Objetivos

- Veremos cómo **planificar** el uso de los recursos para poder **garantizar** los **requisitos temporales**
- Un **método de planificación** tiene dos aspectos importantes:
  - Un **algoritmo de planificación** que determina el orden de acceso de las tareas a los recursos del sistema (en particular al procesador)
  - Un **método de análisis** que permite calcular el **comportamiento temporal** del sistema
    - » Así se pude comprobar si los requisitos temporales están **garantizados** en todos los casos posibles
    - » En general se estudia el **peor comportamiento** posible

Planificación de tareas

- Un método de planificación puede ser
  - **estático**: el análisis se puede hacer antes de la ejecución
  - **dinámico**: el análisis se hace durante la ejecución
- Un método estático muy utilizado es el de **planificación con prioridades fijas y desalojo**
  - La prioridad es un parámetro relacionado con la *urgencia* o la *importancia* de la tarea
  - En cada momento se ejecuta la tarea con mayor prioridad de todas las ejecutables

Parámetros de planificación

N	Número de tareas
T	Período de activación
C	Tiempo de ejecución máximo
D	Plazo de respuesta
R	Tiempo de respuesta máximo
P	Prioridad

En el modelo anterior, para todas las tareas τ_i :

$$C_i \leq D_i = T_i$$

Se trata de asegurar que

$$R_i \leq D_i$$

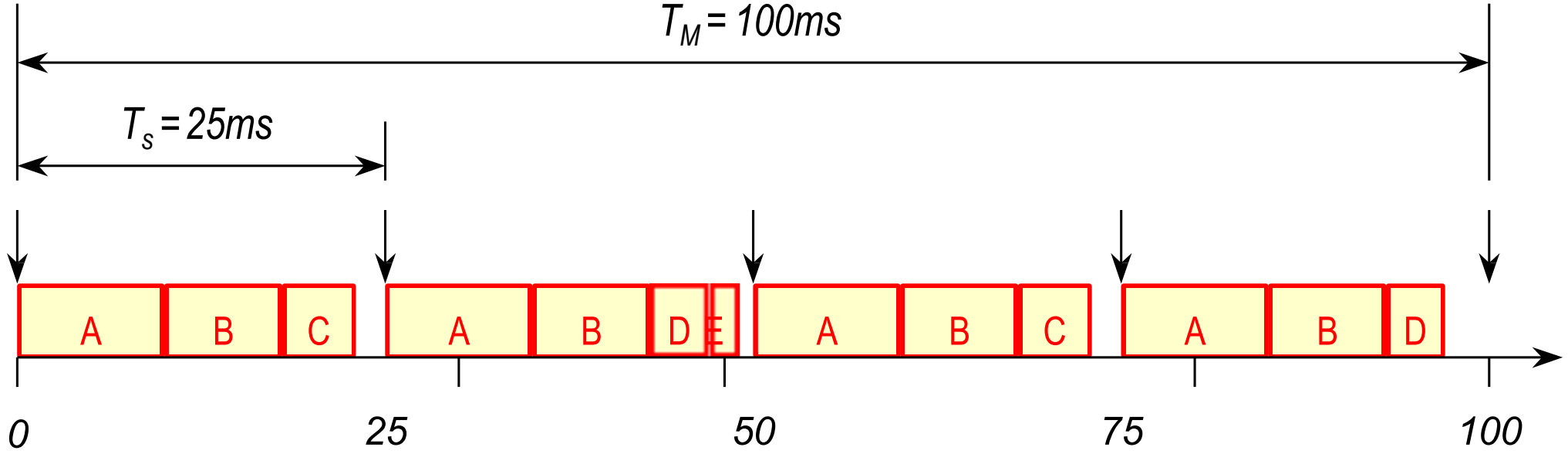
Índice

- Introducción
- **Planificación con ejecutivos cíclicos**
- Planificación con prioridades
  - Tareas periódicas independientes
  - Tareas esporádicas y aperiódicas
  - Interacción entre tareas y bloqueos
  - Modelo de tareas generalizado
- Otros métodos de planificación

Ejemplo

Tarea	T	C
A	25	10
B	25	8
C	50	5
D	50	4
E	100	2

- El sistema es armónico
- El ciclo principal dura 100ms
- Se compone de 4 ciclos secundarios de 25ms cada uno



