Tema 9

Planificación de Recursos

Objetivos

1. Conocer el problema de la **planificación con** acceso concurrente a **recursos compartidos** en modo exclusivo

2. Conocer y entender el funcionamiento de distintos **algoritmos** de planificación de recursos

Índice

Dependencia entre tareas

- Generalmente en un STR existe dependencia entre tareas
- Tipos de dependencias:
  - **Restricciones de precedencia**
    - Sincronización y comunicación
  - **Restricciones de exclusión mútua** para proteger recursos compartidos

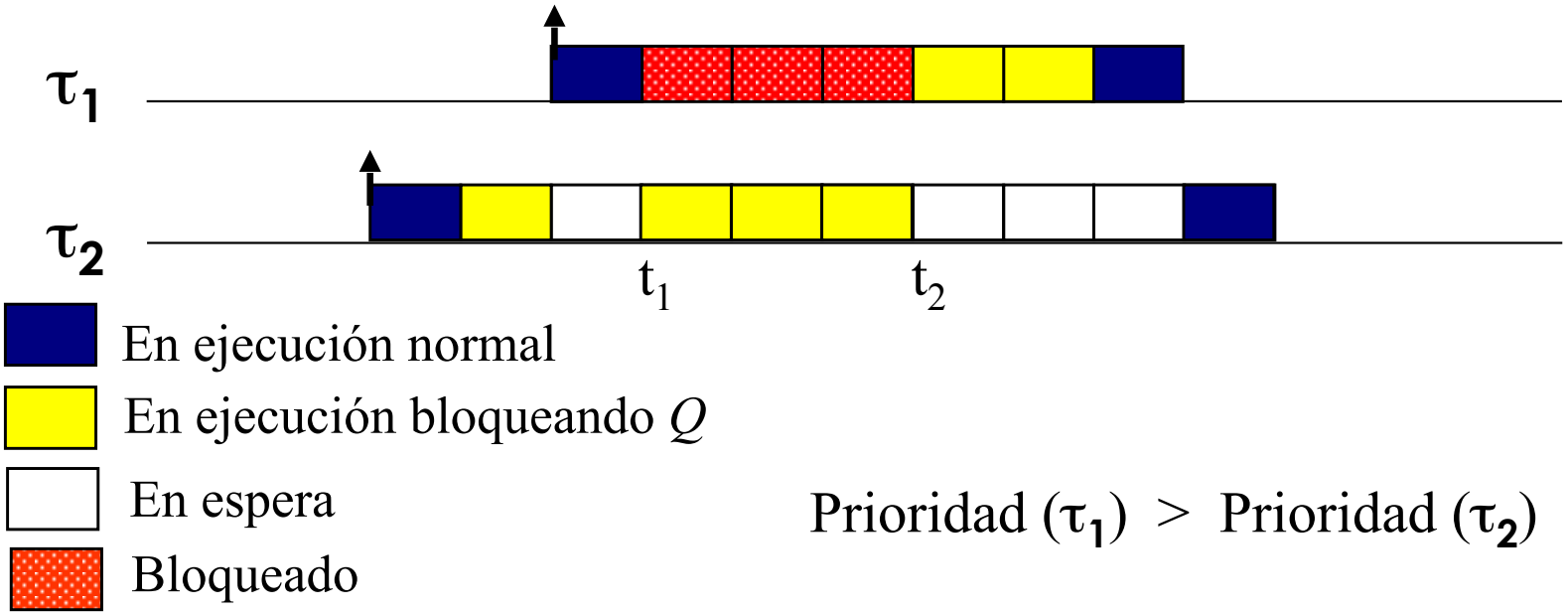
Bloqueo entre tareas

- Los procesos interactúan mediante:
  - Datos compartidos
  - Paso de mensajes
- En ambos casos puede ocurrir que una tarea tenga que esperar un suceso de otra menos prioritaria
- Esta situación se denomina **bloqueo**

