	$J_{2''1}$, J_{11}	$J_{2''1}$, J_{11}
c_3		J_{12}	J_{12}, J_{31}
c_4	$J_{2'2}$	$J_{2'2}$	$J_{2'2}$
c_5	$J_{2''2}$	$J_{2''2}$, J_{13}	$J_{2''2}$, J_{13}
c_6			J_{32}
c_7	$J_{2'3}$	$J_{2'3}$	$J_{2'3}$
c_8	$J_{2''3}$	$J_{2''3}$, J_{14}	$J_{2''3}$, J_{14}

SCHEDULAZIONE “PRIORITY-DRIVEN”

- Ad ogni processo è staticamente o dinamicamente associata una priorità in dipendenza dei corrispondenti requisiti temporali.
- Ad ogni processo è dinamicamente associata una informazione che ne identifica lo stato ai fini della esecuzione:
  - processo IDLE
  - processo READY
  - processo RUNNING
- L’esecuzione di un processo è sospesa se un altro processo di priorità superiore è pronto per l’esecuzione (PREEMPTION).

L'ALGORITMO RATE MONOTONIC PRIORITY ORDERING (RMPO)

Ad ogni processo è staticamente associata una priorità direttamente proporzionale alla corrispondente frequenza di esecuzione:

$p(P_j) = p_j \propto 1 \,/\, T_j = 1 \,/\, D_j \quad (j = 1, 2, \ldots, N)$

OTTIMALITÀ DELL'ALGORITMO

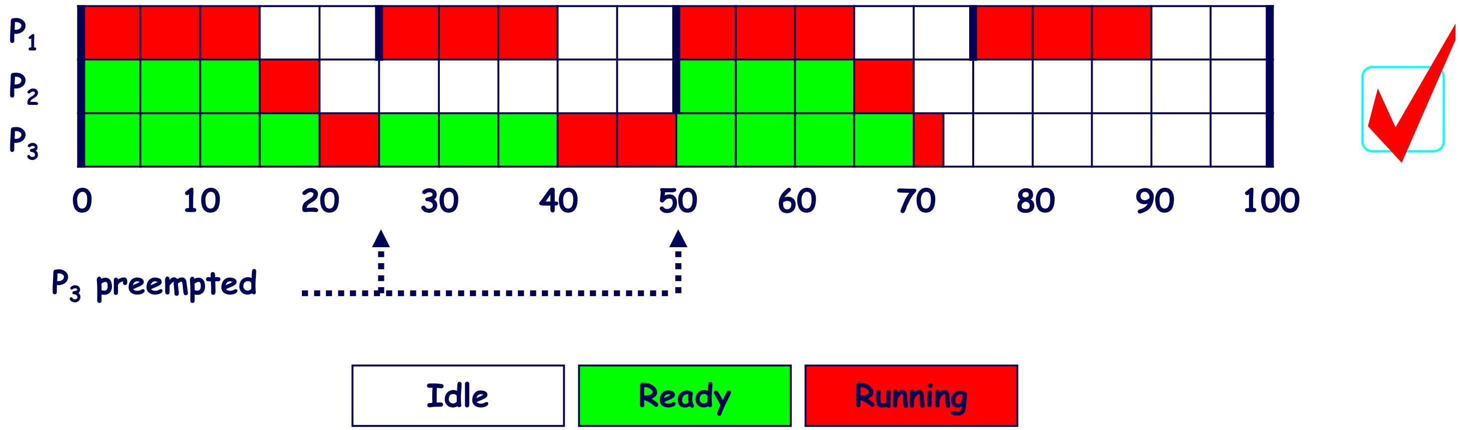
Se un insieme di processi periodici è schedulabile con un qualche algoritmo che prevede un'attribuzione statica di priorità, allora tale insieme è schedulabile anche con RMPO.

Se un insieme di processi periodici non è schedulabile con RMPO, allora tale insieme non è schedulabile con alcun altro algoritmo che preveda un'attribuzione statica di priorità.

IL MECCANISMO DELLA PREEMPTION

A_2	C	T	C/T	p
P_1	15	25	.6	max
P_2	5	50	.1	
P_3	17.5	100	.175	min

$U(A_2) = 0.875$



ANALISI DI SCHEDULABILITÀ

- costruzione di diagrammi temporali che evidenziano l'evoluzione dei processi dal punto di vista dell'esecuzione

- applicazione di criteri basati sul fattore di utilizzazione del processore da parte dei (singoli) processi

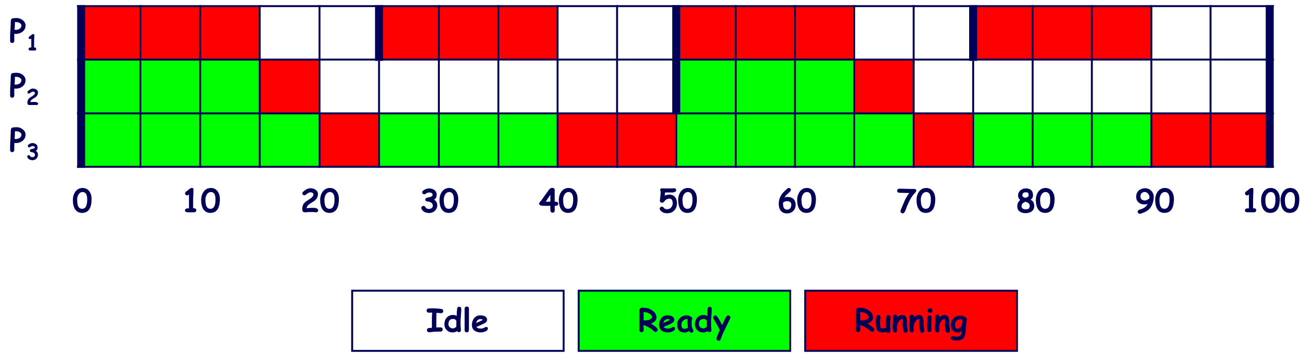
- calcolo dei tempi di risposta dei singoli processi

Esito sempre conclusivo? sì no

ANALISI DI SCHEDULABILITÀ ATTRAVERSO DIAGRAMMI TEMPORALI …

A_3	C	T	C/T	p
P_1	15	25	0.6	max
P_2	5	50	0.1	
P_3	30	100	0.3	min

$U(A_3) = 1$

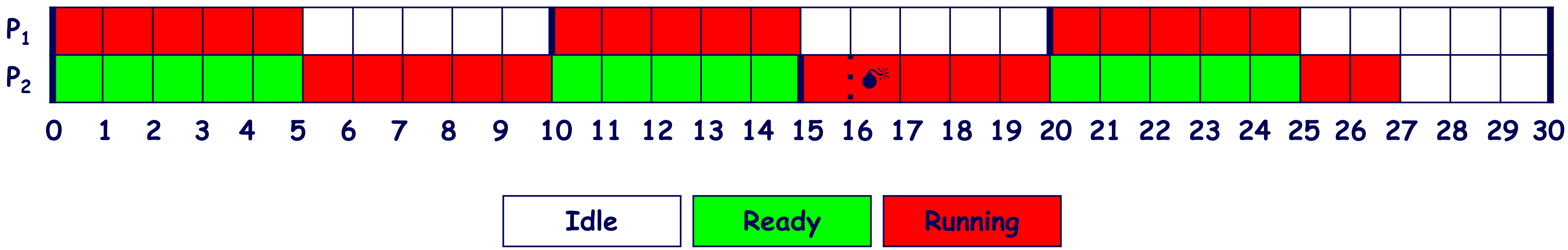


$U \leq 1$ è condizione sufficiente per la schedulabilità di un insieme di processi "semplicemente periodici" (con periodi in relazione armonica).

… ANALISI DI SCHEDULABILITÀ ATTRAVERSO DIAGRAMMI TEMPORALI …

A_4	C	T	C/T	p
P_1	5	10	0.5	max
P_2	6	15	0.4	min

$U(A_4) = 0.9$



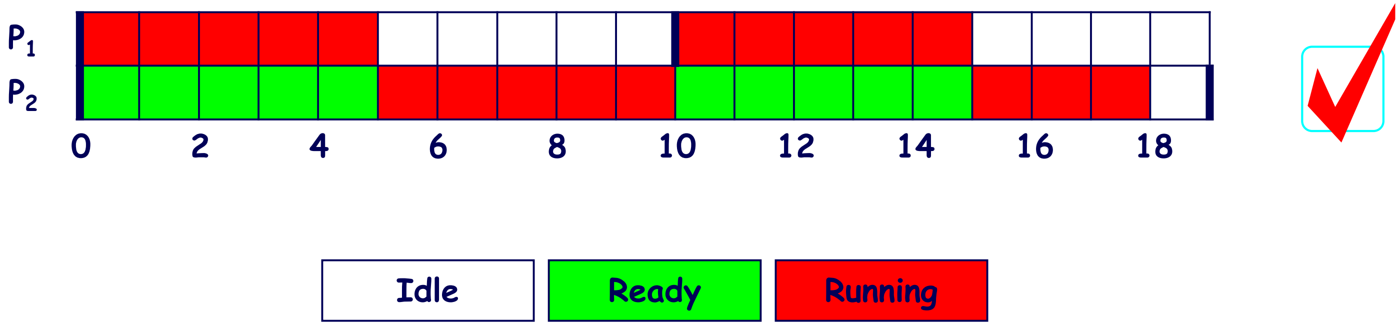
missed deadline (💣): insieme di processi non schedulabile

Quando non sussiste una relazione armonica tra i periodi dei processi, la condizione $U \leq 1$ non è sufficiente per garantire la schedulabilità.

...ANALISI DI SCHEDULABILITÀ ATTRAVERSO DIAGRAMMI TEMPORALI

A_5	C	T	C/T	p
P_1	5	10	0.50	max
P_2	8	19	0.42	min

$U(A_5) = 0.92$

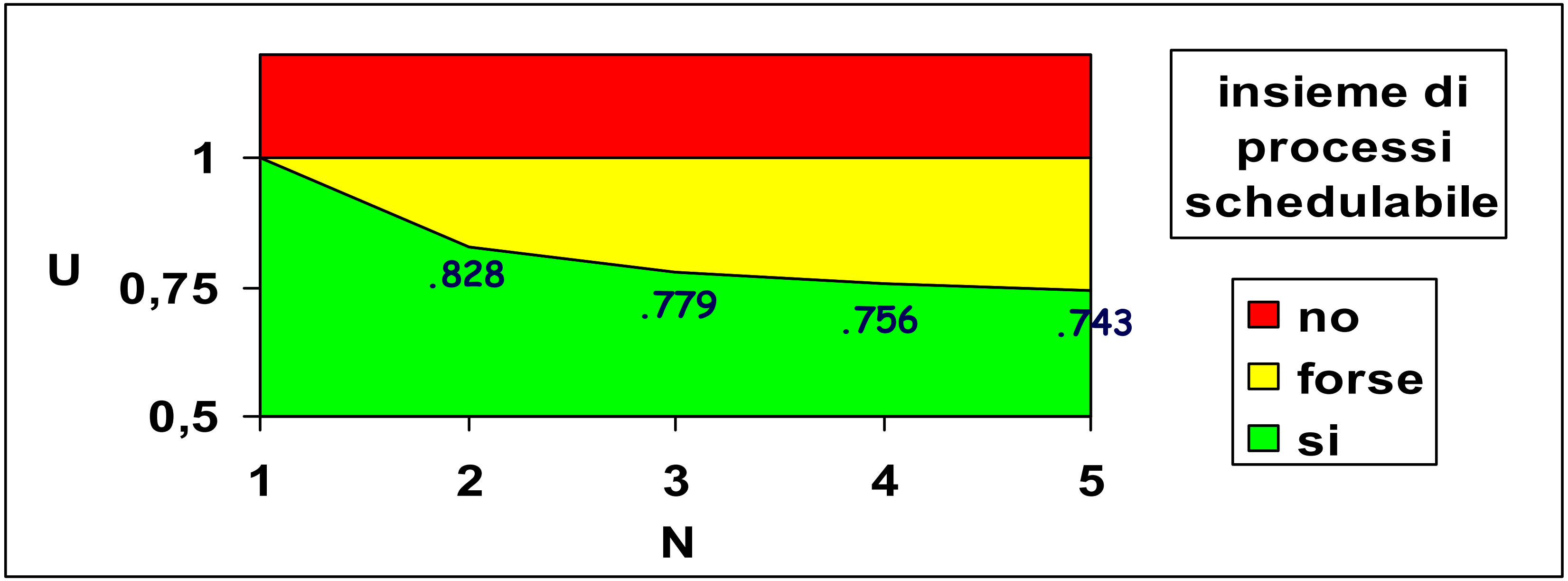


La schedulabilità è garantita se per ogni processo è rispettata la prima deadline a partire dalla condizione più sfavorevole di contemporanea attivazione del processo stesso e di tutti i processi di priorità superiore (istante critico).

TEST DI SCHEDULABILITÀ BASATO SUL FATTORE DI UTILIZZAZIONE DEL PROCESSORE [LIU-LAYLAND (73)]

Condizione sufficiente (ma non necessaria) affinché un insieme di N processi sia schedulabile con l'algoritmo RMPO è che:

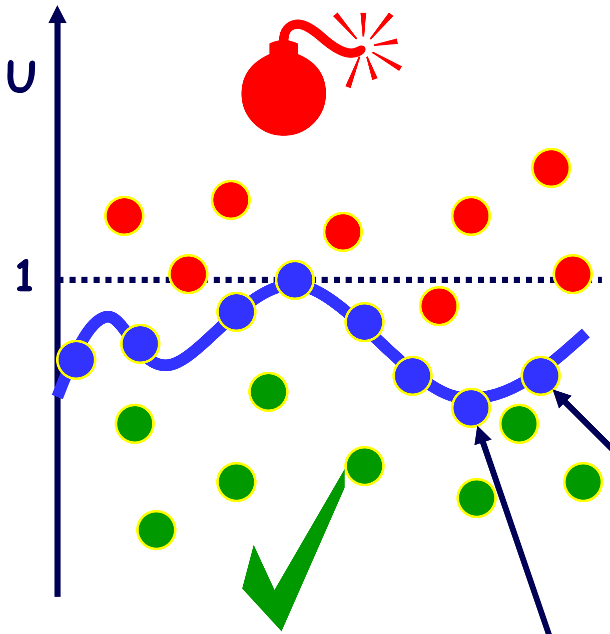
$$U \leq U_{RMPO}(N) = N\left(2^{1/N} - 1\right)$$



$$\lim_{N\to\infty} U_{RMPO}(N) = \ln 2 = 0.693$$

IL LIMITE SUPERIORE MINIMO (LEAST UPPER BOUND) DEL FATTORE DI UTILIZZAZIONE

L'applicazione di un algoritmo di scheduling (A) induce nello spazio dei possibili insiemi di processi $\Sigma \equiv \{\sigma\ (N,\ T_1,\ \ldots,\ T_N,\ C_1,\ \ldots,\ C_N)\}$ una partizione in tre sottospazi disgiunti Σ_1, Σ_2, Σ_3



$\Sigma_1 \equiv \{\sigma \mid$ insieme di processi non schedulabile$\}$

$\Sigma_2 \equiv \{\sigma \mid$ insieme di processi "difficilmente schedulabile" (una variazione di C_j $(C_j+\varepsilon)$ o di T_j $(T_j-\varepsilon)$ per un qualunque P_j in σ conduce ad un nuovo insieme di processi σ' non più schedulabile)$\}$

$\Sigma_3 \equiv \{\sigma \mid$ insieme di processi schedulabile cui non corrisponde la "piena utilizzazione" del processore$\}$

$$U_{lub}\ (A) = \min_{\sigma \in \Sigma_2} \{U_{ub}\ (\sigma,\ A)\}$$

least upper bound — upper bound

CALCOLO DI U_{lub} per RMPO

1) Analisi del caso elementare corrispondente a N = 2 processi

$$T_1$$

$$\frac{T_2}{T_1} = \left\lfloor \frac{T_2}{T_1} \right\rfloor + \frac{T_2 \bmod T_1}{T_1} = i + f$$

$$U_1 = \frac{C_1}{T_1}$$

?

$$U_{2ub}\left(U_1, T_2 / T_1\right)$$

limite superiore
(upper bound) di U_2

?

$$U_{ub}\left(U_1, T_1 / T_2\right) = U_1 + U_{2ub}$$

limite superiore
(upper bound) di U

?

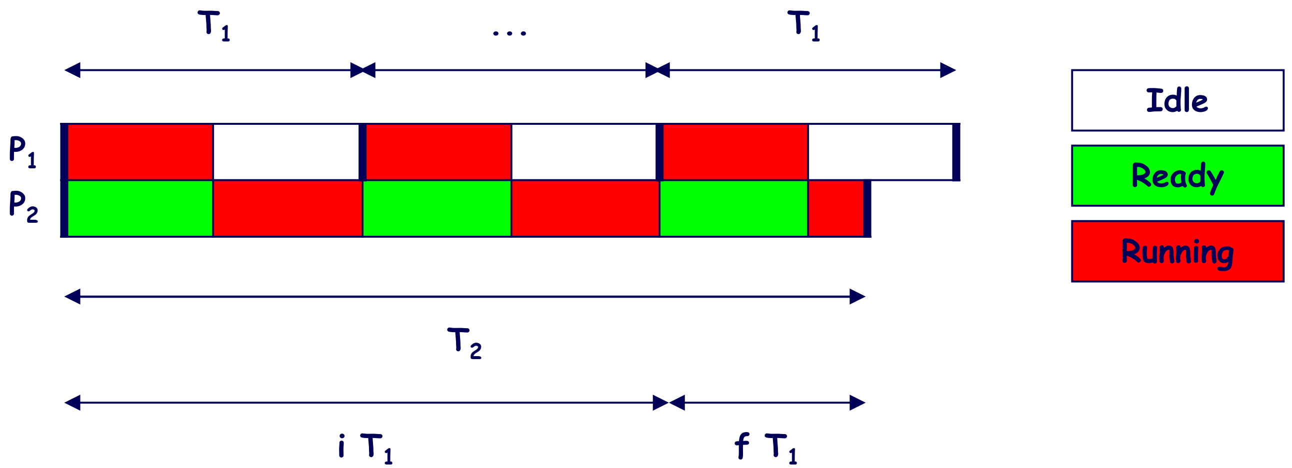
$$U_{lub} = \min\left\{U_{ub}\right\}$$

limite superiore minimo
(least upper bound) di U

2) Generalizzazione dei risultati nel caso di N > 2 processi

CALCOLO DI U_{ub} (N=2) ...

$$1° \text{ caso} : f > U_1, \text{ ovvero } f\,T_1 > C_1$$

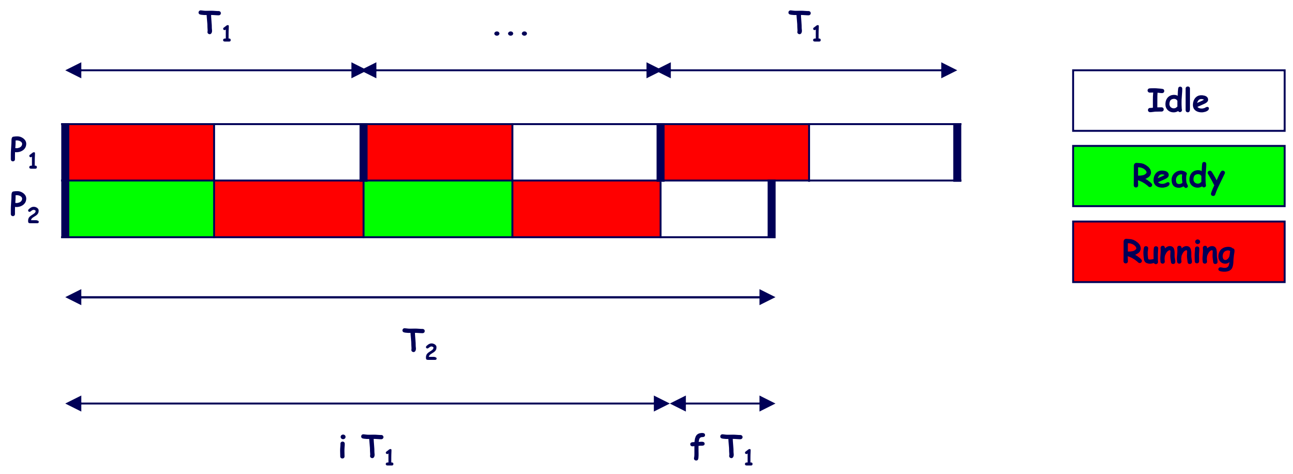


$$C_2 \leq i\left(T_1 - C_1\right) + f\,T_1 - C_1 = \left[i\left(1 - U_1\right) + f - U_1\right]T_1 = C_{2ub}$$

$$U_2 = \frac{C_2}{T_2} \leq \frac{i\left(1 - U_1\right) + f - U_1}{i + f} = U_{2ub}$$