Propiedades

- **No hay concurrencia** en la ejecución
  - Cada ciclo secundario es una secuencia de invocaciones de procedimientos
- Los procedimientos pueden **compartir datos**
  - No hace falta usar mecanismos de exclusión mutua
- Los períodos deben ser **armónicos**
  - Puede ser necesario utilizar períodos más cortos de lo necesario
- **No hace falta analizar** el comportamiento temporal
  - El sistema es correcto por construcción

Índice

- Introducción
- Planificación con ejecutivos cíclicos
- **Planificación con prioridades**
  - **Tareas periódicas independientes**
  - Tareas esporádicas y aperiódicas
  - Interacción entre tareas y bloqueos
  - Modelo de tareas generalizado
- Otros métodos de planificación

Prioridades monótonas en frecuencia

- La asignación de mayor prioridad a las tareas de menor período *(rate monotonic scheduling)* es **óptima** para el modelo de tareas simple (tareas periódicas, independientes, con plazos iguales a los períodos)

Si se pueden garantizar los plazos de un sistema de tareas con otra asignación de prioridades, se pueden garantizar con la asignación monótona en frecuencia

Condición de garantía de los plazos basada en la utilización

- Para el modelo simple, con prioridades monótonas en frecuencia, los plazos están garantizados si

$$U = \sum_{i=1}^{N} \frac{C_i}{T_i} \leq N \cdot \left(2^{1/N} - 1\right)$$

  - La cantidad

$$U_0(N) = N \cdot \left(2^{1/N} - 1\right)$$

es la **utilización mínima garantizada** para N tareas

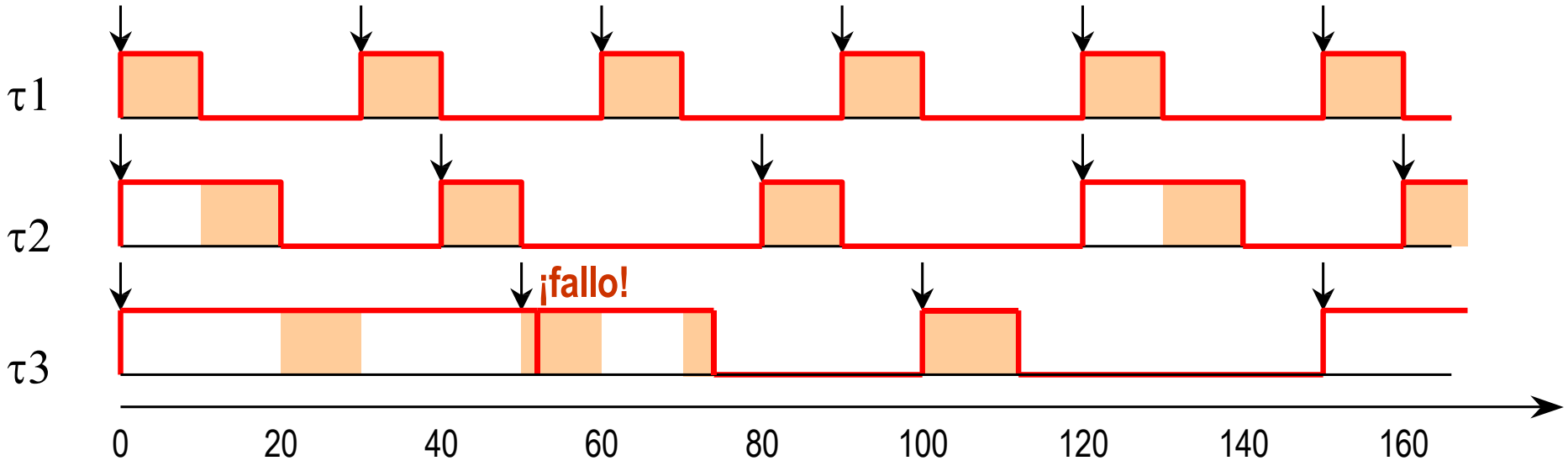
Ejemplo 1

Tarea	T	C	P	U
τ1	30	10	3	0,333
τ2	40	10	2	0,250
τ3	50	12	1	0,240
				0,823

El sistema no cumple la prueba de utilización

(U > 0,779)