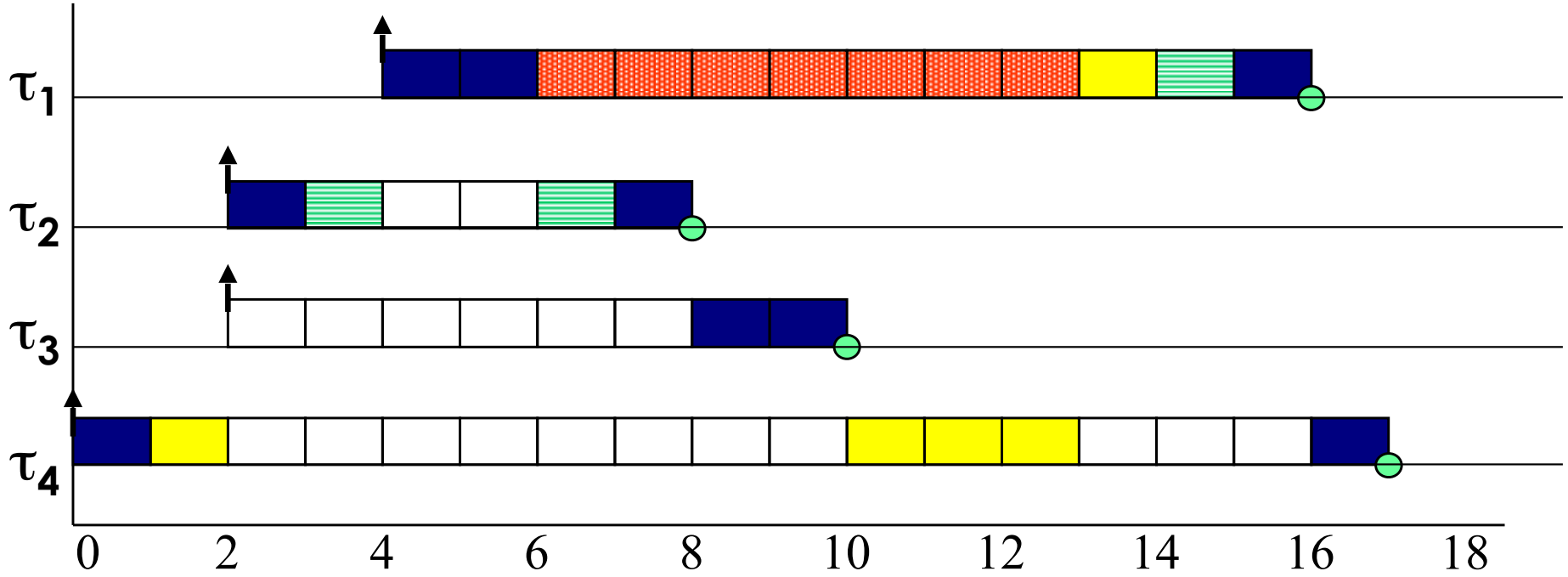
Índice

El problema de la inversión de prioridad

- Un proceso está **bloqueado** cuando está esperando debido a otro de menor prioridad
- A este hecho se le denomina **inversión de prioridad**
- La inversión de prioridad no se puede eliminar completamente, pero es posible limitar su duración