… CALCOLO DI U_{ub} (N=2) …

$$2° \text{ caso} : f \le U_1, \text{ ovvero } \; f\,T_1 \le C_1$$

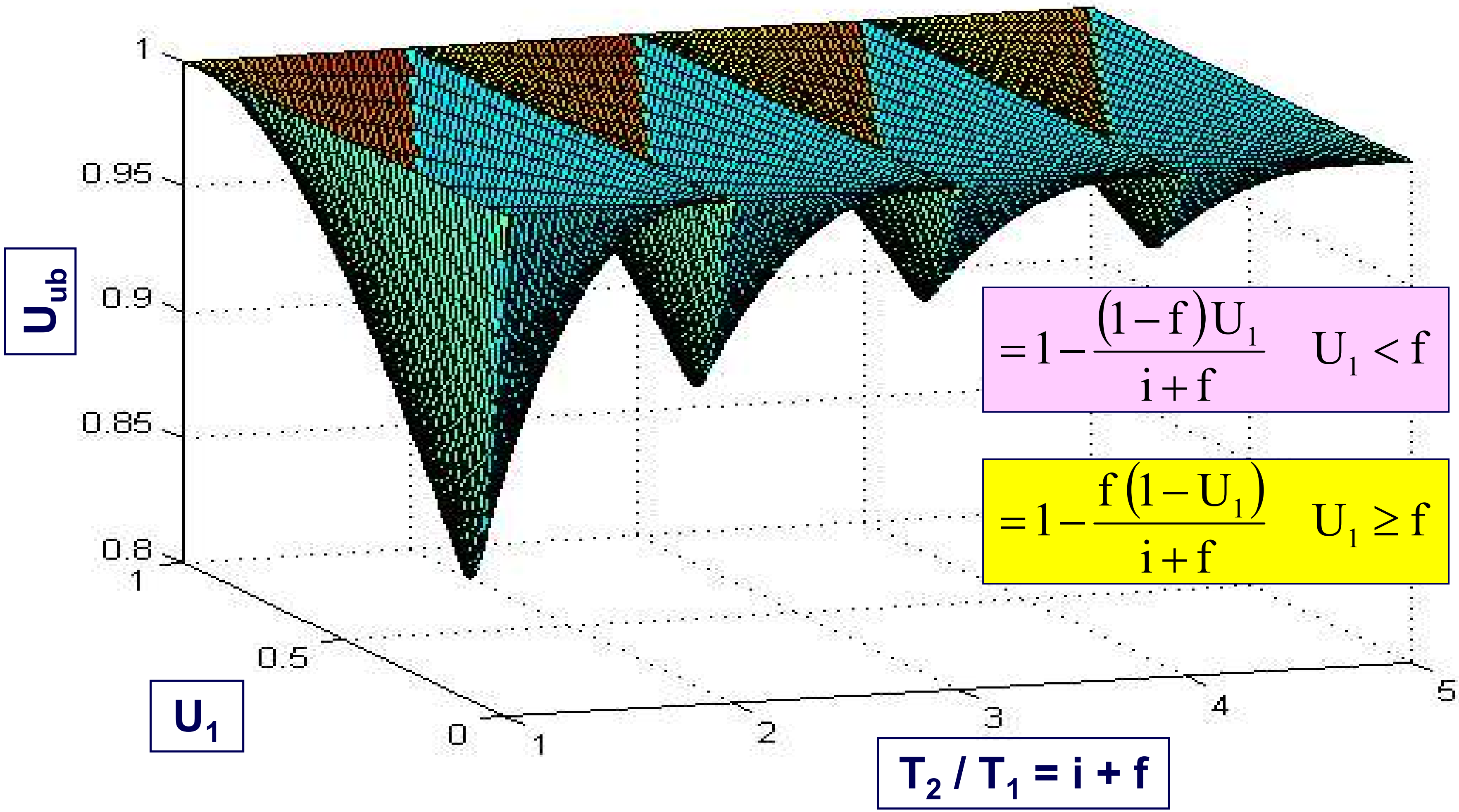


$$C_2 \le i\left(T_1 - C_1\right) = i\left(1 - U_1\right)T_1 = C_{2ub}$$

$$U_2 = \frac{C_2}{T_2} \le \frac{i\left(1 - U_1\right)}{i + f} = U_{2ub}$$

… CALCOLO DI U_{ub} (N=2) …

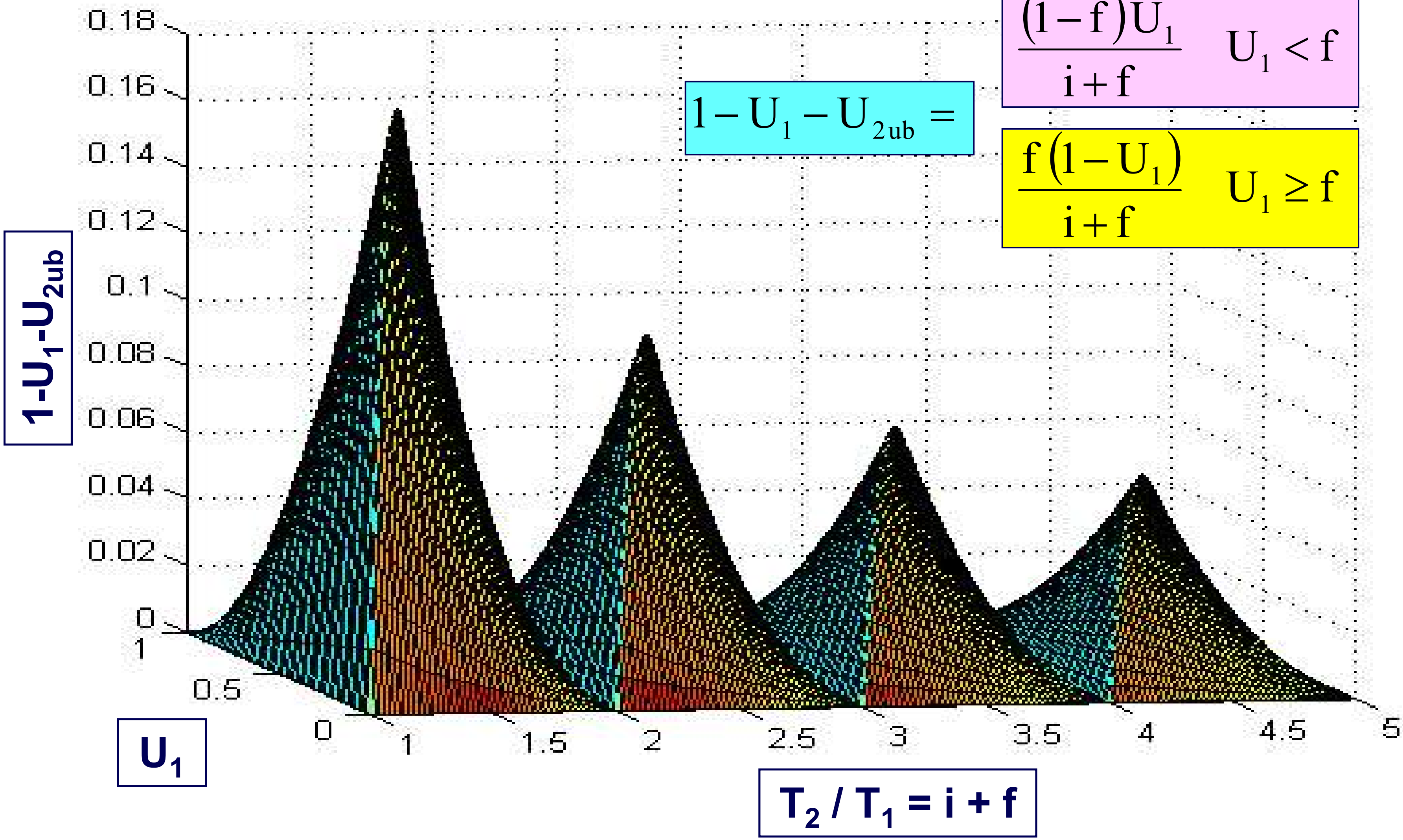
$$U_{ub}(U_1, i, f) = U_1 + U_{2ub}$$



$$= 1 - \frac{(1-f)U_1}{i+f} \qquad U_1 < f$$

$$= 1 - \frac{f(1-U_1)}{i+f} \qquad U_1 \geq f$$

… CALCOLO DI U_{ub} (N=2)



$$1 - U_1 - U_{2ub} = \begin{cases} \dfrac{(1-f)U_1}{i+f} & U_1 < f \\ \dfrac{f(1-U_1)}{i+f} & U_1 \geq f \end{cases}$$

CALCOLO DI U_{lub} (N=2) …

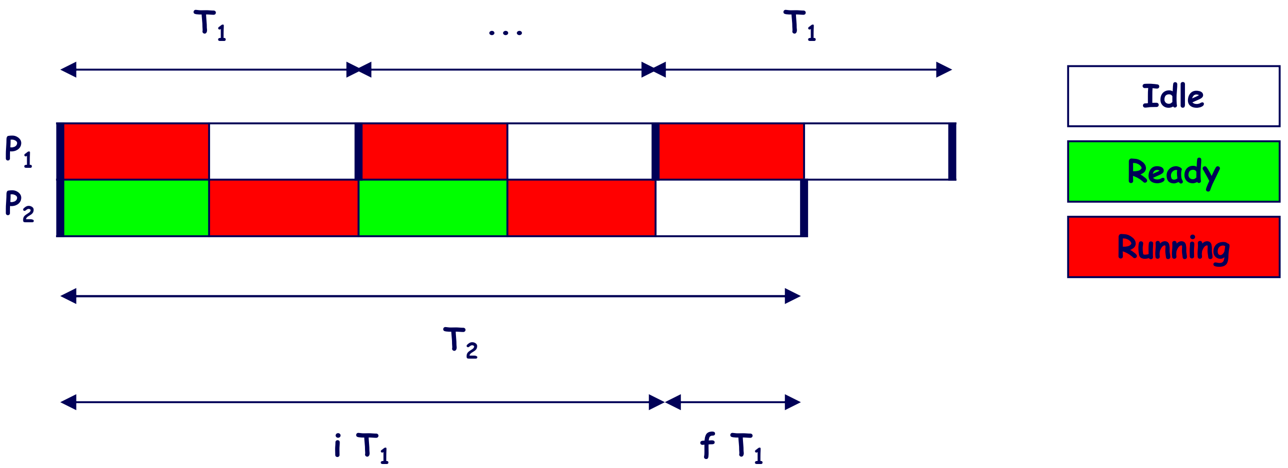
$$U_{ub}(U_1, i, f) = \begin{cases} 1 - \dfrac{(1-f)U_1}{i+f} & U_1 < f \\[2ex] 1 - \dfrac{f(1-U_1)}{i+f} & U_1 \geq f \end{cases}$$

$\uparrow U_1$ $\quad U_{ub}(U_1, i, f)\downarrow$

$\downarrow U_1$ $\quad U_{ub}(U_1, i, f)\downarrow$

$U_1 = f$

$$U_{lub}(i, f) = 1 - \frac{f(1-f)}{i+f}$$

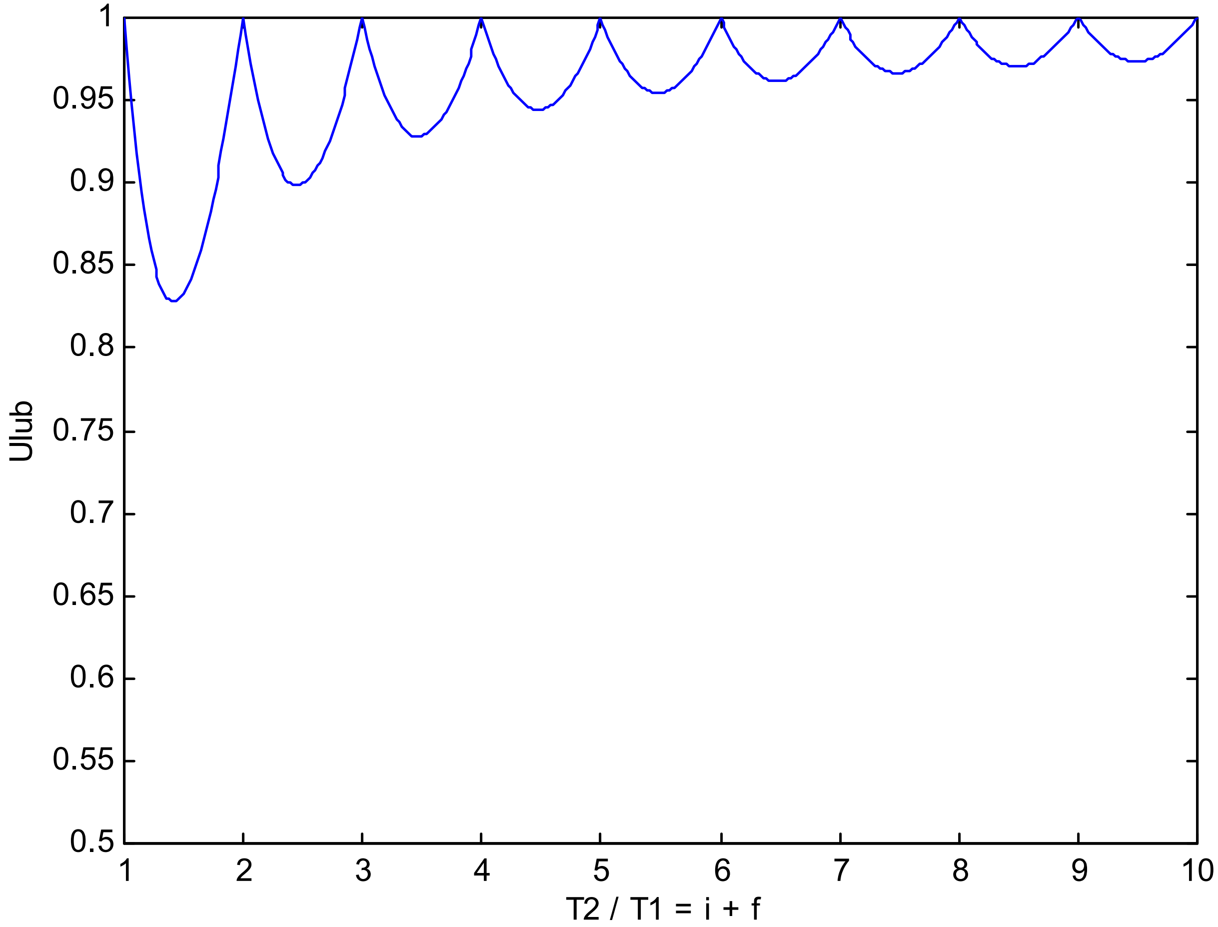


… CALCOLO DI U_{lub} (N=2) …

$$U_{lub}(i, f) = 1 - \frac{f(1-f)}{i+f}$$

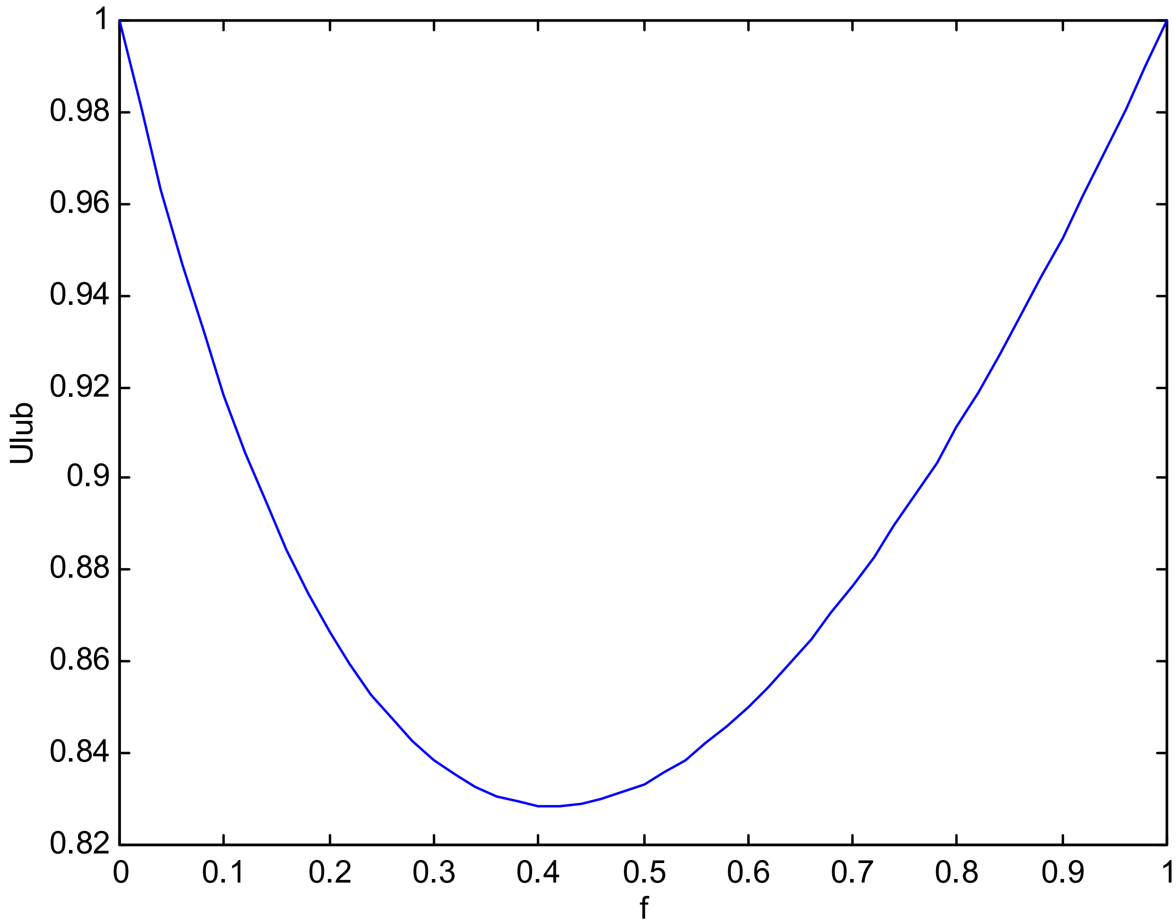
$\downarrow i$ $\quad$ $U_{lub}(i, f)\downarrow$

$i = 1$ $\quad$ $$U_{lub}(f) = \frac{1+f^2}{1+f}$$



… CALCOLO DI U_{lub} (N=2) …

$$U_{lub}(f) = \frac{1+f^2}{1+f}$$



$$\frac{dU_{lub}(f)}{df} = \frac{f^2 + 2f - 1}{(1+f)^2} = 0$$

$$f = \sqrt{2} - 1$$

$$U_{lub} = 2\left(\sqrt{2} - 1\right)$$

… CALCOLO DI U_{lub} (N=2)

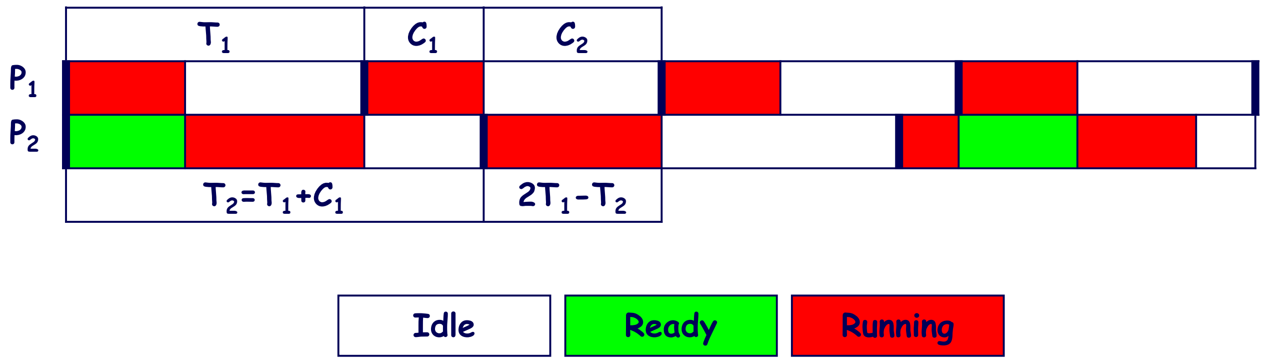
$i = 1$ $\quad$ $T_1 < T_2 < 2\,T_1$

$U_1 = f$ $\quad$ $T_2 = T_1 + C_1$ $\quad$ $C_1 = T_2 - T_1$ $\quad$ $C_2 = T_1 - C_1 = 2\,T_1 - T_2$

