

Tema 5

Planificación de Tareas



Sistemas de Tiempo Real
(3º Grado de Ingeniería Informática)
Área de Ingeniería de Sistemas y Automática
Departamento de Informática
Universidad de Almería



ÍNDICE

Tema 1. Introducción a los sistemas de tiempo real y sistemas de control

Tema 2. Implementación de STR

Tema 3. Modelado de sistemas de tiempo real con Redes de Petri

Tema 4. Tareas, concurrencia y sincronización en STR

Tema 5. Planificación de tareas

Tema 6. Campos de aplicación de los STR

Índice

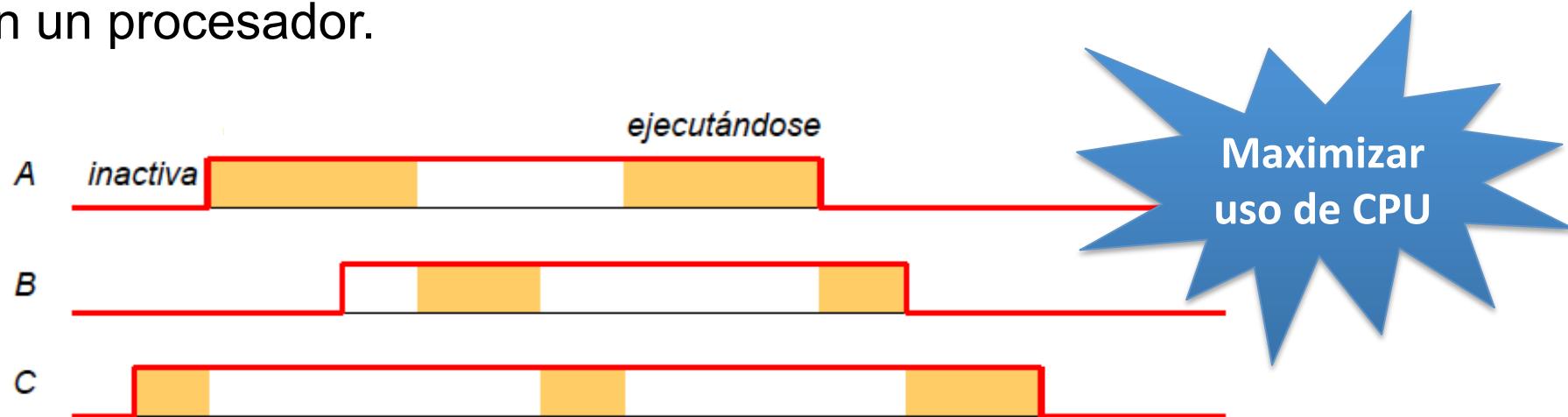
1. Introducción.
2. Planificación con ejecutivos cíclicos.
3. Planificación con prioridades de tareas periódicas independientes.
4. Planificación con prioridades de tareas esporádicas y aperiódicas.
5. Interacción entre tareas y bloqueos en planificación con prioridades.

Índice

- 1. Introducción.**
2. Planificación con ejecutivos cíclicos.
3. Planificación con prioridades de tareas periódicas independientes.
4. Planificación con prioridades de tareas esporádicas y aperiódicas.
5. Interacción entre tareas y bloqueos en planificación con prioridades.

Introducción

Los sistemas de tiempo real controlan actividades del mundo exterior que son simultáneas. Para ello deben ejecutar varias actividades o **tareas** en “paralelo” (concurrentemente). Normalmente, la ejecución de las tareas se multiplexa en el tiempo en un procesador.



Introducción

Planificación: forma o criterio que se sigue a la hora de decidir que proceso debe entrar en ejecución.

Planificador: elemento que se encarga de tomar la decisión de qué proceso entra en ejecución. El planificador DECIDE que proceso sale o entra al procesador.

Objetivos de un buen planificador:

- Equidad
- Eficiencia (100% utilización CPU)
- Minimizar el tiempo de espera
- Aumentar el rendimiento (máximo número de trabajos por unidad de tiempo)

Introducción

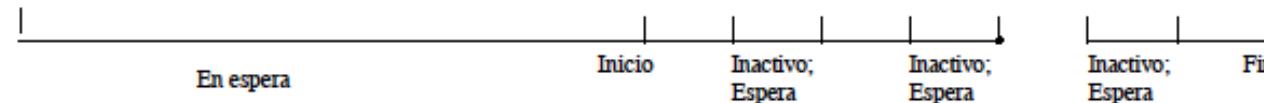
EJEMPLO. Un sistema con dos procesos P₁ y P₂. Cada proceso se ejecuta durante 1 seg. y espera otro seg.

Proceso P₁



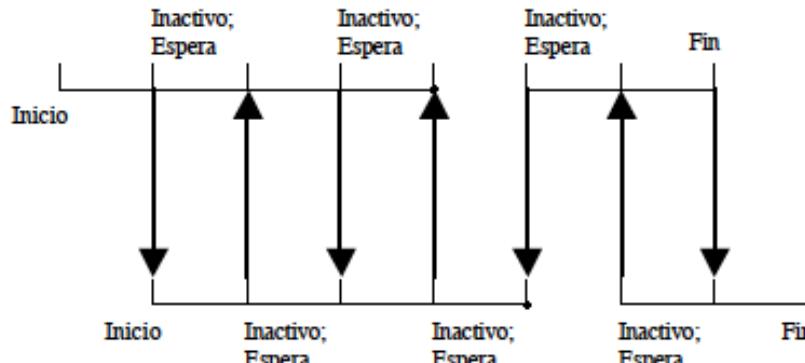
Utilización: 50 %

Proceso P₂



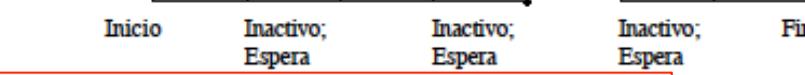
Ejecución de procesos sin multiprogramación

Proceso P₁



Utilización: 100 %

Proceso P₂



Ejecución de procesos con multiprogramación

Fuente: ISA-UMH

Introducción

Los elementos fundamentales de un método de planificación son:

1. Un **algoritmo de planificación** que determina el orden de acceso de las tareas a los recursos del sistema (en particular al procesador).
2. Un **método de análisis** que permita calcular el comportamiento temporal del sistema, de modo que se pueda comprobar si los requisitos temporales están garantizados en todos los casos posibles, estudiando en general el peor comportamiento posible.

Los métodos de planificación se clasifican generalmente en:

- **Estáticos**: donde el análisis se puede hacer antes de la ejecución.
- **Dinámicos**: el análisis se hace durante la ejecución.

Introducción

Por ejemplo, el método estático más utilizado es el de **planificación con prioridades fijas y desalojo**, donde la prioridad es un parámetro relacionado con la urgencia o la importancia de la tarea, de modo que **en cada momento se ejecuta la tarea con mayor prioridad de todas las ejecutables**.

Para el desarrollo de un método de planificación de tareas, es habitualmente preciso disponer de un modelo de tareas. **En una aproximación inicial, consideraremos un *modelo simple* que tiene las siguientes características:**

- El conjunto de tareas es estático.
- Todas las tareas son periódicas y de periodo conocido.
- Las tareas son independientes unas de otras.
- Los plazos de respuesta de todas las tareas son iguales a los periodos respectivos.
- El tiempo de ejecución máximo de cada tarea es conocido.
- Las operaciones del núcleo de multiprogramación son instantáneas.

Introducción

En el desarrollo de métodos de planificación es importante disponer de una nomenclatura :

N	Número de tareas
T	Periodo del proceso (tiempo mínimo entre realizaciones del proceso)
C	Tiempo de ejecución máximo (tiempo de eje. en el peor de los casos)
D	Plazo de respuesta (tiempo de finalización del proceso)
R	Tiempo de respuesta máximo (tiempo de resp. en el peor de los casos)
P	Prioridad (si aplicable)
U	Factor de utilización (C / T)

De modo que en el modelo simple cuyas características se han descrito previamente, se cumple para todas las tareas T_{area_i} que $C_i \leq T_i$ con la finalidad de asegurar $R_i \leq D_i$. En el modelo de tarea simple, el valor $H=mcm(T_i)$ se denomina **hiperperiodo** del sistema (donde mcm es el mínimo común múltiplo del periodo de activación de todas las tareas), de modo que el comportamiento temporal se repite cada hiperperiodo H .