Ejemplo de inversión de prioridad



En ejecución normal

En ejecución bloqueando Q

En ejecución bloqueando V

En espera

Bloqueado

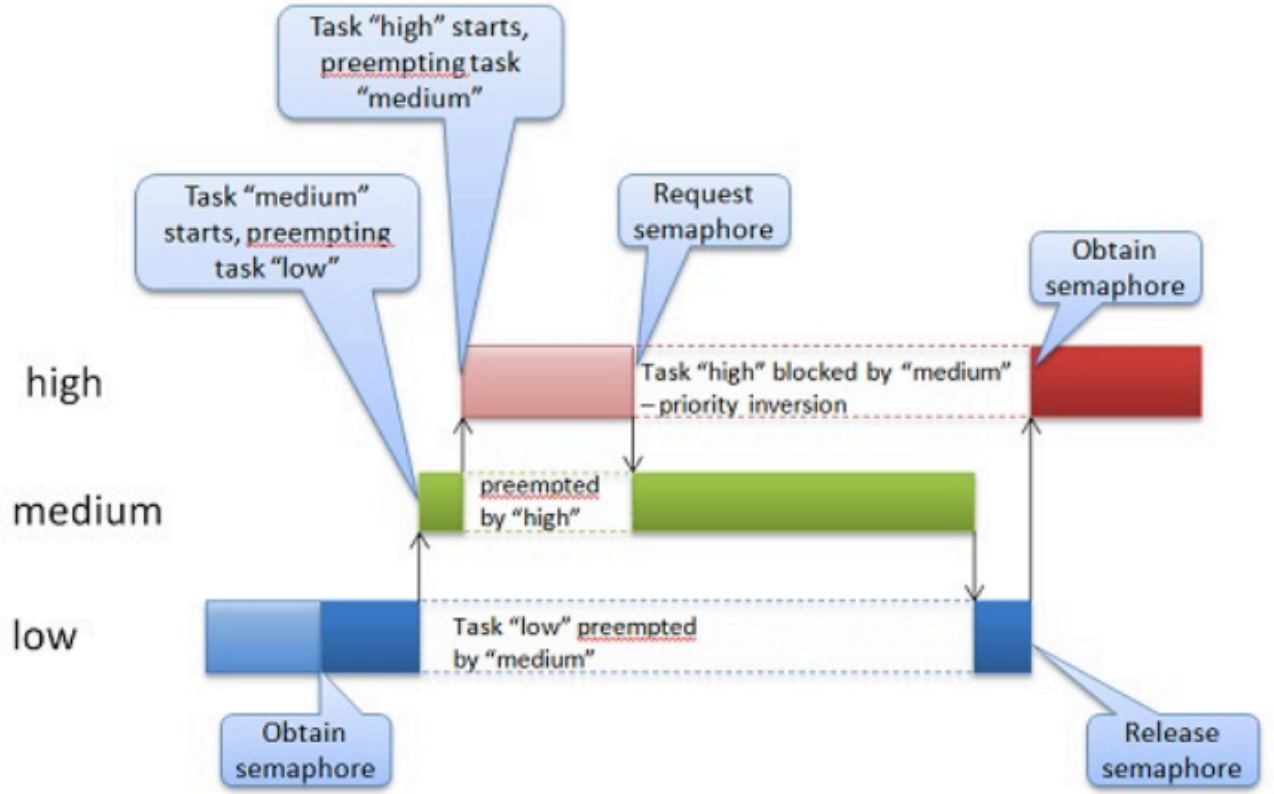
	P_i	Sec. Ejec.	r_i
τ_1	4	EEQVE	4
τ_2	3	EVVE	2
τ_3	2	EE	2
τ_4	1	EQQQQE	0

Caso de estudio : Misión espacial *Mars Pathfinder*

https://mars.nasa.gov/mars-exploration/missions/pathfinder/

https://www.cs.unc.edu/~anderson/teach/comp790/papers/mars_pathfinder_long_version.html