$f = \sqrt{2} - 1$ $\quad$ $T_2 = T_1\,2^{1/2}$ $\quad$ $C_1 = T_1\left(2^{1/2} - 1\right)$ $\quad$ $C_2 = T_2\left(2^{1/2} - 1\right) = C_1\,2^{1/2}$

$$U_1 = U_2 = 2^{1/2} - 1$$

i 2 processi concorrono uniformemente
al fattore di utilizzazione del processore



CALCOLO DI U_{lub} (N>2) …

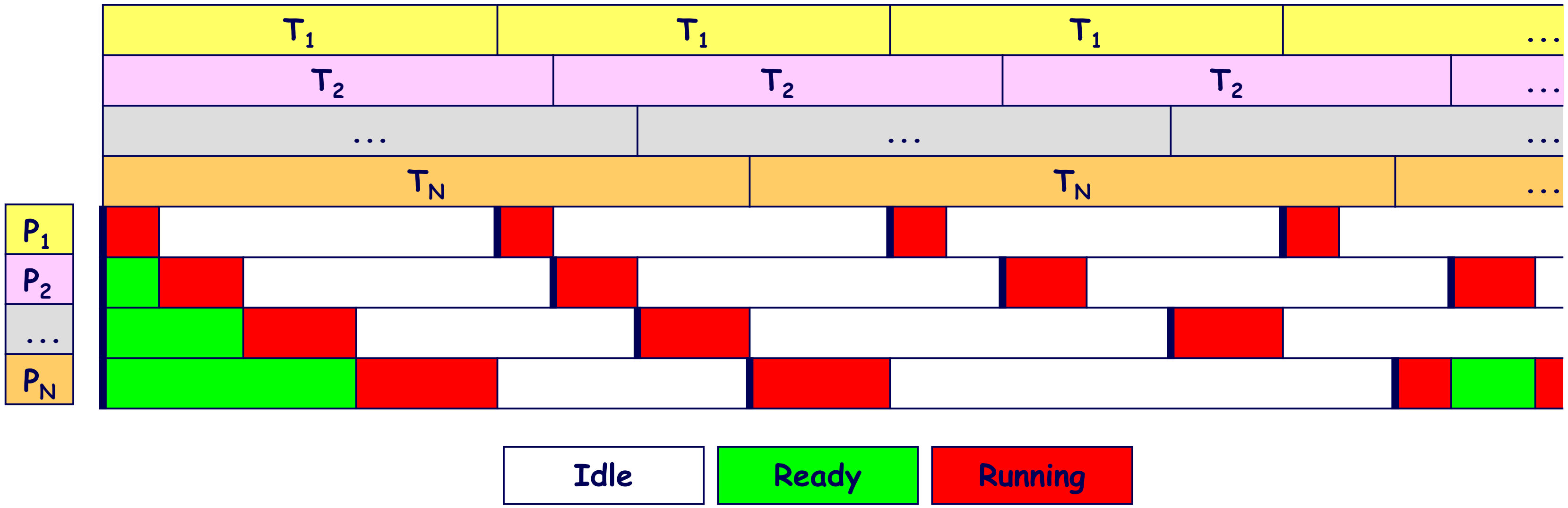
$$C_1 = T_2 - T_1$$

$$C_2 = T_3 - T_2$$

$$\vdots$$

$$C_{N-1} = T_N - T_{N-1}$$

$$C_N \leq T_1 - \sum_{j=1}^{N-1} C_j = 2\,T_1 - T_N = C_{Nub}$$



… CALCOLO DI U_{lub} (N>2) …

$$C_1 = T_2 - T_1 \qquad U_1 = \frac{T_2}{T_1} - 1$$

$$C_2 = T_3 - T_2 \qquad U_2 = \frac{T_3}{T_2} - 1$$

$$\vdots \qquad\qquad \vdots$$

$$C_{N-1} = T_N - T_{N-1} \qquad U_{N-1} = \frac{T_N}{T_{N-1}} - 1$$

$$C_{Nub} = 2T_1 - T_N \qquad U_{Nub} = \frac{2T_1}{T_N} - 1$$

$$\boxed{U_j = \frac{T_{j+1}}{T_j} - 1 = R_j - 1 \quad \forall j \neq N}$$

$$\boxed{U_{Nub} = \frac{2}{R_{N-1}R_{N-2}\cdots R_2 R_1} - 1}$$

$$\boxed{U_{ub} = \sum_{j=1}^{N-1} U_j + U_{Nub} = \sum_{j=1}^{N-1} R_j + \frac{K}{\prod_{j=1}^{N-1} R_j} - N}$$

$$\boxed{K = 2}$$

… CALCOLO DI U_{lub} (N>2) …

$$U_{ub} = \sum_{j=1}^{N-1} R_j + \frac{K}{\prod_{j=1}^{N-1} R_j} - N$$

$$\frac{\partial U_{ub}}{\partial R_j} = 1 - \frac{K}{\prod_{i \neq j} R_i \, R_j^2} = 1 - \frac{K}{R_j \prod_{i=1}^{N-1} R_i} = 0 \qquad j = 1, 2, \cdots, N-1$$

$$R_1 = R_2 = \cdots = R_{N-1} = K^{1/N}$$

$$U_{lub} = (N-1) K^{1/N} + \frac{K}{K^{\frac{N-1}{N}}} - N = N \left(K^{1/N} - 1 \right)$$

… CALCOLO DI U_{lub} (N>2)

$$\boxed{U_{lub} = N\left(2^{1/N} - 1\right)}$$

$$U_j = 2^{1/N} - 1 \quad j = 1, 2, \cdots, N$$

i processi concorrono uniformemente al fattore di utilizzazione del processore

$$T_j = T_{j-1}\, 2^{1/N} = T_1\, 2^{(j-1)/N} \quad j = 2, \ldots, N$$

$$C_1 = T_1\left(2^{1/N} - 1\right) \qquad C_j = C_{j-1}\, 2^{1/N} = C_1\, 2^{(j-1)/N} \quad j = 2, \ldots, N$$

UN COROLLARIO DEL TEOREMA DI LIU-LAYLAND …

Esplicitando l' "utilizzazione schedulabile dell'algoritmo RMPO"
in funzione dei singoli fattori di utilizzazione dei processi
(non semplicemente del numero di processi, come nel teorema),
si perviene ad un test di schedulabilità meno conservativo.

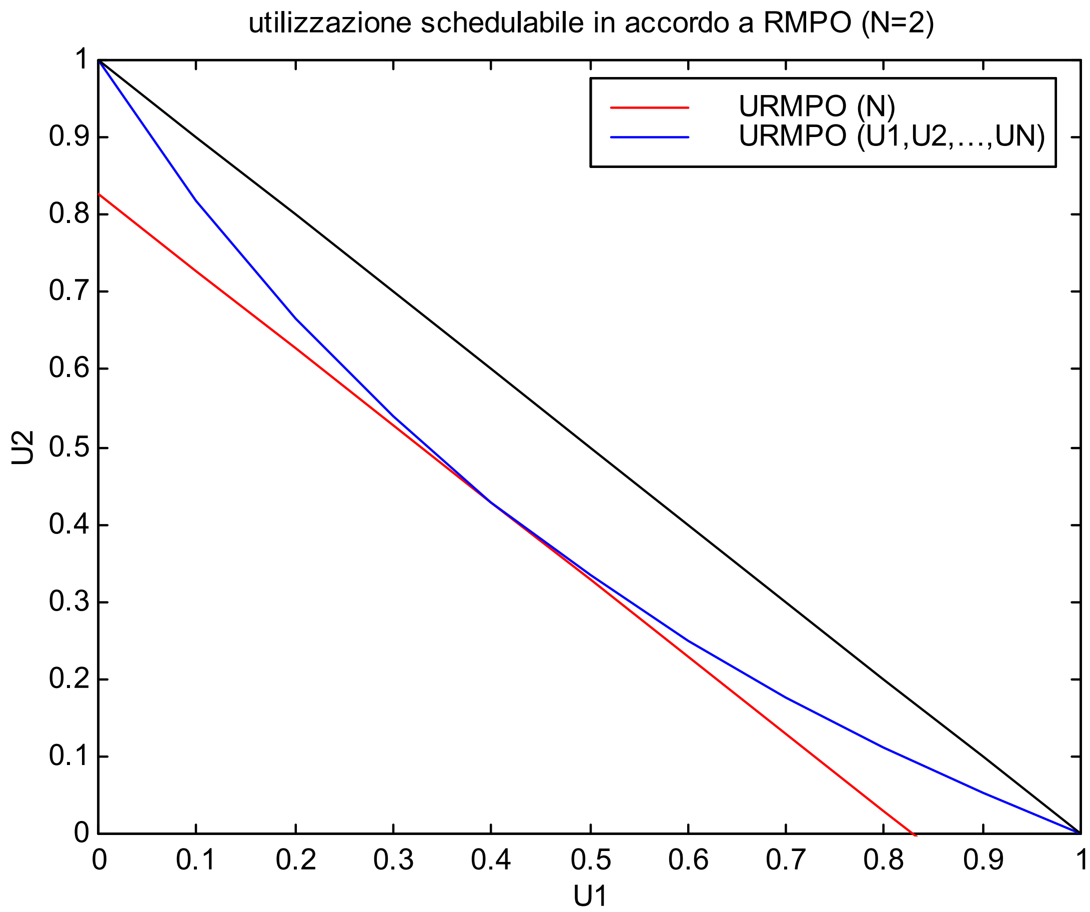
$$
\begin{aligned}
C_1 &= T_2 - T_1 \\
C_2 &= T_3 - T_2 \\
&\cdots \\
C_{N-1} &= T_N - T_{N-1}
\end{aligned}
\quad \Longrightarrow \quad T_{j+1} = T_j\left(1 + U_j\right) \qquad j = 1,2,\ldots,N-1
$$

$$
C_N \leq 2T_1 - T_N \quad \Longrightarrow \quad \frac{T_N}{T_1}\left(1 + U_N\right) \leq 2
$$

$$
\boxed{U_{RMPO}\left(U_1, U_2, \cdots, U_N\right) = \prod_{j=1}^{N}\left(1 + U_j\right) \leq 2}
$$

… UN COROLLARIO DEL TEOREMA DI LIU-LAYLAND …

se $U_1 = U_2 = \cdots = U_N \Rightarrow U_{RMPO}(U_1, U_2, \cdots, U_N) = U_{RMPO}(N)$

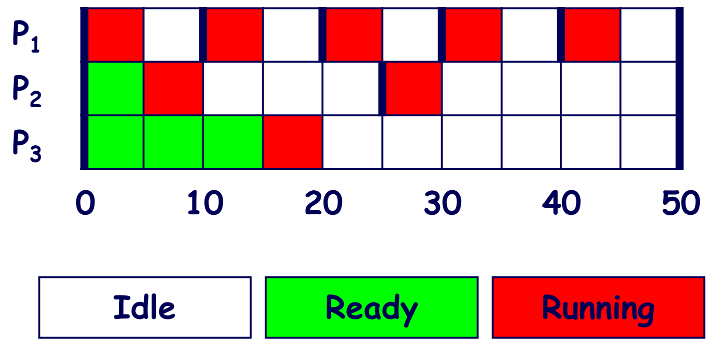


… UN COROLLARIO DEL TEOREMA DI LIU-LAYLAND

	C	T	C/T
P_1	5	10	0.5
P_2	5	25	0.2
P_3	5	50	0.1

$$\sum_{j=1}^{3} U_j = 0.8 > U_{RMPO}(N=3) = 0.78$$

$$U_{RMPO}(U_1 = 0.5, U_2 = 0.2, U_3 = 0.1) = \prod_{j=1}^{3}(1+U_j) = 1.98 \le 2$$



ALTRI TEST DI SCHEDULABILITÀ DIPENDENTI DAI PARAMETRI DEI PROCESSI: IL TEST DI KUO-MOK (91) …

Un insieme S di N processi P_1, P_2, …, P_N contraddistinto da un fattore di utilizzazione del processore U è schedulabile con RMPO se (condizione solo sufficiente) $U \leq U_{RMPO}(K)$, essendo K il numero di sottoinsiemi disgiunti di processi semplicemente periodici in S. La schedulabilità dell'insieme S è garantita se è schedulabile l'insieme S' di K < N processi P_1', P_2', …, P_K' ad esso equivalente. Al processo P_j' (j = 1, 2, …, K) in S', corrispondente al sottoinsieme di processi semplicemente periodici P_x, P_y, …, P_z in S, compete un fattore di utilizzazione del processore $U_j' = U_x + U_y + \ldots + U_z$, un periodo $T_j' = \min(T_x, T_y, \ldots, T_z)$ e un tempo di esecuzione $C_j' = U_j'\, T_j'$.

S	C	T	C/T
P_1	5	10	0.5
P_2	5	25	0.2
P_3	5	50	0.1

$P_1' \equiv P_1$

$P_2' \equiv \{P_2, P_3\}$

S'	C'	T'	C'/T'
P_1'	5	10	0.5
P_2'	7.5	25	0.3

$$\sum_{j=1}^{3} U_j = 0.8 > U_{RMPO}(N=3) = 0.78$$

$$\sum_{j=1}^{2} U_j' = 0.8 \leq U_{RMPO}(N=2) = 0.828$$

… IL TEST DI KUO-MOK …

S	C	T	C/T
P_1	4	10	0.40
P_2	4	20	0.20
P_3	8	40	0.20
P_4	3.6	45	0.08
P_5	1.8	90	0.02

$P_1' \equiv \{P_1, P_2, P_3\}$
$P_2' \equiv \{P_4, P_5\}$

S'	C'	T'	C'/T'
P_1'	8	10	0.8
P_2'	4.5	45	0.1

$$\sum_{j=1}^{5} U_j = 0.9 > U_{RMPO}(N=5) = 0.743$$

$$\sum_{j=1}^{2} U_j' = 0.9 > U_{RMPO}(N=2) = 0.828$$

corollario

$$U_{RMPO}(U_1 = 0.4, \cdots, U_5 = 0.02) = \prod_{j=1}^{5}(1+U_j) = 2.221 > 2$$

$$U_{RMPO}(U_1' = 0.8, U_2' = 0.1) = \prod_{j=1}^{2}(1+U_j') = 1.98 \le 2$$

… IL TEST DI KUO-MOK

Nei casi in cui il partizionamento non sia univoco, conviene optare per quello che induce una distribuzione del fattore di utilizzazione del processore quanto più disuniforme possibile.

S	C	T	C/T
P_1	5.5	10	0.55
P_2	5	25	0.2
P_3	5	50	0.1

S'	C'	T'	C'/T'
P_1'	5.5	10	0.55
P_2'	7.5	25	0.3

$P_1' \equiv P_1$

$P_2' \equiv \{P_2, P_3\}$

$P_1' \equiv \{P_1, P_3\}$

$P_2' \equiv P_2$

S'	C'	T'	C'/T'
P_1'	6.5	10	0.65
P_2'	5	25	0.2

$$\sum_{j=1}^{2} U_j' = 0.85 > U_{RMPO}(N=2) = 0.828$$

$$U_{RMPO}(U_1' = 0.55, U_2' = 0.3) =$$

$$= \prod_{j=1}^{2} (1 + U_j') = 2.015 > 2$$

corollario

$$U_{RMPO}(U_1' = 0.65, U_2' = 0.2) =$$

$$= \prod_{j=1}^{2} (1 + U_j') = 1.98 \leq 2$$

IL TEST DI BURCHARD et al. (96) …

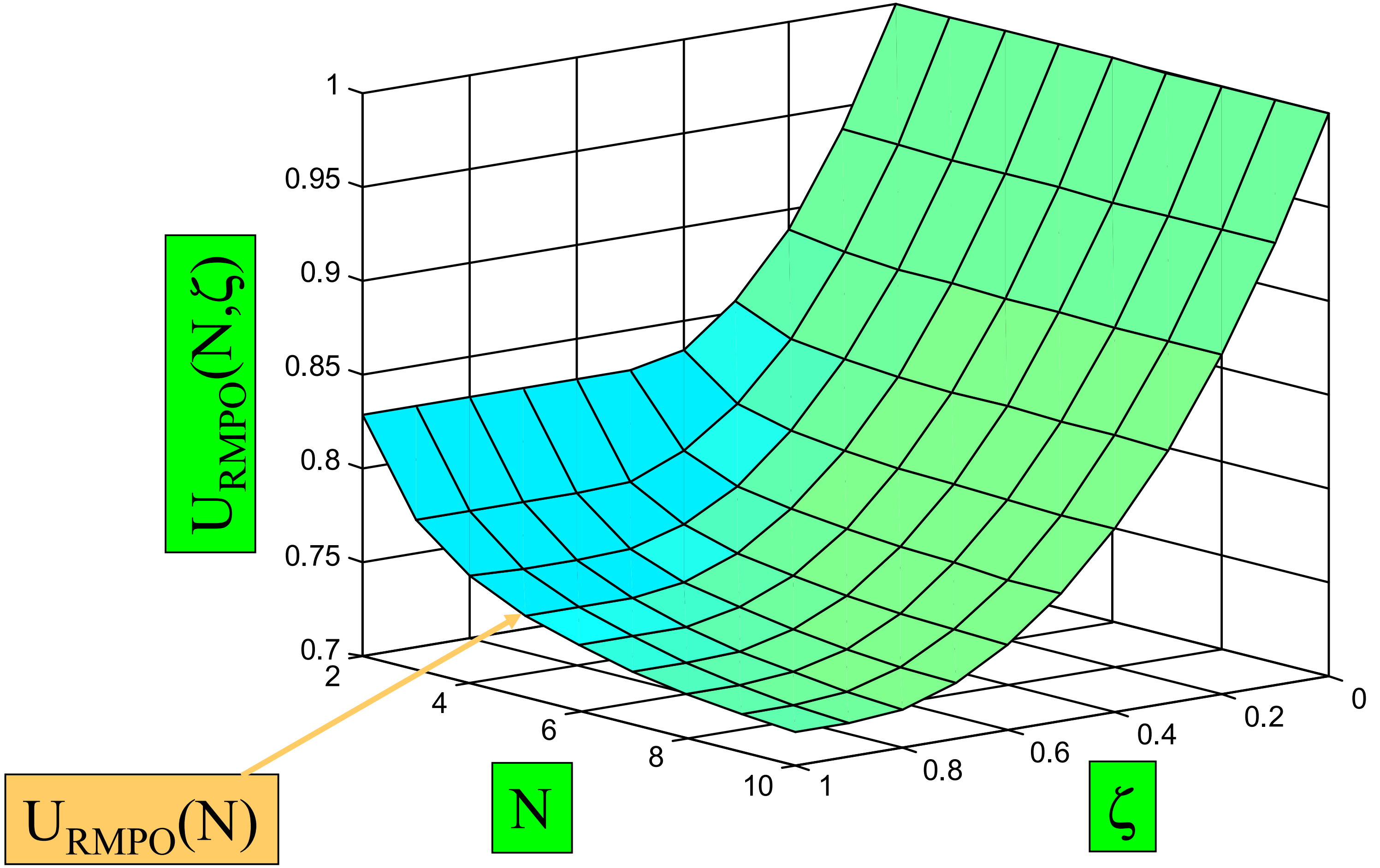
L'"utilizzazione schedulabile dell'algoritmo RMPO" è tanto maggiore (tendendo al valore limite 1 nel caso di processi semplicemente periodici) quanto meno i periodi dei processi si discostano dalla relazione armonica.

$$X_j = \log_2 T_j - \lfloor \log_2 T_j \rfloor \qquad \forall j$$

$$\zeta = \max_{1 \le j \le N} X_j - \min_{1 \le j \le N} X_j$$

$$U_{RMPO}(N,\zeta) = \begin{cases} (N-1)\left(2^{\zeta/(N-1)} - 1\right) + 2^{1-\zeta} - 1 & \zeta < 1 - 1/N \\ N\left(2^{1/N} - 1\right) & \zeta \ge 1 - 1/N \end{cases}$$

… IL TEST DI BURCHARD …



… IL TEST DI BURCHARD

A_5	C	T	C/T
P_1	5	10	0.50
P_2	8	19	0.42

$$\sum_{j=1}^{2} U_j = 0.92 > U_{RMPO}(N=2) = 0.828$$

$$\prod_{j=1}^{2} (1+U_j) = 2.13 = U_{RMPO}(U_1 = 0.5, U_2 = 0.42) > 2$$

$X_1=0.322$

$X_2=0.248$

$\zeta=0.074$

$$\sum_{j=1}^{2} U_j = 0.92 < U_{RMPO}(N=2, \zeta=0.074) = 0.952$$

IL TEST DI HAN (97) …

Un insieme S di N processi P_1, P_2, …, P_N è schedulabile con RMPO
se (condizione solo sufficiente) ad esso corrisponde un "insieme accelerato" S'
di N processi P_1', P_2', …, P_N' semplicemente periodici
e con fattore di utilizzazione del processore $U' = U_1' + U_2' + \ldots + U_N' \leq 1$.

insieme S		insieme accelerato S'
$P_1\ (T_1, C_1)$	⇨	$P_1'\ (T_1' \leq T_1,\ C_1' = C_1)$
$P_2\ (T_2, C_2)$	⇨	$P_2'\ (T_2' \leq T_2,\ C_2' = C_2)$
. . .	. . .	. . .
$P_N\ (T_N, C_N)$	⇨	$P_N'\ (T_N' \leq T_N,\ C_N' = C_N)$

... IL TEST DI HAN ...

