

# Scheduling di processi "soft real-time"

Eugenio Faldella

Dipartimento di Informatica - Scienza e Ingegneria  
Scuola di Ingegneria e Architettura, Università di Bologna



[eugenio.faldella@unibo.it](mailto:eugenio.faldella@unibo.it)  
<http://www.ing.unibo.it>

# ALGORITMI PER LA SCHEDULAZIONE DI PROCESSI APERIODICI

- servizio in background
- servizio tramite server

- ◆ a priorità statica

- polling server
- deferrable server
- priority exchange server
- sporadic server
- ...

- altri algoritmi

- ◆ a priorità statica

- slack stealer
- ...

- ◆ a priorità dinamica

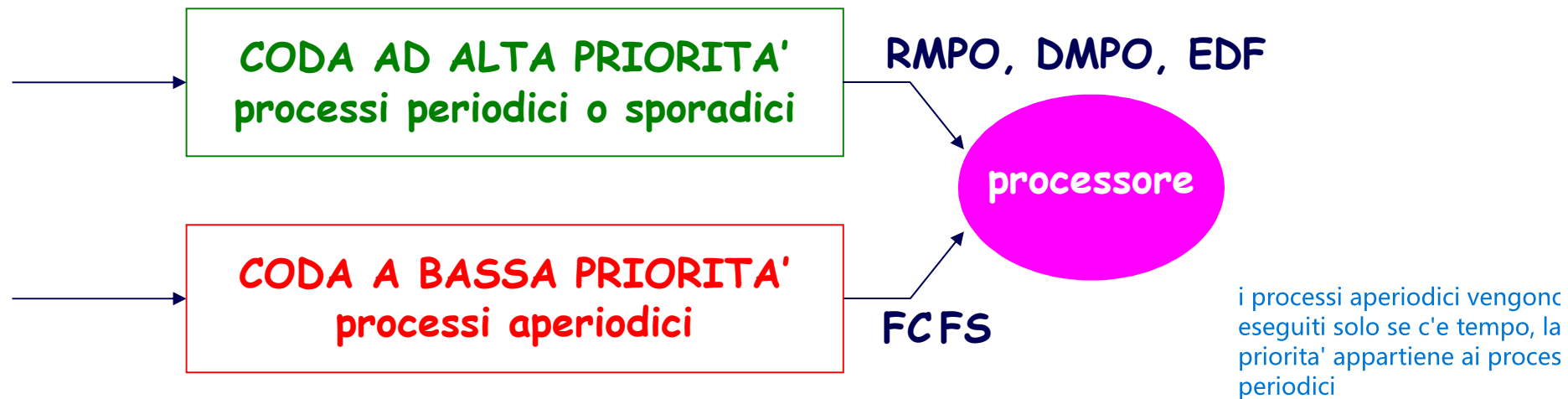
- constant utilization server
- total bandwidth server
- ...

- ◆ a priorità dinamica

- ...

# SERVIZIO IN BACKGROUND ...

Le richieste aperiodiche, caratterizzate da vincoli temporali di tipo soft o non real-time, vengono servite solo se non vi sono processi hard real-time (periodici e/o sporadici) pronti per l'esecuzione.



La strategia di schedulazione dei processi aperiodici (tipicamente FCFS) è indipendente da quella selezionata per gli altri processi (tipicamente RMPO, DMPO o EDF).

La schedulabilità dei processi hard real-time non è influenzata dal servizio dei processi aperiodici.

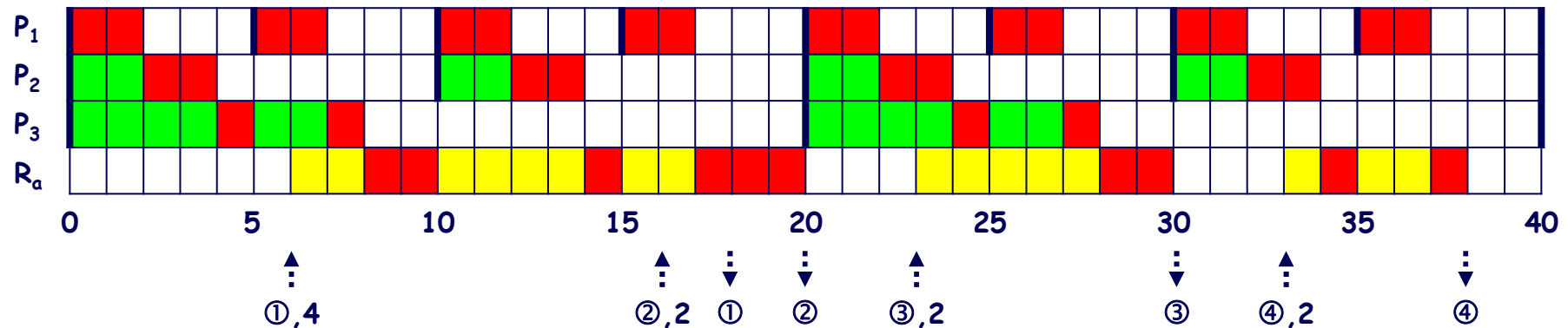
## ... SERVIZIO IN BACKGROUND ...

$A_1$	$C$ [ms]	$T$ [ms]
$P_1$	2	5
$P_2$	2	10
$P_3$	2	20

strategia di schedulazione: RMPO

	$a$ [ms]	$s$ [ms]
$R_{a1}$	6	4
$R_{a2}$	16	2
$R_{a3}$	23	2
$R_{a4}$	33	2

arrival time  $\uparrow$  service time  $\uparrow$



$P_i$	Idle	Ready	Running
$R_a$	None	Pending	Served

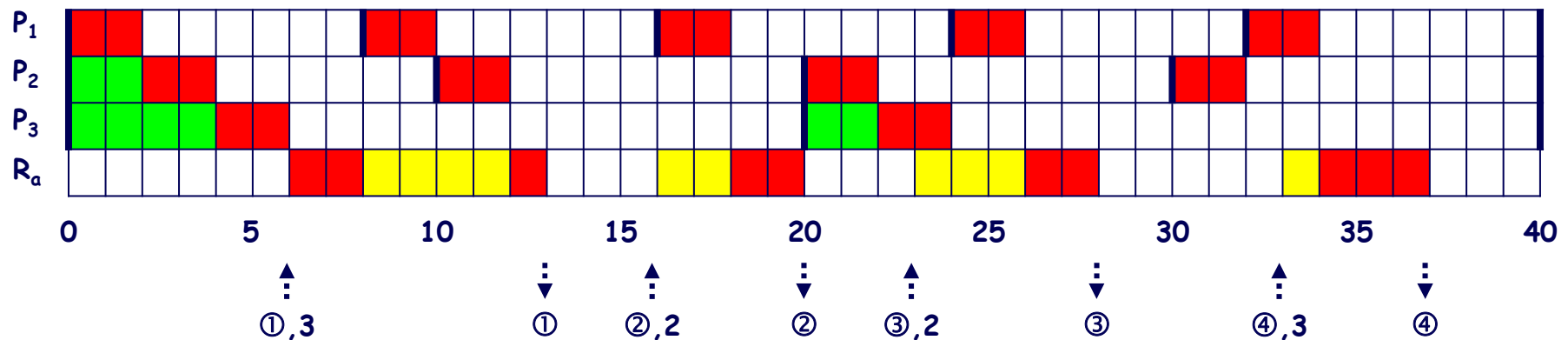
I tempi di risposta delle richieste aperiodiche possono risultare notevoli, soprattutto se il fattore di utilizzazione del processore da parte dei processi hard real-time è elevato.

# ... SERVIZIO IN BACKGROUND

$A_2$	$C$ [ms]	$T$ [ms]
$P_1$	2	8
$P_2$	2	10
$P_3$	2	20

strategia di schedulazione: RMPO

	$a$ [ms]	$s$ [ms]
$R_{a1}$	6	3
$R_{a2}$	16	2
$R_{a3}$	23	2
$R_{a4}$	33	3



$P_i$	Idle	Ready	Running
$R_a$	None	Pending	Served

# CONSIDERAZIONI REALIZZATIVE

Table	a [ms]	s [ms]
$R_{a1}$	$a_1$	$s_1$
...	...	...
$R_{aM}$	$a_M$	$s_M$

Un ambiente di simulazione:

i parametri temporali (tempo di arrivo e tempo di servizio) di un numero arbitrario  $M$  di richieste aperiodiche sono memorizzati in una tabella;

un processo  $P_g$  simula la generazione di ciascuna richiesta scrivendo in una FIFO, in corrispondenza dell'istante di arrivo specificato in tabella, il relativo tempo di servizio;



un processo  $P_e$  simula l'esecuzione del servizio di ciascuna richiesta impegnando la CPU per un tempo pari al valore corrispondentemente letto dalla FIFO.

$P_g$

```

Entry = TableFirstEntry;
LastEntryArrivalTime = 0;
while (Entry <= TableLastEntry)
{
    TaskDelay (Table[Entry]->ArrivalTime - LastEntryArrivalTime);
    put_fifo (Table[Entry]->ServiceTime);
    LastEntryArrivalTime = Table[Entry]->ArrivalTime;
    Entry++;
}
  
```

$P_e$

```

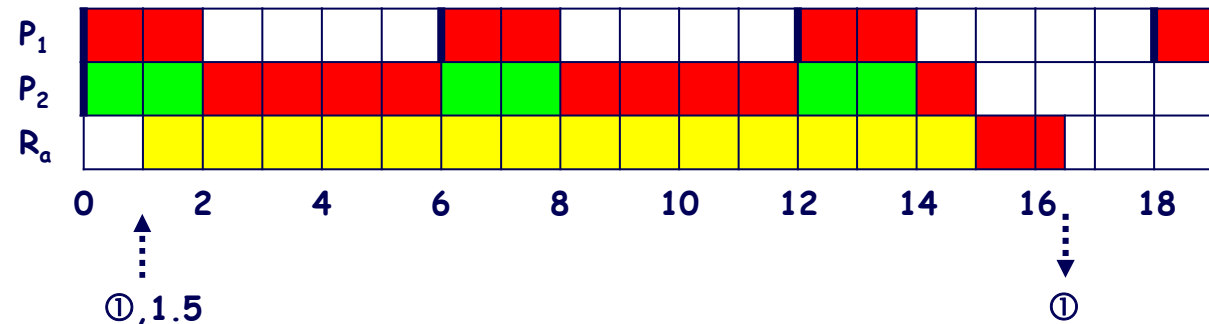
while (true)
{
    while (fifo_empty ( ));
    ServiceTime = get_fifo ( );
    BusyWait (ServiceTime);
}
  
```

# RISULTATI SPERIMENTALI

$A_3$	$C$ [ms]	$T$ [ms]
$P_1$	2	6
$P_2$	9	30

strategia di schedulazione: RMPO

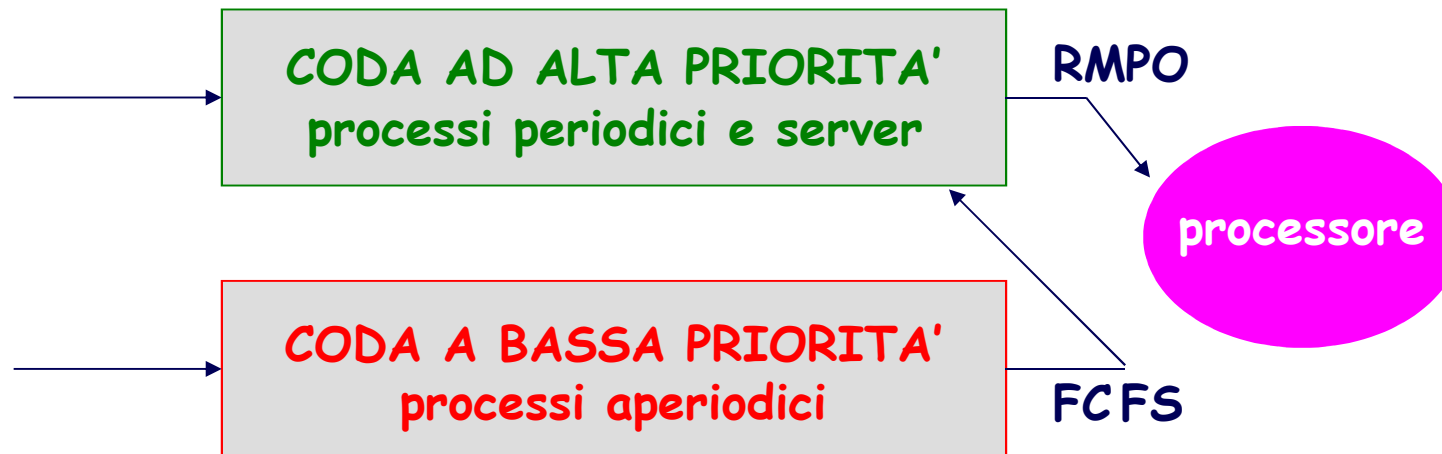
Table	$a$ [ms]	$s$ [ms]
$R_{a1}$	1	1.5



S.O.: RTAI Tool: Linux Trace Toolkit



# SERVIZIO TRAMITE SERVER A PRIORITÀ STATICA



Il server è un processo periodico ( $P_S$ ), con periodo  $T_S$  e tempo massimo di esecuzione  $C_S$  (capacità massima del server) prefissati.

$P_S$  è schedulato secondo la stessa strategia (tipicamente, e nel seguito, RMPO) utilizzata per i processi periodici, in base alla priorità  $p(P_S) \propto 1/T_S$ , di norma elevata, che gli compete.

I tempi di risposta dei processi aperiodici, mediamente sensibilmente inferiori rispetto a quelli derivanti dal servizio in background, dipendono dalle regole, peculiari di ciascuna tipologia di server, secondo cui è operato il riempimento ed il consumo della capacità del server stesso.



# SERVIZIO TRAMITE POLLING SERVER (PS)

Regole di riempimento e di consumo della capacità di  $P_S$

La capacità è ripristinata al valore  $C_S$  all'inizio di ogni periodo  $T_S$ .