Caso de estudio : Misión espacial *Mars Pathfinder*



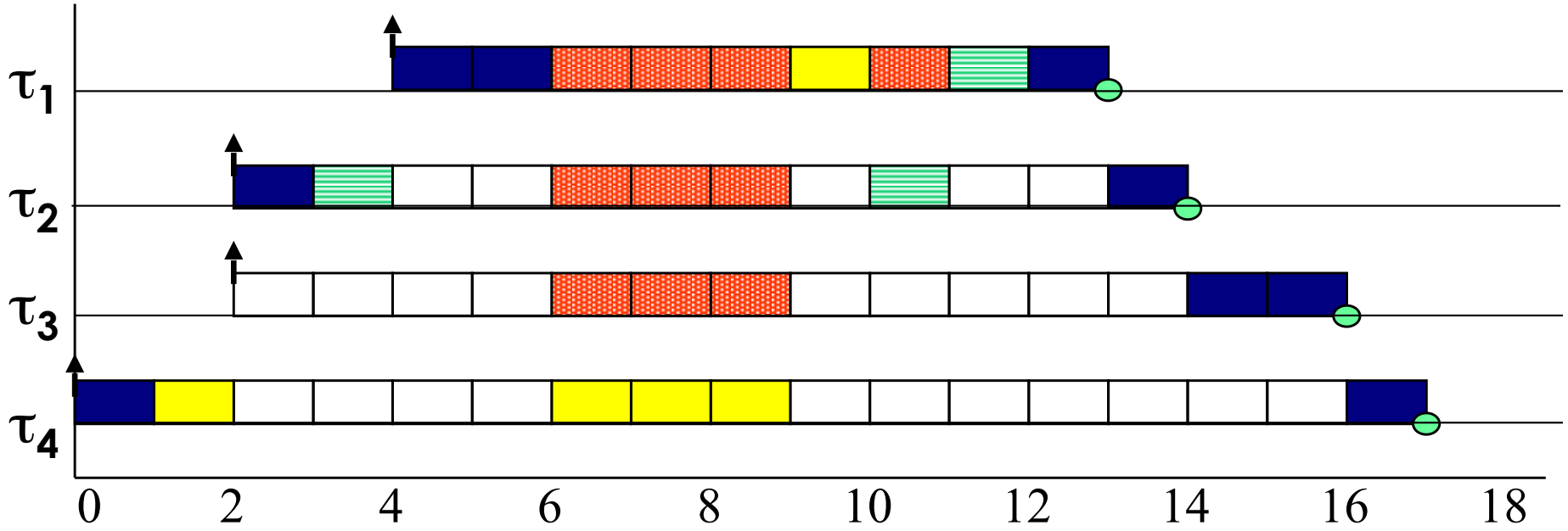
https://www.rapitasystems.com/blog/what-really-happened-to-the-software-on-the-mars-pathfinder-spacecraft

Índice

Herencia de prioridades

- Una forma de reducir la duración de los bloqueos es **variar dinámicamente la prioridad** de las tareas
  - Si la tarea τ_2 está bloqueando a la tarea τ_1, entonces τ_2 se ejecutará **temporalmente** con la prioridad de τ_1
- En cada tarea se distingue entre:
  - una **prioridad fija por defecto** P_i , p.ej. asignada según RM
  - y una **prioridad activa** p_i asignada dinámicamente ($p_i > Pi$)
- La herencia de prioridad es **transitiva**
  - Si la tarea τ_3 bloquea a la tarea τ_2, y τ_2 bloquea a la tarea τ_1, entonces τ_3 hereda la prioridad de τ_1 a través de τ_2

Ejemplo de herencia de prioridades



En ejecución normal

En ejecución bloqueando Q

En ejecución bloqueando V

En espera

Bloqueado

	P_i	Sec. Ejec.	r_i
τ_1	4	EEQVE	4
τ_2	3	EVVE	2
τ_3	2	EE	2
τ_4	1	EQQQQE	0

Herencia de prioridades en Ada

prioridad Activa = Mayor(prioridad Base, prioridad Heredada)

Duración herencia	Prioridad heredada
Activación de la tarea	La del padre
Ejecución de una operación protegida	Techo de prioridad del objeto protegido
Ejecución de una cita	la del cliente

Duración máxima de bloqueo

- El protocolo de herencia de prioridad limita el tiempo máximo de bloqueo de cada tarea
- Una tarea puede bloquearse por recursos a los que no accede
- Una tarea puede bloquearse aunque no acceda a ningún recurso
- La tarea de menor prioridad no sufre bloqueo

Cálculo de duración máxima de bloqueo

$$B_i = \sum_{k=1}^{K} \text{utilizacion}(k,i)\ C(k)$$

$$\text{utilizacion}(k,i) = \begin{cases} \mathbf{1} & \text{si } \exists\ j:\ P_j < P_i \ \text{ y } \ \exists\ x : P_x \geq P_i\ ; \quad j,x \text{ usan recurso } k \\ \mathbf{0} & \text{en otro caso} \end{cases}$$