Introducción

Hay 2 tipos principales de arquitecturas (software) para la planificación de STR:

– Arquitectura síncrona

- » las tareas se ejecutan según un **plan de ejecución fijo** (realizado por el diseñador).
- » el sistema operativo se reemplaza por un **ejecutivo cíclico**.

– Arquitectura asíncrona

- » las tareas se reparten el procesador **de forma dinámica** (invisible para el diseñador).
- » cada tarea tiene una prioridad.
- » en cada momento se ejecuta la **tarea activa de mayor prioridad**.

Índice

1. Introducción.
- 2. Planificación con ejecutivo cílico.**
3. Planificación con prioridades de tareas periódicas independientes.
4. Planificación con prioridades de tareas esporádicas y aperiódicas.
5. Interacción entre tareas y bloqueos en planificación con prioridades.

Ejecutivo cíclico

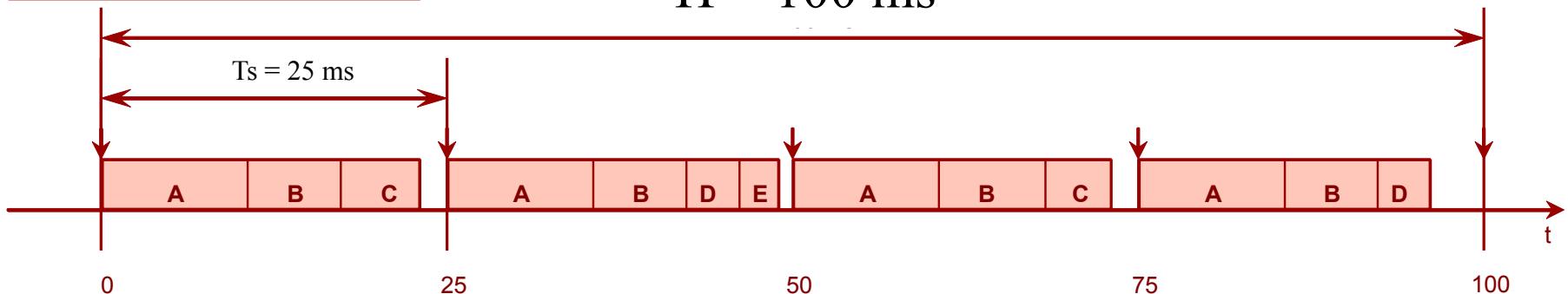
Se utilizan cuando todas las tareas son periódicas, pues se puede confeccionar un plan de ejecución fijo. Se trata de un esquema que se repite cada $H=mcm(T_i)$ (ciclo principal). El ciclo principal se divide en ciclos secundarios, con periodo T_s , ejecutándose las actividades correspondientes a determinadas tareas en cada ciclo secundario.

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

EJEMPLO DE PLANIFICACIÓN CÍCLICA

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

$$H = 100 \text{ ms}$$



Ejecutivo cíclico

```
with Ada.Real_Time; use Ada.Real_Time; -- paquete para trabajar con tiempo (periodo)
with Ada.Text_IO; use Ada.Text_IO;
```

```
procedure Ejecutivo_Ciclico is
```

```
    T: Time;
```

```
    Periodo: Time_Span:= Milliseconds(25);
```

```
    type Ciclo is mod 4;
```

```
    Turno: Ciclo:=0;
```

```
    task A is
```

```
        entry A;
```

```
    end;
```

```
    task body A is
```

```
    begin
```

```
        loop
```

```
            accept A do
```

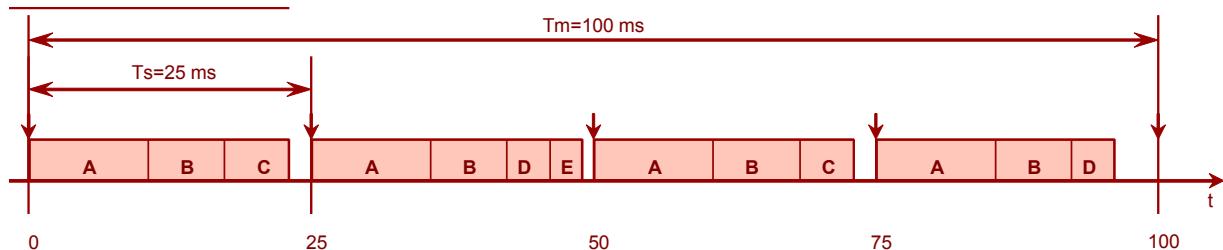
```
                Put("A");      -- aquí irán las acciones de la tarea
```

```
            end A;
```

```
        end loop;
```

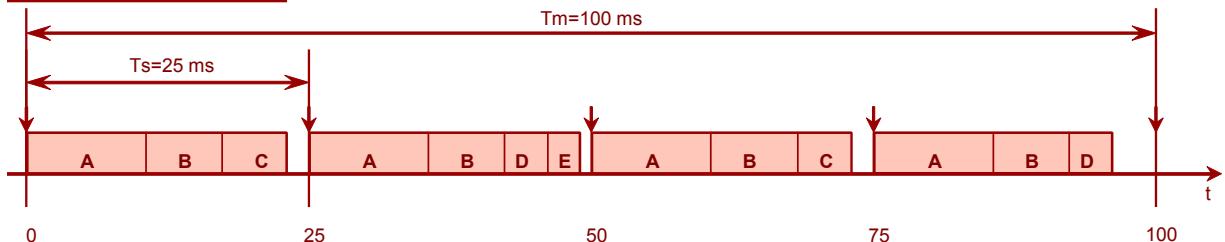
```
    end A;
```

```
-- idem para tareas B,C,D y E
```



Ejecutivo cíclico

```
...
begin
T:=Clock;
loop
  T:=T+Periodo;
  delay until(T);
  A.A; B.B; C.C;
  T:=T+Periodo;
  delay until(T);
  A.A; B.B; D.D; E.E;
  T:=T+Periodo;
  delay until(T);
  A.A; B.B; C.C;
  T:=T+Periodo;
  delay until(T);
  A.A; B.B; D.D;
end loop;
end Ejecutivo_Ciclico;
```



--ciclo 0

--ciclo 1

--ciclo 2

--ciclo 3

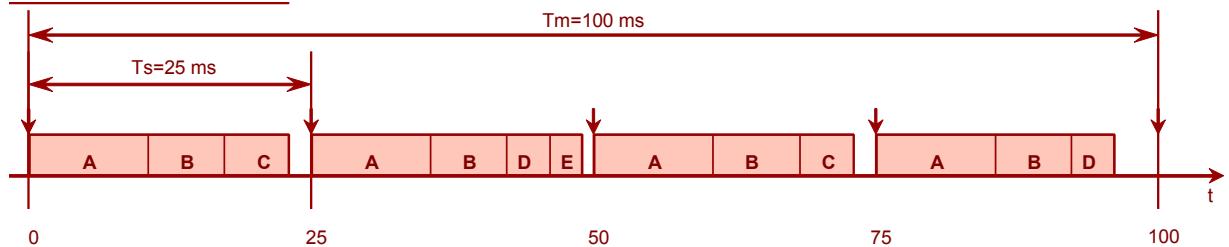
Primera
Implementación

Ejecutivo cíclico

```

...
begin
T:=Clock;
loop
  T:=T+Periodo;
  delay until(T);
  case Turno is
    when 0 => A.A; B.B; C.C;
    when 1 => A.A; B.B; D.D; E.E;
    when 2 => A.A; B.B; C.C;
    when 3 => A.A; B.B; D.D;
  end case;
  Turno:=Turno+1;
end loop;
end Ejecutivo_Ciclico;

```



Ejecutivo cíclico

Las propiedades fundamentales de este esquema de planificación son:

- No hay concurrencia en la ejecución: cada ciclo secundario es una **secuencia de invocaciones de procedimientos**. Sólo hay un hilo de ejecución.
- Los procedimientos pueden compartir datos y **no hace falta usar mecanismos de exclusión mutua. No existen interbloqueos**.
- No existen funciones de sincronización. Mecanismos de comunicación entre tareas simples.
- No hace falta analizar el comportamiento temporal (**el sistema es correcto por construcción**). Es sencillo determinar la ejecutabilidad del sistema.

Ejecutivo cíclico

Los problemas fundamentales que plantea este método son:

- Las tareas esporádicas son difíciles de tratar.
- El plan cíclico es difícil de construir.
- **Todos los períodos de los procesos deben ser múltiplos del tiempo del ciclo menor.**
- Puede ser necesario partir una tarea en varios procedimientos.
- Es poco flexible y difícil de mantener: cada vez que se cambia una tarea hay que rehacer toda la planificación.