La capacità è progressivamente consumata durante il servizio di richieste aperiodiche.

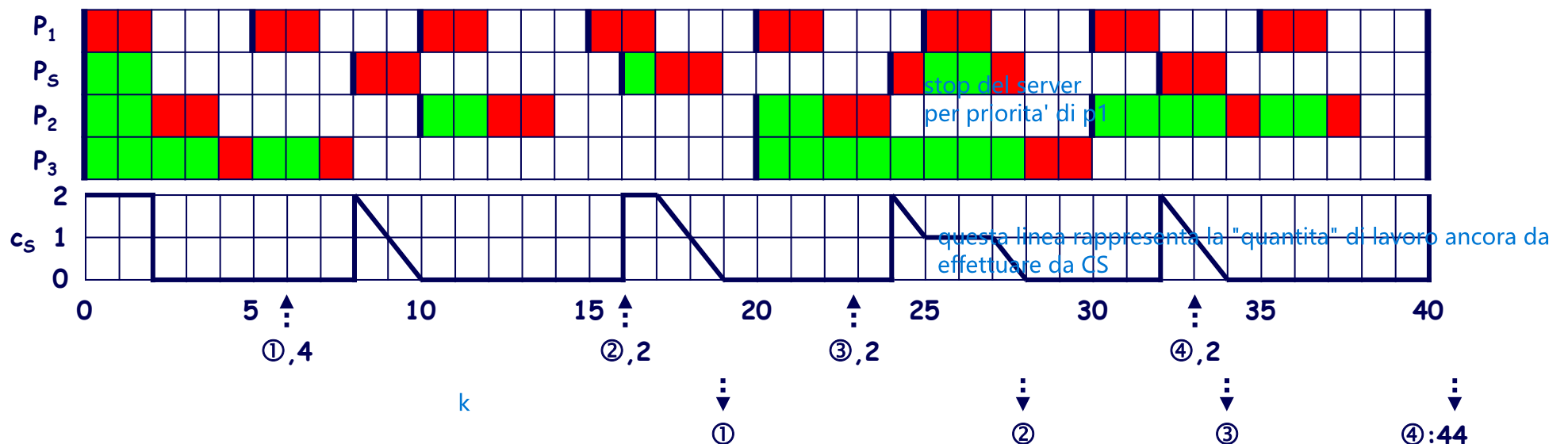
In assenza di (ulteriori) richieste, la capacità (residua) disponibile è scartata.

$A_1$	$C$ [ms]	$T$ [ms]
$P_1$	2	5
$P_2$	2	10
$P_3$	2	20

$P_S : T_S = 8 \text{ ms}, C_S = 2 \text{ ms}$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	6	4
$R_{a2}$	16	2
$R_{a3}$	23	2
$R_{a4}$	33	2

Idle
Ready
Running



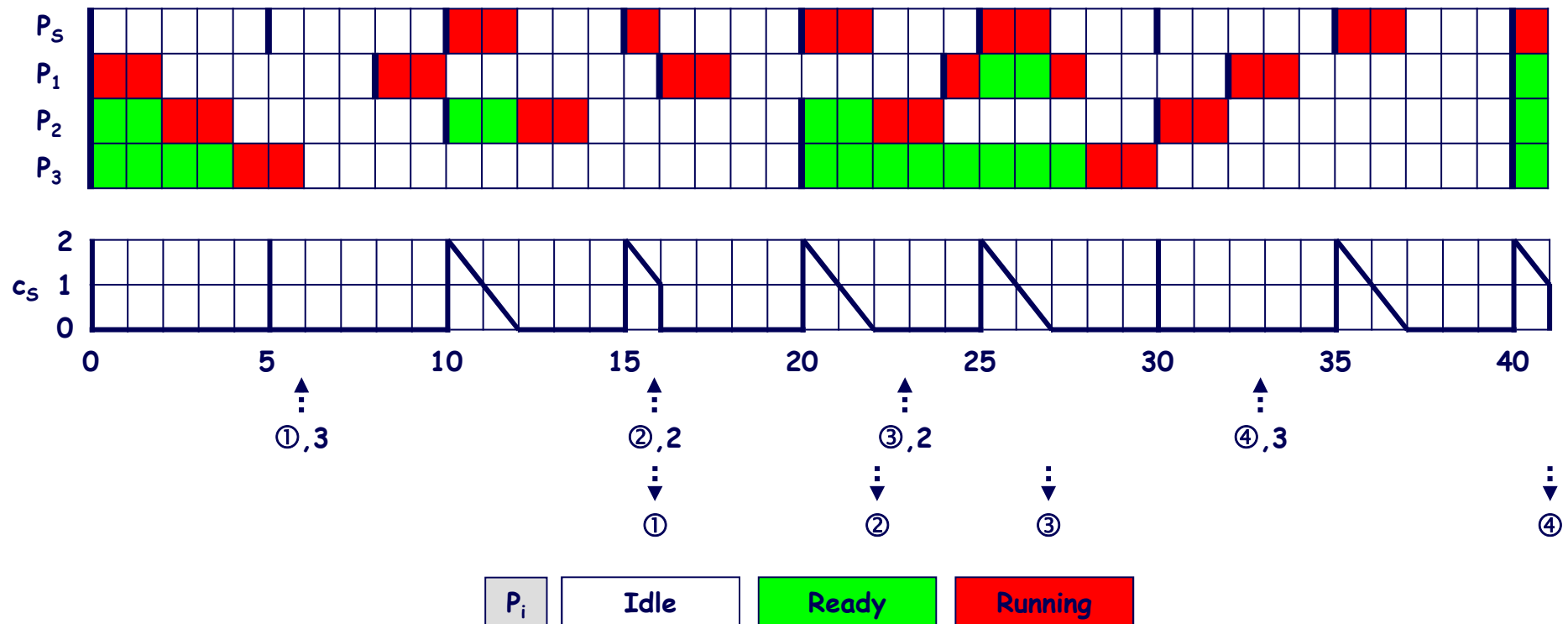
# ... SERVIZIO TRAMITE PS ...

Server a massima priorità:

$A_2$	$C$ [ms]	$T$ [ms]
$P_1$	2	8
$P_2$	2	10
$P_3$	2	20

$P_S : T_S = 5 \text{ ms}, C_S = 2 \text{ ms}$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	6	3
$R_{a2}$	16	2
$R_{a3}$	23	2
$R_{a4}$	33	3



## ... SERVIZIO TRAMITE PS ...

Garanzia di schedulabilità dei processi periodici:

La schedulabilità dei processi periodici è influenzata dal servizio delle richieste aperiodiche.

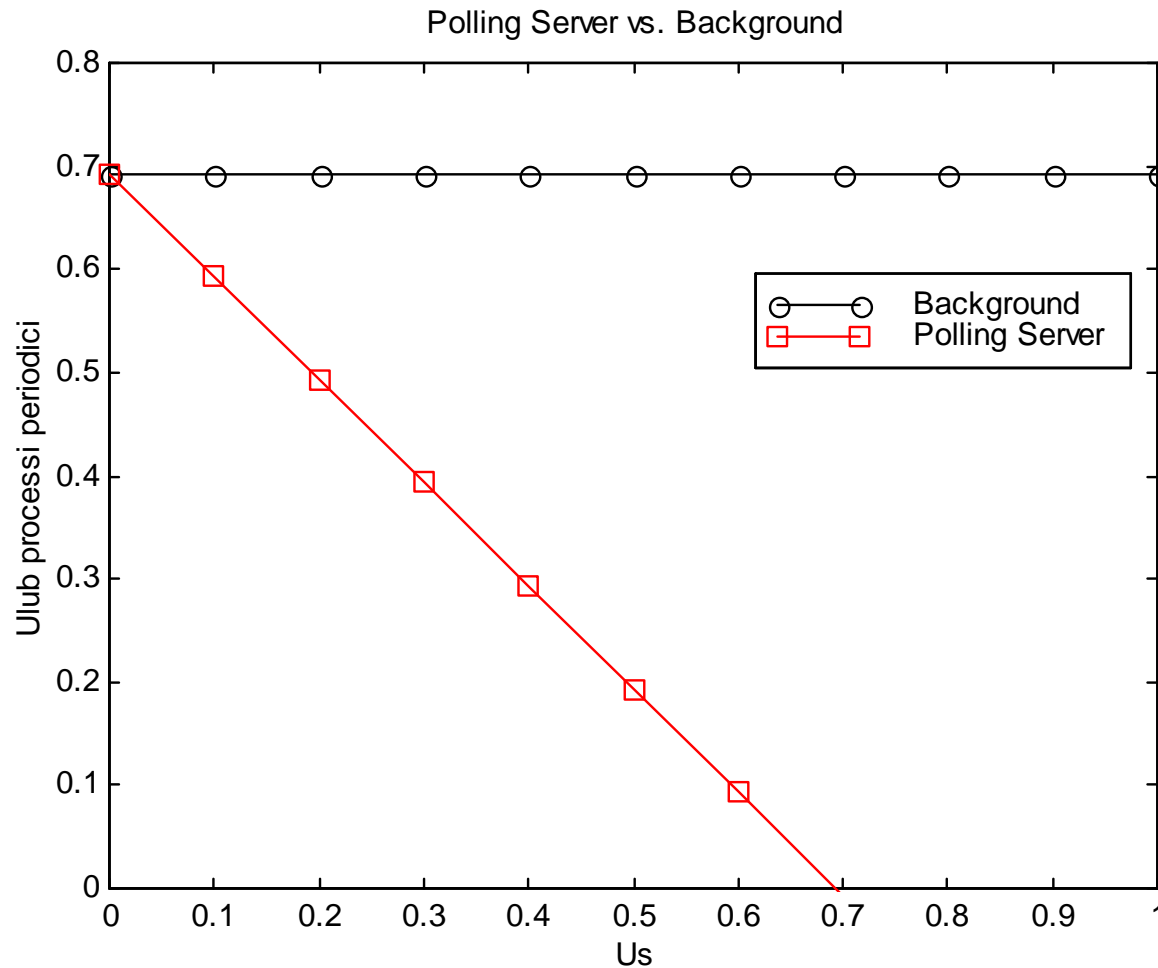
Il massimo carico computazionale indotto da PS è equivalente a quello derivante dall'esecuzione di un processo periodico con fattore di utilizzazione del processore  $U_s = C_s/T_s$ .

La schedulabilità di un insieme di  $N$  processi periodici contraddistinti da un fattore di utilizzazione del processore  $U_p$  è pertanto garantita (condizione sufficiente, ma non necessaria) se:

$$U_p + U_s \leq U_{\text{RMPO}} (N + 1) = (N + 1) \left( 2^{1/(N+1)} - 1 \right)$$

## ... SERVIZIO TRAMITE PS ...

$$U_s \leq (N+1)(2^{1/(N+1)} - 1) - U_p \underset{N \rightarrow \infty}{=} \ln 2 - U_p$$



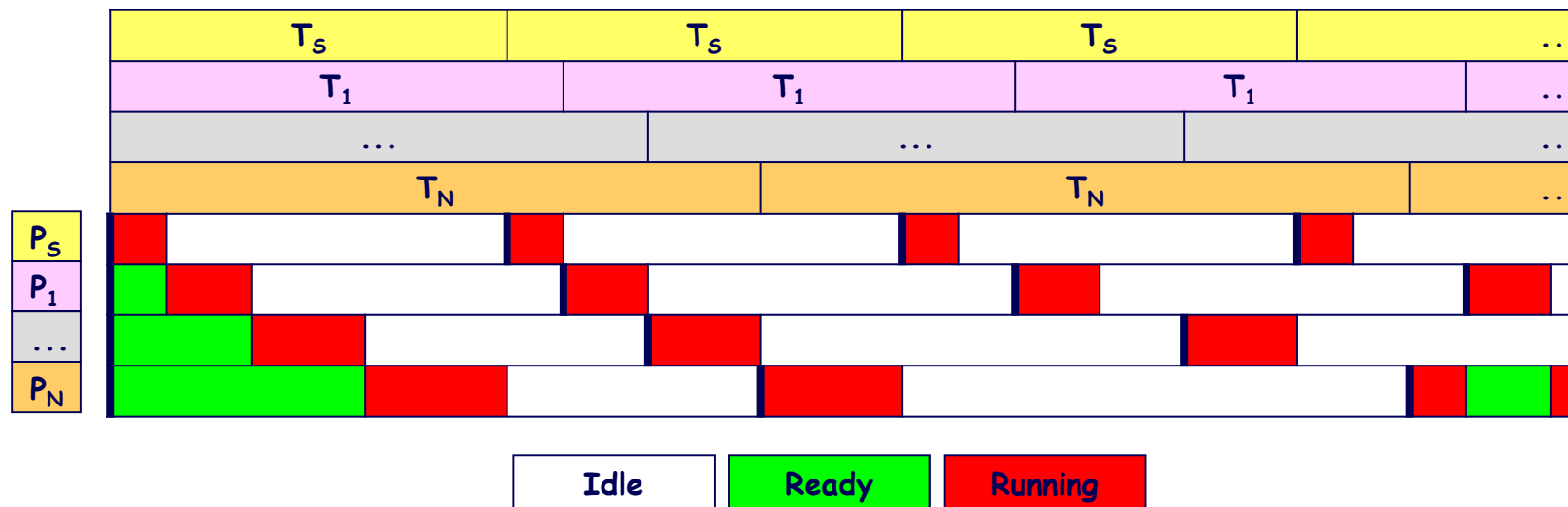
## ... SERVIZIO TRAMITE PS ...