C(k) = duración de bloqueo del recurso k **en el peor caso,** por tareas de menor prioridad

	P_i	Sec. Ejec.	r_i
τ_1	4	EEQVE	4
τ_2	3	EVVE	2
τ_3	2	EE	2
τ_4	1	EQQQQE	0

$B_{t1} = C_{t2,V} + C_{t4,Q} = 2 + 4 = 6$
$B_{t2} = C_{t4,Q} = 4$
$B_{t3} = C_{t4,Q} = 4$
$B_{t4} = 0$

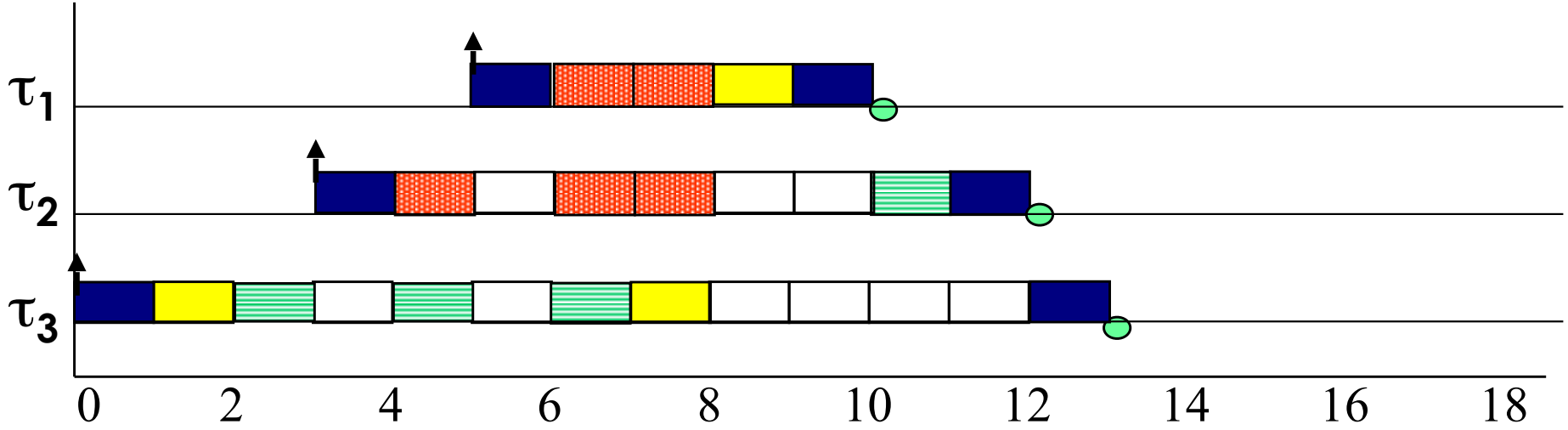
Test de planificabilidad con bloqueos

Test de tiempos de respuesta (cuando $D_i \leq T_i$) :

$$\forall i, R_i = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \cdot C_j \leq D_i$$