Ejercicio: Ejecutivo cíclico

Implemente una aplicación en ADA con 5 actividades periódicas, con períodos (T_i) de 250 ms, 250 ms, 500 ms, 500 ms, 750 ms, respectivamente. Estas actividades se limitarán a mostrar por pantalla su identificador (1, 2, 3, 4, 5) y la hora. Establezca una planificación de la ejecución de esas 5 actividades periódicas. El tiempo de cómputo (C_i) de las actividades es de 50 ms, 80 ms, 50 ms, 40 ms y 20 ms.

Realice un cronograma mostrando la evolución temporal con la ejecución de cada actividad, mostrando los 5 primeros ciclos.

¿Cuál es el factor de utilización?

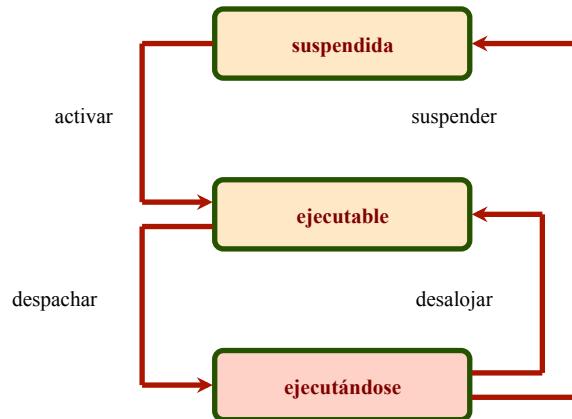
Índice

1. Introducción.
2. Planificación con ejecutivos cíclicos.
- 3. Planificación con prioridades de tareas periódicas independientes.**
4. Planificación con prioridades de tareas esporádicas y aperiódicas.
5. Interacción entre tareas y bloqueos en planificación con prioridades.

Tareas periódicas

En este método de planificación las tareas se realizan como procesos concurrentes que pueden estar en **varios estados**.

Las tareas ejecutables se despachan para su ejecución en **orden de prioridad**, donde el despacho puede hacerse con o sin desalojo, siendo la opción más frecuente el considerar *prioridades fijas con desalojo*.



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

Tareas periódicas

Existe una medida de la carga del procesador para un conjunto de tareas denominada **factor de utilización** del procesador, que viene dado por la expresión:

$$U = \sum_{i=1}^N (C_i / T_i)$$

donde en un sistema monoprocesador debe cumplirse ($U \leq 1$).

Existe una **condición de garantía** de los plazos basada en la utilización, de modo que para el modelo simple, con prioridades monótonas en frecuencia, **los plazos están garantizados si**

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

donde la cantidad $U_0(N) = N(2^{1/N}-1)$ es la **utilización mínima garantizada para N tareas**.

Tareas periódicas

Ejemplo de factor de utilización

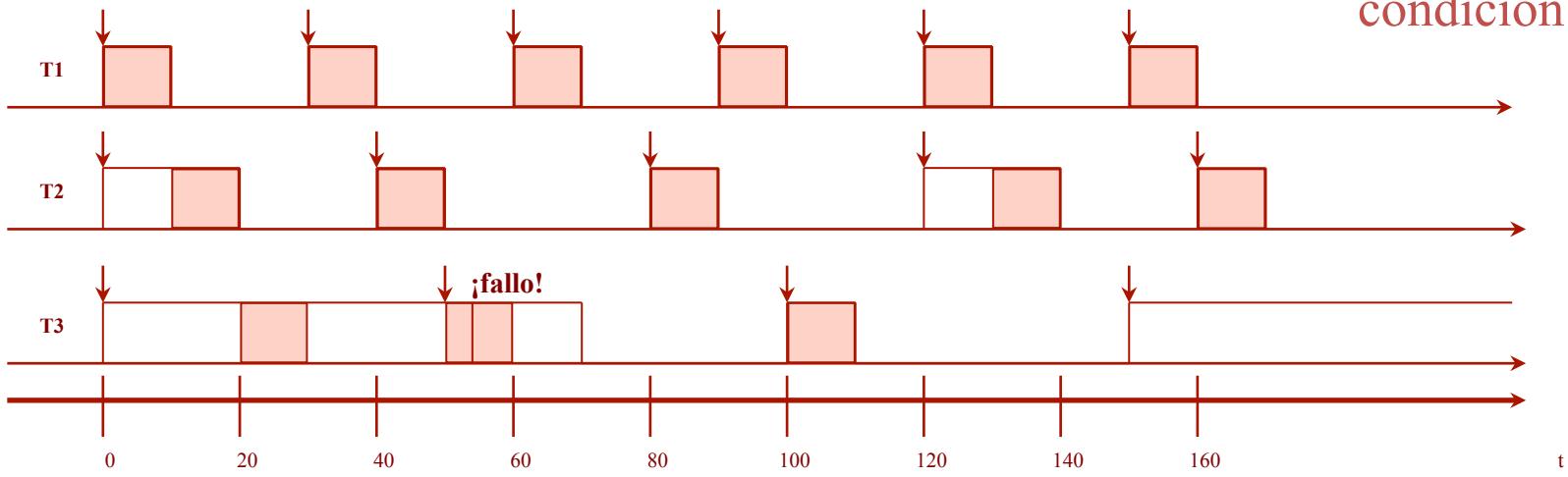
Tarea	T	C	P	U
T1	30	10	3	0.33
T2	40	10	2	0.250
T3	50	12	1	0.240
				0.82

N	$U_0(N)$
1	1.00
2	0.828
3	0.779
4	0.756
5	0.743
$\lim_{N \rightarrow \infty}$	
$\log 2 \approx 0.693$	

- El sistema no cumple la prueba de utilización ($U > 0.779$)
- La tarea 3 falla en $t=50$

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

condición de garantía



Tareas periódicas

Ejemplo de factor de utilización

Tarea	T	C	P	U
T1	16	4	3	0.250
T2	40	5	2	0.125
T3	80	32	1	0.400
				0.775

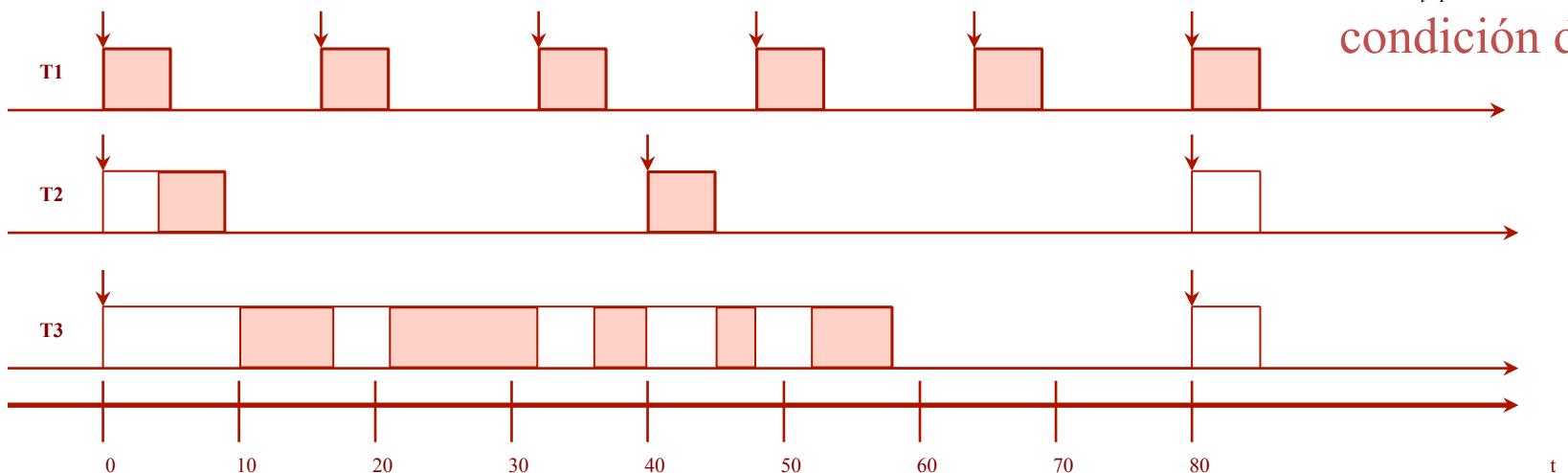
EJEMPLO DE PRUEBA DE UTILIZACIÓN

- Este sistema está garantizado ($U < 0.779$)