S	C	T	C/T
P_1	5	10	.500
P_2	6	16	.375

$$\sum_{j=1}^{2} U_j = 0.875 > U_{RMPO}(N=2) = 0.828$$

$$\prod_{j=1}^{2}(1+U_j) = 2.0625 = U_{RMPO}(U_1 = 0.5, U_2 = 0.375) > 2$$

$X_1=0.322,\ X_2=0$

$\zeta=0.322$

$$\sum_{j=1}^{2} U_j = 0.875 > U_{RMPO}(N=2, \zeta=0.322) = 0.85$$

insieme accelerato

S'	C'	T'	C'/T'
P_1'	5	8	.625
P_2'	6	16	.375

$U_1' + U_2' \leq 1$

… IL TEST DI HAN …

/* Input: $\mathcal{T} = \{\tau_i = (C_i, T_i) \mid 1 \leq i \leq n\}$, where $\mathcal{T}$ is a periodic task set and $T_i \leq T_j, \forall i < j$. */
/* Output: $\mathcal{T}' = \{\tau'_i = (C_i, T'_i) \mid 1 \leq i \leq n\}$, where $T'_i \leq T_i, \forall i$ and $T'_i \mid T'_j, \forall i < j$ */

$min\_f := -1$; $min\_utilization := \infty$;
for $f := 1$ **to** n **do** {
  $Z_f := T_f$;
  for $i := f+1$ **to** n **do** $Z_i := Z_{i-1} \cdot \lfloor T_i / Z_{i-1} \rfloor$;
  for $i := f-1$ **downto** 1 **do** $Z_i := \frac{Z_{i+1}}{\lceil Z_{i+1}/T_i \rceil}$;
  $utilization := \sum_{i=1}^{n} C_i / Z_i$;
  if $utilization < min\_utilization$ **then**
    $min\_utilization := utilization$;
    $min\_f := f$;
    for $i := 1$ **to** n **do** $T'_i := Z_i$;
  endif
}

… IL TEST DI HAN …

Complessità computazionale dell'algoritmo: $O(N^2)$

iterazione

1

P_1	T_1	$T_1'=T_1$
...	...	...
P_k	T_k	T_k'
...	...	...
P_N	T_N	T_N'

⇓ $T_i' = T_{i-1}' * \lfloor T_i / T_{i-1}' \rfloor$ $i = 2, \ldots, N$

$2 \leq k \leq N-1$

P_1	T_1	T_1'
...	...	...
P_k	T_k	$T_k'=T_k$
...	...	...
P_N	T_N	T_N'

⇑ $T_i' = T_{i+1}' / \lceil T_{i+1}' / T_i \rceil$ $i = k-1, \ldots, 1$

⇓ $T_i' = T_{i-1}' * \lfloor T_i / T_{i-1}' \rfloor$ $i = k+1, \ldots, N$

N

P_1	T_1	T_1'
...	...	...
P_k	T_k	T_k'
...	...	...
P_N	T_N	$T_N'=T_N$

⇑ $T_i' = T_{i+1}' / \lceil T_{i+1}' / T_i \rceil$ $i = N-1, \ldots, 1$

… IL TEST DI HAN

Limiti dell'algoritmo

S	C	T	C/T
P_1	8	20	0.4
P_2	6	60	0.1
P_3	27	90	0.3
P_4	18	180	0.1

U = 0.9

S'	C	T'	C/T'
P_1	8	20	0.40
P_2	6	40	0.15
P_3	27	80	0.34
P_4	18	160	0.11

U = 1.0

I iterazione

U = 1.05

P_1	$T_1=20$	$T_1'=20$	
P_2	$T_2=60$	$T_2'=60$	$= T_1' * \lfloor T_2 / T_1' \rfloor = 20 * \lfloor 60 / 20 \rfloor$
P_3	$T_3=90$	$T_3'=60$	$= T_2' * \lfloor T_3 / T_2' \rfloor = 60 * \lfloor 90 / 60 \rfloor$
P_4	$T_4=180$	$T_4'=180$	$= T_3' * \lfloor T_4 / T_3' \rfloor = 60 * \lfloor 180 / 60 \rfloor$

II iterazione

U = 1.05

P_1	$T_1=20$	$T_1'=20$	$= T_2' / \lceil T_2' / T_1 \rceil = 60 / \lceil 60 / 20 \rceil$
P_2	$T_2=60$	$T_2'=60$	
P_3	$T_3=90$	$T_3'=60$	$= T_2' * \lfloor T_3 / T_2' \rfloor = 60 * \lfloor 90 / 60 \rfloor$
P_4	$T_4=180$	$T_4'=180$	$= T_3' * \lfloor T_4 / T_3' \rfloor = 60 * \lfloor 180 / 60 \rfloor$

III iterazione

U = 1.067

P_1	$T_1=20$	$T_1'=15$	$= T_2' / \lceil T_2' / T_1 \rceil = 45 / \lceil 45 / 20 \rceil$
P_2	$T_2=60$	$T_2'=45$	$= T_3' / \lceil T_3' / T_2 \rceil = 90 / \lceil 90 / 60 \rceil$
P_3	$T_3=90$	$T_3'=90$	
P_4	$T_4=180$	$T_4'=180$	$= T_3' * \lfloor T_4 / T_3' \rfloor = 90 * \lfloor 180 / 90 \rfloor$

IV iterazione

U = 1.067

P_1	$T_1=20$	$T_1'=15$	$= T_2' / \lceil T_2' / T_1 \rceil = 45 / \lceil 45 / 20 \rceil$
P_2	$T_2=60$	$T_2'=45$	$= T_3' / \lceil T_3' / T_2 \rceil = 90 / \lceil 90 / 60 \rceil$
P_3	$T_3=90$	$T_3'=90$	$= T_4' / \lceil T_4' / T_3 \rceil = 180 / \lceil 180 / 90 \rceil$
P_4	$T_4=180$	$T_4'=180$	

O PIÙ TEST CONGIUNTAMENTE …

S	C	T	C/T
P_1	2	10	.20
P_2	3	15	.20
P_3	5	20	.25
P_4	9	45	.20

$$\sum_{j=1}^{4} U_j = 0.85 > U_{RMPO}(N=4) = 0.756$$

$$\prod_{j=1}^{4}(1+U_j) = 2.16 > 2$$

Kuo-Mok

$P_1' \equiv \{P_1, P_3\}$

$P_2' \equiv \{P_2, P_4\}$

S'	C'	T'	C'/T'
P_1'	4.5	10	0.45
P_2'	6	15	0.40

$$\sum_{j=1}^{2} U_j' = 0.85 > U_{RMPO}(N=2) = 0.828$$

$$\prod_{j=1}^{2}(1+U_j') = 2.03 > 2$$

Han

S''	C''	T''	C''/T''
P_1''	4.5	7.5	0.6
P_2''	6	15	0.4

$$\sum_{j=1}^{2} U_j'' = 1$$

… O PIÙ TEST CONGIUNTAMENTE …

U = 0.903

S	C	T	C/T
P_1	3	12	.250
P_2	3	18	.167
P_3	5	24	.208
P_4	10	36	.278

$N\left(2^{1/N}-1\right)$ 0.756

$\prod_{j=1}^{N}\left(1+U_j\right)$ 2.16

Kuo-Mok

$P_1' \equiv \{P_1, P_3\}$
$P_2' \equiv \{P_2, P_4\}$

S'	C'	T'	C'/T'
P_1'	5.5	12	.458
P_2'	8	18	.445

$N\left(2^{1/N}-1\right)$ 0.828

$\prod_{j=1}^{N}\left(1+U_j\right)$ 2.11

Han

S''	C''	T''	C''/T''
P_1''	5.5	12	.458
P_2''	8	12	.667

U = 1.125

S''	C''	T''	C''/T''
P_1''	5.5	9	.611
P_2''	8	18	.445

U = 1.056

… O PIÙ TEST CONGIUNTAMENTE

U = 0.903

S	C	T	C/T
P_1	3	12	.250
P_2	3	18	.167
P_3	5	24	.208
P_4	10	36	.278

$N\left(2^{1/N}-1\right)$ 0.756

$\prod_{j=1}^{N}\left(1+U_j\right)$ 2.16

Kuo-Mok

$P_1' \equiv \{P_1, P_4\}$
$P_2' \equiv P_2$
$P_3' \equiv P_3$

S'	C'	T'	C'/T'
P_1'	6.33	12	.528
P_2'	3	18	.167
P_3'	5	24	.208

$N\left(2^{1/N}-1\right)$ 0.780

$\prod_{j=1}^{N}\left(1+U_j\right)$ 2.15

Kuo-Mok

$P_1'' \equiv \{P_1', P_3'\}$
$P_2'' \equiv P_2'$

S''	C''	T''	C''/T''
P_1''	8.83	12	.736
P_2''	3	18	.167

$N\left(2^{1/N}-1\right)$ 0.828

$\prod_{j=1}^{N}\left(1+U_j\right)$ 2.02

Han

S'''	C'''	T'''	C'''/T'''
P_1'''	8.83	12	.736
P_2'''	3	12	.250

U = 0.986

ANALISI DI SCHEDULABILITÀ BASATA SUL CALCOLO DEI TEMPI DI RISPOSTA [AUDSLEY et al. (93)]

La schedulabilità è garantita se il tempo di risposta di ogni processo nelle condizioni più sfavorevoli (cioè a partire da un istante critico) non eccede la corrispondente deadline:

$$R_i = C_i + I_i(R_i) \leq T_i \quad i = 1, 2, \ldots, N$$

I_i rappresenta l'interferenza sul tempo di risposta R_i del processo P_i derivante dall'esecuzione di processi di priorità superiore.

$$I_i(R_i) = \sum_{j \mid p_j > p_i} \left\lceil \frac{R_i}{T_j} \right\rceil C_j$$

R_i può essere calcolato nel seguente modo:

$$R_i^0 = C_i, \quad R_i^n = C_i + I_i\left(R_i^{n-1}\right) \quad n = 1, 2, \ldots$$

L'ALGORITMO DI AUDSLEY ...

A_5	C	T	C/T
P_1	5	10	0.50
P_2	8	19	0.42

$U(A_5) = 0.92$

il processo
a priorità massima
non è soggetto
a interferenza

processo P_1: $R_1^1 = R_1^0 = C_1 = 5 < T_1 = 10$

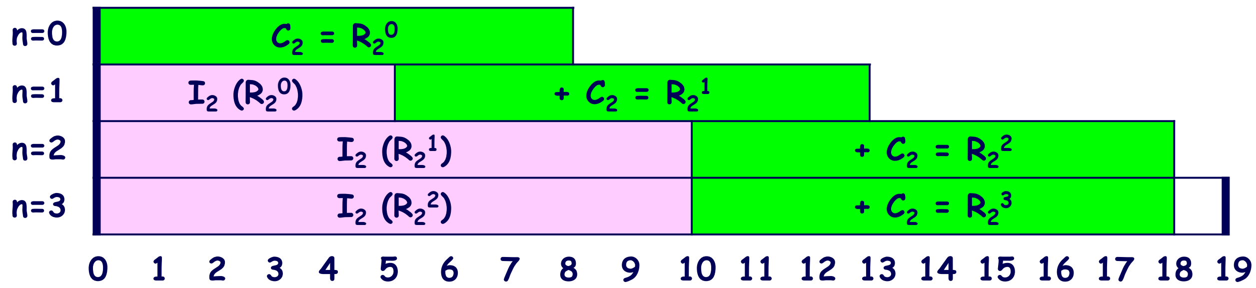
processo P_2: $R_2^0 = C_2 = 8$

$$R_2^1 = C_2 + \lceil R_2^0/T_1 \rceil * C_1 = 8 + \lceil 8/10 \rceil * 5 = 13$$

$$R_2^2 = C_2 + \lceil R_2^1/T_1 \rceil * C_1 = 8 + \lceil 13/10 \rceil * 5 = 18$$

$$R_2^3 = C_2 + \lceil R_2^2/T_1 \rceil * C_1 = 8 + \lceil 18/10 \rceil * 5 = 18$$

$$R_2^3 = R_2^2 = 18 < T_2 = 19$$



A_4	C	T	C/T
P_1	5	10	0.5
P_2	6	15	0.4

$U(A_4) = 0.9$

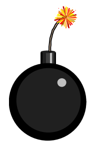
processo P_1: $R_1^1 = R_1^0 = C_1 = 5 < T_1 = 10$

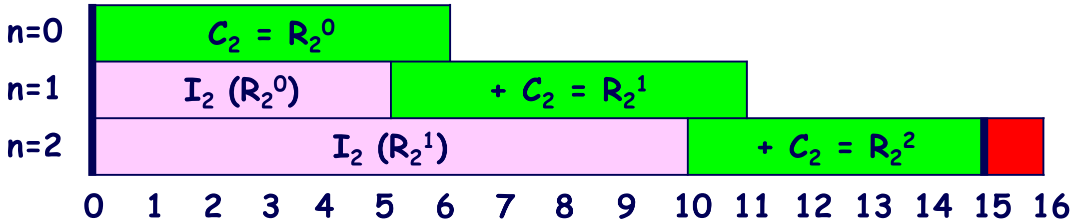
processo P_2: $R_2^0 = C_2 = 6$

$R_2^1 = C_2 + \lceil R_2^0/T_1 \rceil * C_1 = 6 + \lceil 6/10 \rceil * 5 = 11$

$R_2^2 = C_2 + \lceil R_2^1/T_1 \rceil * C_1 = 6 + \lceil 11/10 \rceil * 5 = 16$

$R_2^2 = 16 > T_2 = 15$





… L'ALGORITMO DI AUDSLEY

	C	T	C/T
P_1	6	15	0.400
P_2	5	25	0.200
P_3	7	32	0.219
P_4	3	45	0.067

U = 0.885

	P_1	P_2	P_3	P_4
$R_i^0 = C_i$	6	5	7	3
$R_i^1 = C_i + I_i\ (R_i^0)$	$6 + 0 = 6 \leq 15$	$5 + 6 = 11$	$7 + 6 + 5 = 18$	$3 + 6 + 5 + 7 = 21$
$R_i^2 = C_i + I_i\ (R_i^1)$		$5 + 6 = 11 \leq 25$	$7 + 12 + 5 = 24$	$3 + 12 + 5 + 7 = 27$
$R_i^3 = C_i + I_i\ (R_i^2)$			$7 + 12 + 5 = 24 \leq 32$	$3 + 12 + 10 + 7 = 32$
$R_i^4 = C_i + I_i\ (R_i^3)$				$3 + 18 + 10 + 7 = 38$
$R_i^5 = C_i + I_i\ (R_i^4)$				$3 + 18 + 10 + 14 = 45$
$R_i^6 = C_i + I_i\ (R_i^5)$				$3 + 18 + 10 + 14 = 45 \leq 45$

UN TEST PIÙ EFFICIENTE, MA …

Ai fini della schedulabilità, è sufficiente verificare se per ogni processo, nelle condizioni più sfavorevoli (cioè a partire da un istante critico), risulta:

$$C_i + I_i(T_i) = C_i + \sum_{j \mid p_j > p_i} \left\lceil \frac{T_i}{T_j} \right\rceil C_j \leq T_i \qquad i = 1, 2, \cdots, N$$

A_5	C	T	C/T
P_1	5	10	0.50
P_2	8	19	0.42

$U(A_5) = 0.92$

processo P_1: $C_1 + I_1(T_1) = 5 < T_1 = 10$ ☺

processo P_2: $C_2 + \lceil T_2/T_1 \rceil * C_1 = 8 + \lceil 19/10 \rceil * 5 = 18 < T_2 = 19$ ☺

… MENO EFFICACE

Il test identifica un insieme di condizioni solo sufficienti:

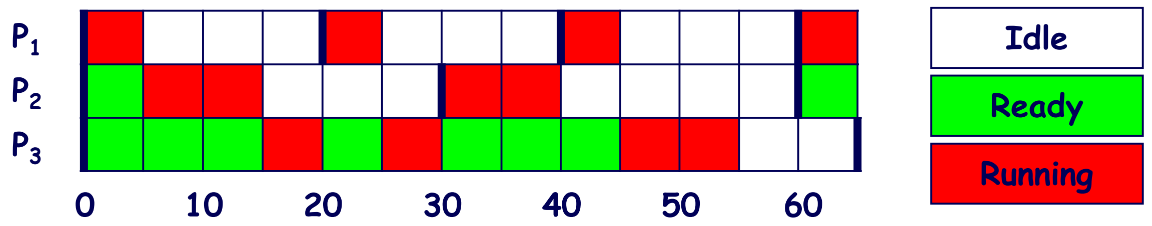
	C	T	C/T
P_1	5	20	.250
P_2	10	30	.333
P_3	20	65	.308

$U = 0.891$

P_1: $C_1 + I_1(T_1) = 5 < T_1 = 20$

P_2: $C_2 + \lceil T_2/T_1 \rceil * C_1 = 10 + \lceil 30/20 \rceil * 5 = 20 < T_2 = 30$

P_3: $C_3 + \lceil T_3/T_1 \rceil * C_1 + \lceil T_3/T_2 \rceil * C_2 = 20 + \lceil 65/20 \rceil * 5 + \lceil 65/30 \rceil * 10 = 70 > T_3 = 65$



P_3: $R_3^0 = C_3 = 20$

$R_3^1 = C_3 + \lceil R_3^0/T_1 \rceil * C_1 + \lceil R_3^0/T_2 \rceil * C_2 = 20 + \lceil 20/20 \rceil * 5 + \lceil 20/30 \rceil * 10 = 35$

$R_3^2 = C_3 + \lceil R_3^1/T_1 \rceil * C_1 + \lceil R_3^1/T_2 \rceil * C_2 = 20 + \lceil 35/20 \rceil * 5 + \lceil 35/30 \rceil * 10 = 50$

$R_3^3 = C_3 + \lceil R_3^2/T_1 \rceil * C_1 + \lceil R_3^2/T_2 \rceil * C_2 = 20 + \lceil 50/20 \rceil * 5 + \lceil 50/30 \rceil * 10 = 55$

$R_3^4 = C_3 + \lceil R_3^3/T_1 \rceil * C_1 + \lceil R_3^3/T_2 \rceil * C_2 = 20 + \lceil 55/20 \rceil * 5 + \lceil 55/30 \rceil * 10 = 55$

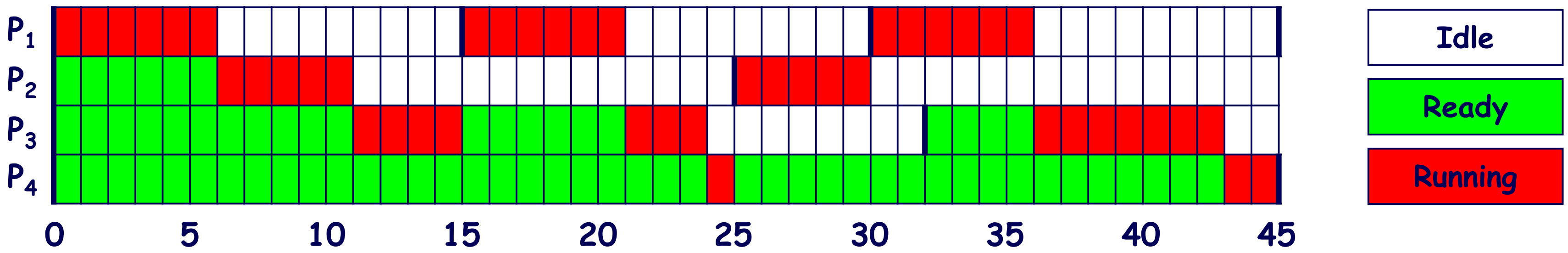
$R_3^4 = R_3^3 = 55 < T_3 = 65$

UN TEST PIÙ EFFICIENTE MA MENO EFFICACE

	C	T	C/T
P_1	6	15	0.400
P_2	5	25	0.200
P_3	7	32	0.219
P_4	3	45	0.067

$U = 0.885$

	P_1	P_2	P_3	P_4
$C_i + I_i(T_i) \le T_i$	$6 + 0 = 6$ $6 \le 15$ ok	$5 + 12 = 17$ $17 \le 25$ ok	$7 + 18 + 10 = 35$ $35 \le 32$ ko	$3 + 18 + 10 + 14 = 45$ $45 \le 45$ ok
$R_i^0 = C_i$			7	
$R_i^1 = C_i + I_i(R_i^0)$			$7 + 6 + 5 = 18$	
$R_i^2 = C_i + I_i(R_i^1)$			$7 + 12 + 5 = 24$	
$R_i^3 = C_i + I_i(R_i^2)$			$7 + 12 + 5 = 24$	



UN'ESTENSIONE DEL MODELLO PER LA SCHEDULAZIONE DI PROCESSI SIA PERIODICI CHE SPORADICI

I processi sporadici, pur essendo tipicamente caratterizzati da una bassa frequenza di esecuzione, impongono vincoli più stringenti per quanto concerne il tempo di completamento della loro esecuzione (processi urgenti):

$$\exists j \;\; D_j < T_j$$

Ai fini della schedulazione è sufficiente estendere il precedente modello, prevedendo che ogni processo P_j sia esplicitamente caratterizzato da tre parametri:
T_j (MIT nel caso di processi sporadici, periodo nel caso di processi periodici),
$D_j \leq T_j$ (< nel caso di processi sporadici, = nel caso di processi periodici)
e $C_j < D_j$.

L'ALGORITMO DEADLINE MONOTONIC PRIORITY ORDERING (DMPO) [LEUNG-WHITEHEAD (82)]

Ad ogni processo P_j ($\forall$ j) è staticamente associata una priorità inversamente proporzionale alla corrispondente deadline relativa:
$p_j \propto 1 / D_j$.

OTTIMALITÀ DELL'ALGORITMO

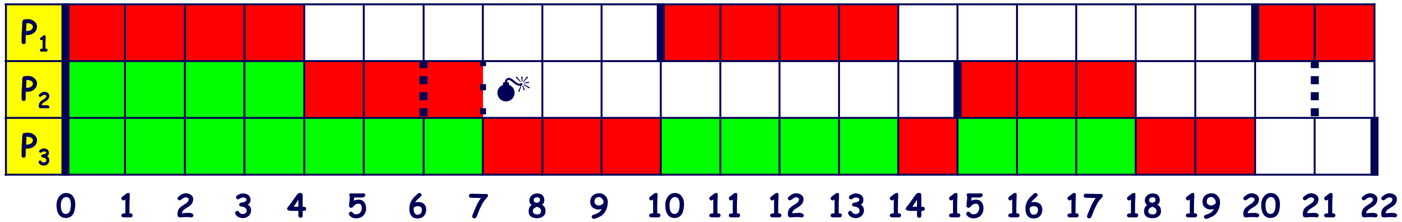
Se un insieme di processi periodici e/o sporadici è schedulabile con un qualche algoritmo che prevede un'attribuzione statica di priorità, allora tale insieme è schedulabile anche con DMPO.

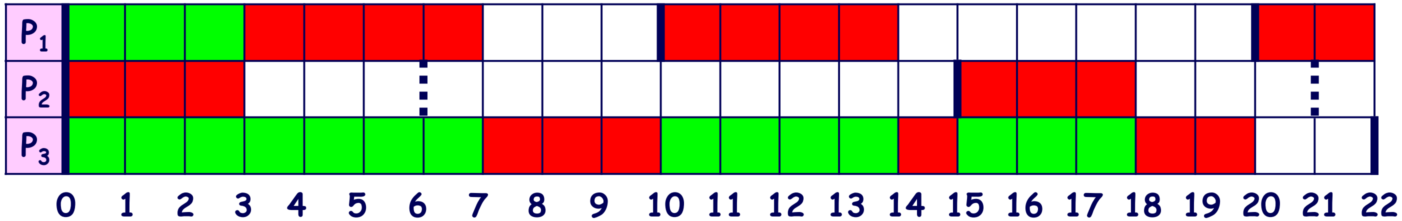
Se un insieme di processi periodici e/o sporadici non è schedulabile con DMPO, allora tale insieme non è schedulabile con alcun altro algoritmo che preveda un'attribuzione statica di priorità.