Garanzia di schedulabilità con server a massima priorità:

$$C_s = T_1 - T_s$$

$$C_1 = T_2 - T_1 \quad \dots \quad C_{N-1} = T_N - T_{N-1}$$

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



## ... SERVIZIO TRAMITE PS ...

$$R_j = \frac{T_{j+1}}{T_j} \quad j = 1, 2, \dots, N-1$$

$$K = \frac{2 T_s}{T_1} = \frac{2}{U_s + 1}$$

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

$$\frac{\partial U_{p \text{ ub}}}{\partial R_j} = 0 \quad \forall j$$

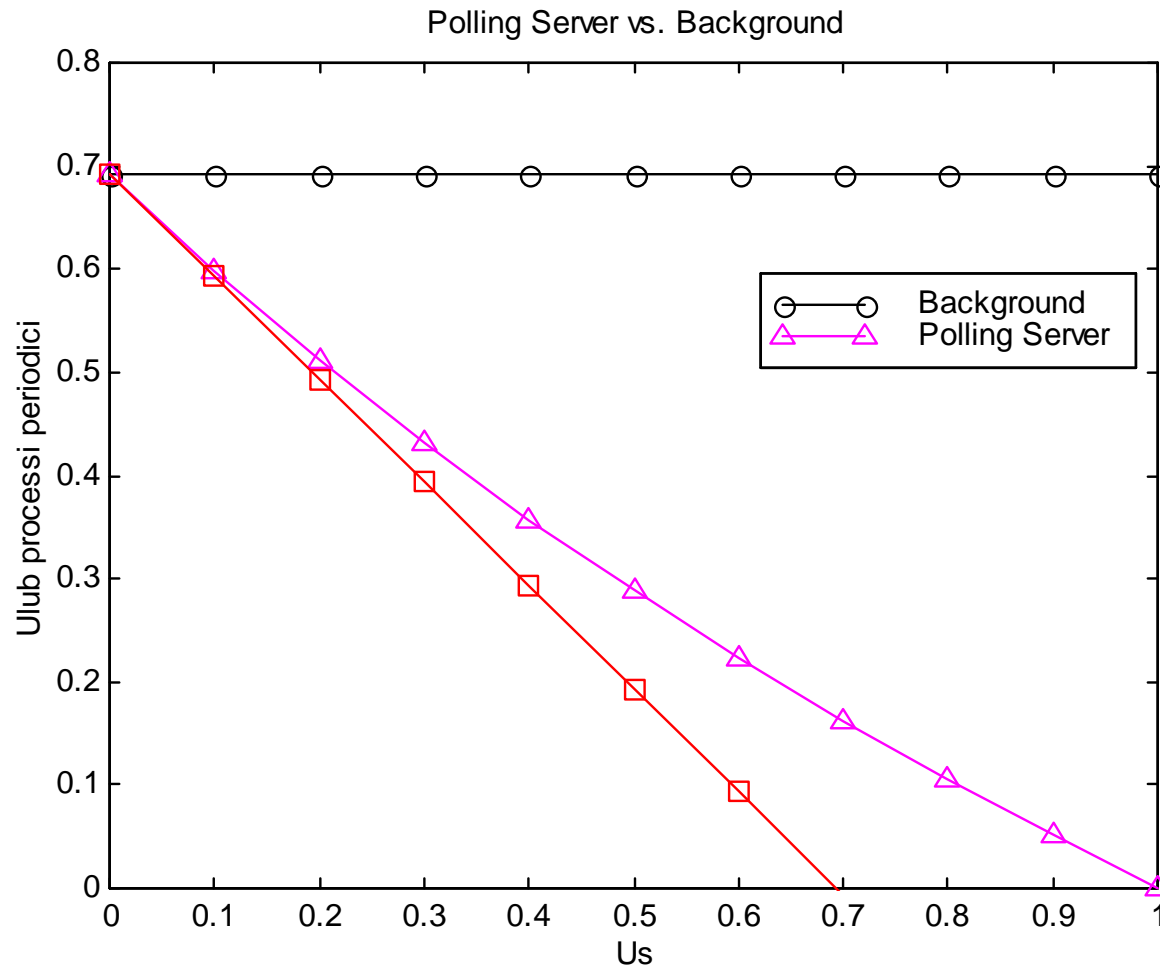
$$R_j = K^{1/N} \quad \forall j$$

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

$$U_{p \text{ lub}} = N \left( \left( \frac{2}{U_s + 1} \right)^{1/N} - 1 \right) \xrightarrow{N \rightarrow \infty} \ln \frac{2}{U_s + 1}$$

## ... SERVIZIO TRAMITE PS

$$U_s \leq \frac{2}{\left(1 + U_p/N\right)^N} - 1 \underset{N \rightarrow \infty}{=} \frac{2}{e^{U_p}} - 1$$



# CONSIDERAZIONI REALIZZATIVE

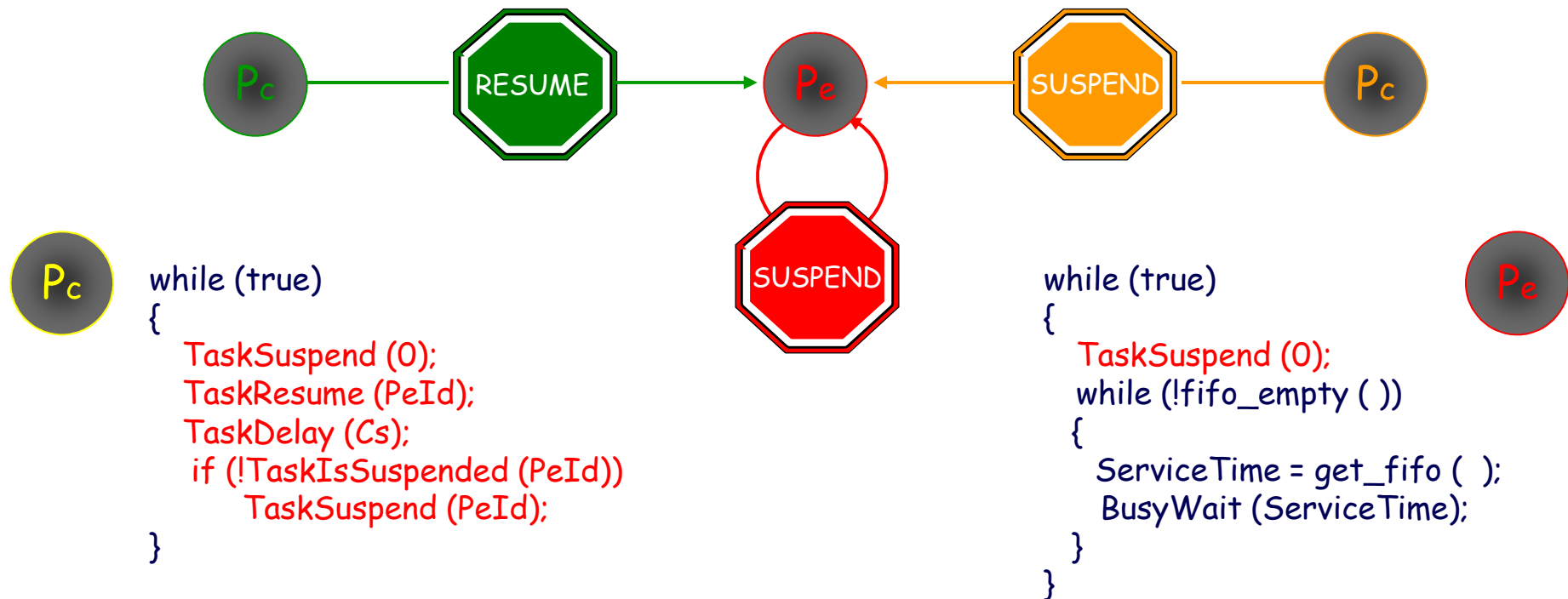
## Server a massima priorità

2 processi ( $P_c$ ,  $P_e$ ) che operano in sinergia:

$P_c$ , periodico con periodo  $T_s$ , (ripristina la capacità al valore massimo  $C_s$  ed) attiva l'esecuzione di  $P_e$

$P_e$  serve le eventuali richieste pendenti e si sospende (scartando così la capacità residua)

$P_c$ , se necessario, sospende  $P_e$  dopo un tempo pari a  $C_s$





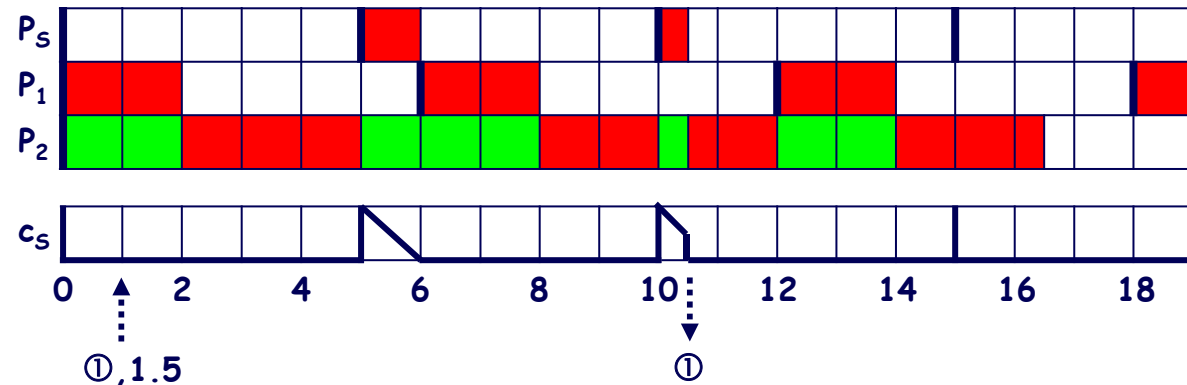
# RISULTATI SPERIMENTALI

$A_3$	$C$ [ms]	$T$ [ms]
$P_1$	2	6
$P_2$	9	30

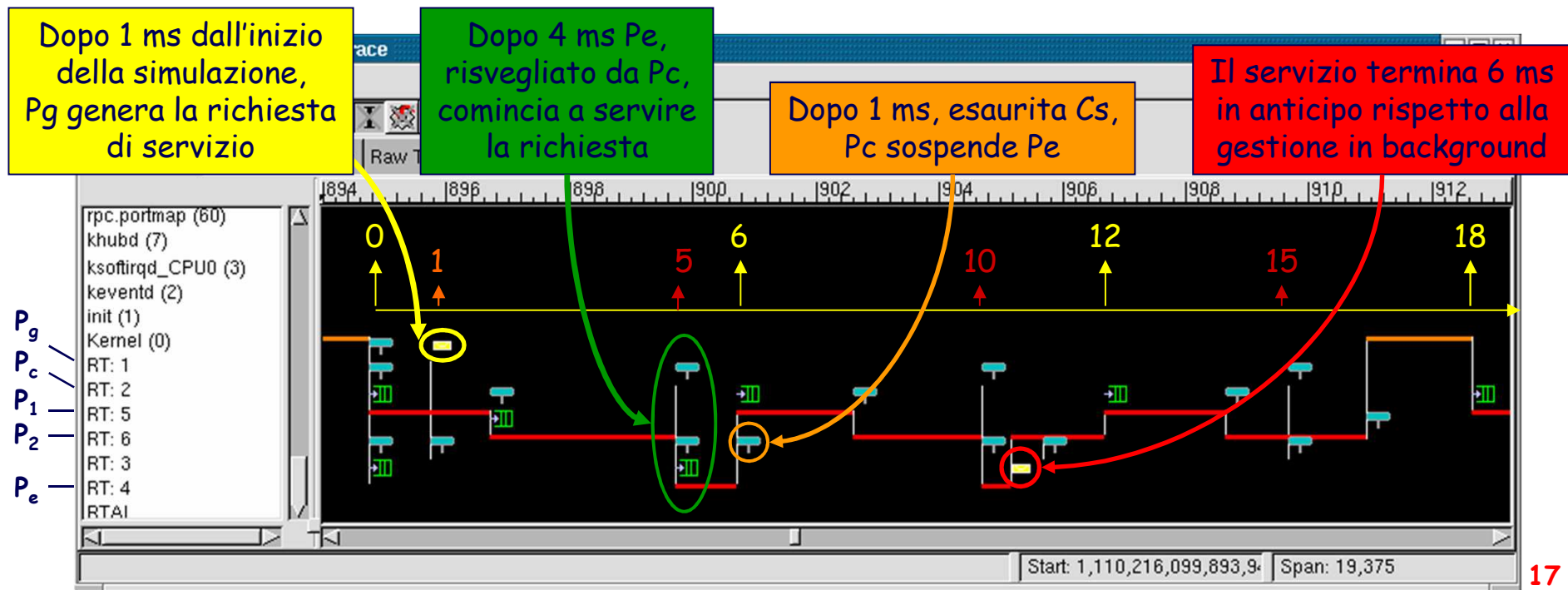
$P_S: T_S = 5 \text{ ms}, C_S = 1 \text{ ms}$