Este test es condición suficiente

Ejemplo con acceso anidado



En ejecución normal

En ejecución bloqueando Q

En ejecución bloqueando V

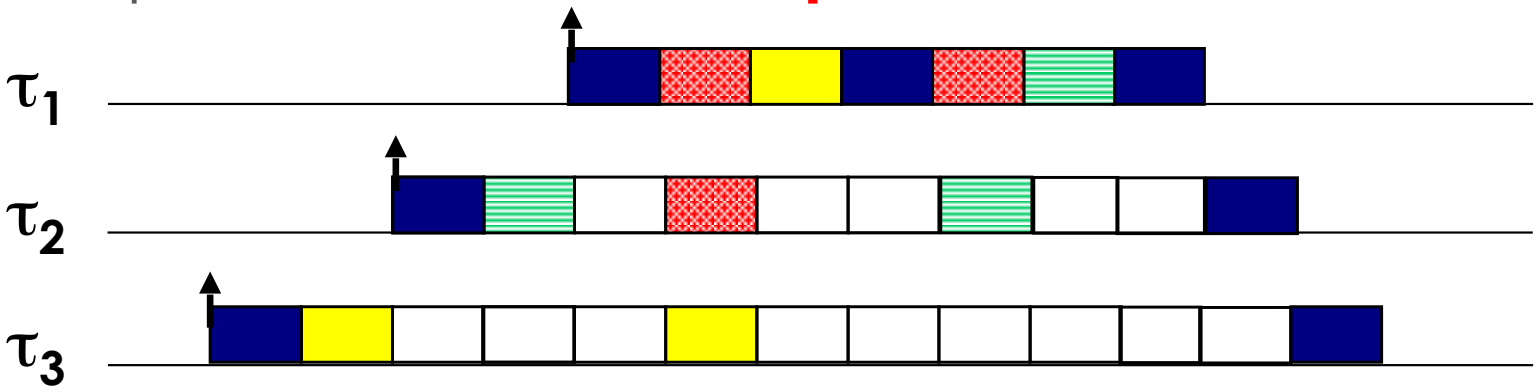
En espera

Bloqueado

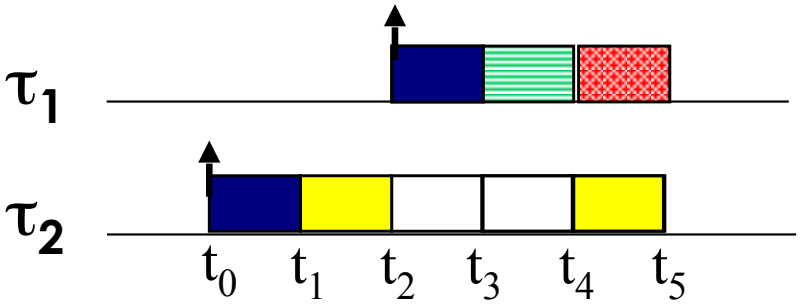
	P_i	Sec. Ejec.	r_i
τ_1	3	EQE	5
τ_2	2	EVE	3
τ_3	1	EQVVVQE	0

Inconvenientes del protocolo

- Se pueden formar **bloqueos encadenados**



- No previene **interbloqueos** (***deadlocks***)



	P_i	Sec. Ejec.	r_i
τ_1	2	EVQVE	t_1
τ_2	1	EQQVQE	t_0

Índice

1. Interacción entre tareas
2. Inversión de prioridades
3. Protocolo de herencia de prioridades
4. **Protocolos de techo de prioridad**