ANALISI DI SCHEDULABILITÀ ATTRAVERSO DIAGRAMMI TEMPORALI …

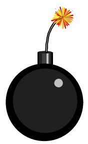
A_6	C	T	D	C/T	p(RMPO)	p(DMPO)
P_1	4	10	10	0.40	max	
P_2	3	15	6	0.20		max
P_3	6	22	22	0.27	min	min

$U(A_6) = 0.87$





Idle	Ready	Running



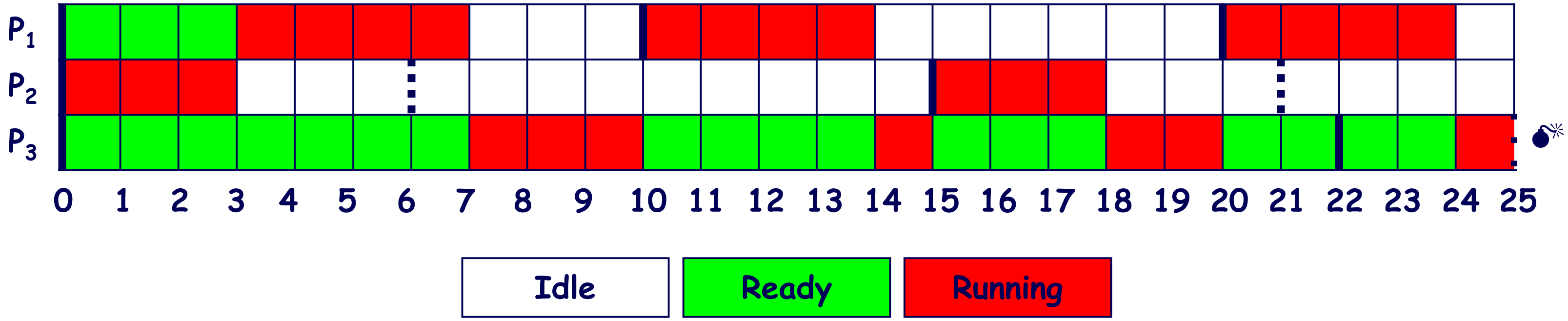
RMPO: missed deadline

DMPO: no missed deadline

… ANALISI DI SCHEDULABILITÀ ATTRAVERSO DIAGRAMMI TEMPORALI

A_7	C	T	D	C/T	p
P_1	4	10	10	0.40	
P_2	3	15	6	0.20	max
P_3	7	22	22	0.32	min

$U(A_7) = 0.92$



missed deadline (💣): insieme di processi non schedulabile

Per poter schedulare applicazioni con un elevato fattore di utilizzazione del processore (come A_7 o A_4), occorre utilizzare un algoritmo che attribuisca le priorità ai processi non staticamente, bensì dinamicamente.

ANALISI DI SCHEDULABILITÀ: CALCOLO DEI TEMPI DI RISPOSTA

Algoritmo di Audsley: condizione necessaria e sufficiente affinché un insieme di N processi periodici e sporadici sia schedulabile con l'algoritmo DMPO è che:

$$R_i = C_i + I_i(R_i) = C_i + \sum_{j \mid p_j > p_i} \left\lceil \frac{R_i}{T_j} \right\rceil C_j \le D_i \qquad i = 1, 2, \cdots, N$$

A_6	C	T	D	C/T
P_1	4	10	10	0.40
P_2	3	15	6	0.20
P_3	6	22	22	0.27

$U(A_6) = 0.87$

P_2: $R_2^1 = R_2^0 = C_2 = 3 < D_2 = 6$

P_1: $R_1^0 = C_1 = 4$

$R_1^1 = C_1 + \lceil R_1^0/T_2 \rceil * C_2 = 4 + \lceil 4/15 \rceil * 3 = 7$

$R_1^2 = C_1 + \lceil R_1^1/T_2 \rceil * C_2 = 4 + \lceil 7/15 \rceil * 3 = 7$

$R_1^2 = R_1^1 = 7 < D_1 = 10$

P_3: $R_3^0 = C_3 = 6$

$R_3^1 = C_3 + \lceil R_3^0/T_2 \rceil * C_2 + \lceil R_3^0/T_1 \rceil * C_1 = 6 + \lceil 6/15 \rceil * 3 + \lceil 6/10 \rceil * 4 = 13$

$R_3^2 = C_3 + \lceil R_3^1/T_2 \rceil * C_2 + \lceil R_3^1/T_1 \rceil * C_1 = 6 + \lceil 13/15 \rceil * 3 + \lceil 13/10 \rceil * 4 = 17$

$R_3^3 = C_3 + \lceil R_3^2/T_2 \rceil * C_2 + \lceil R_3^2/T_1 \rceil * C_1 = 6 + \lceil 17/15 \rceil * 3 + \lceil 17/10 \rceil * 4 = 20$

$R_3^4 = C_3 + \lceil R_3^3/T_2 \rceil * C_2 + \lceil R_3^3/T_1 \rceil * C_1 = 6 + \lceil 20/15 \rceil * 3 + \lceil 20/10 \rceil * 4 = 20$

$R_3^4 = R_3^3 = 20 < D_3 = 22$

TEST DI SCHEDULABILITÀ …

Condizione sufficiente (ma non necessaria) affinché un insieme di N processi periodici e sporadici sia schedulabile con l'algoritmo DMPO è che:

$$C_i + I_i(D_i) = C_i + \sum_{j \mid p_j > p_i} \left\lceil \frac{D_i}{T_j} \right\rceil C_j \le D_i \qquad i = 1, 2, \cdots, N$$

A_6	C	T	D	C/T
P_1	4	10	10	0.40
P_2	3	15	6	0.20
P_3	6	22	22	0.27

$U(A_6) = 0.87$

P_2: $C_2 + I_2(D_2) = 3 < D_2 = 6$

P_1: $C_1 + \lceil D_1/T_2 \rceil * C_2 = 4 + \lceil 10/15 \rceil * 3 = 7 < D_1 = 10$

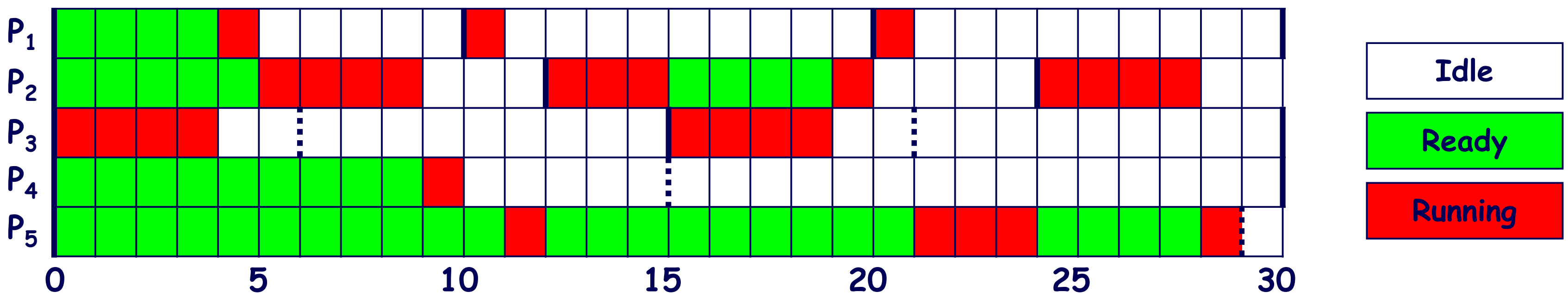
P_3: $C_3 + \lceil D_3/T_2 \rceil * C_2 + \lceil D_3/T_1 \rceil * C_1 = 6 + \lceil 22/15 \rceil * 3 + \lceil 22/10 \rceil * 4 = 24 > D_3 = 22$

… TEST DI SCHEDULABILITÀ …

	C	T	D	C/T
P_1	1	10	10	0.100
P_2	4	12	12	0.333
P_3	4	15	6	0.267
P_4	1	30	15	0.033
P_5	5	60	29	0.083

U = 0.817

	P_3	P_1	P_2	P_4	P_5
$C_i + I_i(D_i) \le D_i$	4+0 = 4 $4 \le 6$ ok	1+4 = 5 $5 \le 10$ ok	4+4+2 = 10 $10 \le 12$ ok	1+4+2+8 = 15 $15 \le 15$ ok	5+8+3+12+1 = 29 $29 \le 29$ ok
$R_i^0 = C_i$					
$R_i^1 = C_i + I_i(R_i^0)$					
$R_i^2 = C_i + I_i(R_i^1)$					
$R_i^3 = C_i + I_i(R_i^2)$					
$R_i^4 = C_i + I_i(R_i^3)$					
$R_i^5 = C_i + I_i(R_i^4)$					
$R_i^6 = C_i + I_i(R_i^5)$					



… TEST DI SCHEDULABILITÀ …

Test basato sulla "densità di utilizzazione" del processore

Condizione sufficiente (ma non necessaria) affinché un insieme di N processi periodici e sporadici sia schedulabile con l'algoritmo DMPO è che:

$$\Delta = \sum_{j=1}^{N} \frac{C_j}{D_j} \leq U_{RMPO}(N) = N\left(2^{1/N} - 1\right)$$

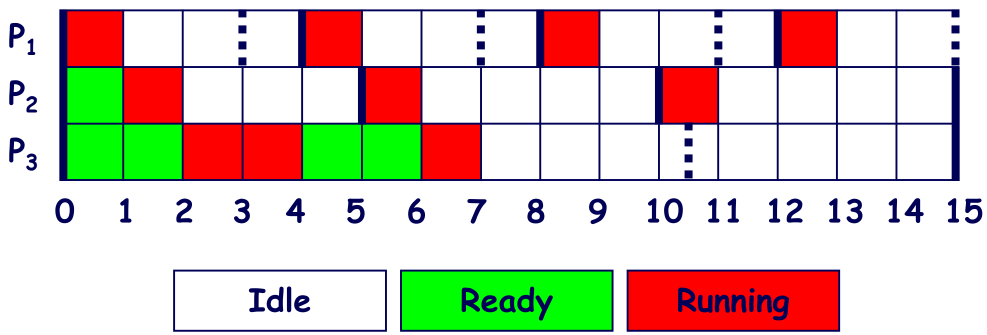
Condizione fortemente conservativa se esistono uno o più processi con deadline relativa << periodo.

… TEST DI SCHEDULABILITÀ

	C	T	D	C/D
P_1	1	4	3	0.33
P_2	1	5	5	0.20
P_3	3	15	10.5	0.29

$$\sum_{j=1}^{3} \frac{C_j}{D_j} = 0.82 > U_{RMPO}(N=3) = 0.78$$

Il test fallisce nonostante l'insieme dei processi sia schedulabile con l'algoritmo DMPO:



IL TEST DI LEHOCZKY (90) …

Condizione sufficiente (ma non necessaria) affinché un insieme di N processi periodici e sporadici, ciascuno contraddistinto da una deadline relativa $D_j = \delta_j * T_j$, j = 1, 2, …, N, sia schedulabile con l'algoritmo DMPO è che:

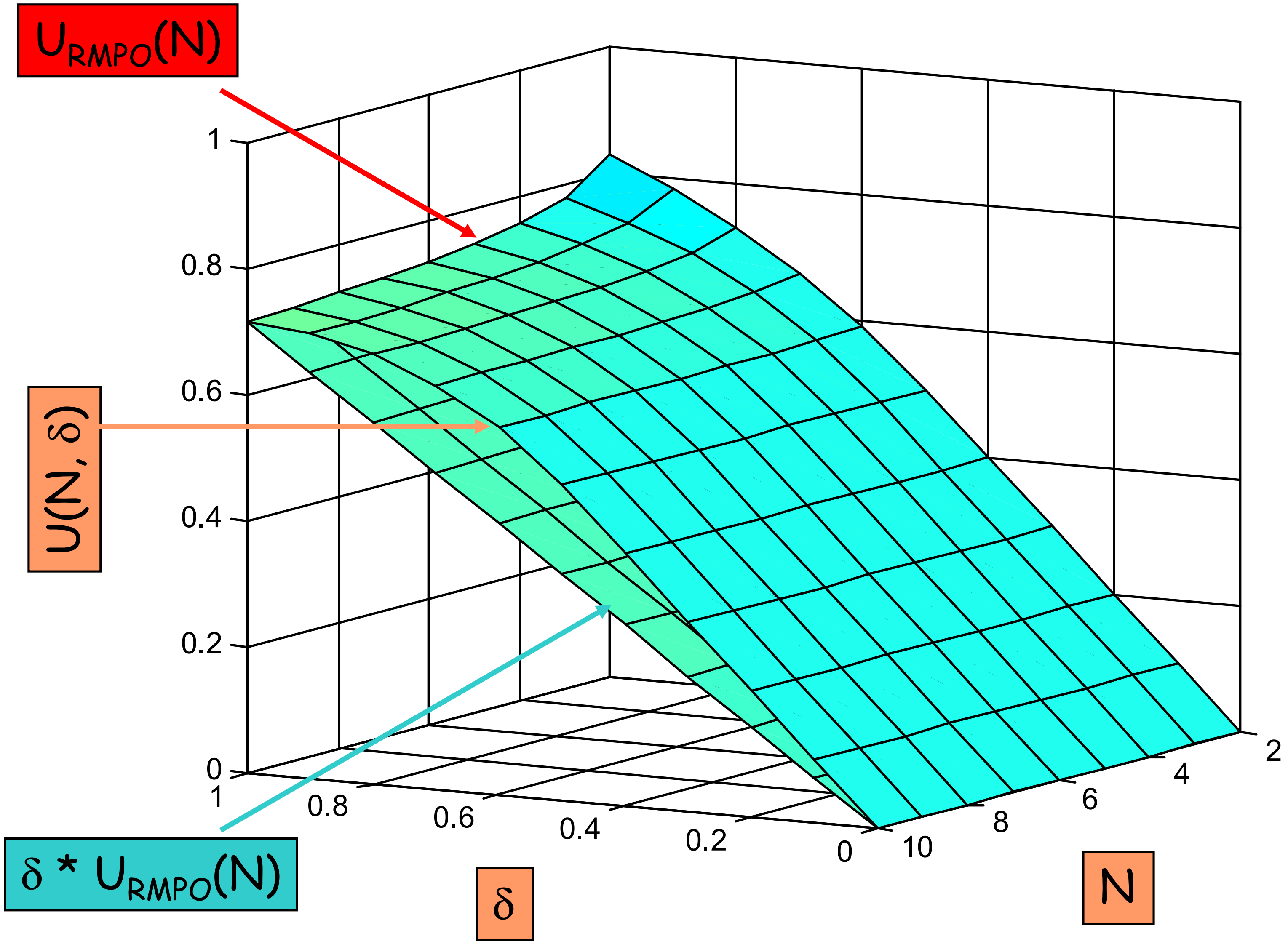
$$\sum_{j=1}^{N} \frac{C_j}{T_j} \le U(N,\delta) = \begin{cases} N\left((2\delta)^{1/N} - 1\right) + 1 - \delta & 0.5 \le \delta \le 1 \\ \delta & 0 \le \delta \le 0.5 \end{cases} \qquad \delta = \min_j \{\delta_j\}$$

	C	T	D	C/T	D/T
P_1	1	4	3	0.25	0.75
P_2	1	5	5	0.20	1
P_3	3	15	10.5	0.20	0.7

$$\boxed{\sum_{j=1}^{3} \frac{C_j}{T_j} = 0.65 \le U(N = 3, \delta = 0.7) = 0.656}$$

☺

… IL TEST DI LEHOCZKY



IL TEST BASATO SUL FATTORE DI UTILIZZAZIONE EFFICACE DEI PROCESSI …

Il test di Lehoczky nella formulazione precedente risulta ancora decisamente conservativo se i δ_j sono alquanto diversi fra loro.

In tal caso è conveniente applicare il test calcolando individualmente per ogni processo P_j, in ordine di priorità decrescente, il "fattore di utilizzazione efficace" f_j, definito da:

$$f_j = \left(\sum_{i \in H_n} \frac{C_i}{T_i} \right) + \frac{1}{T_j} \left(C_j + \sum_{k \in H_1} C_k \right)$$

essendo H_1 e H_n i due sottoinsiemi di processi di priorità $\geq p(P_j)$ con periodo, rispettivamente, $\geq D_j$ e $< D_j$,

e verificando se risulta:

$$f_j \leq U\left(N = |H_n| + 1, \delta = \delta_j\right)$$

L'insieme di processi è schedulabile se tale condizione (sufficiente, non necessaria) è soddisfatta per $\forall$ P_j, $j = 1, 2, \ldots, N$.

… IL TEST BASATO SUL FATTORE DI UTILIZZAZIONE EFFICACE DEI PROCESSI …

	C	T	D	C/T	C/D	D/T
P_1	1	10	10	0.100	0.100	1
P_2	4	12	12	0.333	0.333	1
P_3	4	15	6	0.267	0.667	0.400
P_4	1	30	15	0.033	0.067	0.500
P_5	5	60	29	0.083	0.172	0.483

$\Delta = 1.339 > U_{RMPO}(N=5) = 0.743$

$U = 0.817 > U(N=5, \delta=0.4) = 0.4$

	H_n	H_1	f	$U(N,\delta)$
P_3	{ }	{ }	4/15=0.267	U(1,0.4)=0.4
P_1	{ }	$\{P_3\}$	(1+4)/10=0.5	U(1,1)=1
P_2	$\{P_1\}$	$\{P_3\}$	1/10+(4+4)/12=0.767	U(2,1)=0.828
P_4	$\{P_1,P_2\}$	$\{P_3\}$	1/10+4/12+(1+4)/30=0.6	U(3,0.5)=0.5
P_5	$\{P_3,P_1,P_2\}$	$\{P_4\}$	4/15+1/10+4/12+(5+1)/60=0.8	U(4,0.483)=0.483

… IL TEST BASATO SUL FATTORE DI UTILIZZAZIONE EFFICACE DEI PROCESSI

	C	T	D	C/T	C/D	D/T
P_1	1	4	4	0.250	0.250	1
P_2	1	6	5	0.167	0.200	0.833
P_3	1	12	7	0.083	0.143	0.583
P_4	2	9	9	0.222	0.222	1

$$\sum_{j=1}^{4} \frac{C_j}{D_j} = 0.815 > U_{RMPO}(N=4) = 0.757$$

$$\sum_{j=1}^{4} \frac{C_j}{T_j} = 0.722 > U(N=4, \delta=0.583) = 0.574$$

	H_n	H_1	f	$U(N,\delta)$
P_1	{ }	{ }	1/4 = 0.25	U(1,1) = 1
P_2	$\{P_1\}$	{ }	1/4 + 1/6 = 0.417	U(2,5/6) = 0.749
P_3	$\{P_1,P_2\}$	{ }	1/4 + 1/6 + 1/12 = 0.5	U(3,7/12) = 0.575
P_4	$\{P_1,P_2\}$	$\{P_3\}$	1/4 + 1/6 + (2+1)/9 = 0.75	U(3,1) = 0.78

SCHEDULABILITA' IN PRESENZA DI INTERRUPT SERVICE ROUTINES …

Il test basato sul fattore di utilizzazione efficace
può essere convenientemente applicato per verificare la schedulabilità
di un insieme di processi periodici (e/o sporadici)
gestiti in accordo alla strategia RMPO (o DMPO)
in presenza di interrupt service routines (ISR).

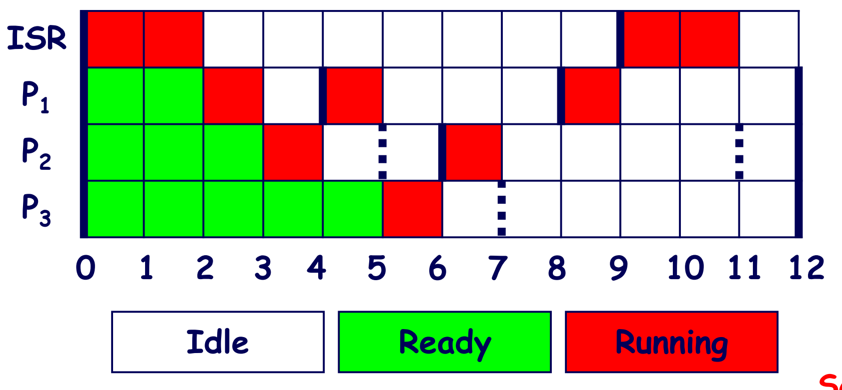
E' sufficiente a tale scopo procedere al calcolo
del fattore di utilizzazione efficace di ogni ISR e di ogni processo,
tenendo conto che le ISR vengono eseguite
ad una priorità in generale più elevata di quella che loro deriverebbe
dalla corrispondente frequenza di esecuzione.

Esempio: la precedente applicazione risulterebbe ancora schedulabile se,
al posto di P_4, considerassimo una ISR di pari frequenza e tempo di esecuzione ?

