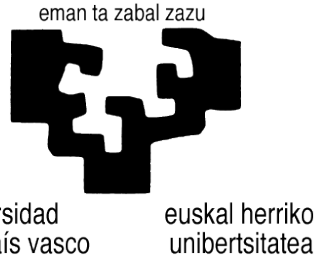
UPV / EHU



Programación Concurrente en Linux

El problema del interbloqueo

Alberto Lafuente, Dep. KAT/ATC de la UPV/EHU, bajo Licencia Creative Commons

UPV / EHU

Contenido

1. Inanición e interbloqueo
2. Modelo del interbloqueo
3. Soluciones al interbloqueo

OCW OpenCourseWare

UPV / EHU

1. Inanición e interbloqueo

OCW OpenCourseWare

Inanición e interbloqueo

- El acceso a recursos compartidos no está libre de problemas:
  - *Inanición*: un proceso/hilo espera indefinidamente para entrar a la sección crítica.
    - Posible escenario: Algunas primitivas de sincronización pueden estar implementadas con criterios de prioridad para seleccionar el siguiente proceso a entrar en la sección crítica.
  - *Interbloqueo*: es una situación de reciprocidad en el acceso a recursos múltiples.
    - Un conjunto de procesos está interbloqueado cuando cada proceso está esperando por un recurso que está siendo usado por otro proceso del conjunto.
    - ¡Aunque las primitivas cumplan rigurosamente las propiedades del acceso exclusivo!

Interbloqueo
Un escenario sencillo

- El siguiente código puede conducir a un interbloqueo entre P1 y P2

Proceso P1

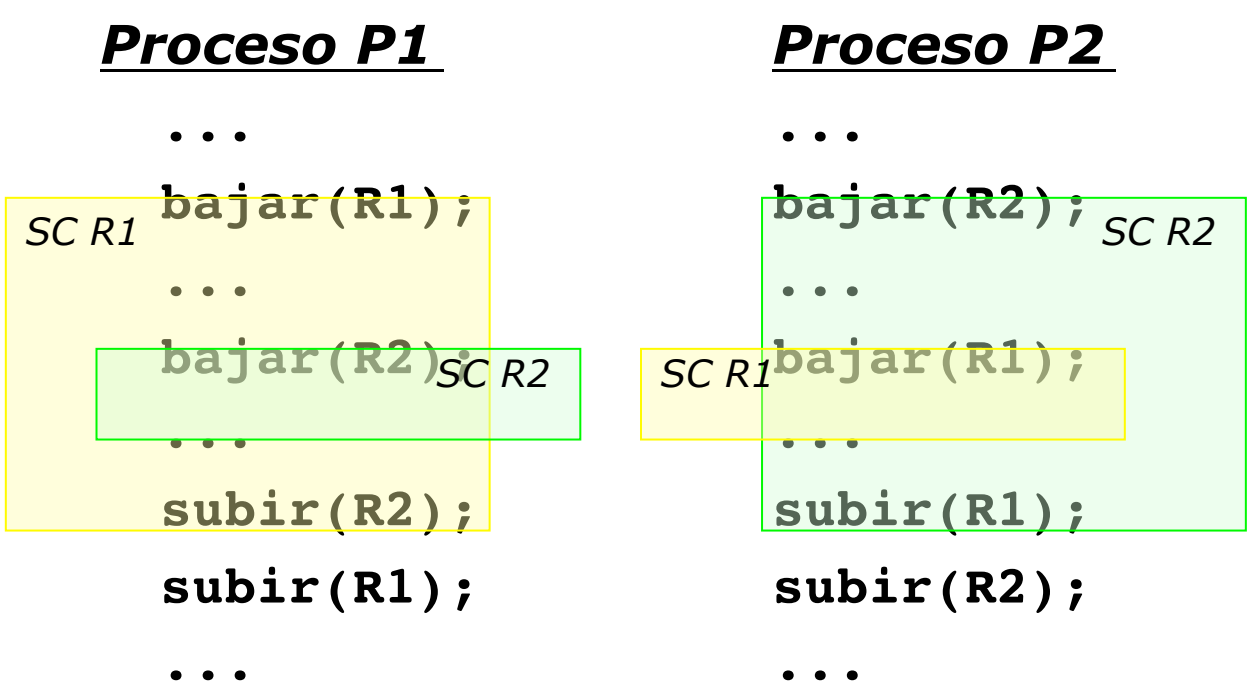
```
...
bajar(R1);
...
bajar(R2);
...
subir(R2);
subir(R1);
...
```