La tarea 3 falla en $t = 50$

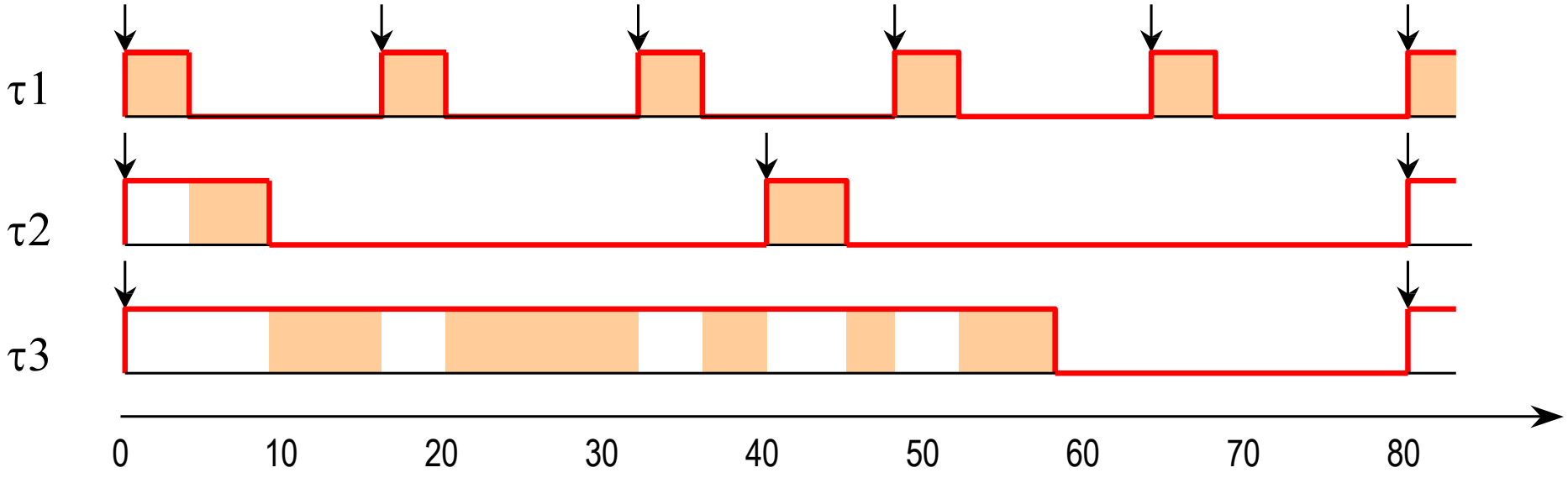


Ejemplo 2

Tarea	T	C	P	U
τ1	**16**	**4**	**3**	0,250
τ2	**40**	**5**	**2**	0,125
τ3	**80**	**32**	**1**	0,400
				0,775

Este sistema está garantizado

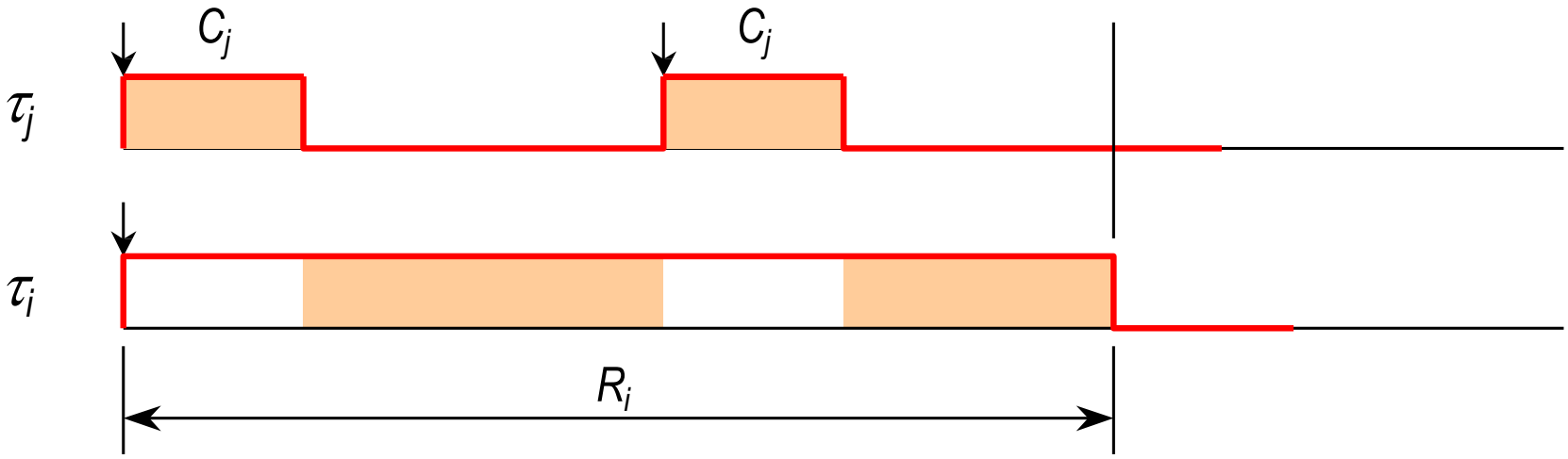
(U < 0,779)