… SCHEDULABILITA' IN PRESENZA DI INTERRUPT SERVICE ROUTINES

	C	T	D	C/T	C/D	D/T
P_1	1	4	4	0.250	0.250	1
P_2	1	6	5	0.167	0.200	0.833
P_3	1	12	7	0.083	0.143	0.583
ISR → P_4	2	9	9	0.222	0.222	1

	H_n	H_1	f	U(N,δ)
ISR	{ }	{ }	2/9 = 0.222	U(1,1) = 1
P_1	{ }	{ISR}	(1+2)/4 = 0.75	U(1,1) = 1
P_2	$\{P_1\}$	{ISR}	1/4 + (1+2)/6 = 0.75	U(2,5/6) = 0.749
P_3	$\{P_1,P_2\}$	{ISR}	1/4 + 1/6 + (1+2)/12 = 0.667	U(3,7/12) = 0.575



condizione sufficiente, non necessaria

L'ALGORITMO EARLIEST DEADLINE FIRST (EDF)

Ad ogni processo è dinamicamente associata una priorità tanto maggiore quanto più imminente è la corrispondente deadline assoluta.

A_4	C	T	D	C/T
P_1	5	10	10	0.5
P_2	6	15	15	0.4

$U(A_4) = 0.9$

"task-level dynamic-priority algorithm"

"job-level fixed-priority algorithm"

JOB-LEVEL FIXED vs. DYNAMIC PRIORITY

	C	T	D	C/T
P_1	0.8	2	2	0.40
P_2	1.5	5	5	0.30
P_3	1.5	6	6	0.25

U = 0.95

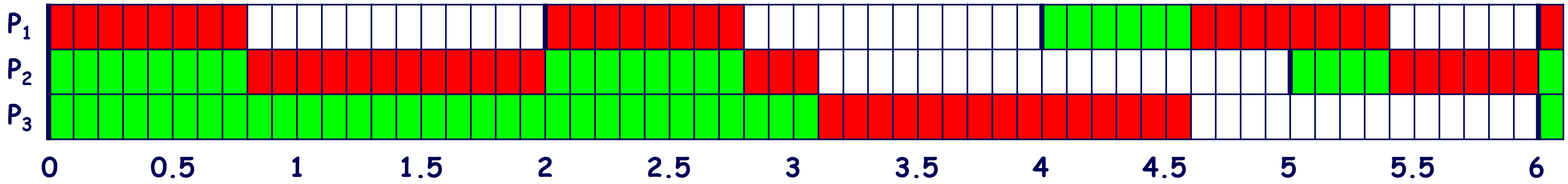
Idle

Ready

Running

EDF

$p(J_{21}) > p(J_{31})$



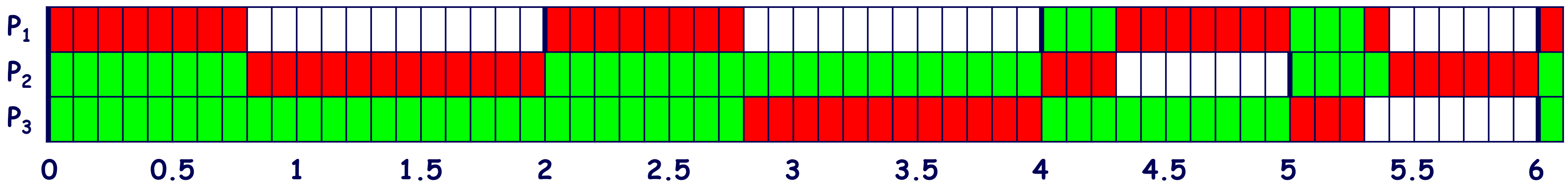
(nonstrict) Least Slack Time First: “task-level and job-level dynamic-priority algorithm”

istanti decisionali:
job release time

	t = 0		t = 2		t = 4		t = 5		t = 6	
	slack	priority	slack	priority	slack	priority	slack	priority	slack	priority
P_1	1.2	p_H	1.2	p_H	1.2	p_M	0.9	p_M	1.2	p_H
P_2	3.5	p_M	2.7	p_L	0.7	p_H	3.5	p_L	3.1	p_M
P_3	4.5	p_L	2.5	p_M	1.7	p_L	0.7	p_H	4.5	p_L

LST

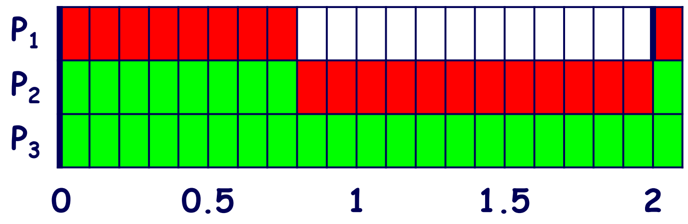
$p(J_{21}) > p(J_{31})$ $p(J_{21}) < p(J_{31})$ $p(J_{21}) > p(J_{31})$



NONSTRICT LST

	C	T	D	C/T
P_1	0.8	2	2	0.40
P_2	1.5	5	5	0.30
P_3	1.5	6	6	0.25

	t = 0		t = 2	
	slack	priority	slack	priority
P_1	1.2	p_H	1.2	p_H
P_2	3.5	p_M	2.7	p_L
P_3	4.5	p_L	2.5	p_M

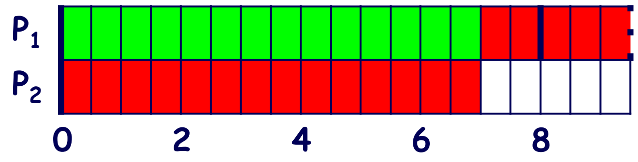


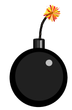
STRICT vs. NONSTRICT LST

	C	T	D	C/T
P_1	2.5	8	8	0.31
P_2	7	12	12	0.58

nonstrict LST

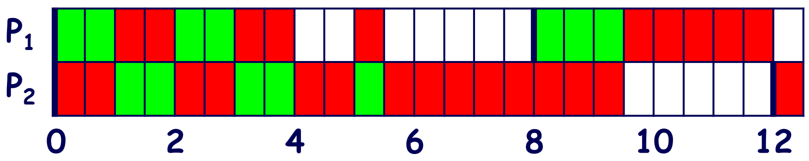
	t = 0	t = 7
P_1	5.5	- 1.5
P_2	5	

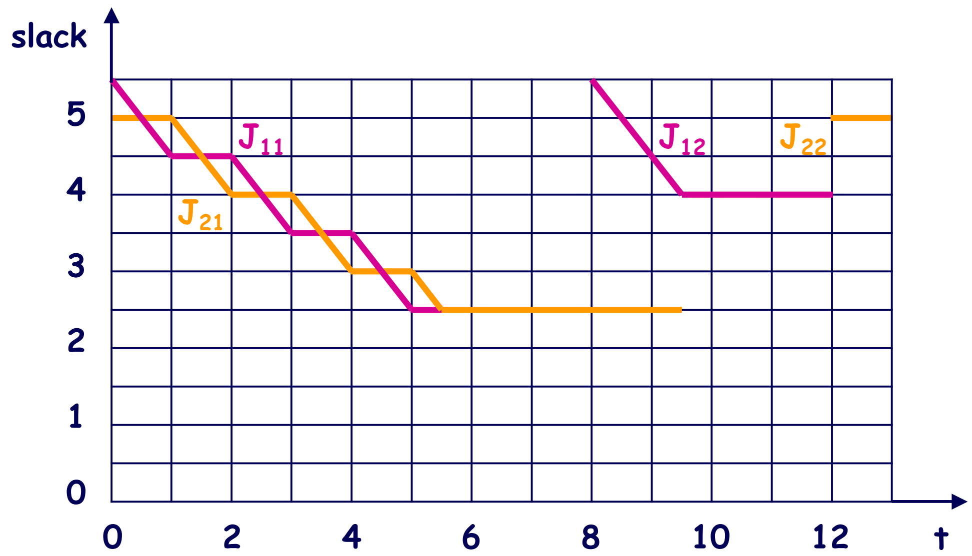




strict LST

$\Delta t = 1$





EDF: CONDIZIONI DI SCHEDULABILITÀ

Condizione necessaria e sufficiente affinché un insieme di N processi periodici sia schedulabile con l'algoritmo EDF è che:

$$U = \sum_{i=1}^{N} \frac{C_i}{T_i} \leq U_{EDF} = 1$$

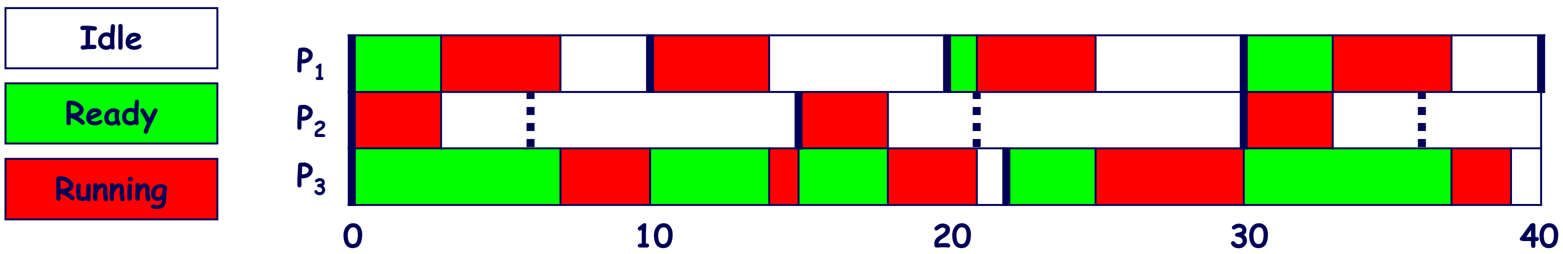
Condizione sufficiente (ma non necessaria) affinché un insieme di N processi periodici e sporadici sia schedulabile con l'algoritmo EDF è che:

$$\Delta = \sum_{i=1}^{N} \frac{C_i}{D_i} \leq 1$$

A_7	C	T	D	C/T	C/D
P_1	4	10	10	0.40	0.40
P_2	3	15	6	0.20	0.50
P_3	7	22	22	0.32	0.32

$U(A_7) = 0.92$

$\Delta(A_7) = 1.22$



ANALISI DI SCHEDULABILITÀ: L'APPROCCIO "PROCESSOR DEMAND" [BARUAH et al. (90)] …

La schedulabilità di un insieme di N processi periodici e sporadici, contraddistinti da un fattore di utilizzazione del processore $U \leq 1$ ed attivati contemporaneamente all'istante 0, è garantita se in ogni intervallo [0,t] il tempo di elaborazione cumulativamente richiesto per completare l'esecuzione di tutti i job aventi deadline $\leq$ t non eccede il tempo disponibile t:

$$C_P(0,t) = \sum_{i=1}^{N} C_i(0,t) = \sum_{i=1}^{N} \left(\left\lfloor \frac{t - D_i}{T_i} \right\rfloor + 1 \right) C_i \leq t$$

Ai fini della schedulabilità, è sufficiente verificare se tale condizione è soddisfatta per $\forall\, t \in \mathcal{D} \equiv \{d_{ik} \mid d_{ik} = (k-1)\, T_i + D_i,\ d_{ik} < BI,\ 1 \leq i \leq N,\ k \geq 1\}$, essendo [0,BI] ("busy interval") il più piccolo intervallo entro il quale termina l'esecuzione di tutti i job attivati in esso (ovvero all'istante BI il processore risulta idle). BI può essere calcolato iterativamente nel seguente modo:

$$BI^0 = \sum_{i=1}^{N} C_i \qquad\qquad BI^n = \sum_{i=1}^{N} \left\lceil \frac{BI^{n-1}}{T_i} \right\rceil C_i \quad n = 1,2,\ldots$$

(se $U \leq 1$, $\exists$ n | $BI^n = BI^{n-1} \leq H$ (iperperiodo); solo in caso contrario $BI \rightarrow \infty$)

… L'APPROCCIO "PROCESSOR DEMAND" …

A_7	C	T	D	C/T	C/D
P_1	4	10	10	0.40	0.40
P_2	3	15	6	0.20	0.50
P_3	7	22	22	0.32	0.32

$U(A_7) = 0.92$

$\Delta(A_7) = 1.22$

$BI^0 = C_1 + C_2 + C_3 = 14$

$BI^1 = \lceil BI^0/T_1 \rceil C_1 + \lceil BI^0/T_2 \rceil C_2 + \lceil BI^0/T_3 \rceil C_3 = \lceil 14/10 \rceil 4 + \lceil 14/15 \rceil 3 + \lceil 14/22 \rceil 7 = 18$

$BI^2 = \lceil BI^1/T_1 \rceil C_1 + \lceil BI^1/T_2 \rceil C_2 + \lceil BI^1/T_3 \rceil C_3 = \lceil 18/10 \rceil 4 + \lceil 18/15 \rceil 3 + \lceil 18/22 \rceil 7 = 21$

$BI^3 = \lceil BI^2/T_1 \rceil C_1 + \lceil BI^2/T_2 \rceil C_2 + \lceil BI^2/T_3 \rceil C_3 = \lceil 21/10 \rceil 4 + \lceil 21/15 \rceil 3 + \lceil 21/22 \rceil 7 = 25$

$BI^4 = \lceil BI^3/T_1 \rceil C_1 + \lceil BI^3/T_2 \rceil C_2 + \lceil BI^3/T_3 \rceil C_3 = \lceil 25/10 \rceil 4 + \lceil 25/15 \rceil 3 + \lceil 25/22 \rceil 7 = 32$

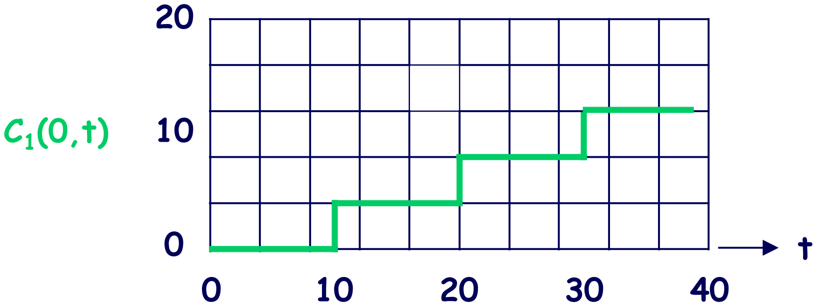
$BI^5 = \lceil BI^4/T_1 \rceil C_1 + \lceil BI^4/T_2 \rceil C_2 + \lceil BI^4/T_3 \rceil C_3 = \lceil 32/10 \rceil 4 + \lceil 32/15 \rceil 3 + \lceil 32/22 \rceil 7 = 39$

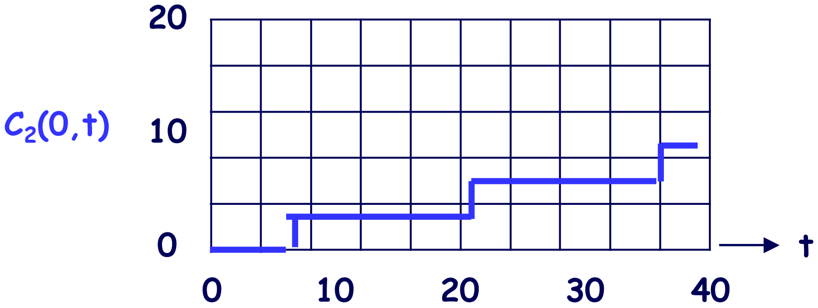
$BI^6 = \lceil BI^5/T_1 \rceil C_1 + \lceil BI^5/T_2 \rceil C_2 + \lceil BI^5/T_3 \rceil C_3 = \lceil 39/10 \rceil 4 + \lceil 39/15 \rceil 3 + \lceil 39/22 \rceil 7 = 39 = BI^5$

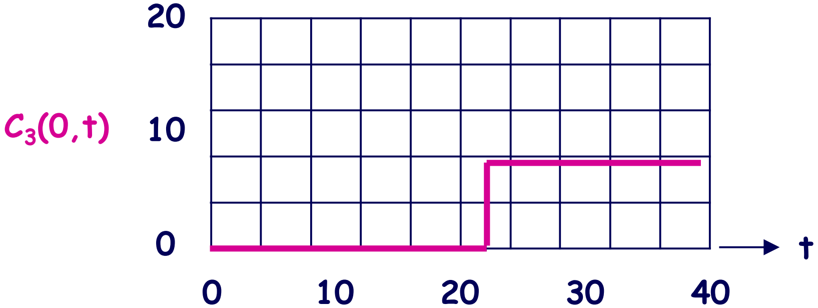
$BI = 39 < H = \text{mcm}\ (10, 15, 22) = 330$

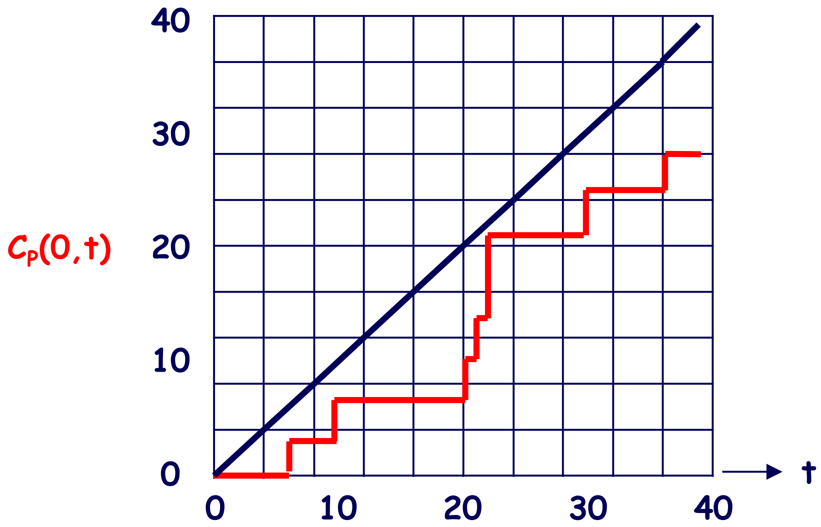
$\mathcal{D} \equiv \{6, 10, 20, 21, 22, 30, 36\}$

… L'APPROCCIO "PROCESSOR DEMAND" …



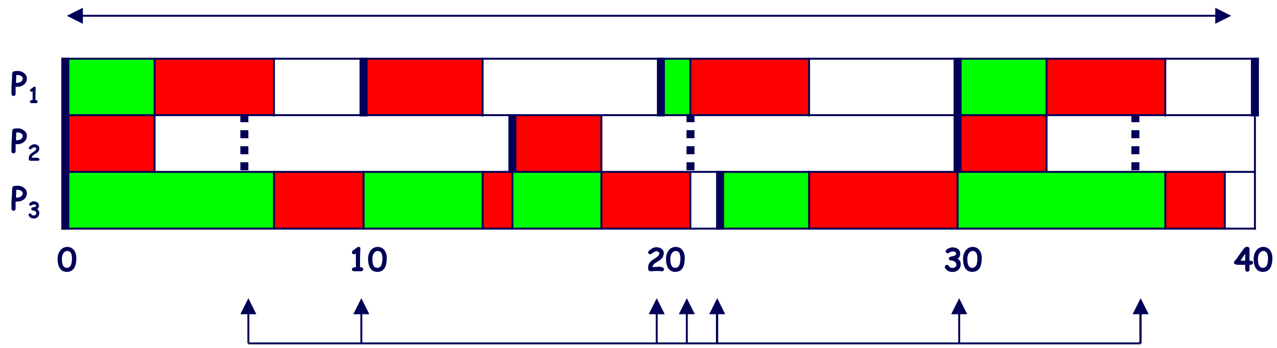






… L'APPROCCIO "PROCESSOR DEMAND" …

BI = 39 < H = 330



$\mathcal{D} \equiv \{6,\ 10,\ 20,\ 21,\ 22,\ 30,\ 36\}$

t	$C_1(0,t)$	$C_2(0,t)$	$C_3(0,t)$	$C_P(0,t)$	$\leq$ t
6	0	3	0	3	👍
10	4	3	0	7	👍
20	8	3	0	11	👍
21	8	6	0	14	👍
22	8	6	7	21	👍
30	12	6	7	25	👍
36	12	9	7	28	👍

… L'APPROCCIO "PROCESSOR DEMAND" …

A_8	C	T	D	C/T	C/D
P_1	4	10	10	0.40	0.40
P_2	3	15	6.5	0.20	0.46
P_3	8	21	21	0.38	0.38

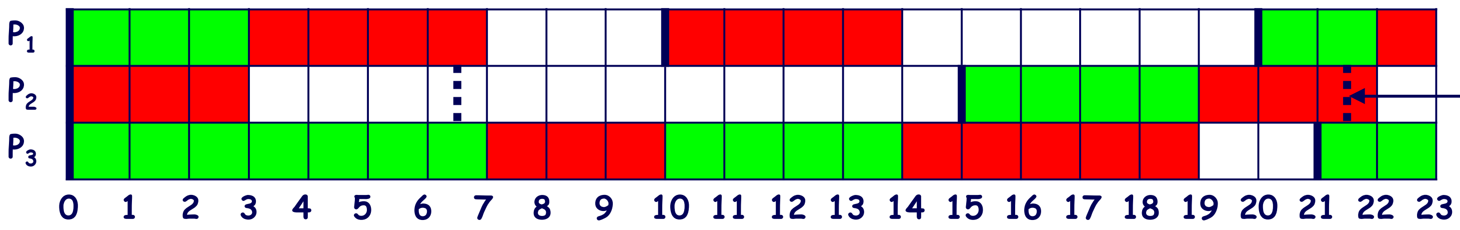
$U(A_8) = 0.98$

$\Delta(A_8) = 1.24$

n	BI^n
0	15
1	19
2	22
...	...