Proceso P2

```
...
bajar(R2);
...
bajar(R1);
...
subir(R1);
subir(R2);
...
```

Interbloqueo
Un escenario sencillo

- El siguiente código puede conducir a un interbloqueo entre P1 y P2:



Interbloqueo
Otro escenario: recursos múltiples

- Un proceso/hilo necesita *n* unidades (bloques) de memoria:

```
proceso_que_usa_memoria(int n) {
    for (i=0; i<n; i++)
        b[i]= dame_un_bloque(); // en exc. mutua

    // Se usan los bloques...

    for (i=0; i<n; i++)
        liberar_bloque(b[i]); // en exc. mutua
}
```

- Supongamos la siguiente situación:
  - Un proceso P va a usar M bloques
  - Un proceso Q va a usar N bloques
  - En el sistema quedan B bloques libres, tal que B<M+N

UPV / EHU

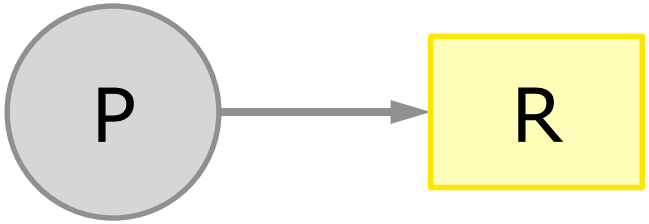
2. Modelo del interbloqueo

OCW OpenCourseWare

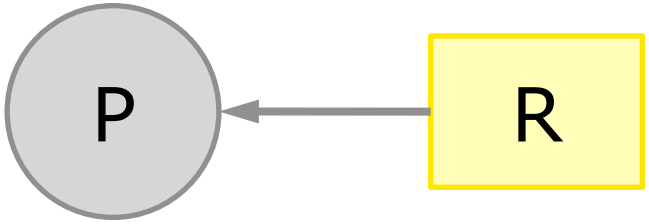
Modelo del interbloqueo

Grafo de asignación de recursos a procesos

Notación:



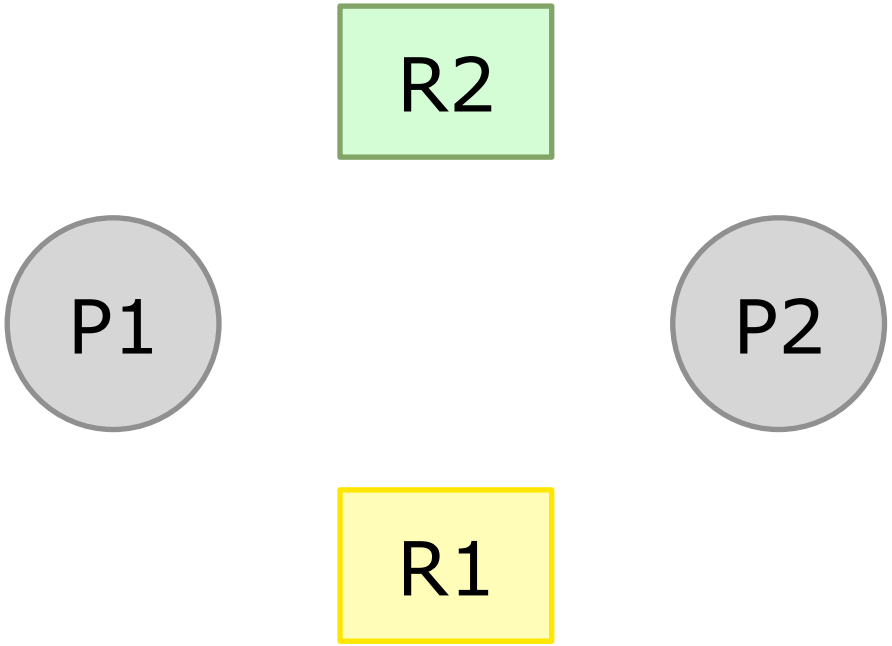
El proceso P *ha solicitado* usar el recurso R