strategia di schedulazione: RMPO

Table	$a$ [ms]	$s$ [ms]
$R_{a1}$	1	1.5



S.O.: RTAI Tool: Linux Trace Toolkit



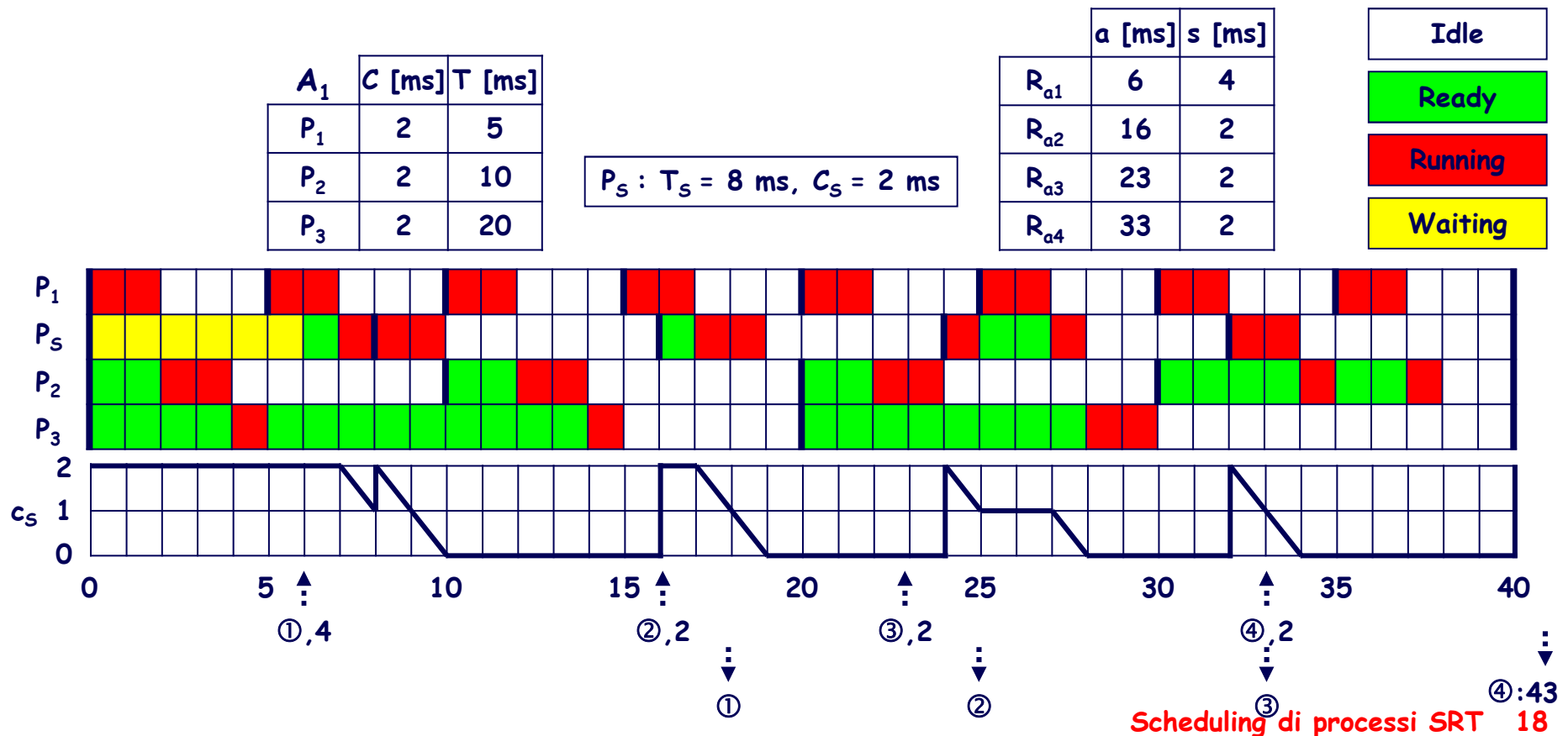
# SERVIZIO TRAMITE DEFERRABLE SERVER (DS) [Lehoczky, Sha, Strosnider (87)] ...

Regole di riempimento e di consumo della capacità di  $P_S$

La capacità viene ripristinata al valore  $C_S$  all'inizio di ogni periodo  $T_S$ .

La capacità disponibile è conservata in assenza di richieste aperiodiche pendenti:  
il servizio può pertanto essere differito (bandwidth-preserving server).

La capacità è progressivamente consumata durante il servizio di richieste aperiodiche.



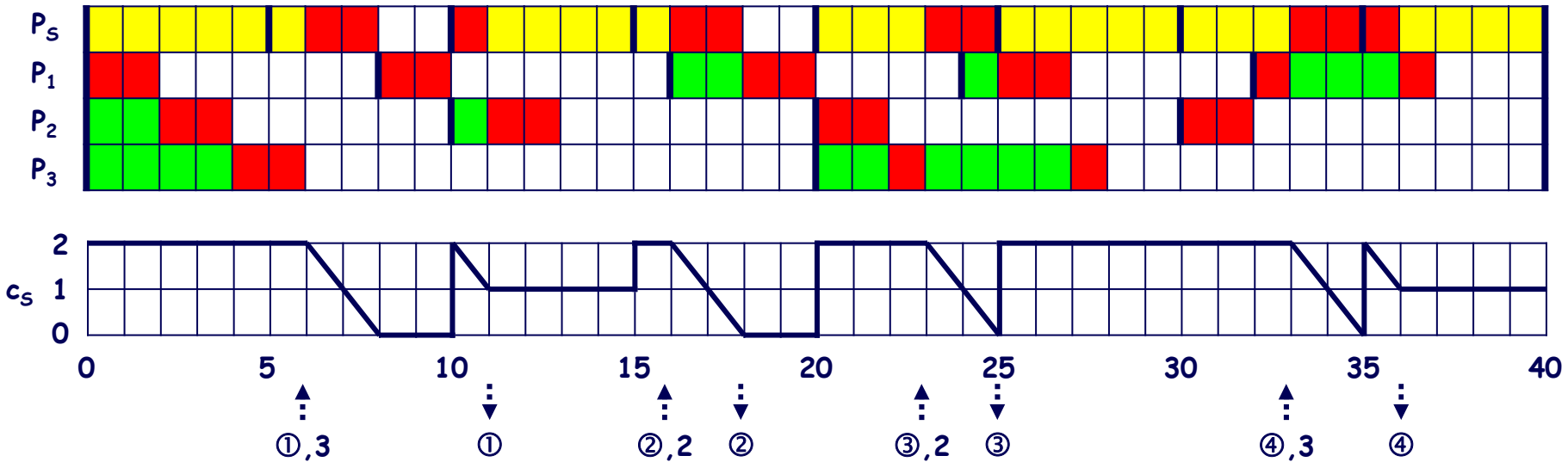
... SERVIZIO TRAMITE DS ...

## Server a massima priorità:

$A_2$	C [ms]	T [ms]
$P_1$	2	8
$P_2$	2	10
$P_3$	2	20

$P_S : T_S = 5 \text{ ms}, C_S = 2 \text{ ms}$

	$\alpha$ [ms]	$s$ [ms]
$R_{a1}$	6	3
$R_{a2}$	16	2
$R_{a3}$	23	2
$R_{a4}$	33	3

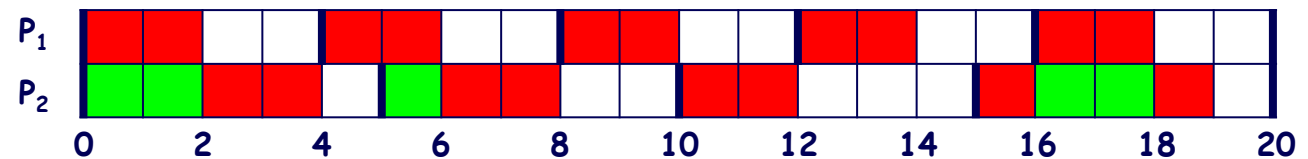


# ... SERVIZIO TRAMITE DS ...

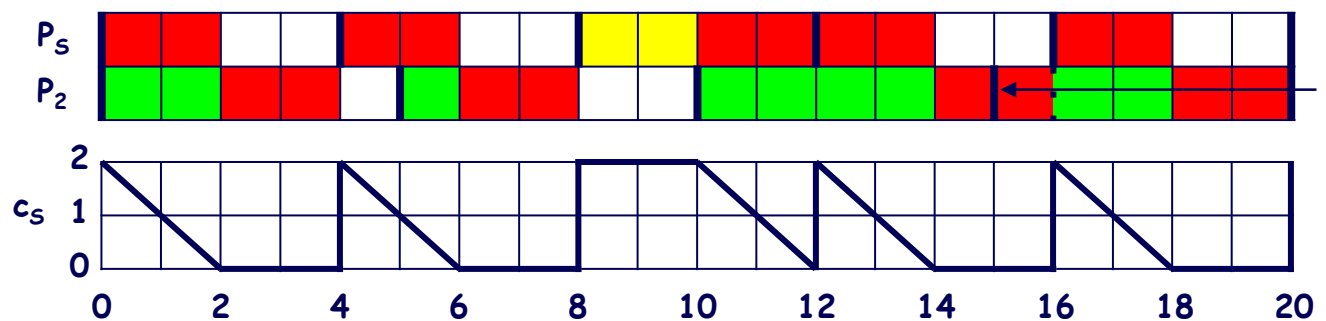
Garanzia di schedulabilità dei processi periodici:

Il contributo al fattore di utilizzazione del processore imputabile a DS è, in condizioni di massimo carico, maggiore di quello derivante dall'esecuzione di un processo periodico con periodo  $T_s$  e tempo di esecuzione  $C_s$ , ovvero  $U_s > C_s/T_s$ . Infatti:

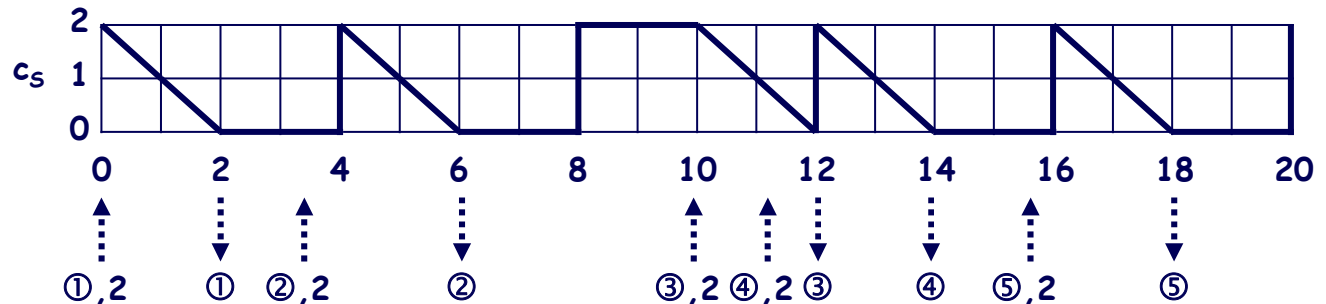
$A_4$	$C$ [ms]	$T$ [ms]
$P_1$	2	4
$P_2$	2	5



$A_4'$	$C$ [ms]	$T$ [ms]
$P_s$	2	4
$P_2$	2	5



	$a$ [ms]	$s$ [ms]
$R_{a1}$	0	2
$R_{a2}$	3.5	2
$R_{a3}$	10	2
$R_{a4}$	11.2	2
$R_{a5}$	15.5	2



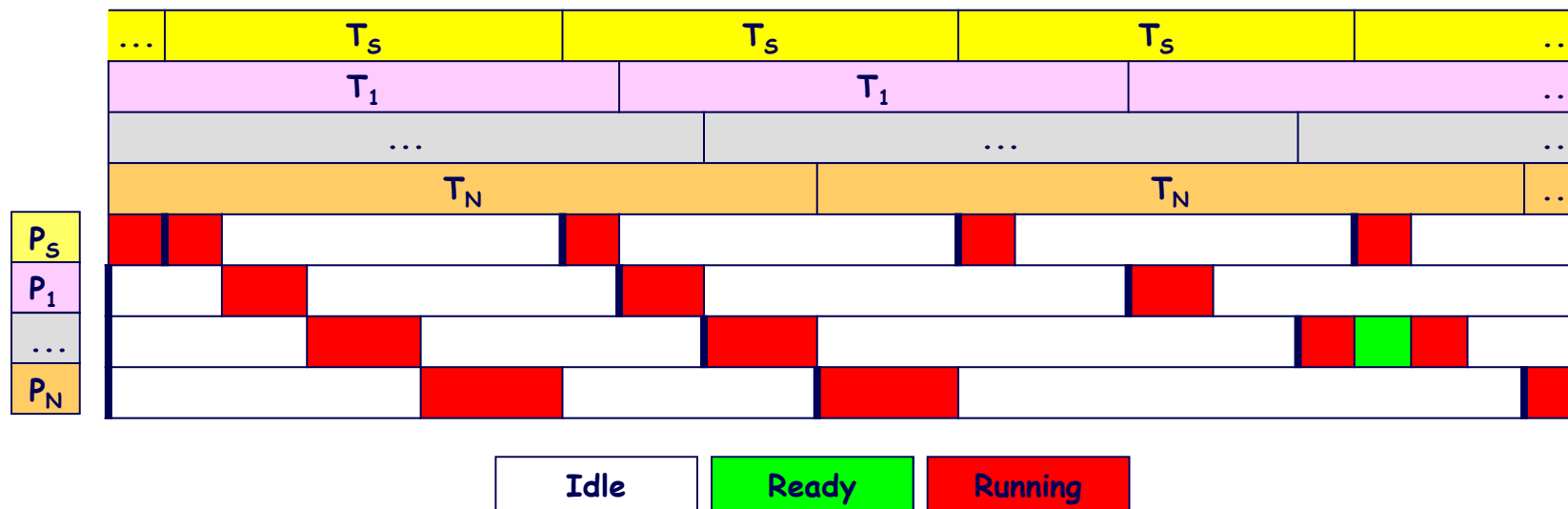
# ... SERVIZIO TRAMITE DS ...