Análisis del tiempo de respuesta

- La prueba del factor de utilización no es exacta, ni se puede generalizar a modelos de tareas más complejos
- La construcción de un cronograma es compleja, incluso considerando que el instante inicial es crítico
- Veremos una prueba basada en el cálculo del tiempo de respuesta de cada tarea

Cálculo de la interferencia máxima



Para una tarea
de prioridad superior

$$I_i^j = \left\lceil \frac{R_i}{T_j} \right\rceil \bullet C_j$$

Para todas las tareas
de prioridad superior

$$I_i = \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \cdot C_j$$

Iteración lineal

La ecuación del tiempo de respuesta se puede resolver mediante la relación de recurrencia

$$w_i^{n+1} = C_i + \sum_{j \in hp(i)} \left\lceil \frac{w_i^n}{T_j} \right\rceil \cdot C_j$$

Un valor inicial aceptable es

$$w_i^0 = C_i + \sum_{j \in hp(i)} C_j$$

Se termina cuando

a) $w^{n+1} = w^n$, o bien

b) $w^{n+1} > T_i$ (no se cumple el plazo)

Ejemplo

Tarea	T	C	P	R
T1	7	3	3	3
T2	12	3	2	6
T3	20	5	1	20

T1: $W_1^0 = 3$ O.K.

T2: $W_2^0 = 3 + 3 = 6$

$W_2^1 = 3 + \lceil (6/7) \rceil \cdot 3 = 6$ O.K.

T3: $W_3^0 = 5 + 3 + 3 = 11$

$W_3^1 = 5 + \lceil (11/7) \rceil \cdot 3 + \lceil (11/12) \rceil \cdot 3 = 14$

$W_3^2 = 5 + \lceil (14/7) \rceil \cdot 3 + \lceil (14/12) \rceil \cdot 3 = 17$

$W_3^3 = 5 + \lceil (17/7) \rceil \cdot 3 + \lceil (17/12) \rceil \cdot 3 = 20$

$W_3^4 = 5 + \lceil (20/7) \rceil \cdot 3 + \lceil (20/12) \rceil \cdot 3 = 20$ O.K.

Tiempo de cómputo

Hay dos formas de obtener el valor de *C* para una tarea:

- **Medida del tiempo de ejecución**
  - no es fácil saber cuándo se mide el tiempo máximo
- **Análisis del código ejecutable**
  - » se descompone el código en un grafo de bloques secuenciales
  - » se calcula el tiempo de ejecución de cada bloque
  - » se busca el camino más largo
  - la estimación de *C* puede ser muy pesimista
  - es difícil tener en cuenta los efectos de los dispositivos de hardware (*caches, pipelines,* etc..)
  - se puede mejorar con análisis estático

Índice

- Introducción
- Planificación con ejecutivos cíclicos
- **Planificación con prioridades**
  - Modelo de tareas básico
  - **Tareas esporádicas y aperiódicas**
  - Interacción entre tareas y bloqueos
  - Modelo de tareas generalizado
- Otros métodos de planificación