N	$U_0(N)$
1	1.00
2	0.828
3	0.779
4	0.756
5	0.743
$\lim_{N \rightarrow \infty} U_0(N) = \log 2 \approx 0.693$	

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

condición de garantía



Tareas periódicas

Ejemplo de factor de utilización

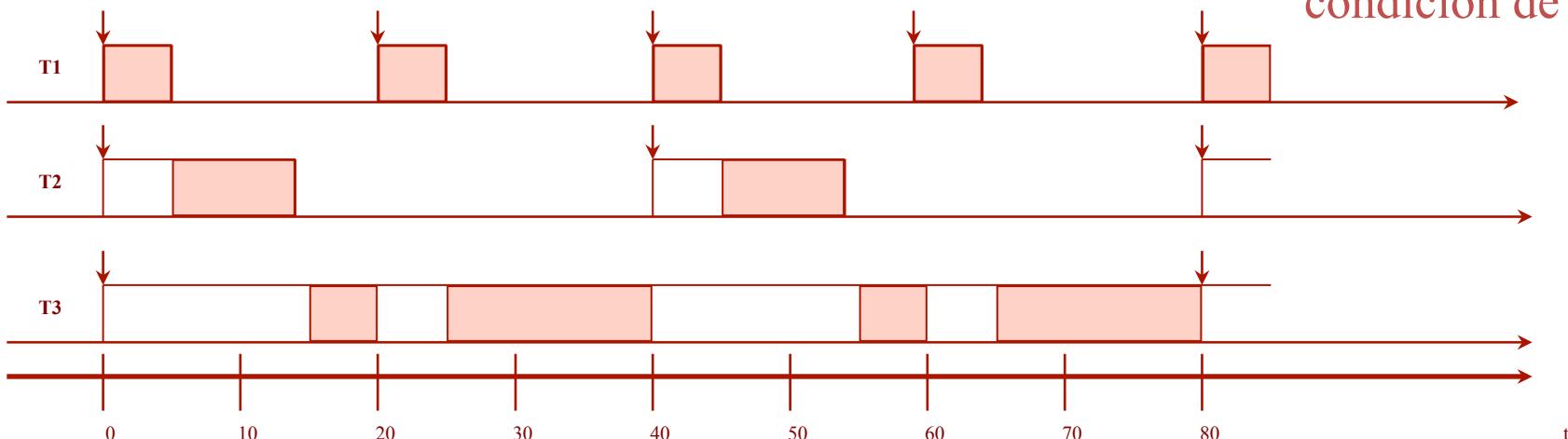
Tarea	T	C	P	U
T1	20	5	3	0.250
T2	40	10	2	0.250
T3	80	40	1	0.500
				1.000

EJEMPLO DE PRUEBA DE UTILIZACIÓN

- Este sistema no pasa la prueba ($U > 0.779$) pero se cumplen los plazos

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

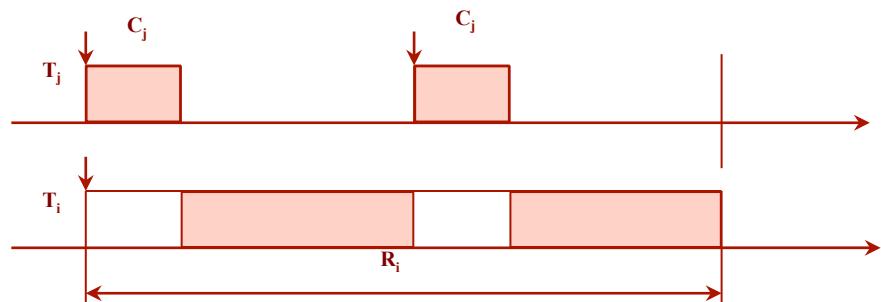
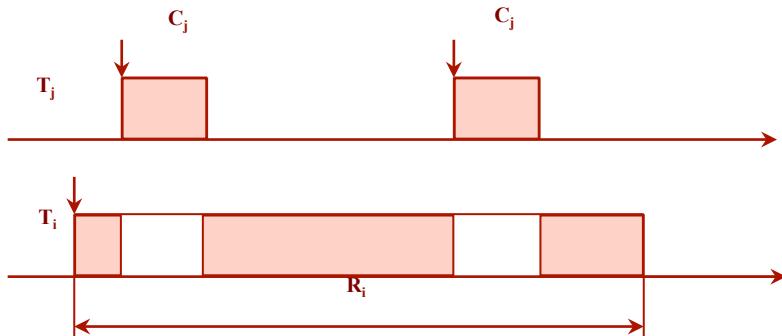
condición de garantía



Tareas periódicas

La condición de garantía indica una condición que sí se cumple los plazos están garantizados pero si no se cumple, los plazo se pueden cumplir o no, como ha pasado en el ejemplo anterior.

Se analiza a continuación una prueba basada en el **cálculo del tiempo de respuesta de cada tarea**. **El tiempo de respuesta de una tarea es la suma de su tiempo de cómputo más la *interferencia* que sufre por la ejecución de tareas más prioritarias ($R_i = C_i + I_i$).**



Tareas periódicas

El cálculo de la **interferencia máxima** que sufre una tarea se realiza usando las siguientes fórmulas ($hp(i)$ indica todas las tareas con prioridad mayor que la tarea i (**higher priority**) y la función techo ($\lceil \rceil$) da un redondeo superior:

- Para una tarea de prioridad superior:
- Para todas las tareas de prioridad superior:

$$I_i^j = \lceil R_i / T_j \rceil C_j$$

$$I_i = \sum_{j \in hp(i)} \lceil R_i / T_j \rceil C_j$$

La ecuación del **tiempo de respuesta** queda así:

$$R_i = C_i + \sum_{j \in hp(i)} \lceil R_i / T_j \rceil C_j$$

Tareas periódicas

R_i es la solución mínima de la ecuación:

$$w = C_i + \sum_{j \in hp(i)} \lceil w / T_j \rceil C_j$$

Como se observa, la ecuación no es continua ni lineal (debido a la función techo que acota superiormente los cálculos y que hace que existan infinitos valores que la cumplan), y no se puede resolver analíticamente.

Sin embargo se puede resolver mediante una ecuación de recurrencia:

$$w_i^{n+1} = C_i + \sum_{j \in hp(i)} \lceil w_i^n / T_j \rceil C_j \quad w_i^0 = C_i + \sum_{j \in hp(i)} C_j$$

Las iteraciones se terminan cuando $w^{n+1} = w^n$ (el conjunto de valores es monótonamente creciente) o bien $w^{n+1} > T_i$ (no se cumple el plazo).

Tareas periódicas

EJEMPLO CÁLCULO DE TIEMPO DE RESPUESTA:

$$w_i^{n+1} = C_i + \sum_{j \in hp(i)} \lceil w_i^n / T_j \rceil C_j \quad w_i^0 = C_i + \sum_{j \in hp(i)} C_j$$

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

T1: $w_1^0 = 3;$

T2: $w_2^0 = 3 + 3 = 6;$
 $w_2^1 = 3 + \lceil 6/7 \rceil 3 = 6;$

Recordar que la función techo ($\lceil \rceil$) da un redondeo superior.

T3: $w_3^0 = 5 + 3 + 3 = 11;$
 $w_3^1 = 5 + \lceil 11/7 \rceil 3 + \lceil 11/12 \rceil 3 = 14;$
 $w_3^2 = 5 + \lceil 14/7 \rceil 3 + \lceil 14/12 \rceil 3 = 17;$
 $w_3^3 = 5 + \lceil 17/7 \rceil 3 + \lceil 17/12 \rceil 3 = 20;$
 $w_3^4 = 5 + \lceil 20/7 \rceil 3 + \lceil 20/12 \rceil 3 = 20;$

Tareas periódicas

EJEMPLO CÁLCULO DE TIEMPO DE RESPUESTA:

$$w_i^{n+1} = C_i + \sum_{j \in hp(i)} \lceil w_i^n / T_j \rceil C_j \quad w_i^0 = C_i + \sum_{j \in hp(i)} C_j$$

Tarea	T	C	P	R
T1	20	5	3	?
T2	40	10	2	?
T3	80	40	1	?