$(BI^8 = BI^7 = 60)$

t	$C_1(0,t)$	$C_2(0,t)$	$C_3(0,t)$	$C_P(0,t)$	$\le t$
6.5	0	3	0	3	👍
10	4	3	0	7	👍
20	8	3	0	11	👍
21	8	3	8	19	👍
21.5	8	6	8	22	👎



Idle Ready Running

… L'APPROCCIO "PROCESSOR DEMAND" …

Indicato con:

$$t^* = \frac{\sum_{i=1}^{N}\left(1-\frac{D_i}{T_i}\right)C_i}{1-U}$$

è possibile, qualora risulti $t^* < BI$, migliorare l'efficienza dell'approccio limitandosi a verificare se è soddisfatta la condizione $C_P(0,t) \le t$ solo per il seguente insieme $\mathcal{D}^*$ di istanti t:

$$\mathcal{D}^* \equiv \{d_{ik} \mid d_{ik} = (k-1)\,T_i + D_i,\ d_{ik} < t^*,\ 1 \le i \le N,\ k \ge 1\}.$$

Ciò deriva dalla constatazione che:

$$C_P(0,t) \le C^*(0,t) = \sum_{i=1}^{N}\left(\frac{t-D_i}{T_i}+1\right)C_i = U\,t + \sum_{i=1}^{N}\left(1-\frac{D_i}{T_i}\right)C_i,\ \forall t$$

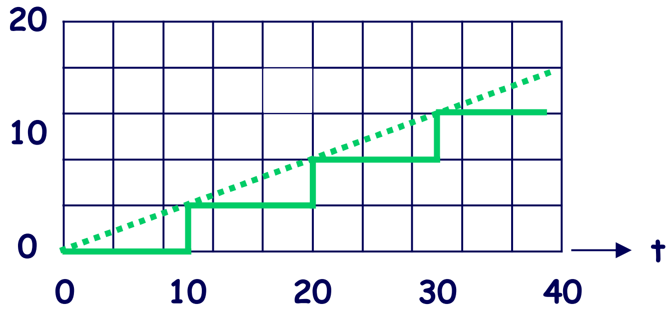
e quindi che $C_P(0,t) \le t$ per $\forall\ t \ge t^*$.

… L'APPROCCIO "PROCESSOR DEMAND" …

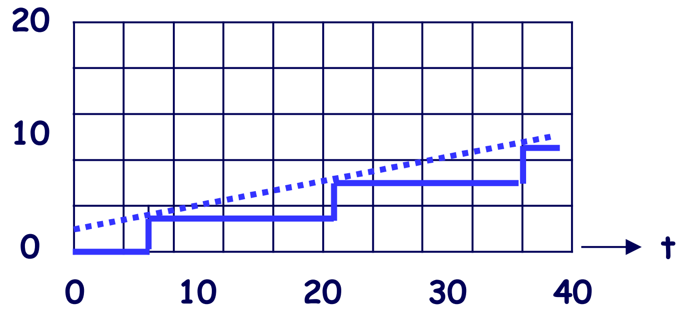
La diversa modellazione del tempo di elaborazione complessivamente richiesto in [0,t]:

A_7	C	T	D	C/T	C/D
P_1	4	10	10	0.40	0.40
P_2	3	15	6	0.20	0.50
P_3	7	22	22	0.32	0.32

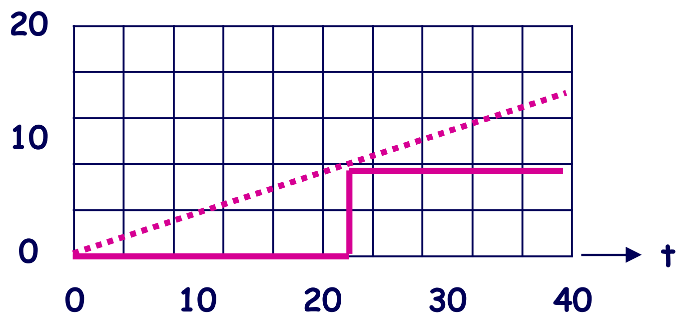
$C_1(0,t)$



$C_2(0,t)$



$C_3(0,t)$



$C_P(0,t)$

$BI = 39 \rightarrow \mathfrak{D} \equiv \{6,\ 10,\ 20,\ 21,\ 22,\ 30,\ 36\}$

$t^* = 22 \rightarrow \mathfrak{D}^* \equiv \{6,\ 10,\ 20,\ 21\}$

$t^* < BI \rightarrow \mathfrak{D}^* \subset \mathfrak{D}$

… L'APPROCCIO "PROCESSOR DEMAND"

A_9	C	T	D
P_1	4	10	5
P_2	3.5	15	8
P_3	5	22	21

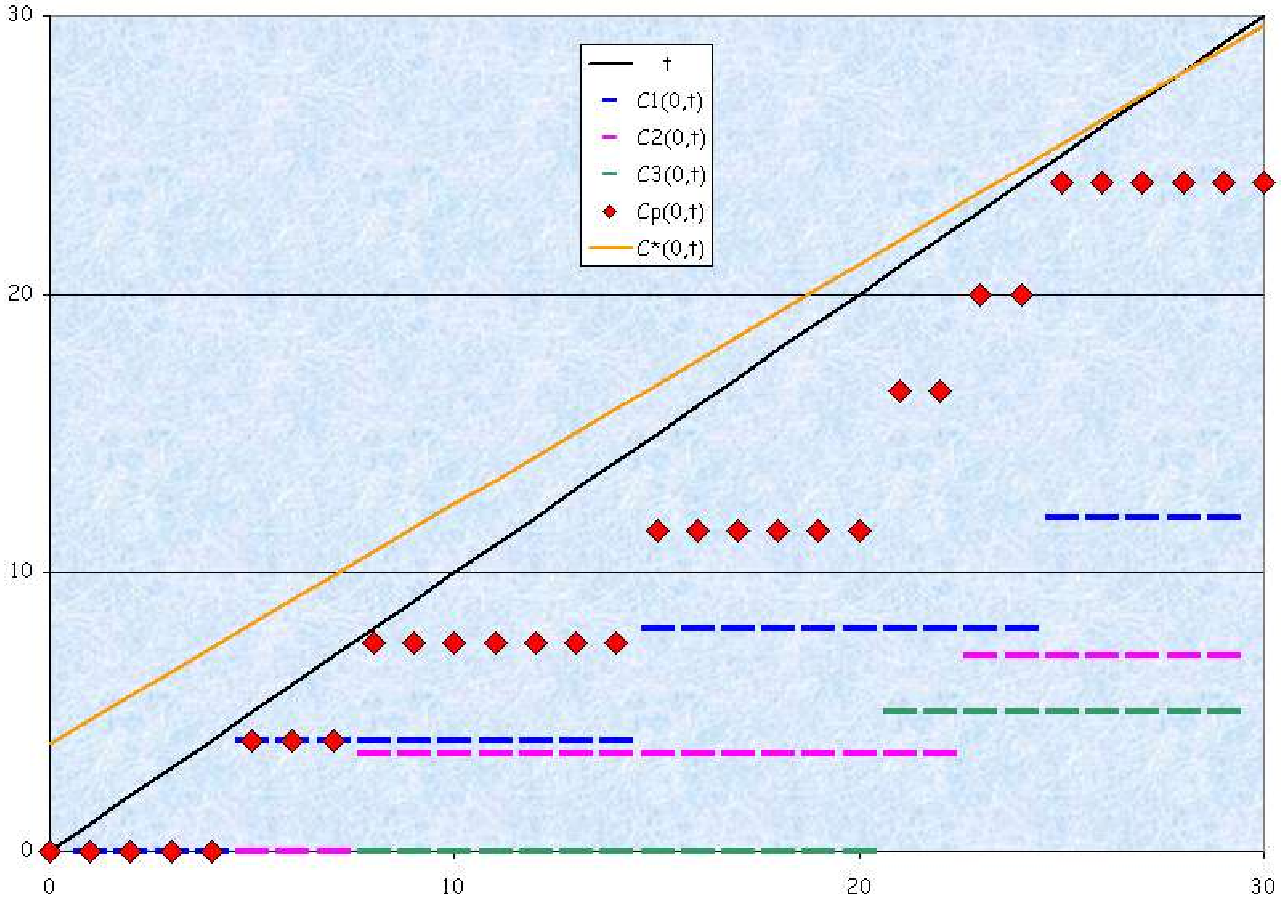
BI = 20

$\mathfrak{D} \equiv \{5,\ 8,\ 15\}$

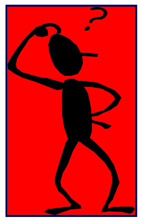
$t^* = 27.7$

$\mathfrak{D}^* \equiv \{5,\ 8,\ 15,\ 21,\ 23,\ 25\}$

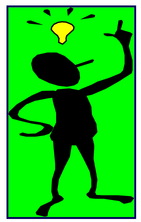
$t^* > BI \rightarrow \mathfrak{D}^* \supset \mathfrak{D}$



t	$C_1(0,t)$	$C_2(0,t)$	$C_3(0,t)$	$C_p(0,t)$	$\leq t$
5	4	0	0	4	
8	4	3.5	0	7.5	
15	8	3.5	0	11.5	



In generale:
t^* o BI ?



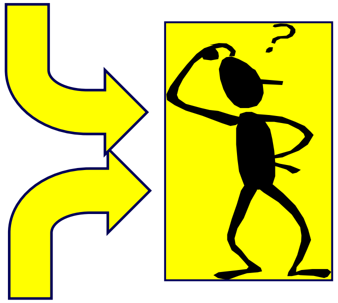
BI^n

min (t^*, BI)

RMPO, DMPO, EDF: ASPETTI REALIZZATIVI

Separazione tra aspetti funzionali e temporali

Name	P_1	...	P_N
Period	P_1Period	...	P_NPeriod
Relative Deadline	P_1Deadline	...	P_NDeadline
InitializationEntryPoint	P_1ResetHandler	...	P_NResetHandler
JobEntryPoint	P_1Handler	...	P_NHandler
MissedDeadlineEntryPoint	P_1MissedDeadlineHandler	...	P_NMissedDeadlineHandler



Specifiche a livello di "application design"

Primitive del Sistema Operativo (VxWorks)

SystemTickRate ( ) /* real-time clock frequency */

SystemTickConnect (EntryPoint) /* hook to real-time clock interrupt handler */

TaskSpawn (Name, Priority, Options, StackSize, EntryPoint, Arg1, …, Arg10) /* creation & activation */

SetTaskPriority (TaskId, Priority) /* TaskId = 0: calling task */

TaskSuspend (TaskId) /* TaskId = 0: calling task */

TaskResume (TaskId)

TaskIsSuspended (TaskId)

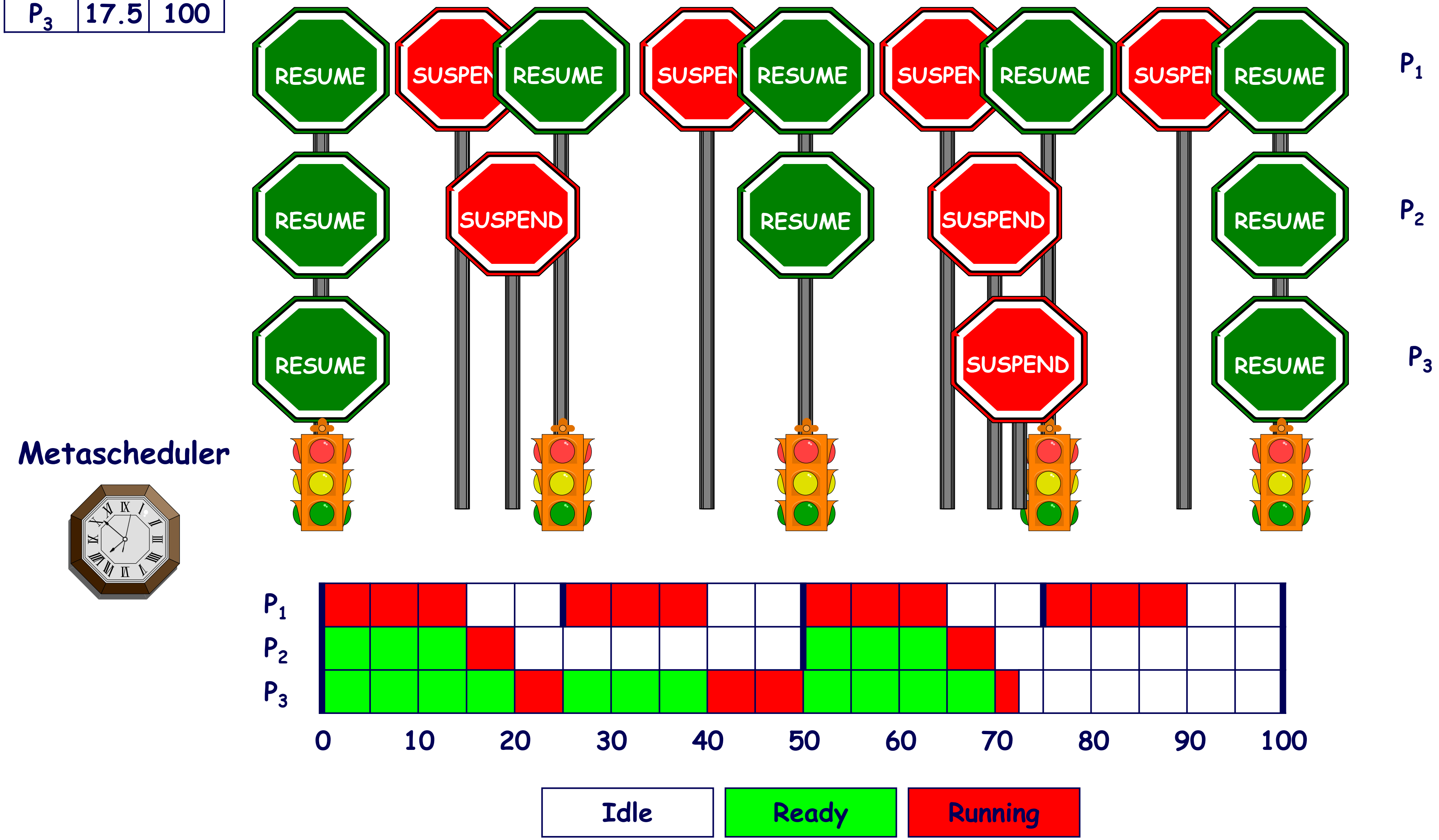
…

Distinzione tra politiche e meccanismi

RMPO, DMPO: IL RUOLO DEL METASCHEDULER

A_2	C	T
P_1	15	25
P_2	5	50
P_3	17.5	100

Ad ogni processo è attribuita una priorità (statica) in base alla corrispondente deadline relativa



ESECUZIONE PERIODICA DEL METASCHEDULER

Real-Time Clock Interrupt

```
void SystemTickInterruptHandler (void)
{
   …
   SystemTickHandler ( );
   …
}
```

SystemTickConnect (SystemTickHandler);

```
private void SystemTickHandler (void)
{
  if ((--MetaschedulerReleaseTimer) == 0)
  {
     TaskResume (MetaschedulerId);
     MetaschedulerReleaseTimer = MetaschedulerReleaseTime; /* = 50 se
  }
}
```

T_{RTC} = 500 μs

$T_{METASCHEDULER}$ = mcd (25 (P_1), 50 (P_2), 100 (P_3)) = 25 ms */

ESECUZIONE PERIODICA DEI PROCESSI …

```
MetaschedulerId = TaskSpawn ('Metascheduler', HighPriority,
                             DefaultOptions, DefaultStackSize,
                             MetaschedulerShell, 0);
```

```
private void MetaschedulerShell (void)
{
  while (true)
  {
    TaskSuspend (0);
    Metascheduler ( );
  }
}
```

```
private void Metascheduler (void)
{
  repeat for each Pj
  {
    if ((--PjReleaseTimer) == 0)
    {
      TaskResume (PjId);
      PjReleaseTimer = PjReleaseTime; /* = 1 (P1), 2 (P2), 4 (P3) */
    }
  }
}
```

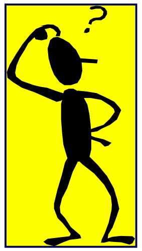
… ESECUZIONE PERIODICA DEI PROCESSI

/* for each Pj */

PjId = TaskSpawn (‘Pj’, PjPriority, DefaultOptions, DefaultStackSize, TaskShell, PjHandler);

```
private void TaskShell (void (* PjHandler) (void))
{
      while (true)
      {
         TaskSuspend (0);
         PjHandler ( );
      }
}
```

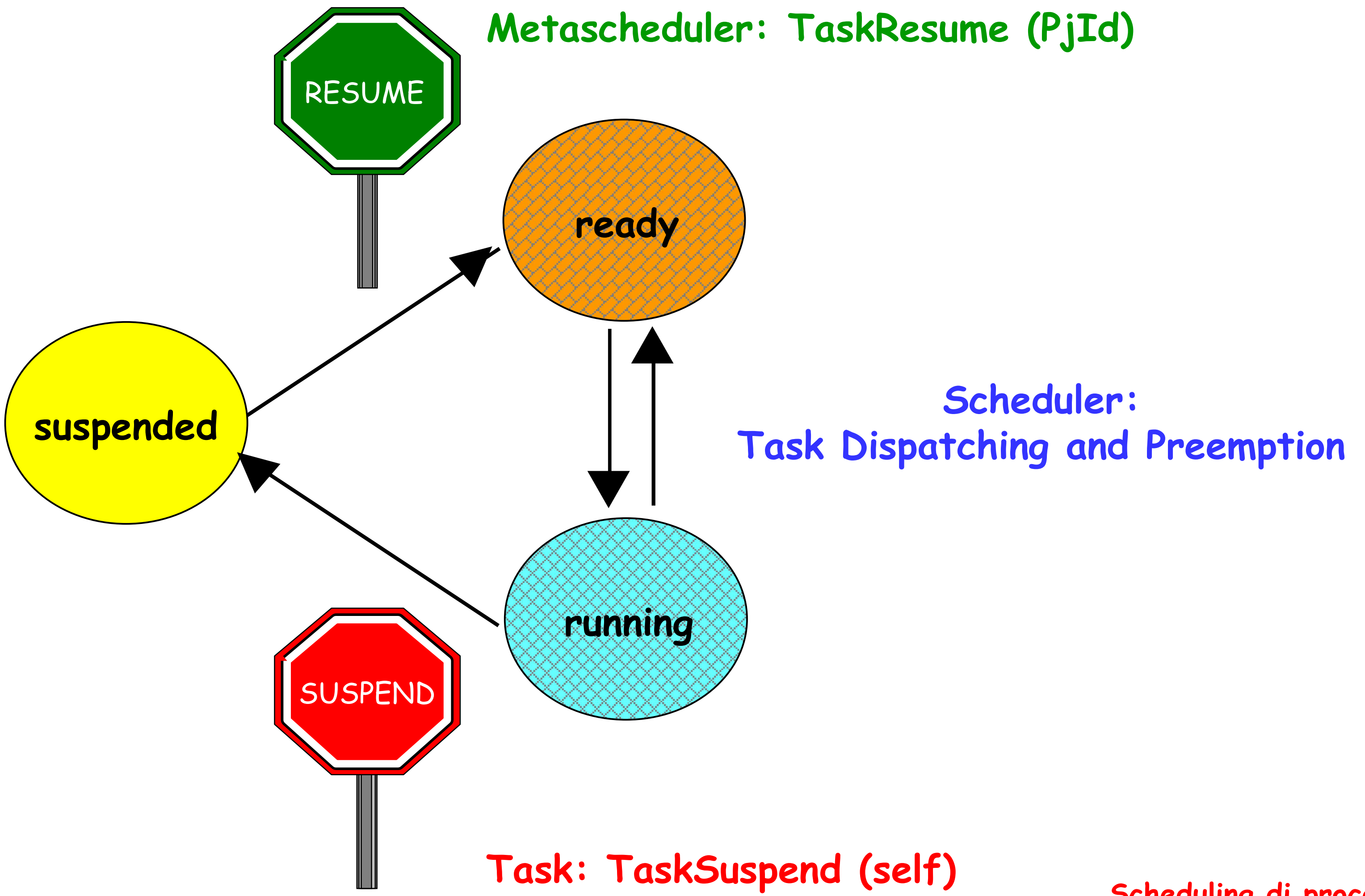
missed deadline



GESTIONE DEI SOVRACCARICHI ...