Garanzia di schedulabilità con server a massima priorità:

$$C_s = T_1 - (T_s + C_s) = \frac{T_1 - T_s}{2}$$

$$C_1 = T_2 - T_1 \quad \dots \quad C_{N-1} = T_N - T_{N-1}$$

$$C_N \leq T_s - C_s - \sum_{j=1}^{N-1} C_j = C_s + 2T_s - T_N = \frac{3T_s + T_1}{2} - T_N = C_{Nub}$$



## ... SERVIZIO TRAMITE DS ...

$$R_j = \frac{T_{j+1}}{T_j} \quad j = 1, 2, \dots, N-1$$

$$K = \frac{3 T_s / T_1 + 1}{2} = \frac{U_s + 2}{2 U_s + 1}$$

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

$$\frac{\partial U_{p \text{ ub}}}{\partial R_j} = 0 \quad \forall j$$

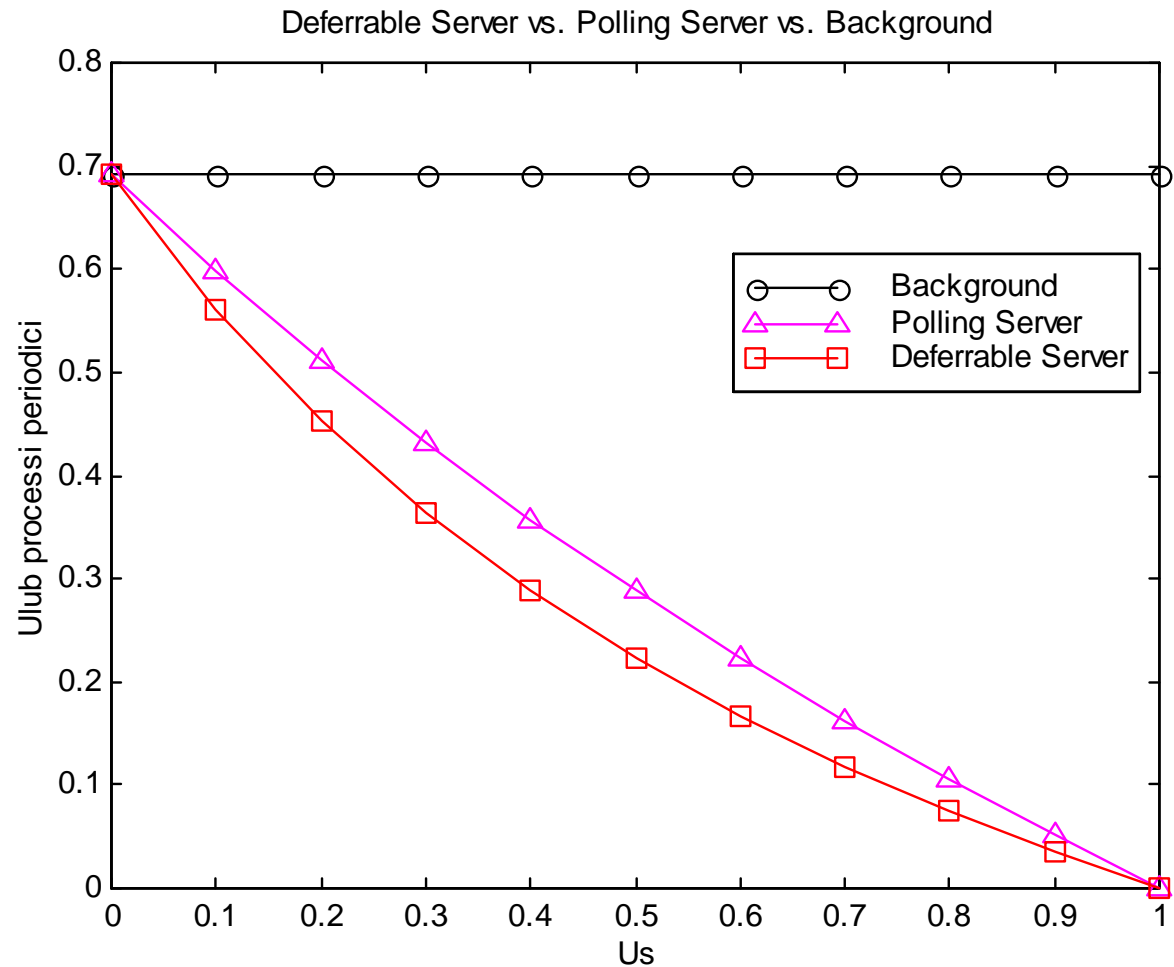
$$R_j = K^{1/N} \quad \forall j$$

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

$$U_{p \text{ lub}} = N \left( \left( \frac{U_s + 2}{2U_s + 1} \right)^{1/N} - 1 \right) \xrightarrow{N \rightarrow \infty} \ln \frac{U_s + 2}{2U_s + 1}$$

## ... SERVIZIO TRAMITE DS

$$U_s \leq \frac{2 - (1 + U_p/N)^N}{2(1 + U_p/N)^N - 1} \stackrel{N \rightarrow \infty}{=} \frac{2 - e^{U_p}}{2e^{U_p} - 1}$$



# SERVIZIO TRAMITE PRIORITY EXCHANGE SERVER (PES) [Lehoczky, Sha, Strosnider (87)] ...

Regole di riempimento e di consumo della capacità di  $P_S$

La capacità può essere accumulata, oltre che al livello di priorità  $p(P_S)$  proprio di  $P_S$ , al livello di priorità che compete a ciascun processo periodico  $P_j$  di priorità  $p(P_j) < p(P_S)$ .

La capacità al livello di priorità  $p(P_S)$  è ripristinata al valore  $C_S$  all'inizio di ogni periodo  $T_S$ .

La capacità disponibile ad un qualunque livello di priorità è conservata durante l'esecuzione di un processo periodico di priorità non inferiore.

La capacità disponibile al massimo livello di priorità è progressivamente consumata sia durante il servizio di richieste aperiodiche, sia in assenza di richieste aperiodiche qualora non vi siano processi periodici di priorità inferiore pronti per l'esecuzione.

La capacità disponibile al massimo livello di priorità, in assenza di richieste aperiodiche ed in presenza di uno o più processi periodici di priorità inferiore pronti per l'esecuzione, è progressivamente trasferita durante l'esecuzione di ciascun processo al corrispondente livello di priorità.

In presenza di richieste aperiodiche,  $P_S$ , se dispone di capacità ad un qualche livello di priorità, ha la precedenza rispetto ad un processo periodico di pari priorità.

Da tali regole discende che il servizio delle richieste aperiodiche, ancorché talora con priorità  $< p(P_S)$ , può essere differito (bandwidth-preserving server).



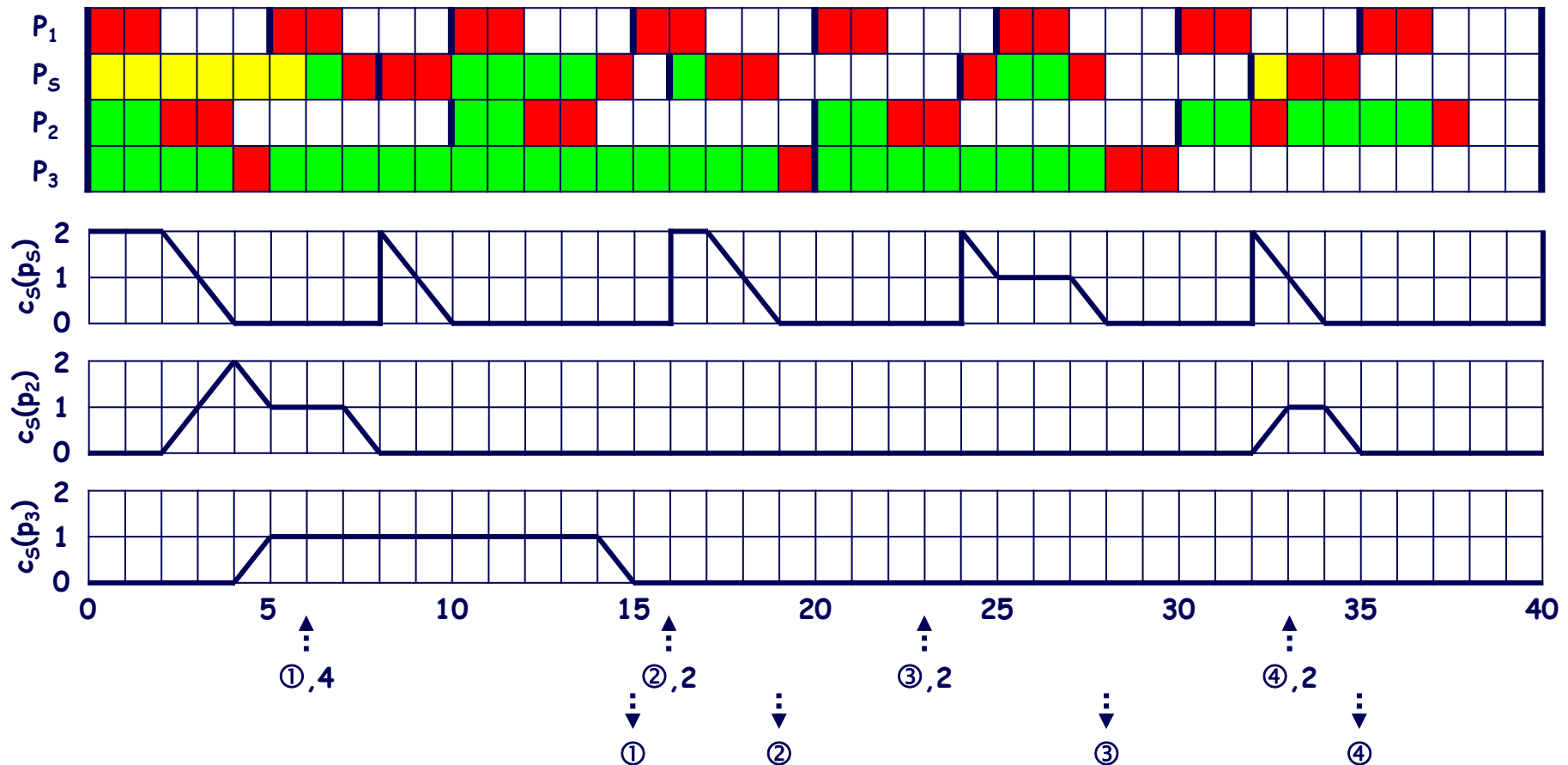
# ... SERVIZIO TRAMITE PES ...

$A_1$	$C$ [ms]	$T$ [ms]
$P_1$	2	5
$P_2$	2	10
$P_3$	2	20

$P_S : T_S = 8 \text{ ms}, C_S = 2 \text{ ms}$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	6	4
$R_{a2}$	16	2
$R_{a3}$	23	2
$R_{a4}$	33	2

Idle
Ready
Running
Waiting



# ... SERVIZIO TRAMITE PES ...

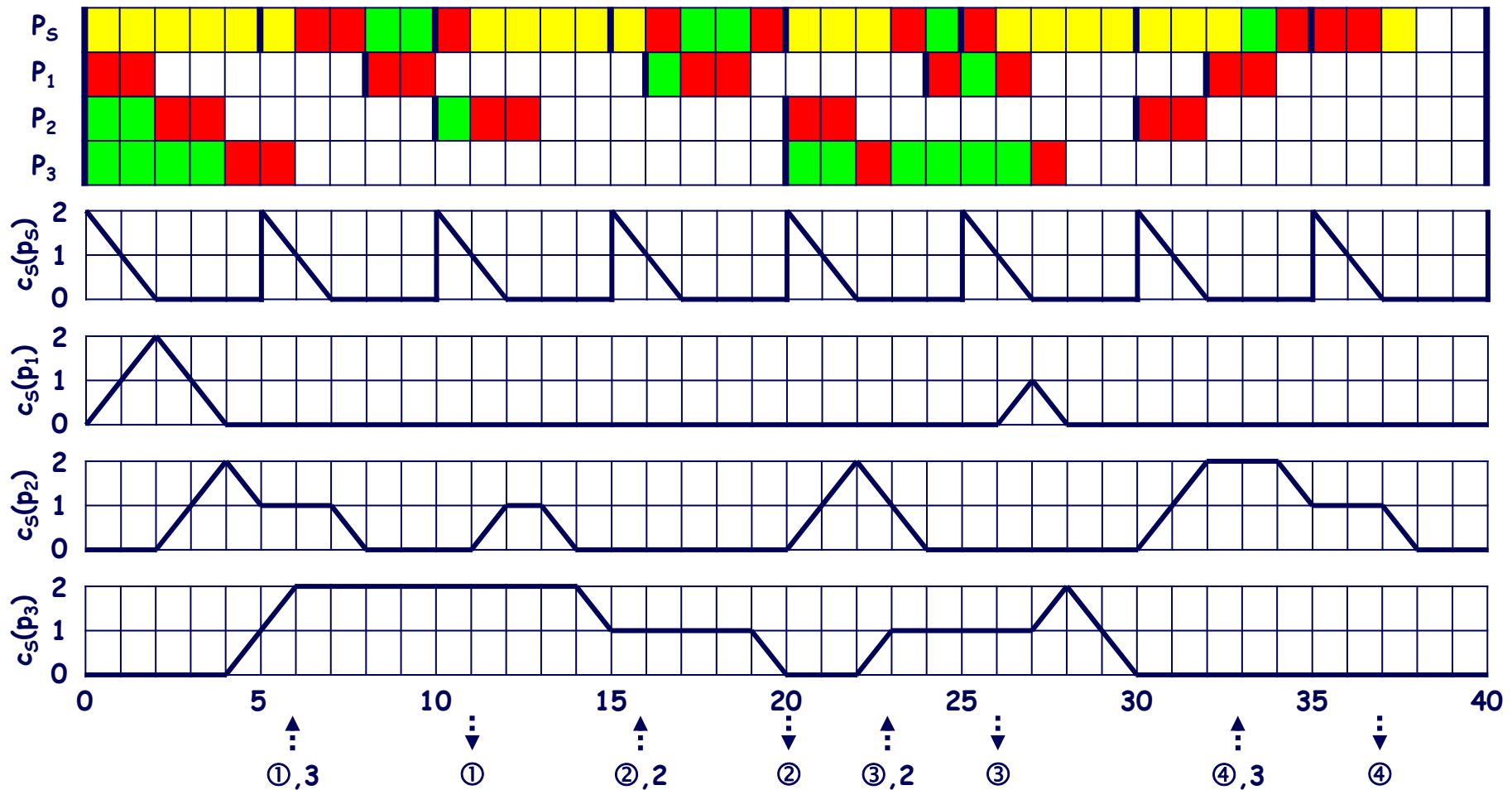
Server a massima priorità:

$A_2$	$C$ [ms]	$T$ [ms]
$P_1$	2	8
$P_2$	2	10
$P_3$	2	20

$P_S : T_S = 5 \text{ ms}, C_S = 2 \text{ ms}$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	6	3
$R_{a2}$	16	2
$R_{a3}$	23	2
$R_{a4}$	33	3

Idle
Ready
Running
Waiting



## ... SERVIZIO TRAMITE PES

Il massimo carico computazionale indotto da PES  
è equivalente a quello derivante dall'esecuzione di un processo periodico  
con fattore di utilizzazione del processore  $U_s = C_s/T_s$ .