T1: $w_1^0 = 5;$

T2: $w_2^0 = 10 + 5 = 15;$
 $w_2^1 = 10 + \lceil 15/20 \rceil 5 = 15;$

T3: $w_3^0 = 40 + 5 + 10 = 55;$
 $w_3^1 = 40 + \lceil 55/20 \rceil 5 + \lceil 55/40 \rceil 10 = 75;$
 $w_3^2 = 40 + \lceil 75/20 \rceil 5 + \lceil 75/40 \rceil 10 = 80;$
 $w_3^2 = 40 + \lceil 80/20 \rceil 5 + \lceil 80/40 \rceil 10 = 80;$

Tareas periódicas

EJEMPLO CÁLCULO DE TIEMPO DE RESPUESTA:

$$w_i^{n+1} = C_i + \sum_{j \in hp(i)} \left\lceil w_i^n / T_j \right\rceil C_j \quad w_i^0 = C_i + \sum_{j \in hp(i)} C_j$$

Tarea	T	C	P	R
T1	30	10	3	?
T2	40	10	2	?
T3	50	12	1	?

Tareas periódicas

EJEMPLO CÁLCULO DE TIEMPO DE RESPUESTA:

$$w_i^{n+1} = C_i + \sum_{j \in hp(i)} \left\lceil w_i^n / T_j \right\rceil C_j \quad w_i^0 = C_i + \sum_{j \in hp(i)} C_j$$

Tarea	T	C	P	R
T1	30	10	3	?
T2	40	10	2	?
T3	50	12	1	?

T1: $w_1^0 = 10;$

T2: $w_2^0 = 10 + 10 = 20;$
 $w_2^1 = 10 + [20/30] 10 = 20;$

T3: $w_3^0 = 12 + 10 + 10 = 32;$
 $w_3^1 = 12 + [32/30] 10 + [32/40] 10 = 42;$
 $w_3^2 = 12 + [42/30] 10 + [42/40] 10 = 52;$

Plazo no garantizado (ver transp. 23)

Índice

1. Introducción.
2. Planificación con ejecutivos cíclicos.
3. Planificación con prioridades de tareas periódicas independientes.
- 4. Planificación con prioridades de tareas esporádicas y aperiódicas.**
5. Interacción entre tareas y bloqueos en planificación con prioridades.

Tareas esporádicas y aperiódicas

Las tareas aperiódicas pueden ser:

- **críticas** (hard) o **esporádicas**: tienen un plazo de respuesta crítico (no puede fallar) y se caracterizan por una separación mínima entre dos sucesos de activación consecutivos.
- **acríticas** (soft): pueden responder tarde ocasionalmente y se caracterizan por un esquema de activación estocástico.

Deben garantizarse los plazos de todas las tareas en condiciones normales y los plazos de las tareas críticas en las peores condiciones. 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 periodo T).**
- El plazo de respuesta puede ser menor que el periodo ($D \leq T$).

Tareas esporádicas y aperiódicas

El análisis de tiempo de respuesta sigue siendo válido bajo esos supuestos y el esquema funciona bien *con cualquier orden de prioridad*.

Cuando los plazos son menores o iguales que los periodos, la asignación de mayor prioridad a las tareas de **menor plazo de respuesta** (*deadline monotonic scheduling*) es óptima.

El tiempo de respuesta se calcula de la misma forma que con la asignación monótona en frecuencia: se termina cuando $w^{n+1} = w^n$ o bien $w^{n+1} > D_i$.

Deadline monotonic scheduling

Tarea	T	D	C	P	R
T1	20	5	3	4	3
T2	15	7	3	3	6
T3	10	10	4	2	10
T4	20	20	3	1	20

Rate monotonic scheduling

Tarea	T	D	C	P	R
T1	10	10	4	4	4
T2	15	7	3	3	7
T3	20	5	3	2	10
T4	20	20	3	1	20

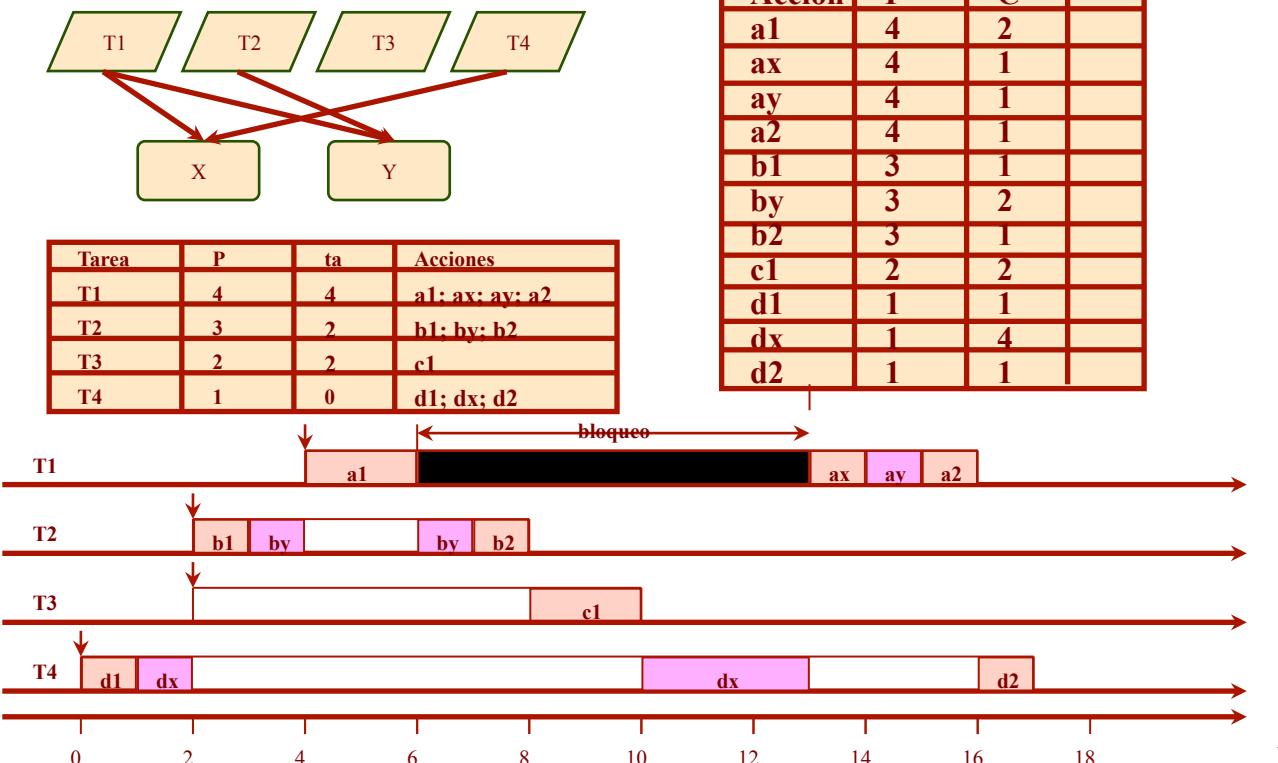
Índice

1. Introducción.
2. Planificación con ejecutivos cíclicos.
3. Planificación con prioridades de tareas periódicas independientes.
4. Planificación con prioridades de tareas esporádicas y aperiódicas.
- 5. Interacción entre tareas y bloqueos en planificación con prioridades.**

Interacción tareas y bloques

En la mayoría de los sistemas de interés práctico las tareas interaccionan mediante datos comunes (protegidos) y citas o mensajes.

En todos estos casos puede ocurrir que la tarea tenga que esperar un suceso de otra menos prioritaria. Esta situación se denomina **bloqueo** y produce una **inversión de prioridad** indeseable, que no se puede eliminar completamente, pero que es posible limitar su duración.