Tareas esporádicas

- Para incluir tareas esporádicas hace falta modificar el modelo simple de tareas:
  - El parámetro ***T*** representa la **separación** mínima entre dos sucesos de activación consecutivos
  - Suponemos que en el peor caso la activación es pseudoperiódica (con período *T)*
  - El plazo de respuesta puede ser menor que el período ($D \leq T$)
- El análisis de tiempo de respuesta sigue siendo válido
- Funciona bien con cualquier orden de prioridad

Ejemplo :

Tarea	T	D	C	P	R
τ1	20	5	3	4	3
τ2	15	7	3	3	6
τ3	10	10	4	2	10
τ4	20	20	3	1	20

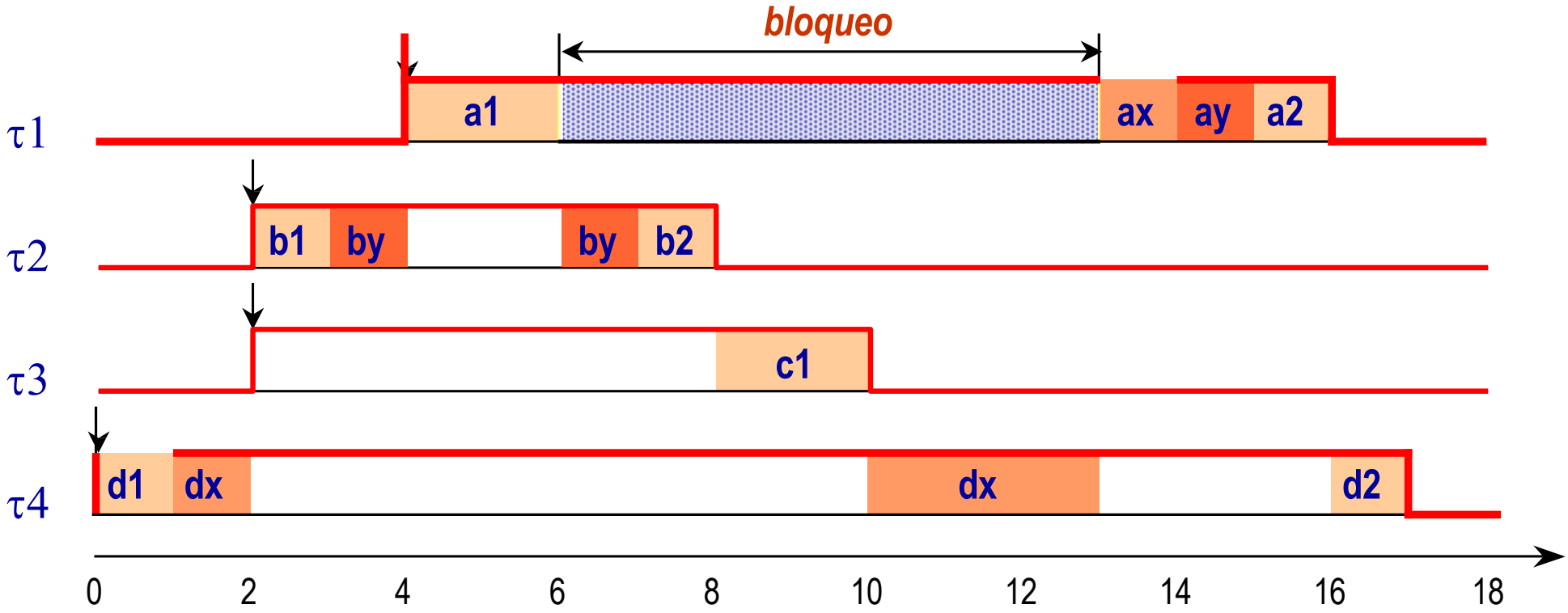
Comprobar si planificamos con prioridades monótonas en frecuencia se garantizan los pazos. Que sucede si hacemos el análisis tomando en cuenta el tiempo de respuesta R?