La schedulabilità di un insieme di  $N$  processi periodici  
contraddistinti da un fattore di utilizzazione del processore  $U_p$   
è pertanto garantita (condizione sufficiente, ma non necessaria) se:

$$U_s \leq (N+1) \left( 2^{1/(N+1)} - 1 \right) - U_p \underset{N \rightarrow \infty}{=} \ln 2 - U_p$$

o, nel caso PES abbia massima priorità, se:

$$U_s \leq \frac{2}{\left(1 + U_p/N\right)^N} - 1 \underset{N \rightarrow \infty}{=} \frac{2}{e^{U_p}} - 1$$

# SERVIZIO TRAMITE SPORADIC SERVER (SS) [Sprunt, Sha, Lehoczky (89)] ...

Regole di riempimento e di consumo della capacità di  $P_s$

La capacità disponibile è conservata in assenza di richieste aperiodiche pendenti:  
il servizio può pertanto essere differito (bandwidth-preserving server).

La capacità è progressivamente consumata soltanto  
durante il servizio di richieste aperiodiche.

La capacità viene reintegrata solo dopo essere stata consumata  
e nella misura in cui è stata effettivamente utilizzata.

Indicando con  $t_A$  l'istante in cui si verifica la condizione  $c_s > 0$  e  $P_s$  attivo  
( $P_s$  si dice attivo all'istante  $t$  se il processo in esecuzione ha priorità  $\geq p(P_s)$ ),  
e con  $t_D$  il successivo istante in cui si verifica la condizione  $c_s = 0$  o  $P_s$  non attivo,  
l'entità RA (replenishment amount) e l'istante RT (replenishment time)  
del successivo reintegro sono stabiliti come segue:

$RA = \text{capacità consumata in } [t_A, t_D],$

$RT = \max \{t_A + T_s, t_D\}.$

Il comportamento di  $P_s$  è (!?) equivalente a quello di una molteplicità di processi periodici  
aventi tutti lo stesso periodo  $T_s$  e tempo di esecuzione uguale complessivamente a  $C_s$ .

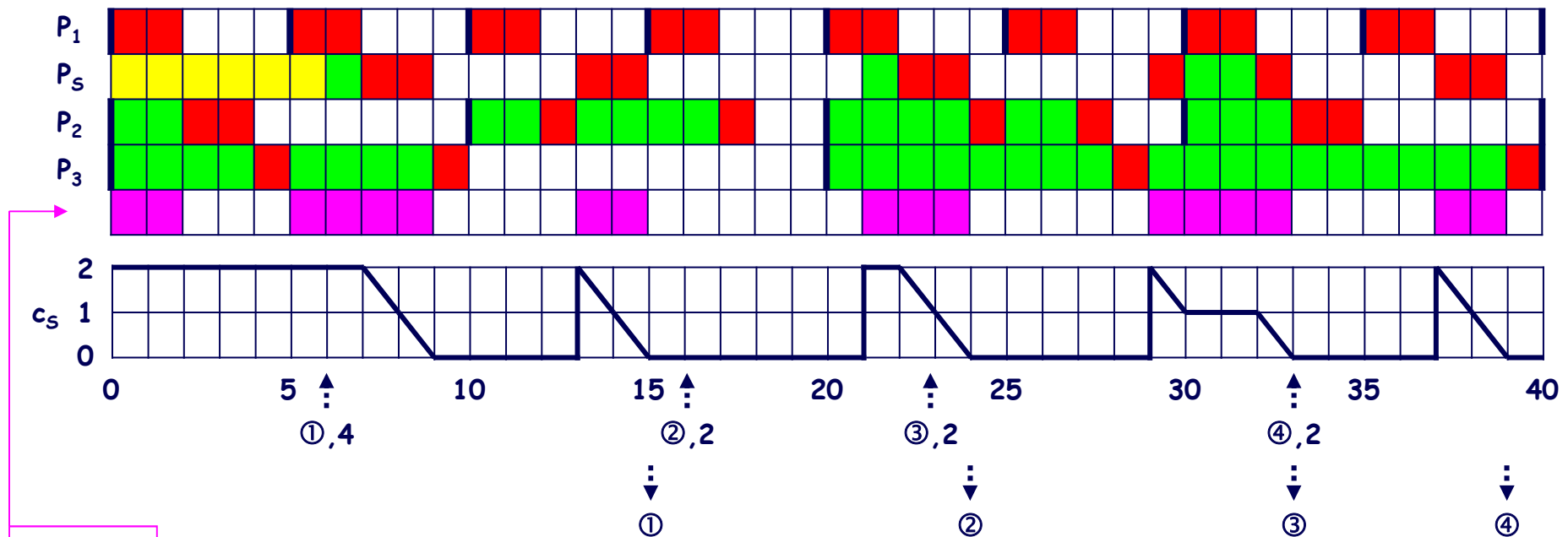
# ... SERVIZIO TRAMITE SS ...

$A_1$	$C$ [ms]	$T$ [ms]
$P_1$	2	5
$P_2$	2	10
$P_3$	2	20

$P_S : T_S = 8 \text{ ms}, C_S = 2 \text{ ms}$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	6	4
$R_{a2}$	16	2
$R_{a3}$	23	2
$R_{a4}$	33	2

Idle
Ready
Running
Waiting



$P_S$  attivo  
&  
 $c_S > 0$

$t_A$	0	5	13	21	29	37
$t_D$	2	9	15	24	33	39
RA	0	2	2	2	2	2
RT	-	13	21	29	37	45

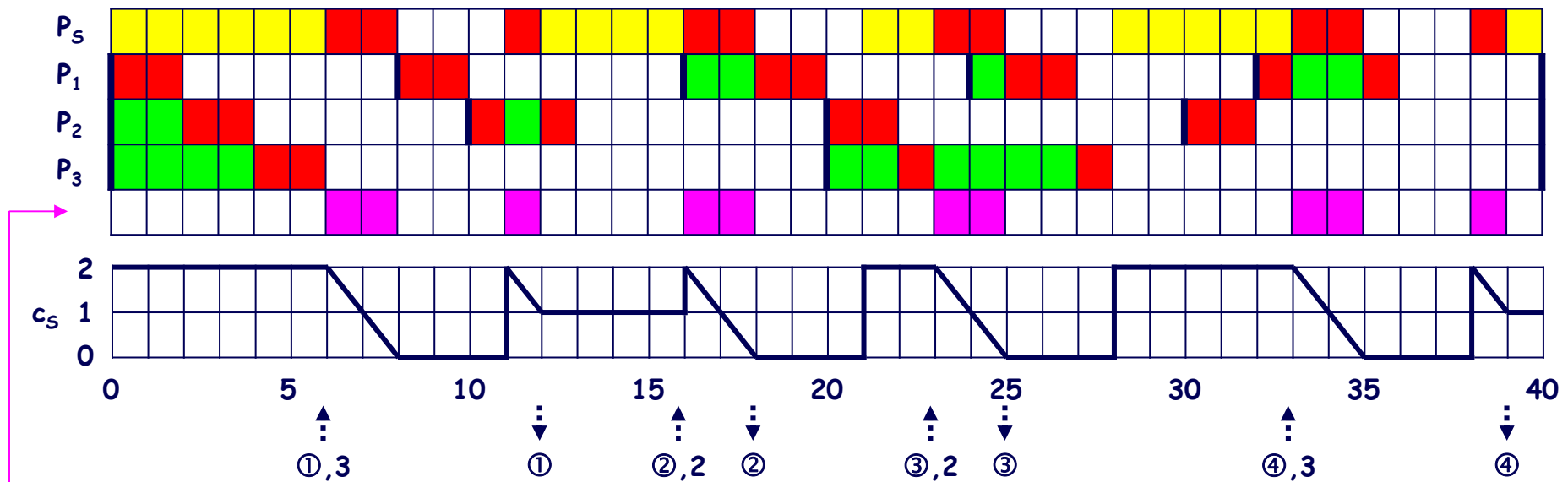
# ... SERVIZIO TRAMITE SS ...

Server a massima priorità:

$A_2$	$C$ [ms]	$T$ [ms]
$P_1$	2	8
$P_2$	2	10
$P_3$	2	20

$P_S : T_S = 5 \text{ ms}, C_S = 2 \text{ ms}$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	6	3
$R_{a2}$	16	2
$R_{a3}$	23	2
$R_{a4}$	33	3



$P_S$  attivo  
&  
 $c_S > 0$

$t_A$	6	11	16	23	33	38
$t_D$	8	12	18	25	35	39
RA	2	1	2	2	2	1
RT	11	16	21	28	38	43

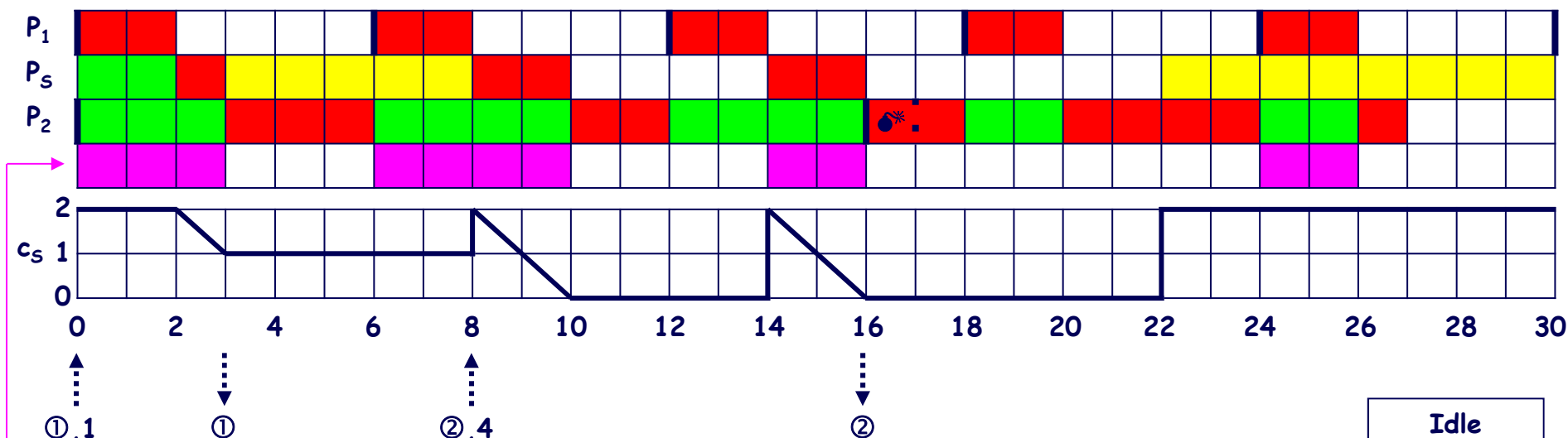
Idle
Ready
Running
Waiting

# ... SERVIZIO TRAMITE SS ...

$A_5$	$C$ [ms]	$T$ [ms]
$P_1$	2	6
$P_2$	6	16

$P_S : T_S = 8 \text{ ms}, C_S = 2 \text{ ms}$

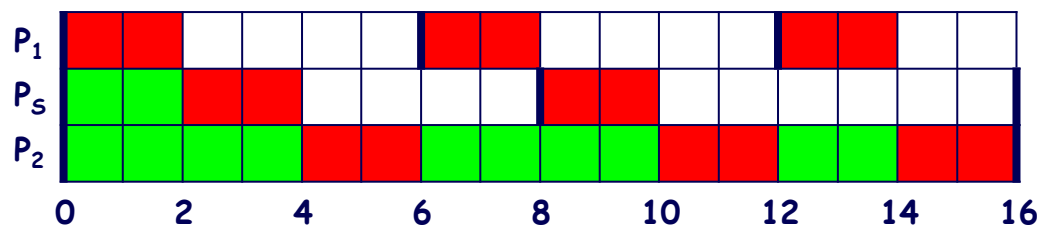
	$a$ [ms]	$s$ [ms]
$R_{a1}$	0	1
$R_{a2}$	8	4



$P_S$  attivo  
&  
 $c_S > 0$

$t_A$	0	6	14	24
$t_D$	3	10	16	26
RA	1	2	2	0
RT	8	14	22	-

Eppure, se  $P_S$  fosse un normale processo periodico, l'insieme dei 3 processi  $P_1, P_S, P_2$  risulterebbe schedulabile.



## ... SERVIZIO TRAMITE SPORADIC SERVER ...

... Regole di riempimento e di consumo della capacità di  $P_s$

Quando il carico derivante da richieste aperiodiche è tale da non comportare il totale consumo della capacità disponibile, le distinte porzioni (chunks) di capacità che ne derivano, quella residua non utilizzata e quella consumata ed in seguito reintegrata, devono essere gestite separatamente, tenendo traccia per ciascuna di esse del corrispondente tempo di reintegro  $RT$ .

Le regole di reintegro dei singoli chunks, di volta in volta utilizzati per il servizio delle richieste aperiodiche in ordine di  $RT$  crescente, sono pertanto generalizzate come segue:

$$t_E \text{ (effective replenishment time)} = \max \{RT, t_A\},$$

$$RA = \text{capacità consumata in } [t_E, t_D],$$

$$RT = \max \{t_E + T_S, t_D\}.$$

Due o più chunks sono consolidati in un unico chunk quando il loro tempo efficace di reintegro  $t_E$  coincide.