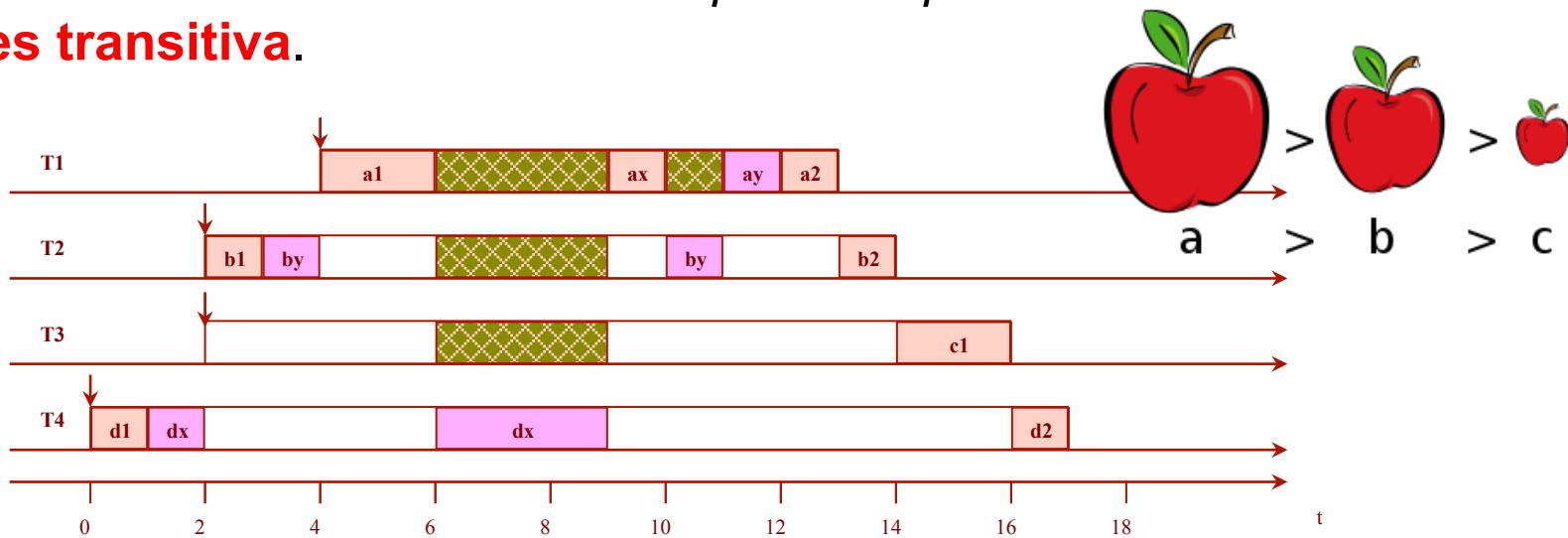


Interacción tareas y bloques

Herencia de prioridad

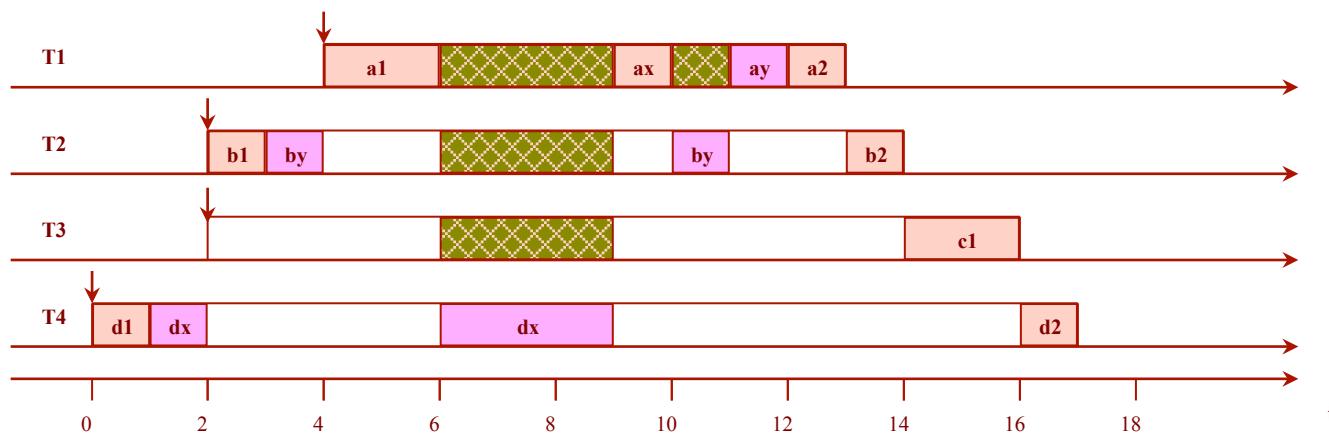
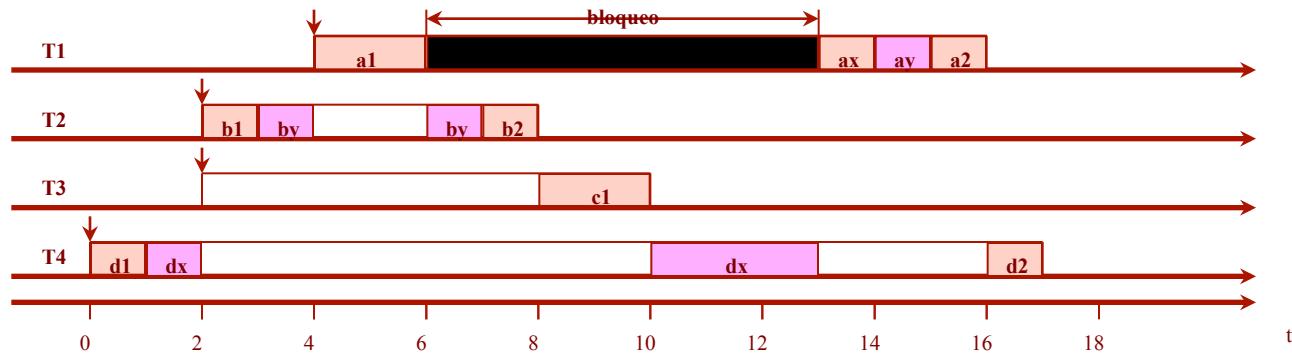
Una forma de reducir la duración de los bloqueos es *variando dinámicamente la prioridad de las tareas*. Cuando una tarea está bloqueando a otra más prioritaria, **hereda** la prioridad de ésta.

La prioridad dinámica de una tarea es el *máximo de su prioridad básica o de las prioridades de todas las tareas bloqueadas por ella*. **La herencia de prioridad es transitiva**.



Interacción tareas y bloques

Herencia de prioridad



Interacción tareas y bloques

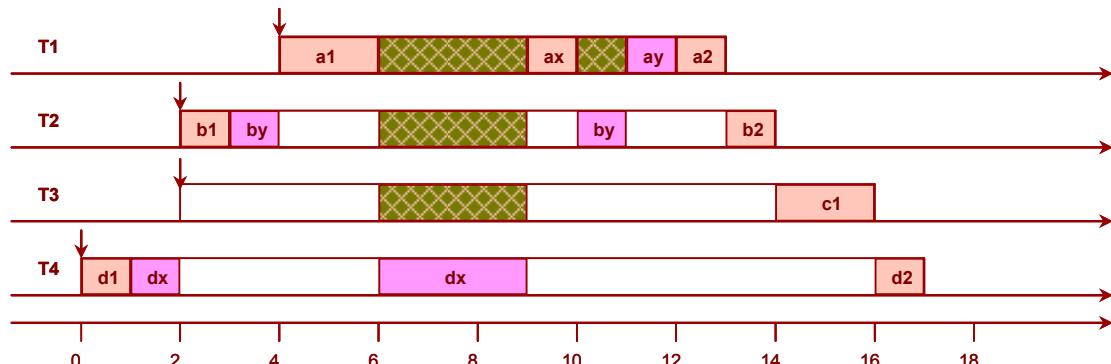
Herencia de prioridad

La duración total máxima de los bloqueos es: $B_i = \sum_{j \in lp(i)} C_{j,k}$

donde $lp(i)$ son las tareas con menor prioridad (*less priority*) que la tarea i , k es el número de recursos compartidos y $C_{j,k}$ es el tiempo durante el cual la tarea j accede al recurso k (igual a cero si no se usa el recurso).

Tarea	P	ta	Acciones
T1	4	4	a1; ax; av; a2
T2	3	2	b1; bv; b2
T3	2	2	c1
T4	1	0	d1; dx; d2

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



Interacción tareas y bloques

Herencia de prioridad

Una tarea puede bloquearse por recursos a los que no accede (*por ejemplo T2*). Además, una tarea puede sufrir bloqueo aunque no acceda a recursos compartidos (*por ejemplo T3*). La tarea de menor prioridad (*T4*) no sufre bloqueo.