Tarea	T	D	C	P	R
τ3	10	10	4	4	4
τ2	15	7	3	3	7
τ1	20	5	3	2	10
τ4	20	20	3	1	20

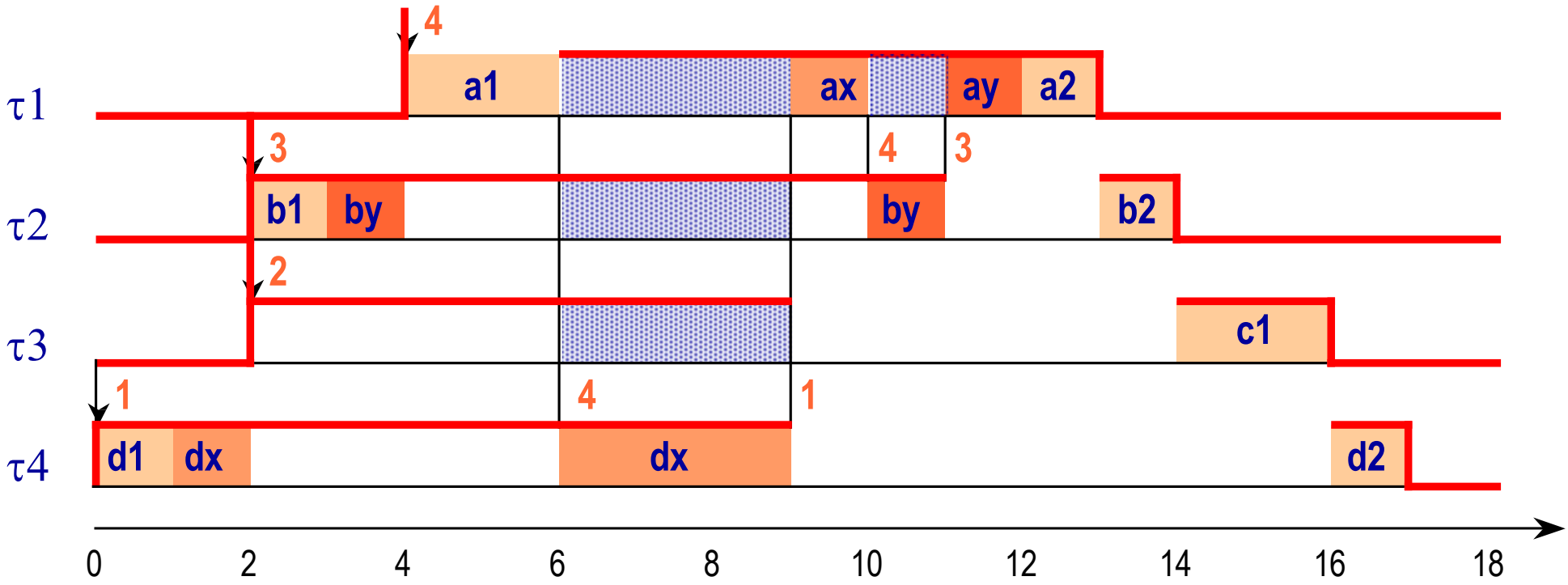
Interacción entre tareas

- En la mayoría de los sistemas de interés práctico las tareas interaccionan mediante
  - datos comunes (protegidos)
  - citas o mensajes
- En todos estos casos puede ocurrir que una tarea tenga que esperar un suceso de otra menos prioritaria
- Esta situación se denomina **bloqueo**, y produce una **inversión de prioridad** indeseable
- La inversión de prioridad no se puede eliminar completamente, pero es posible limitar su duración

Ejemplo 6 : inversión de prioridad



Ejemplo 6 : herencia de prioridad



Ejemplo 6 : cálculo del bloqueo

$$B_1 = C_{2,Y} + C_{4,X} = 2 + 4 = 6$$

$$B_2 = C_{4,X} = 4$$

$$B_3 = C_{4,X} = 4$$

$$B_4 = 0$$

- Una tarea puede bloquearse por recursos a los que no accede (por ejemplo, $\tau 2$)
- Una tarea puede sufrir bloqueo aunque no acceda a recursos compartidos (por ejemplo, $\tau 3$)
- La tarea de menor prioridad ($\tau 4$) no sufre bloqueo

Protocolos de techo de prioridad

- El **techo de prioridad** de un recurso es la máxima prioridad de las tareas que lo usan

- El **protocolo del techo de prioridad** consiste en :
  - la prioridad dinámica de una tarea es el máximo de su prioridad básica y las prioridades de las tareas a las que bloquea
  - una tarea sólo puede usar un recurso si su prioridad dinámica es mayor que el techo de todos los recursos en uso por otras tareas