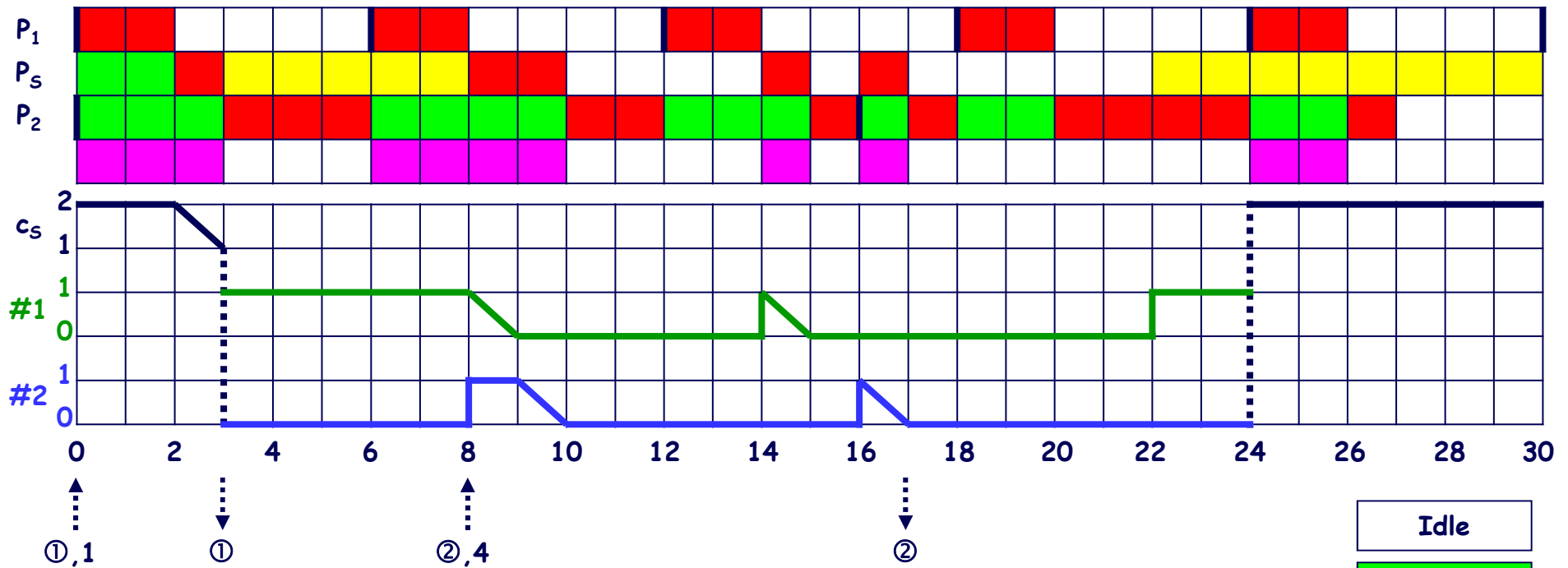


# ... SERVIZIO TRAMITE SS ...

$A_5$	$C$ [ms]	$T$ [ms]
$P_1$	2	6
$P_2$	6	16

$P_S : T_S = 8 \text{ ms}, C_S = 2 \text{ ms}$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	0	1
$R_{a2}$	8	4



		#1	#2	#1	#2	
$t_A$	0	6	14	16	24	
$t_E$	0	6	8	14	16	24
$t_D$	3	10	15	17	26	
RA	1	1	1	1	1	0
RT	8	14	16	22	24	-

Idle
Ready
Running
Waiting

## ... SERVIZIO TRAMITE SS

Il massimo carico computazionale indotto da SS  
è equivalente a quello derivante da una molteplicità di processi periodici aventi tutti  
lo stesso periodo  $T_s$  e tempo massimo di esecuzione uguale complessivamente a  $C_s$ .  
Ne deriva che il fattore di utilizzazione del processore da parte di SS è  $U_s = C_s/T_s$ .

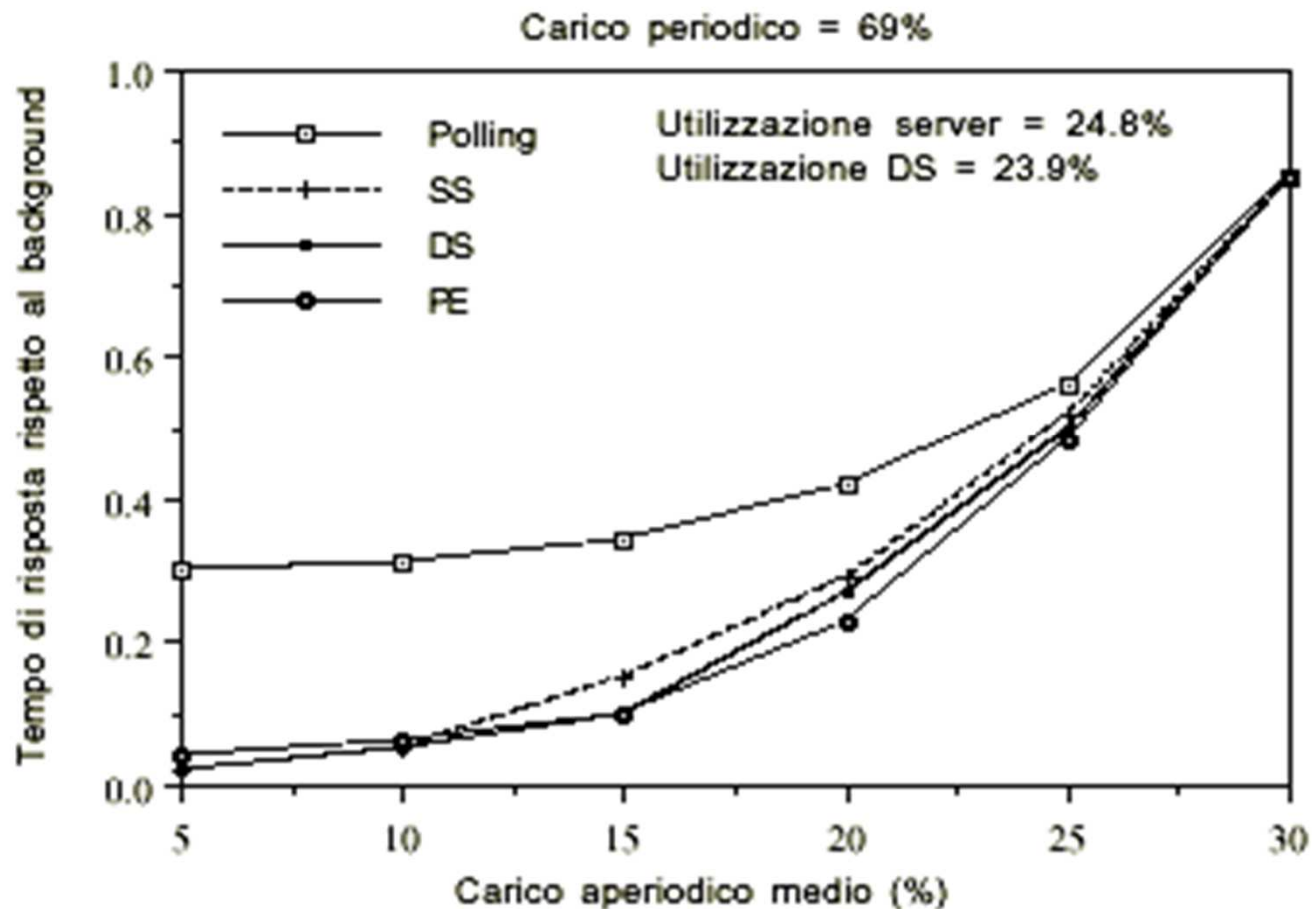
La schedulabilità di un insieme di  $N$  processi periodici  
contraddistinti da un fattore di utilizzazione del processore  $U_p$   
è pertanto garantita (condizione sufficiente, ma non necessaria) se:

$$U_s \leq (N+1) \left( 2^{1/(N+1)} - 1 \right) - U_p \underset{N \rightarrow \infty}{=} \ln 2 - U_p$$

o, nel caso SS abbia massima priorità, se:

$$U_s \leq \frac{2}{\left( 1 + U_p/N \right)^N} - 1 \underset{N \rightarrow \infty}{=} \frac{2}{e^{U_p}} - 1$$

## Confronto delle prestazioni: risultati di una simulazione disponibili in rete

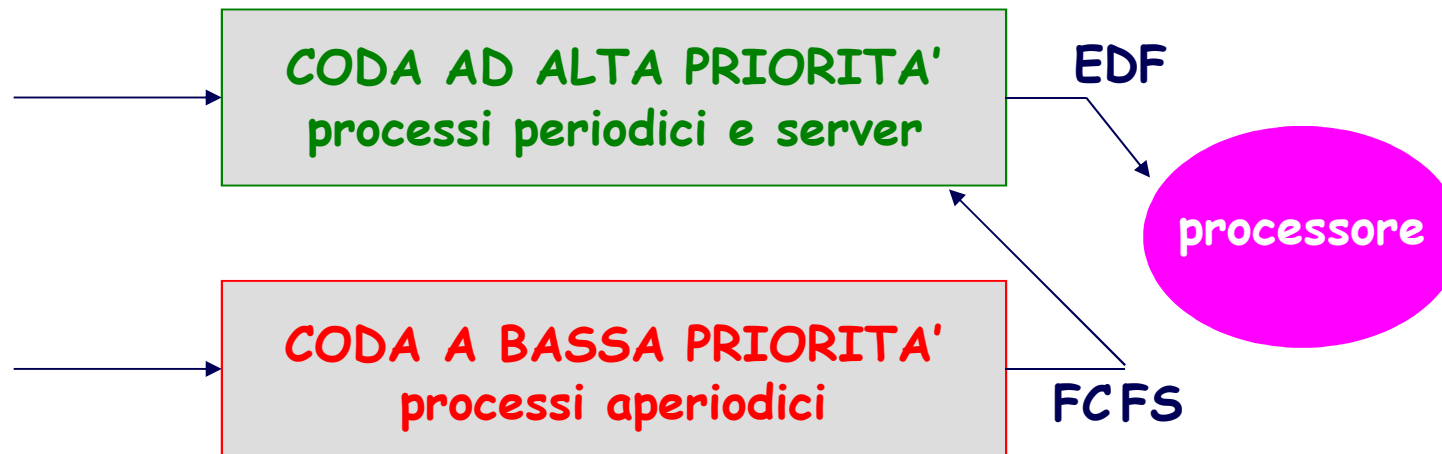


## Un confronto delle prestazioni

algoritmo	complessità	prestazioni (tempi di risposta)	schedulabilità processi hard RT
background service			
polling server			
deferrable server			
priority exchange server			
sporadic server			

Lo sporadic server fornisce un buon compromesso  
tra prestazioni, schedulabilità e complessità realizzativa.

# SERVIZIO TRAMITE SERVER A PRIORITÀ DINAMICA



Il server ( $P_s$ ) viene schedulato secondo la stessa strategia (tipicamente, e nel seguito, EDF) utilizzata per gli altri processi, in accordo alla sua priorità dinamica correlata alla deadline assoluta  $d_s$ .

$P_s$  emula un processo sporadico con fattore di utilizzazione scelto in modo tale da non compromettere la schedulabilità dei processi periodici:  $U_s \leq 1 - U_p$ .

I tempi di risposta dei processi aperiodici, mediamente inferiori rispetto a quelli derivanti dal servizio con server a priorità statica in conseguenza del più elevato valore di  $U_s$  selezionabile, dipendono dalle regole, peculiari di ciascuna tipologia di server, secondo cui è operato il riempimento ed il consumo della capacità del server stesso.

# SERVIZIO TRAMITE CONSTANT UTILIZATION SERVER (CUS) [Deng, Liu, Sun (97)] ...

Regole di riempimento e di consumo della capacità di  $P_S$

$P_S$ , inizialmente posto nello stato di attesa di richieste aperiodiche, diviene pronto per l'esecuzione non appena si presenta all'istante  $t_{Ra}$  una richiesta. Inserita la richiesta in testa alla coda,  $c_S$  e  $d_S$  vengono calcolati come segue:

$$c_S = C_{Ra}, d_S = t_{Ra} + c_S / U_S,$$

essendo  $C_{Ra}$  il tempo di servizio della richiesta.

$P_S$  viene eseguito in accordo alla priorità dinamica correlata alla deadline  $d_S$ , e  $c_S$  progressivamente consumata durante l'espletamento del servizio.

Eventuali richieste che si presentano prima di  $d_S$  vengono accodate.

Completato il servizio, la richiesta viene rimossa dalla coda e  $P_S$  posto nello stato idle fino all'istante  $d_S$ .

All'istante  $d_S$  se una o più richieste sono già pendenti, oppure all'istante di arrivo  $t_{Ra} > d_S$  di una nuova richiesta, viene identificato il tempo di servizio  $C_{Ra}$  della richiesta in testa alla coda e, posto

$$c_S = C_{Ra}, d_S = \max(d_S, t_{Ra}) + c_S / U_S,$$

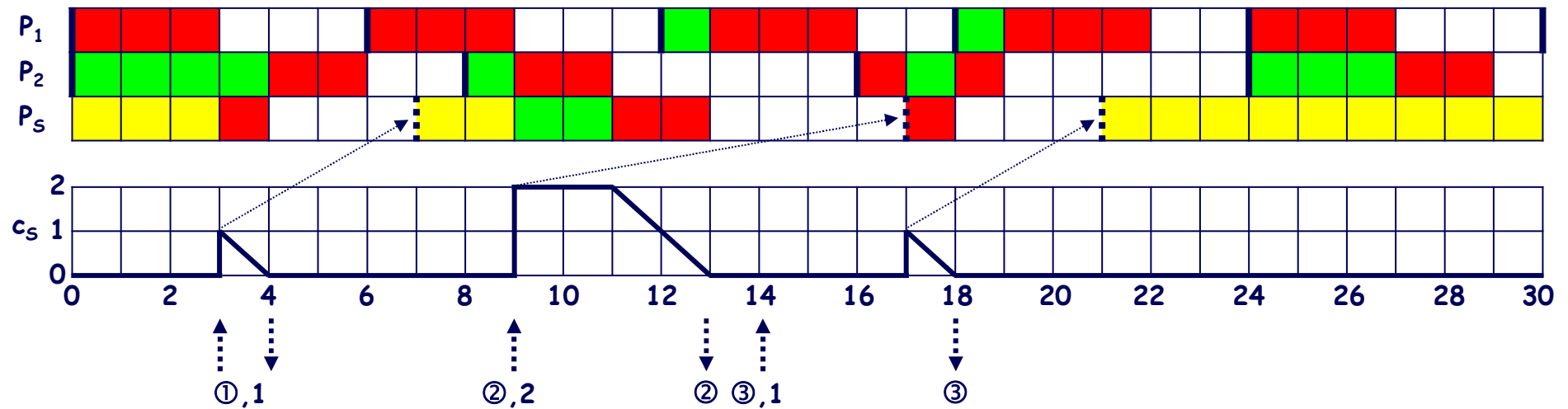
$P_S$  nuovamente dichiarato pronto per l'esecuzione.