Cuando hay bloqueos, la ecuación del tiempo de respuesta queda así:

$$R_i = C_i + B_i + \sum_{j \in hp(i)} \lceil R_i / T_j \rceil C_j \quad w_i^{n+1} = C_i + B_i + \sum_{j \in hp(i)} \lceil w_i^n / T_j \rceil C_j \quad B_i = \sum_{j \in p(i)} C_{j,k}$$

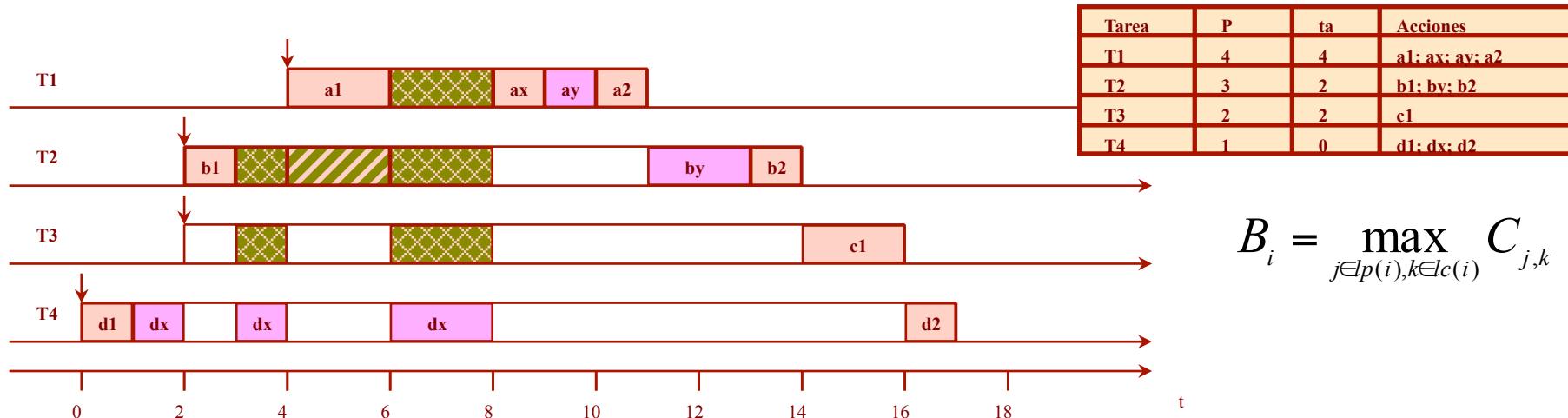
$$w_i^0 = C_i + B_i + \sum_{j \in hp(i)} C_j$$

Interacción tareas y bloques

Techo de prioridad

El techo de prioridad de un recurso es la máxima prioridad de las tareas que lo usan. El protocolo se define de la siguiente forma:

- 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.

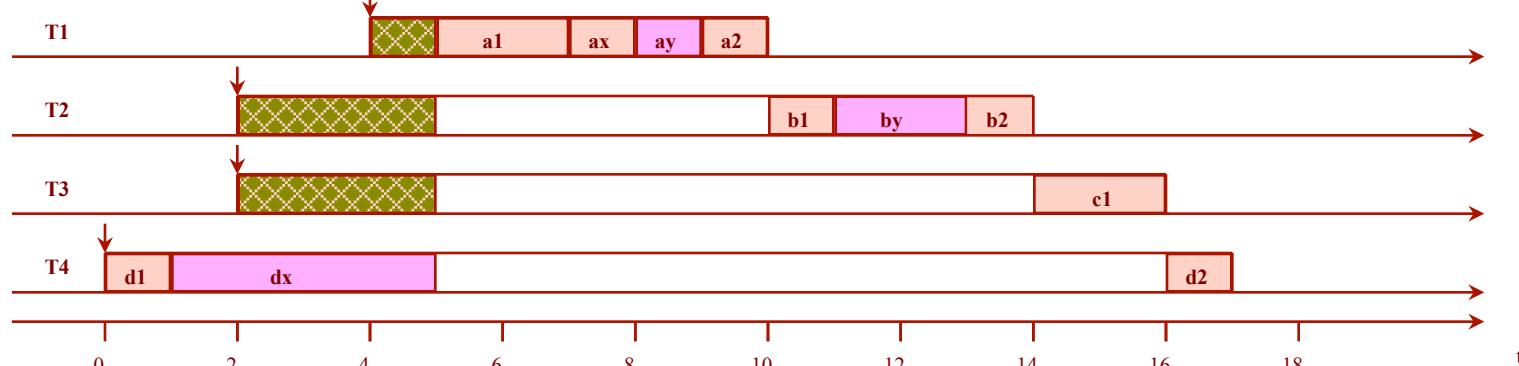


Interacción tareas y bloques

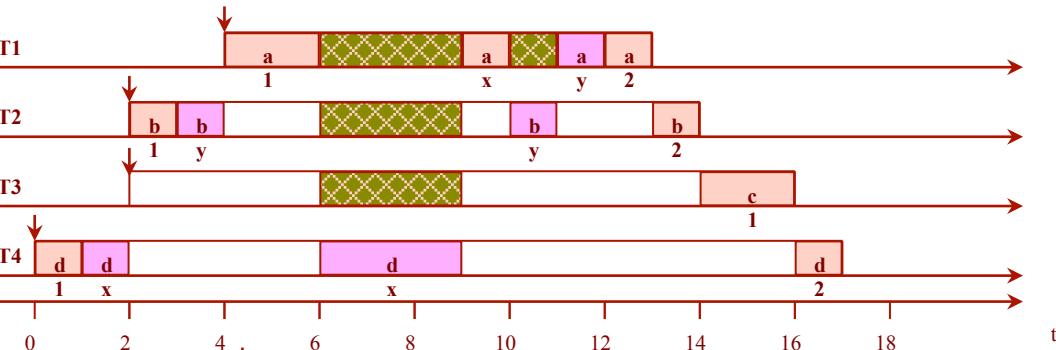
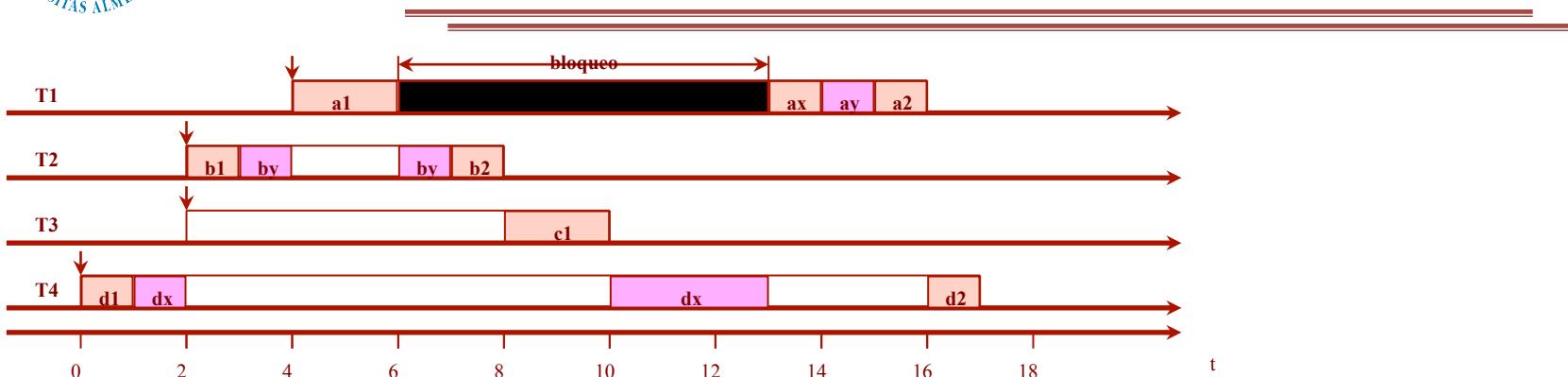
Techo de prioridad

Existe una modificación al protocolo anterior denominado **protocolo del techo de prioridad inmediato**, que tiene las siguientes propiedades:

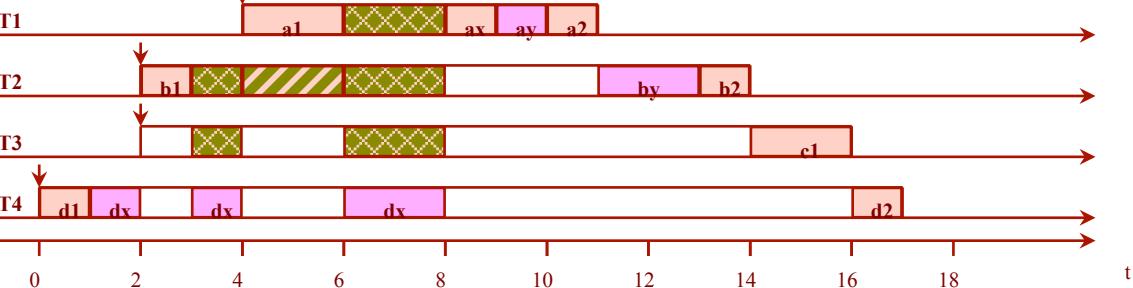
- Con este protocolo, una tarea que accede a un recurso hereda inmediatamente el techo de prioridad del recurso (la prioridad dinámica de la tarea es el máximo de su prioridad básica y los techos de prioridad de los recursos que usa).
- Las propiedades son las mismas que las del protocolo del techo de prioridad, y además, si una tarea se bloquea, lo hace al principio del ciclo.
- Es más fácil de implementar que el protocolo básico y más eficiente (menos cambios de contexto).