Gli stati di un processo
a livello di:

metascheduler	scheduler
idle	suspended
not_idle	ready / running



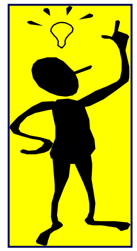
… GESTIONE DEI SOVRACCARICHI …

```
private void Metascheduler (void)
{
   repeat for each Pj
   {
      if (PjState != idle)
      {
         PjExecutionTimer++;
         if (TaskIsSuspended (PjId))
         {
            ExecutionTimeLogHandler (Pj, PjExecutionTimer) /* min, max, average, … */
            PjState = idle;
         }
         else
         {
            if (PjExecutionTimer == PjCompletionTime) /* missed deadline */
            {
                PjMissedDeadlineCounter++;
                PjMissedDeadlineHandler; /* application dependent */
            }
         }
      }
      …
```

… GESTIONE DEI SOVRACCARICHI …

```
…
if ((--PjReleaseTimer) == 0)
{
  if (PjState == idle)
  {
    TaskResume (PjId);
    PjReleaseTimer = PjReleaseTime;
    PjExecutionTimer = 0;
    PjState = not_idle;
  }
  else  /* 💣 overrun */
  {
    PjOverrunCounter++;
    if (PjOverrunHandlingPolicy == ASAP)
        PjReleaseTimer = 1; /* release next job As Soon As Possible */
    else PjReleaseTimer = PjReleaseTime; /* skip next job execution */
  }
}
}
}
```

MetaSchedulerReleaseTime	PjReleaseTime		
1	50	100	200
2	25	50	100
	P_1	P_2	P_3



risoluzione

☹

… GESTIONE DEI SOVRACCARICHI

```
private void SystemTickHandler (void)
{
   if ((--MetaschedulerReleaseTimer) == 0)
   {
      if (TaskIsSuspended (MetaschedulerId))
            TaskResume (MetaschedulerId);
      else MetaschedulerOverrunCounter++;
      MetaschedulerReleaseTimer = MetaschedulerReleaseTime;
   }
}
```

RMPO, DMPO: INIZIALIZZAZIONE

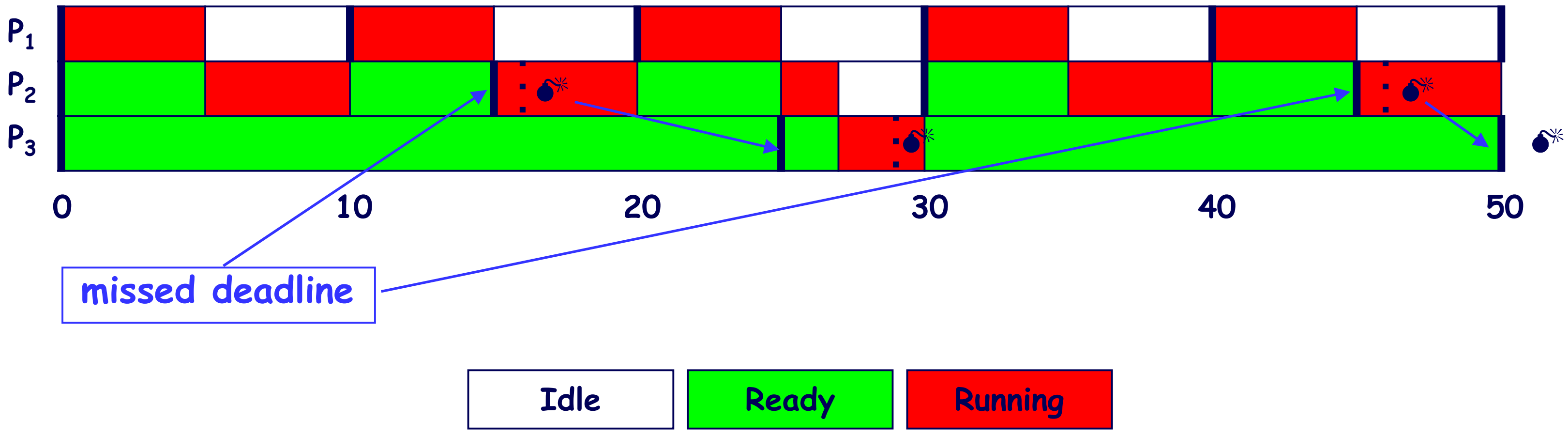
```
void MetaschedulerAndTasksInitialization (void)
{
  MetaschedulerReleaseTime /* [SystemTicks] */ = SystemTickRate /* [SystemTicks/s] */
                                                 * MetaschedulerPeriod /* [µs] */
                                                 / 1000000;
  repeat for each Pj
  {
    PjResetHandler ( );
    PjReleaseTime /* [MetaschedulerTicks] */ = PjPeriod /* [µs] */ / MetaschedulerPeriod;
    PjCompletionTime      /* idem */        = PjDeadline /* [µs] */ / MetaschedulerPeriod;
    PjPriority = PriorityHandler (PjCompletionTime);
    PjId = TaskSpawn ('Pj', PjPriority, DefaultOptions, DefaultStackSize,
                      TaskShell, PjHandler);
    PjReleaseTimer = 1; PjState = idle; PjMissedDeadlineCounter = 0; PjOverrunCounter = 0;
    ExecutionTimeLogResetHandler (Pj); /* min, max, average, … */
  }

  MetaschedulerId = TaskSpawn ('MetaScheduler', HighPriority, DefaultOptions,
                               DefaultStackSize, MetaschedulerShell, 0);
  MetaschedulerReleaseTimer = 1; MetaschedulerOverrunCounter = 0;

  SystemTickConnect (SystemTickHandler);
}
```

RMPO: RISULTATI SPERIMENTALI …

	C	T	C/T	p
P_1	5	10	0.50	max
P_2	6	15	0.40	
P_3	2	25	0.08	min

U = 0.98

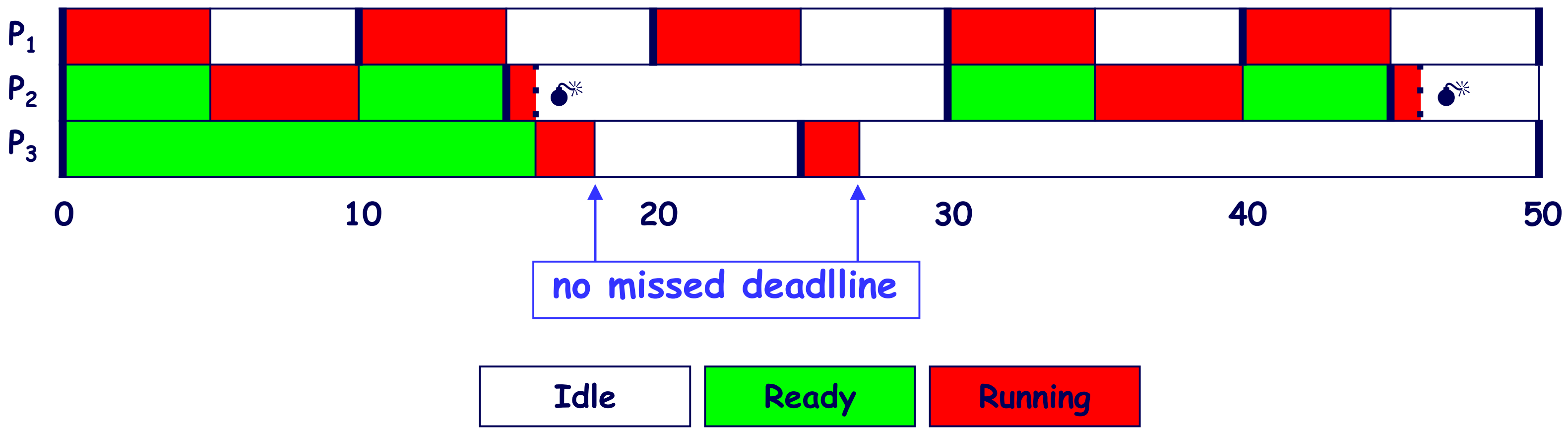
overrun handling policy: *ASAP*

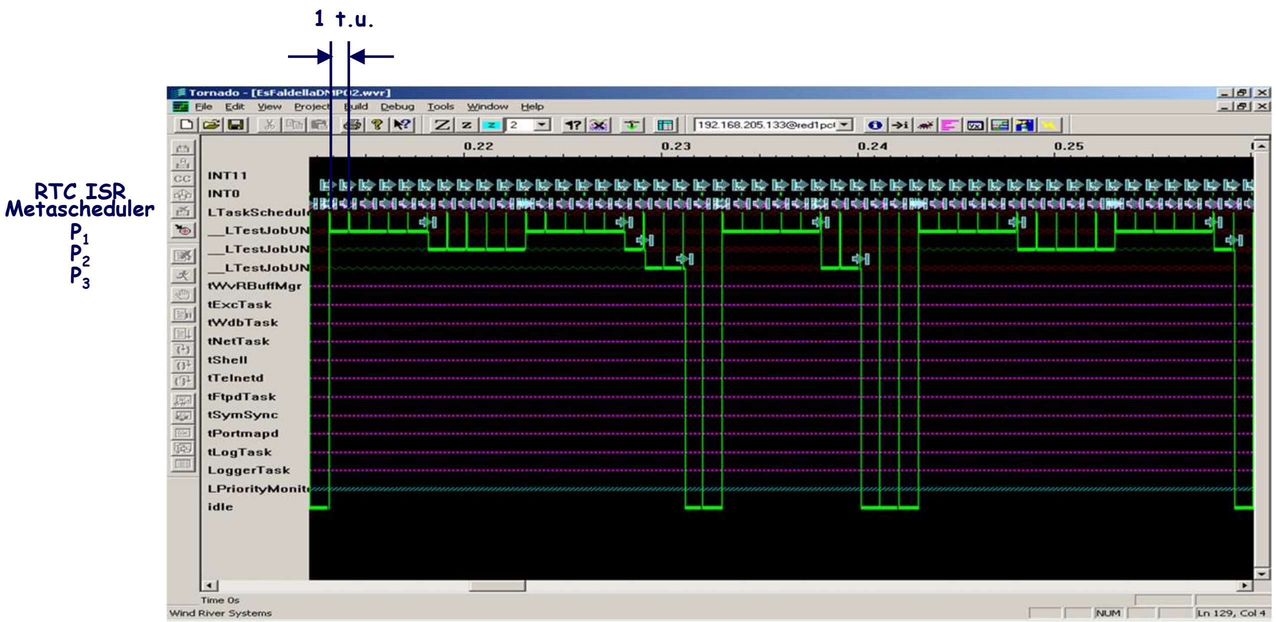


"effetto domino" sui processi di priorità inferiore

… RMPO: RISULTATI SPERIMENTALI

overrun handling policy: SKIP





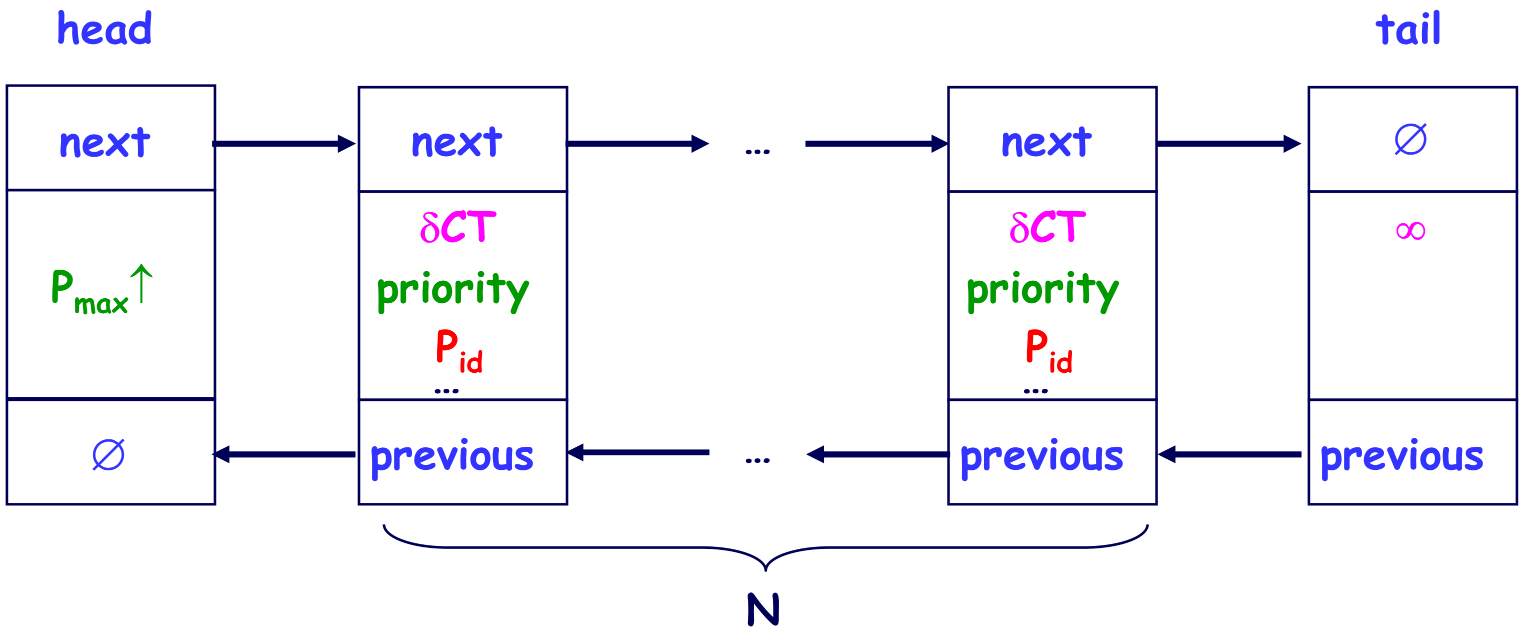
S.O.: VxWorks

Tool: WindView

EDF: ASPETTI REALIZZATIVI

Gestione dinamica della priorità di ciascun processo
in base alla corrispondente deadline assoluta

not-idle tasks queue



CT: Completion Time   δCT: delta Completion Time

$\delta CT(next(head)) = CT(next(head))$

$\delta CT(j) = CT(j) - CT(previous(j)),\ j = next(\ldots next(head))$

inizializzazione: next(head) = tail, previous(tail) = head

GESTIONE DINAMICA DELLE PRIORITA' ...

Enqueue (j, CT(j))

```
δCT(j) = CT(j);
c = next(head);
while (δCT(j) >= δCT(c))
{
      δCT(j) -= δCT(c);
      c = next(c);
}
if (c != tail) δCT(c) -= δCT(j);
p = previous(c);
next(p) = j;
next(j) = c;
previous(j) = p;
previous(c) = j;
UpdatePriorities (j);
```

Dequeue (j)

```
p = previous(j);
c = next(j);
if (c != tail) δCT(c) += δCT(j);
next(p) = c;
previous(c) = p;
UpdatePriorities (c);
```

UpdatePriorities (i)

```
while (i != tail)
{
      priority(i) = ↓(priority(previous(i)));
      SetTaskPriority (P_id(i), priority(i));
      i = next(i);
}
```

… GESTIONE DINAMICA DELLE PRIORITA'

```
private void Metascheduler (void)
{
  UpdateCompletionTimes ( );
  repeat for each Pj
  {
      idem
      Enqueue (Pj, PjCompletionTime);
      TaskResume (PjId);
      idem
  }
}
```

```
UpdateCompletionTimes ( )
c = next(head);
while (δCT(c) == 0)    c = next(c);
if (c != tail) δCT(c)--;
```

```
private void TaskShell (void (* PjHandler) (void))
{
      while (true)
      {
        TaskSuspend (0);
        PjHandler ( );
        SetTaskPriority (0, MetaSchedulerPriority);
        Dequeue (Pj);
      }
}
```

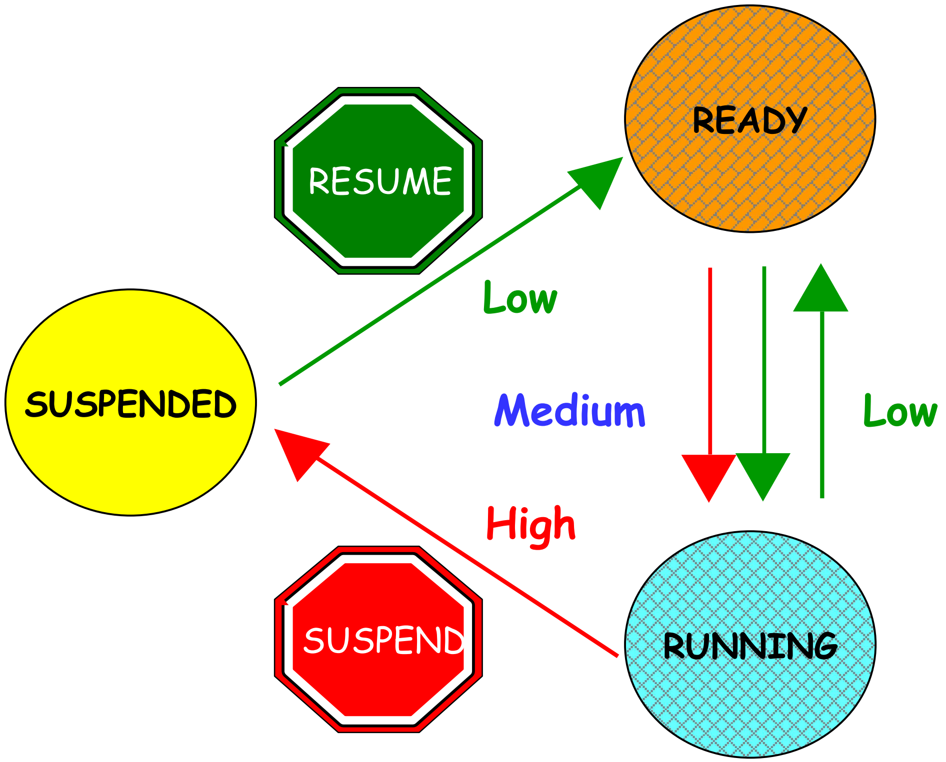
UNA EFFICIENTE GESTIONE ALTERNATIVA …

3 livelli di priorità (Low, Medium, High), condivisi da tutti i processi.

Medium: priorità di un processo allorché running, ovvero con deadline più imminente.

Low: priorità di un processo allorché pronto, ovvero con deadline meno imminente.

High: priorità (coincidente con quella del metascheduler) di un processo allorché, al termine di un job e prima di autosospendersi, promuove l'esecuzione del processo con deadline ora più imminente, modificandone la priorità da Low a Medium.



… UNA EFFICIENTE GESTIONE ALTERNATIVA

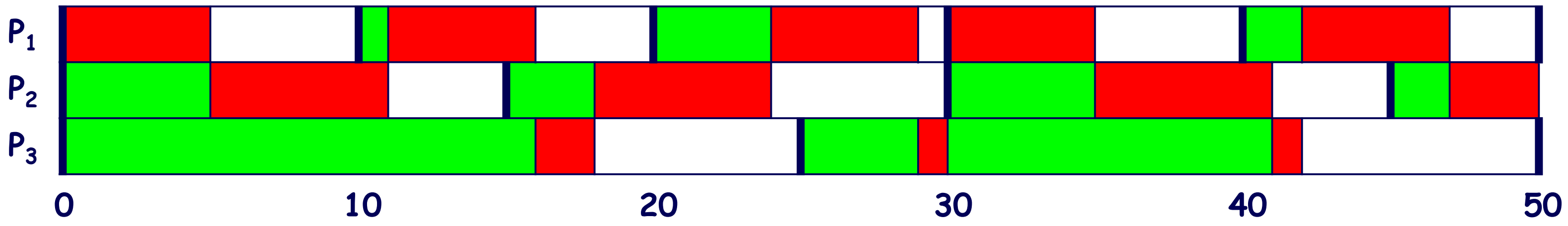
```
private void Metascheduler (void)
{
   UpdateCompletionTimes ( );
   r = next(head);
   repeat for each Pj
   {
      idem
      EnqueueOnly (Pj, PjCompletionTime);
      SetTaskPriority (PjId, Low);
      TaskResume (PjId);
      idem
   }
   if (next(head) != r)
   {
      if (r != tail) SetTaskPriority (Pid(r), Low);
      SetTaskPriority (Pid(next(head)), Medium);
   }
}
```

```
private void TaskShell (void (* PjHandler) (void))
{
   while (true)
   {
      TaskSuspend (0);
      PjHandler ( );
      SetTaskPriority (0, High);
      DequeueOnly (Pj);
      if (next(head) != tail)
         SetTaskPriority (Pid(next(head)), Medium);
   }
}
```

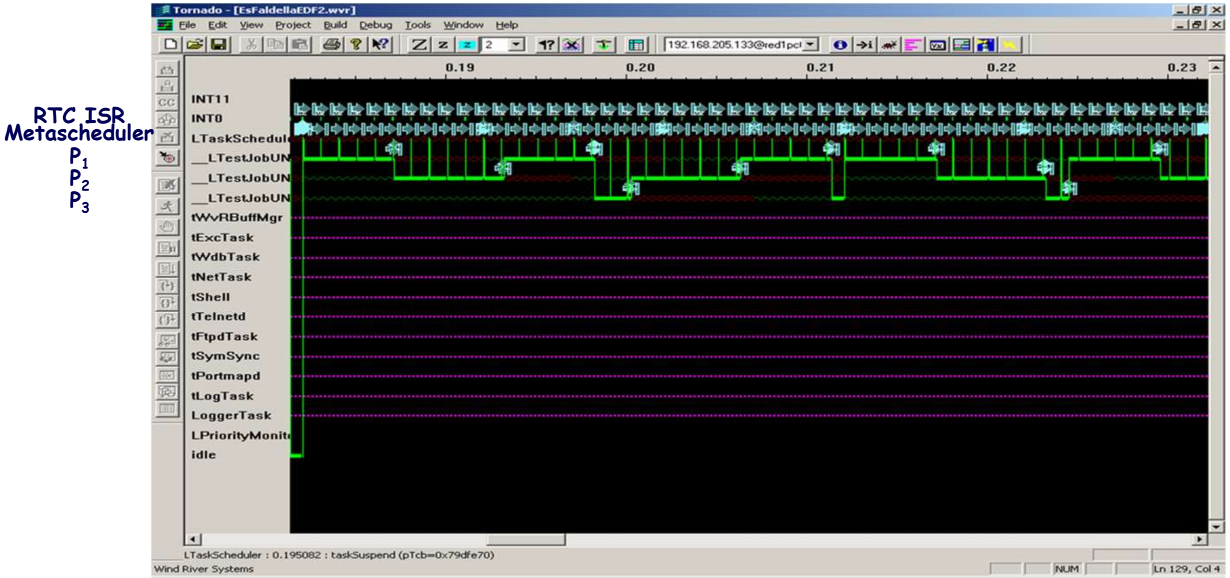
EDF: RISULTATI SPERIMENTALI

	C	T	C/T
P_1	5	10	0.50
P_2	6	15	0.40
P_3	2	25	0.08

U = 0.98



Idle | Ready | Running



S.O.: VxWorks

Tool: WindView