## ... SERVIZIO TRAMITE CUS

$A_6$	$C$ [ms]	$T$ [ms]	$C/T$
$P_1$	3	6	0.50
$P_2$	2	8	0.25

$$P_S : U_S = 1 - U_p = 0.25$$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	3	1
$R_{a2}$	9	2
$R_{a3}$	14	1



# SERVIZIO TRAMITE TOTAL BANDWIDTH SERVER (TBS) [Spuri, Buttazzo (96)] ...

Regole di riempimento e di consumo della capacità di  $P_S$

$P_S$ , inizialmente posto nello stato di attesa di richieste aperiodiche, diviene pronto per l'esecuzione non appena si presenta all'istante  $t_{Ra}$  una richiesta. Inserita la richiesta in testa alla coda,  $c_S$  e  $d_S$  vengono calcolati come segue:

$$c_S = C_{Ra}, d_S = t_{Ra} + c_S / U_S,$$

essendo  $C_{Ra}$  il tempo di servizio della richiesta.

$P_S$  viene eseguito in accordo alla priorità dinamica correlata alla deadline  $d_S$ , e  $c_S$  progressivamente consumata durante l'espletamento del servizio.

Eventuali richieste che si presentano prima del completamento del servizio vengono accodate.

Terminato il servizio, la richiesta viene rimossa dalla coda.

Immediatamente se una o più richieste sono già pendenti, oppure all'istante di arrivo  $t_{Ra}$  di una nuova richiesta, viene identificato il tempo di servizio  $C_{Ra}$  della richiesta in testa alla coda e, posto

$$c_S = C_{Ra}, d_S = \max(d_S, t_{Ra}) + c_S / U_S,$$

$P_S$  nuovamente dichiarato pronto per l'esecuzione.

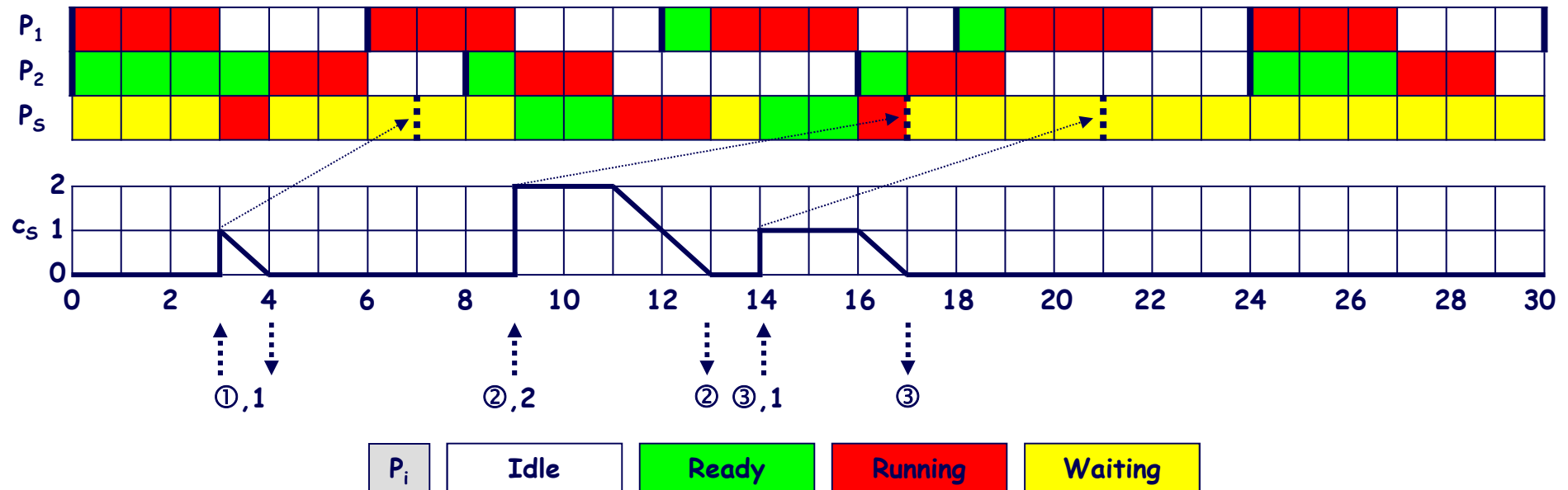


## ... SERVIZIO TRAMITE TBS

$A_6$	$C$ [ms]	$T$ [ms]	$C/T$
$P_1$	3	6	0.50
$P_2$	2	8	0.25

$$P_S : U_S = 1 - U_p = 0.25$$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	3	1
$R_{a2}$	9	2
$R_{a3}$	14	1



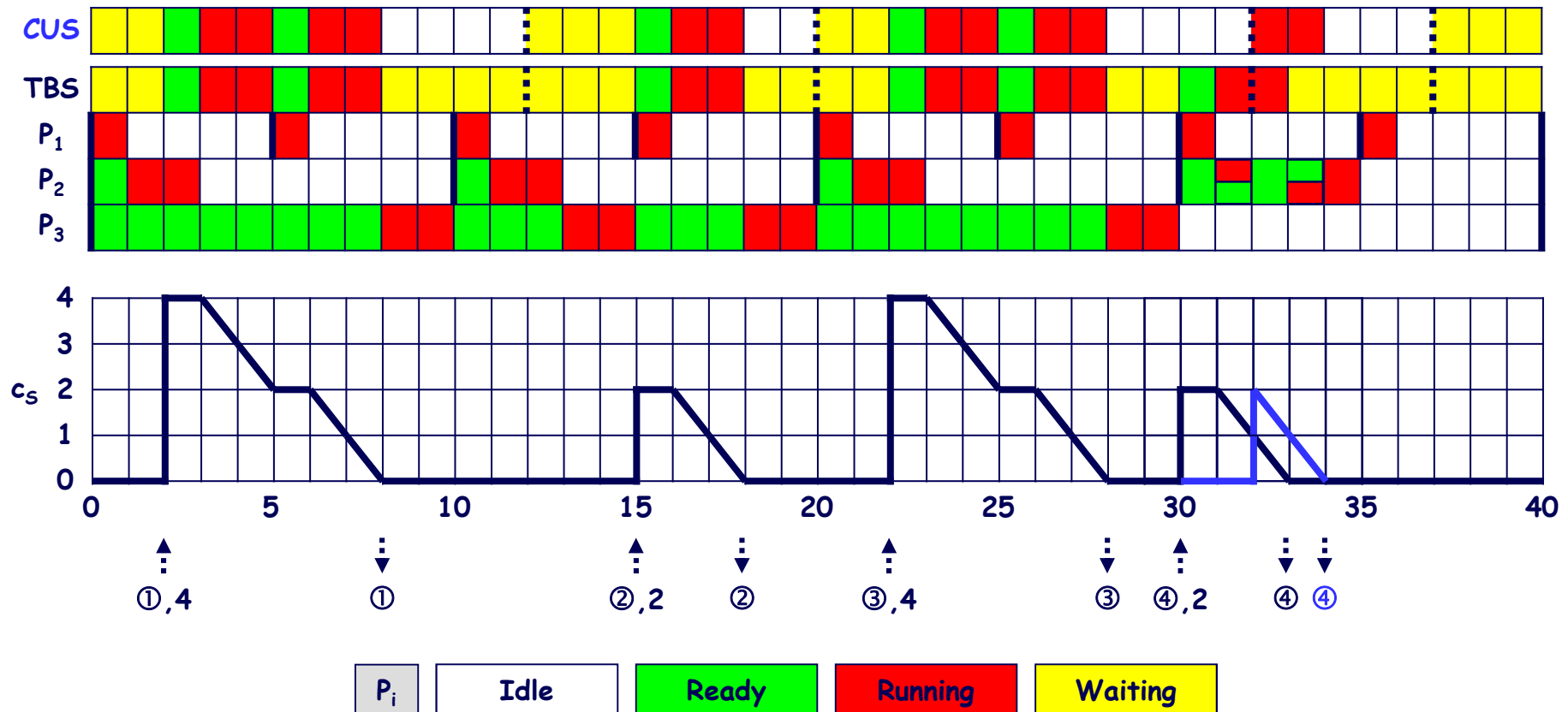
# CUS vs. TBS ...

$A_7$	$C$ [ms]	$T$ [ms]
$P_1$	1	5
$P_2$	2	10
$P_3$	8	40

$$U_p = 0.6 \rightarrow U_s = 0.4$$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	2	4
$R_{a2}$	15	2
$R_{a3}$	22	4
$R_{a4}$	30	2

$d$ [ms]
$2 + 4 / 0.4 = 12$
$15 + 2 / 0.4 = 20$
$22 + 4 / 0.4 = 32$
$32 + 2 / 0.4 = 37$



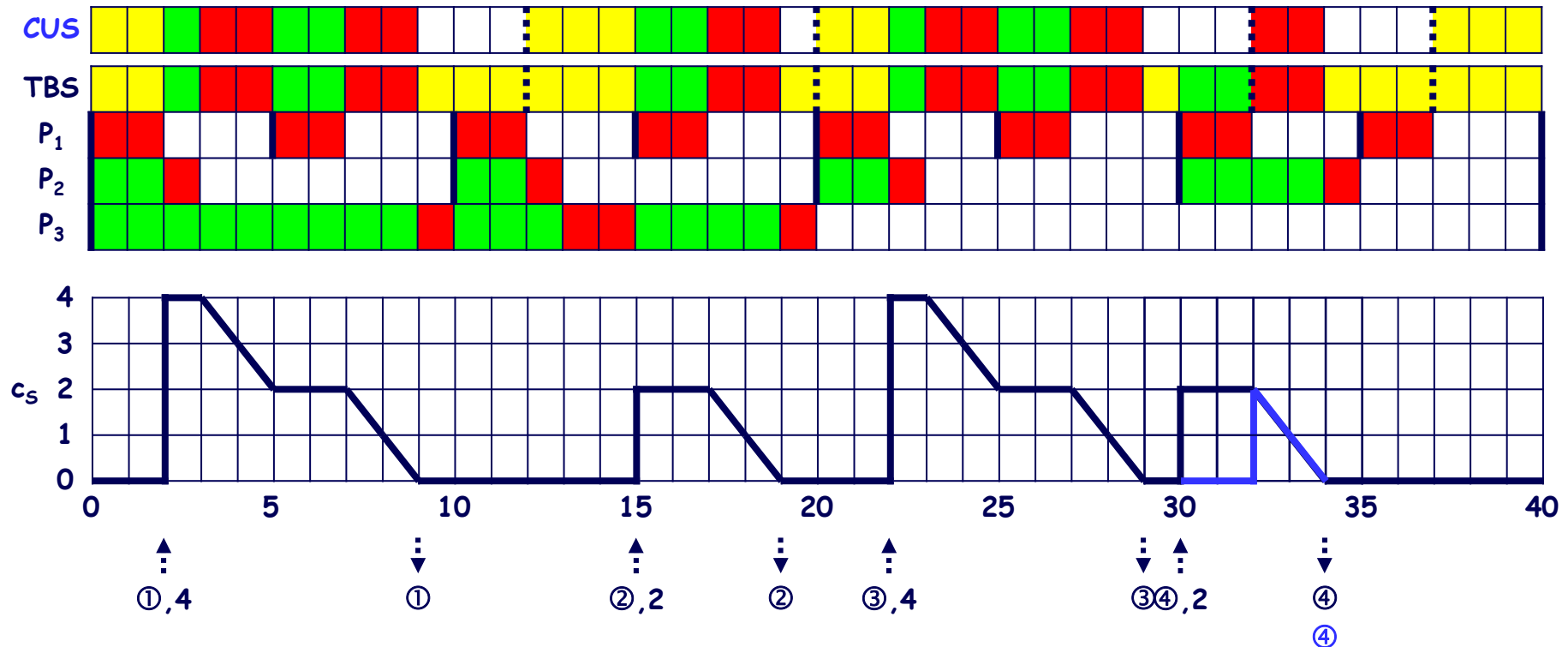
## ... CUS vs. TBS

$A_7$	$C$ [ms]	$T$ [ms]
$P_1$	2	5
$P_2$	1	10
$P_3$	4	40

$$U_p = 0.6 \rightarrow U_s = 0.4$$

	$a$ [ms]	$s$ [ms]
$R_{a1}$	2	4
$R_{a2}$	15	2
$R_{a3}$	22	4
$R_{a4}$	30	2

$d$ [ms]
$2 + 4 / 0.4 = 12$
$15 + 2 / 0.4 = 20$
$22 + 4 / 0.4 = 32$
$32 + 2 / 0.4 = 37$



## ... SERVIZIO TRAMITE CUS O TBS

Se il tempo di servizio  $C_{Ra}$  di una richiesta aperiodica non è noto all'istante di arrivo  $t_{Ra}$ , si può porre:

$$c_S = C_S, d_S = \max(d_S, t_{Ra}) + c_S / U_S,$$

essendo  $C_S$  un valore di default per la capacità del server.

Se il servizio non è completato all'istante  $d_S$  preventivato, ovvero se  $C_{Ra} > C_S$ , si ripete contestualmente il riempimento di  $c_S$  e si ricalcola  $d_S$  come segue:

$$c_S = C_S, d_S = d_S + c_S / U_S.$$

In caso contrario, ovvero se  $C_{Ra} < C_S$  o, più in generale, se  $C_{Ra} < n C_S$  ( $n = 1, 2, \dots$ ), si procede al completamento del servizio a ridimensionare, in base alla capacità residua del server  $c_S$ ,

l'ultimo valore di  $d_S$  preventivato, di fatto eccessivo, ponendo:

$$d_S = d_S - c_S / U_S, c_S = 0.$$