Interacción tareas y bloques

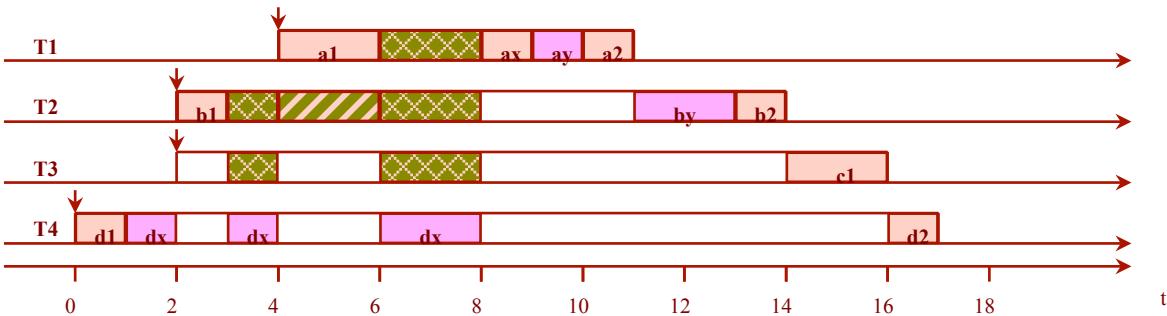
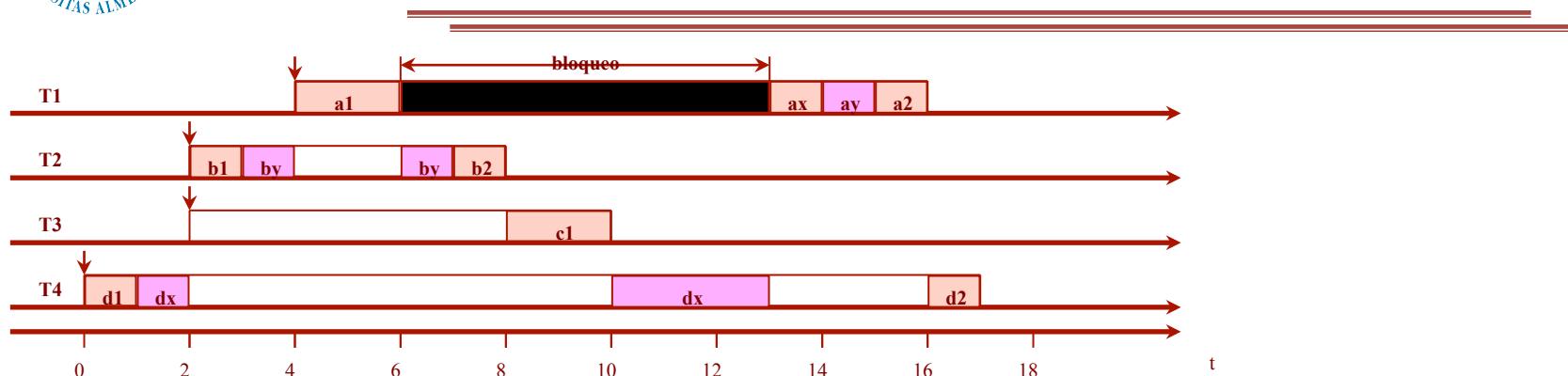


Herencia de prioridad

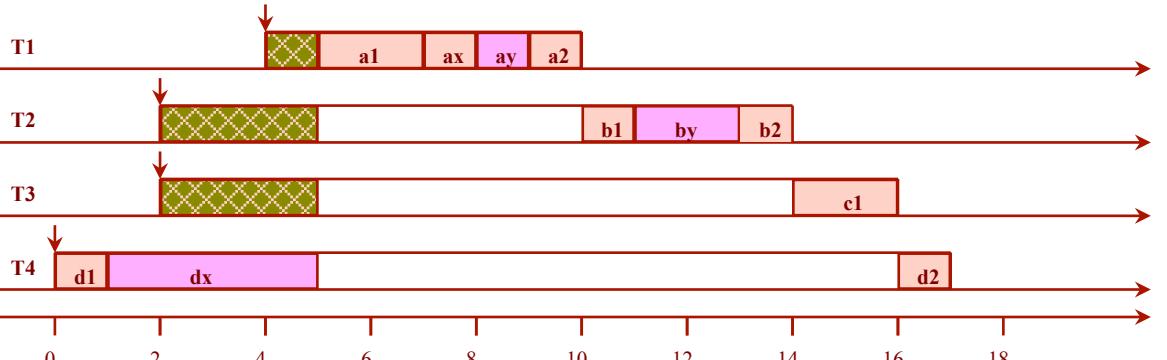


Techo de prioridad

Interacción tareas y bloques



Techo de prioridad



Techo de prioridad
inmediato

Ejercicio: Herencia de prioridad

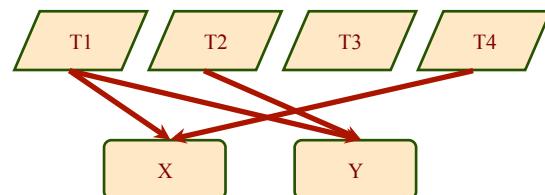
$$R_i = C_i + B_i + \sum_{j \in hp(i)} \lceil R_i / T_j \rceil C_j$$

$$w_i^{n+1} = C_i + B_i + \sum_{j \in hp(i)} \lceil w_i^n / T_j \rceil C_j$$

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

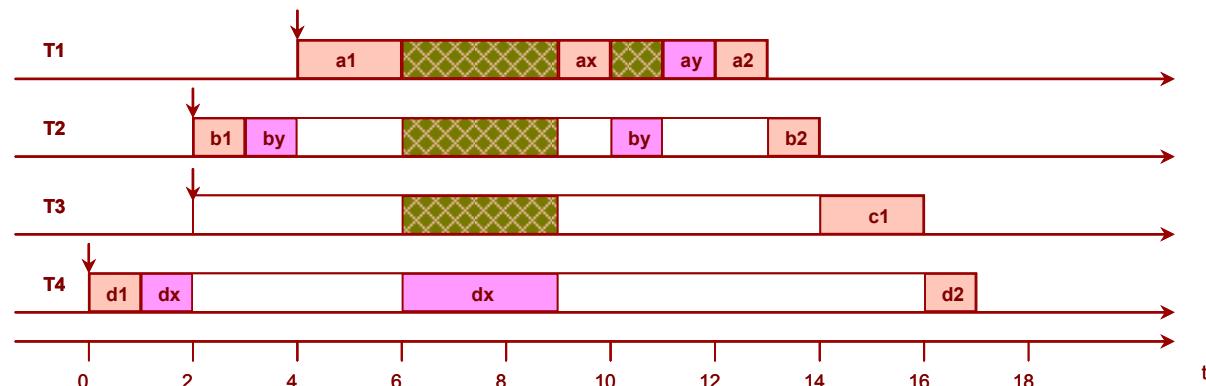
$$B_i = \sum_{j \in p(i)} C_{j,k}$$

Calcular el tiempo de respuesta para las 4 tareas



I	Tarea	P	ta	Acciones
15	T1	4	4	a1; ax; ay; a2
15	T2	3	2	b1; by; b2
20	T3	2	2	c1
40	T4	1	0	d1; dx; d2

Acción	P	C
a1	4	2
ax	4	1
ay	4	1
a2	4	1
b1	3	1
by	3	2
b2	3	1
c1	2	2
d1	1	1
dx	1	4
d2	1	1



Tema 5

Planificación de Tareas



Sistemas de Tiempo Real
(3º Grado de Ingeniería Informática)
Área de Ingeniería de Sistemas y Automática
Departamento de Informática
Universidad de Almería