Ventajas

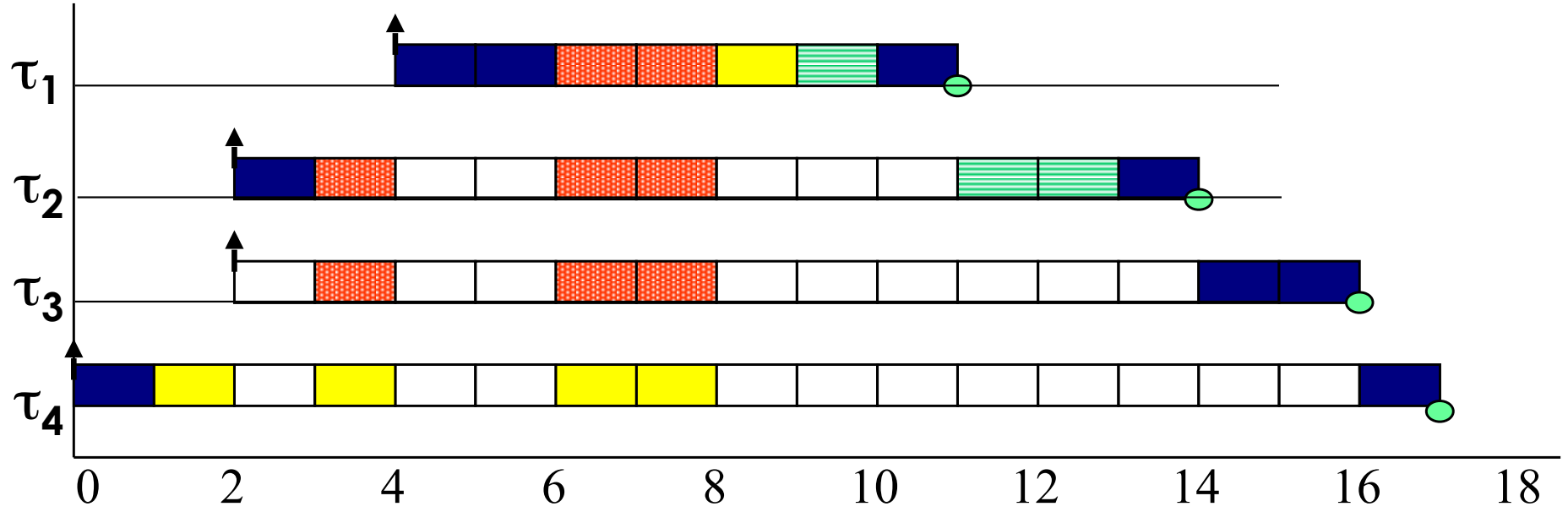
- Previene la formación de
  - bloqueos encadenados
  - *deadlocks*
- Un proceso de alta prioridad puede ser bloqueado como máximo en una sola ocasión en cada activación

OCPP (*Original Ceiling Priority Protocol*)

- Cada tarea tiene una **prioridad estática**
  - asignada por defecto
- Cada recurso tiene un **techo de prioridad** asignado
  - máxima prioridad de las tareas que lo pueden usar
- Cada tarea tiene una **prioridad dinámica**
  - máximo entre su propia prioridad estática y cualquiera que herede debido a que bloquea a tareas más prioritarias

Una tarea puede **bloquear** un recurso solamente si su prioridad dinámica es mayor que el techo de cualquier recurso actualmente bloqueado por otras tareas

Ejemplo de OCPP



En ejecución normal

En ejecución bloqueando Q

En ejecución bloqueando V

En espera

Bloqueado

	P_i	Sec. Ejec.	r_i
τ_1	4	EEQVE	4
τ_2	3	EVVE	2
τ_3	2	EE	2
τ_4	1	EQQQQE	0

Duración máxima de bloqueo

- La duración máxima del bloqueo con un protocolo OCPP de una determinada tarea es el tiempo de ejecución de la sección crítica mayor que puede bloquear dicha tarea