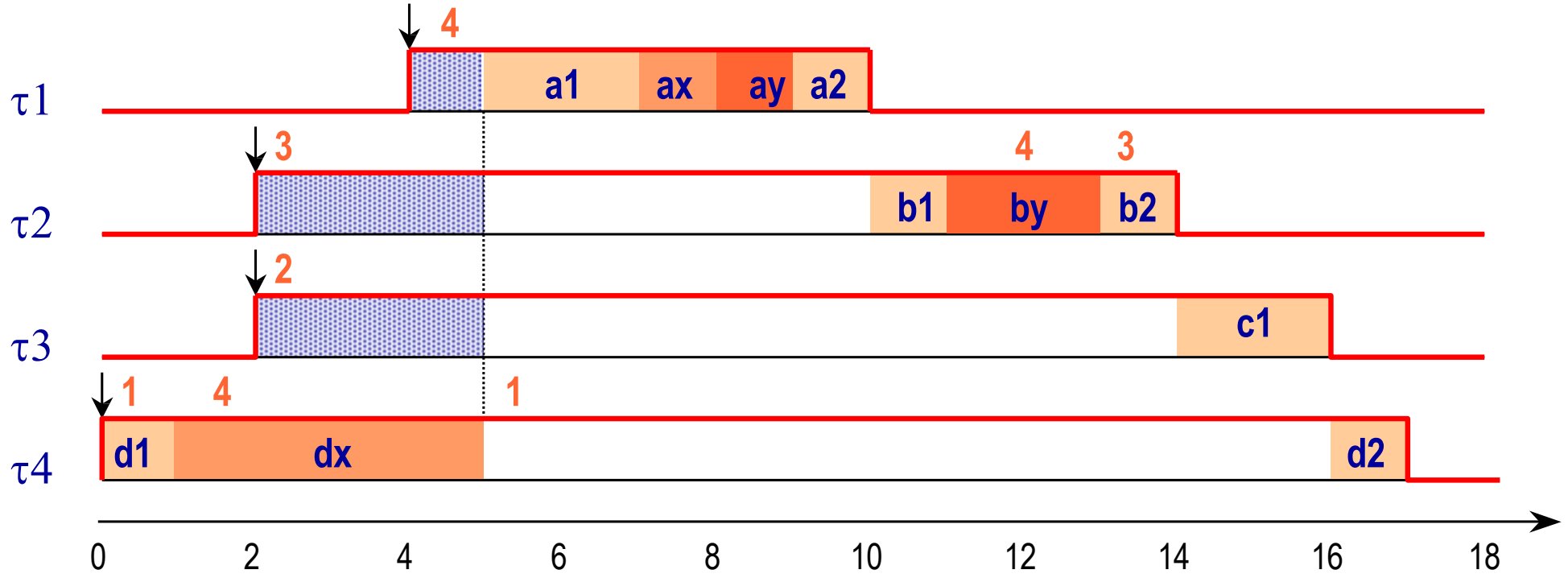
Propiedades

- Cuando se usa el protocolo del techo de prioridad en un sistema monoprocesador,
  - Cada tarea se puede bloquear una vez, como máximo, en cada ciclo
  - No puede haber interbloqueos
  - No puede haber bloqueos encadenados
  - Se consigue la exclusión mutua sin mecanismos de protección adicionales
- La duración máxima del bloqueo es ahora