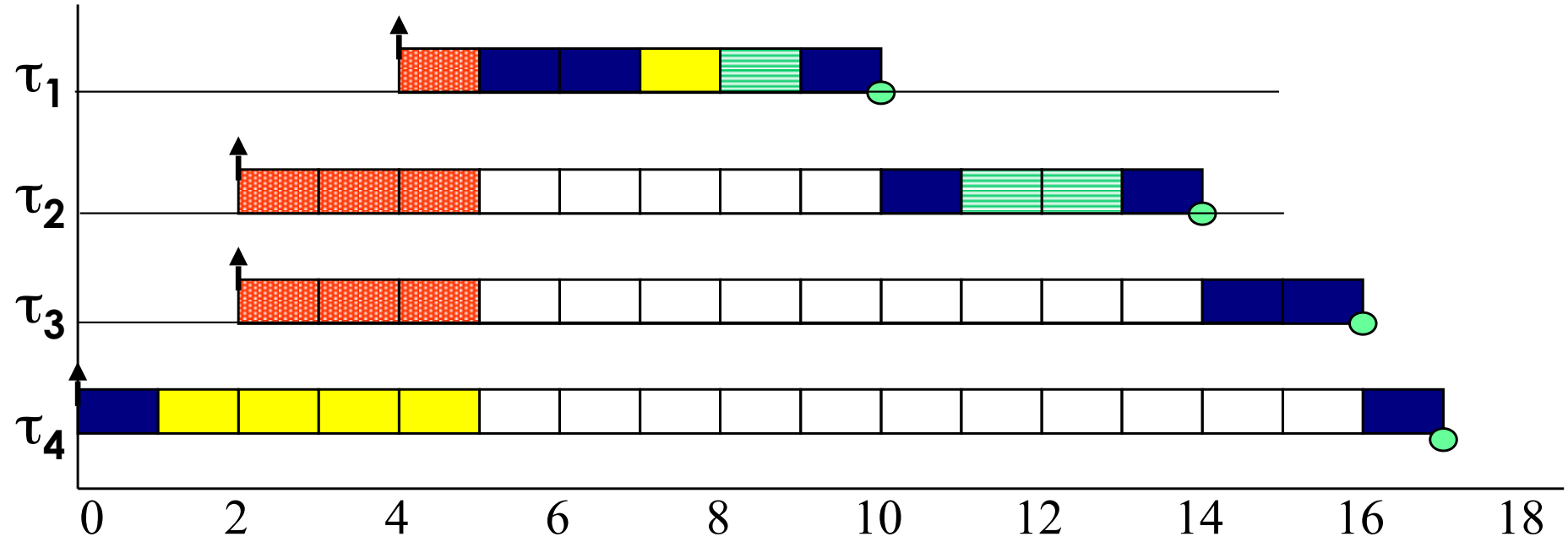
$$B_i = \max_{k=1}^{K} \text{utilizacion}(k,i)\ C(k)$$

Ejercicio: calcula el bloqueo máximo para cada una de las tareas del ejemplo anterior

ICPP (*Immediate Ceiling Priority Protocol*)

- Cada tarea tiene una **prioridad estática**
  - asignada por defecto
- Cada recurso tiene un **techo de prioridad** asignado
  - máxima prioridad de las tareas que lo pueden usar
- Cada tarea tiene una **prioridad dinámica**
  - máximo entre su prioridad estática y los valores "techo" de cualquier recurso que en ese momento tenga bloqueado

Ejemplo de ICPP



En ejecución normal

En ejecución bloqueando Q

En ejecución bloqueando V

En espera

Bloqueado

	P_i	Sec. Ejec.	r_i
τ_1	4	EEQVE	4
τ_2	3	EVVE	2
τ_3	2	EE	2
τ_4	1	EQQQQE	0

ICPP en Ada

`Pragma` Locking_Policy(*Ceiling_Locking*)

- A los objetos protegidos se les puede asignar una prioridad utilizando el pragma Priority
- Esta prioridad debe ser mayor o igual que las prioridades de las tareas que usan el objeto protegido

ICPP vs OCPP

- El tiempo de respuesta en el peor de los casos se calcula igual que para OCPP
- Es más fácil de implementar que un OCPP
- Provoca menos cambios de contexto
- Requiere más "cambios" de prioridad (OCPP solamente cambia la prioridad si ocurre un bloqueo real)

Conclusiones

- Generalmente en STR las dependencias entre tareas son necesarias para realizar actividades de control
- Los bloqueos no se pueden evitar pero utilizando determinados protocolos se puede limitar el tiempo máximo de bloqueo de cada tarea
- Los protocolos de techo de prioridad previenen la formación de bloqueos encadenados e interbloqueos

Bibliografía Recomendada

Sistemas de tiempo real y lenguajes de programación (**3ª edición**)
Alan Burns and Andy Wellings
Addison Wesley (2002)

- Capítulo 13 (Apartados 13.10 y 13.11)

Hard real-time computing systems (**Second edition**)
Giorgio C. Buttazzo
Kluwer Academic Publishers (2004)

- Capítulo 7 (excepto apartado 7.5)

Otras fuentes de información

Manual de Referencia de Ada2005

? Anexo: D.3