$$B_i = \max_{j \in lp(i), k \in lc(i)} C_{j,k}$$

lc(i) es el conjunto de recursos que pueden bloquear *i*

Ejemplo 6 : techo de prioridad inmediato



Modelo de tareas generalizado

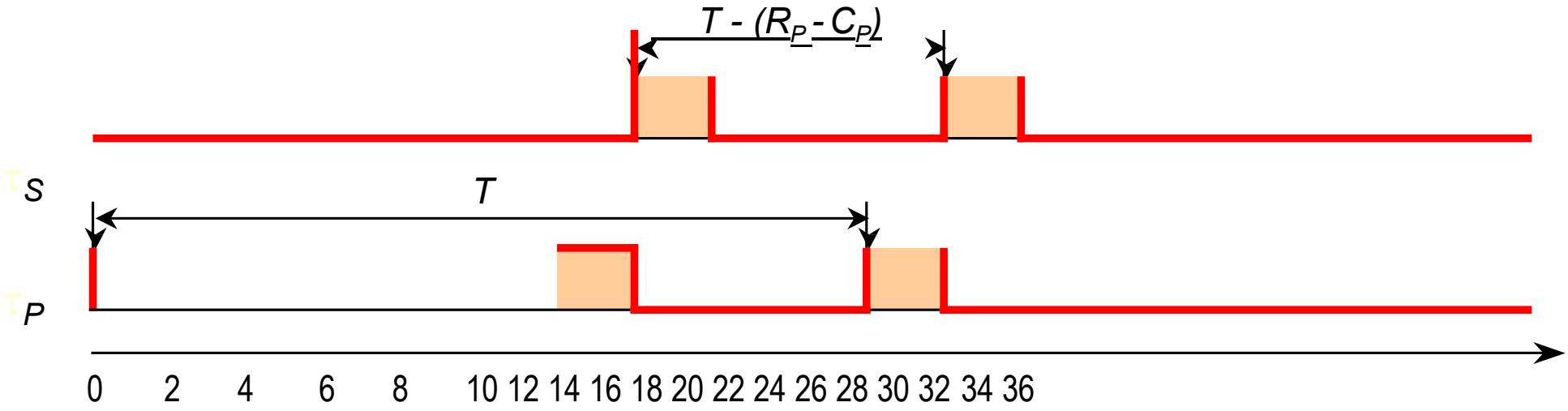
- El modelo de tareas básico que hemos visto incluye
  - tareas periódicas y esporádicas
  - plazos menores o iguales que los períodos
  - interacción mediante secciones críticas
- El análisis del tiempo de respuesta se puede extender con
  - planificación cooperativa
  - variación *(jitter)* en el esquema de activación
  - plazos arbitrarios (también mayores que los períodos)

Ventajas

- Ventajas
  - Mejora la planificabilidad
  - Se pueden reducir los tiempos de cómputo
  - Los tiempos de cómputo de los bloques son más fáciles de calcular
- Problemas
  - el coste de ofrecer un cambio de contexto puede ser alto (e innecesario)

Activación irregular

- Puede haber variaciones en los esquemas de activación periódicos y esporádicos
- Por ejemplo, sea una tarea esporádica activada por un suceso producido por una tarea periódica situada en otro procesador



Tareas periódicas con *jitter*

- Las tareas periódicas no suelen tener *jitter*
- Puede haberlo si la planificación se hace con una granularidad apreciable
- El tiempo de respuesta se calcula con

$$w_i^{n+1} = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{w_i^n + J_j}{T_j} \right\rceil \cdot C_j$$

$$R_i = w_i^n + J_i, \text{ para } w_i^{n+1} = w_i^n$$

Tiempo de respuesta máximo

- Se calcula R_i(q) para q = 0, 1, ... , q_{max}, donde q_{max} es el menor valor que cumple

$$R_i(q_{max}) \leq T_i$$

En esta situación ya no hay solape con la siguiente activación

- El tiempo de respuesta en el peor caso es

$$R_i = \max_{q=0,..q_{max}} R_i(q)$$

Asignación de prioridades

- Con el modelo general que hemos, la ordenación de prioridades monótona en frecuencia (RMS) o en plazos **no es óptima**
- Se aplica el siguiente **teorema**:

> *Si una tarea τ cumple su plazo con la prioridad más baja del sistema, y existe un orden de prioridades admisible para todo el sistema, entonces hay un orden de prioridades que garantiza todas las tareas en el cual τ tiene la prioridad más baja*

Otros métodos de planificación

- Se pueden conseguir mejores resultados realizando la planificación **dinámicamente** en función de los requisitos temporales y de los recursos disponibles
- Dos métodos dinámicos interesantes son:
  - **Primero el más urgente** *(earliest deadline first, EDS)*
  - **Primero el menos holgado** *(least slack first, LSF)*
- Ambos son óptimos para tareas independientes
  - se garantizan los plazos hasta el 100% de utilización
  - se comportan bien cuando hay muchas tareas esporádicas
- Problemas:
  - el comportamiento en caso de sobrecarga es imprevisible
  - no está bien resuelta la interacción entre tareas