El proceso P *está usando* el recurso R

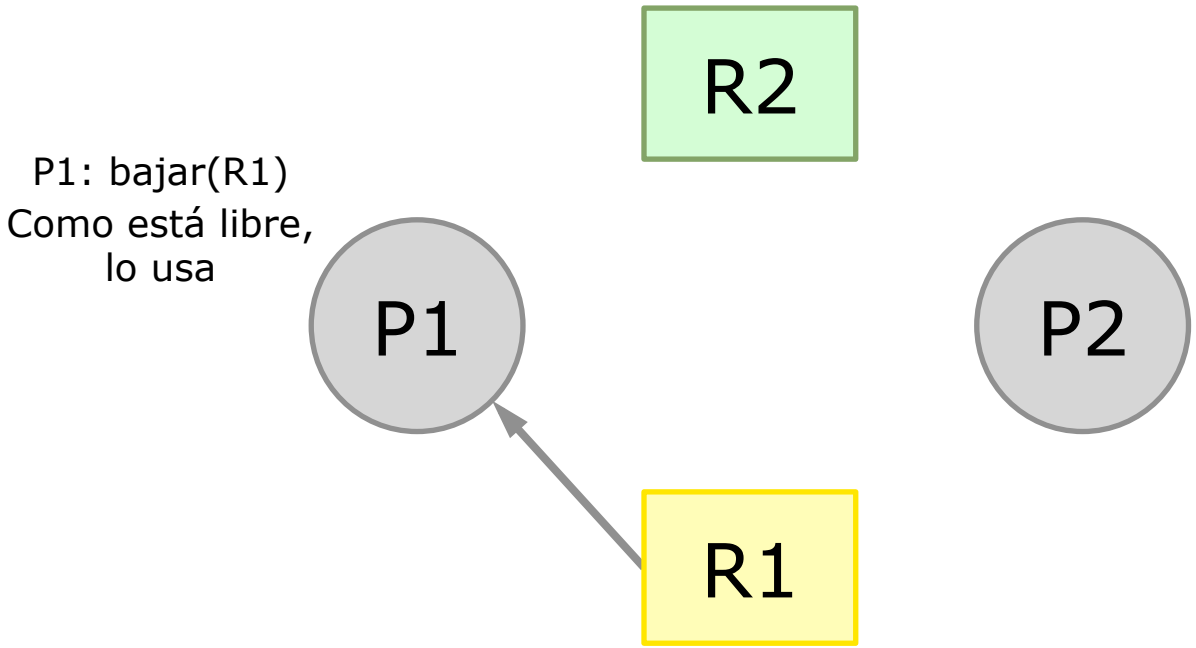
Modelo del interbloqueo

- El escenario sencillo anterior. Una posible secuencia:



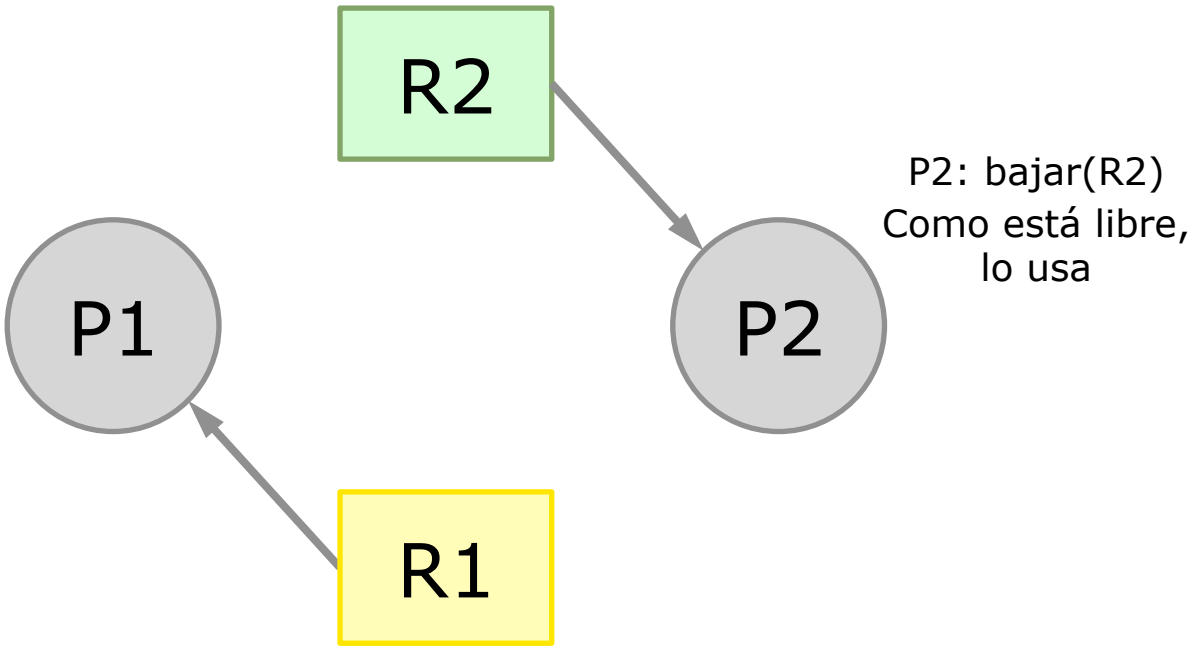
Modelo del interbloqueo

- El escenario sencillo anterior. Una posible secuencia:



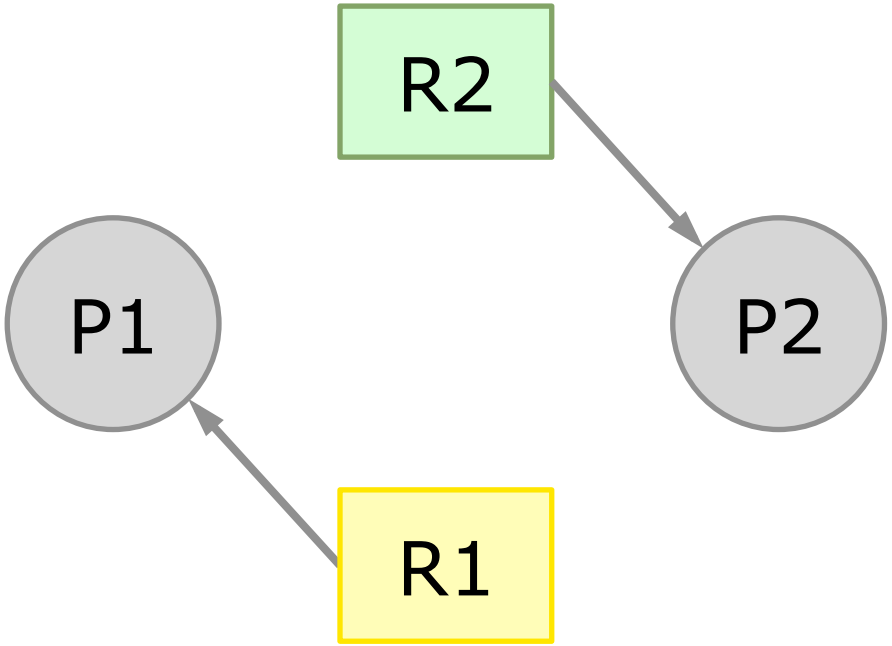
Modelo del interbloqueo

- El escenario sencillo anterior. Una posible secuencia:



Modelo del interbloqueo

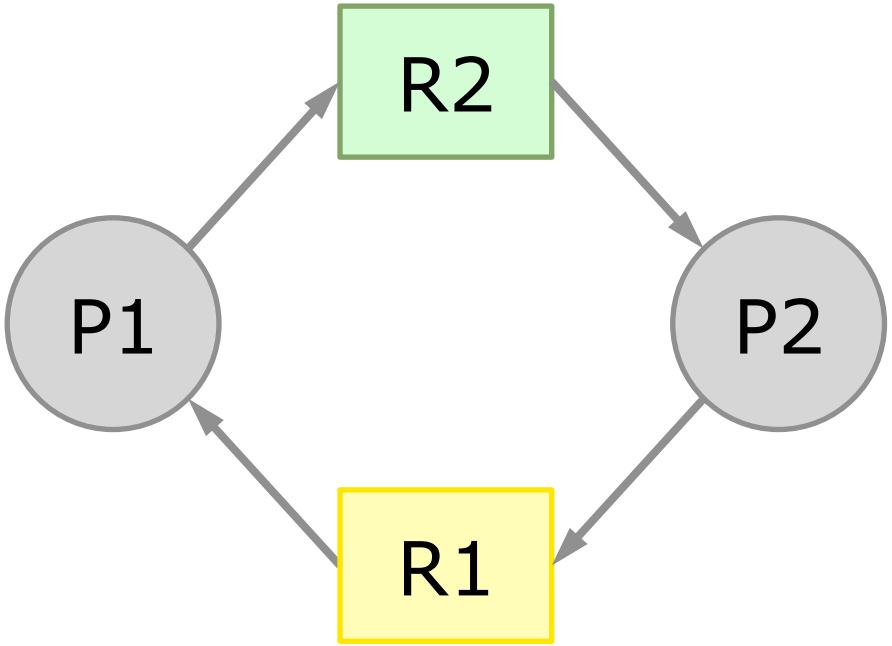
- El escenario sencillo anterior. Una posible secuencia:



El interbloqueo ya es inevitable…

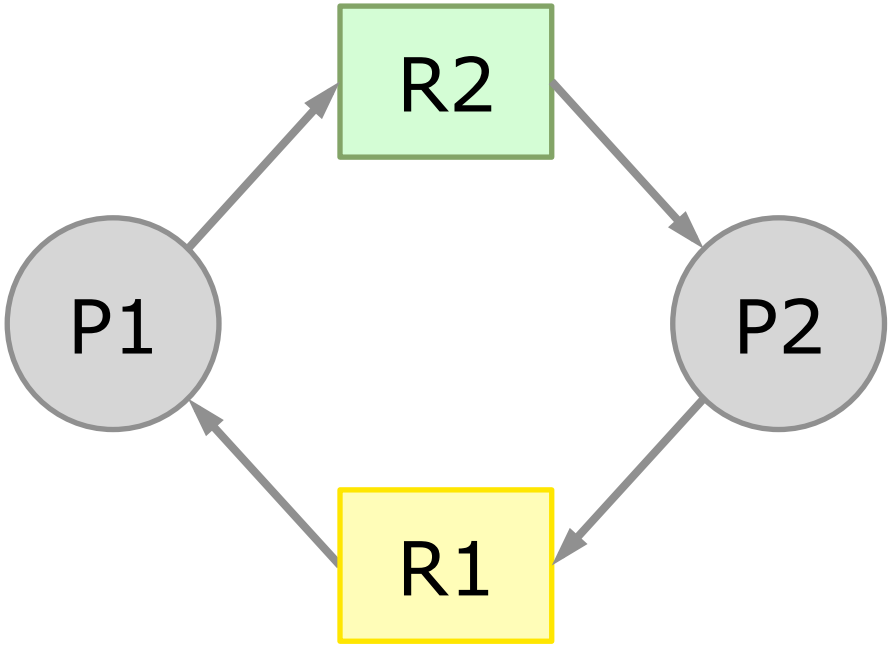
Modelo del interbloqueo

- El escenario sencillo anterior. Una posible secuencia:



Modelo del interbloqueo

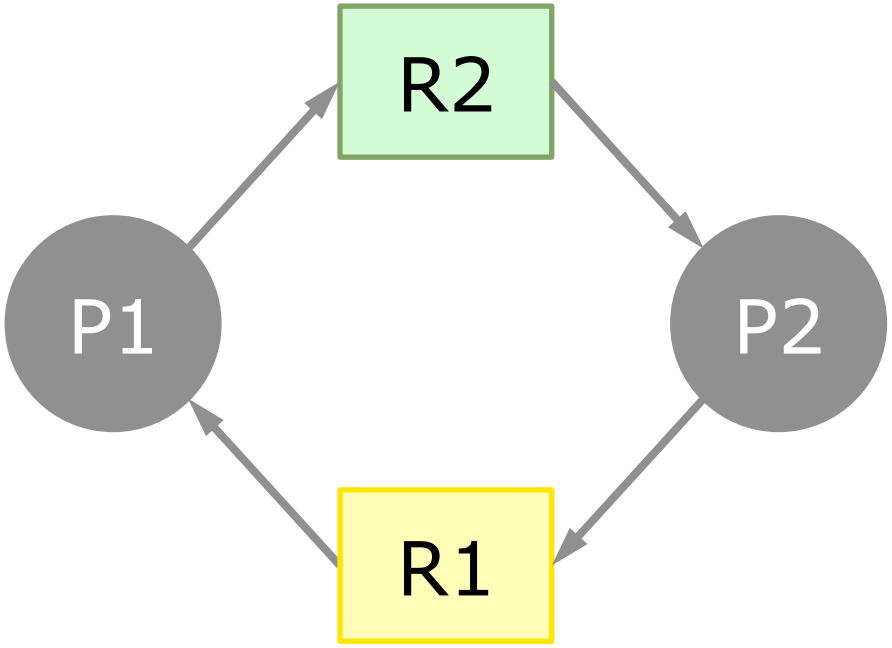
- El escenario sencillo anterior. Una posible secuencia:



…se forma un ciclo en el grafo de asignación

Modelo del interbloqueo

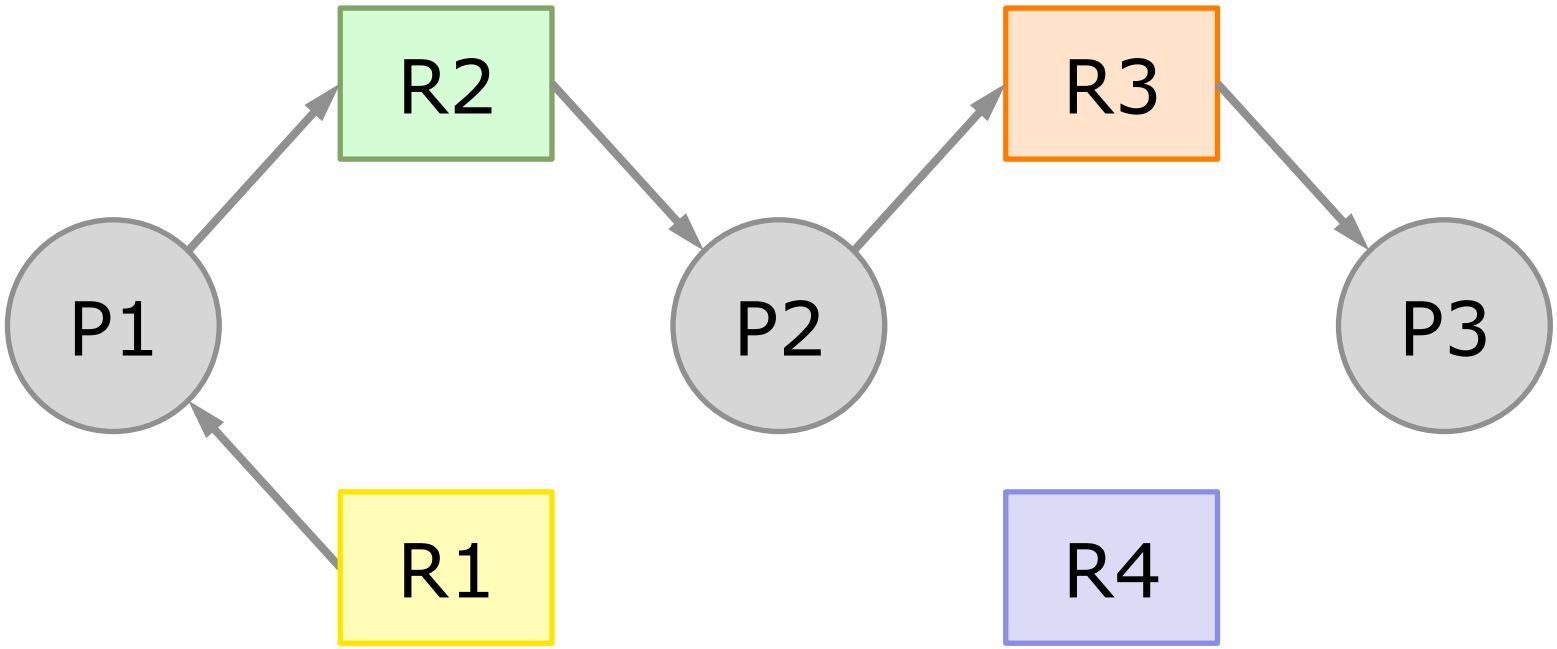
- El escenario sencillo anterior. Una posible secuencia:



P1 y P2 están interbloqueados

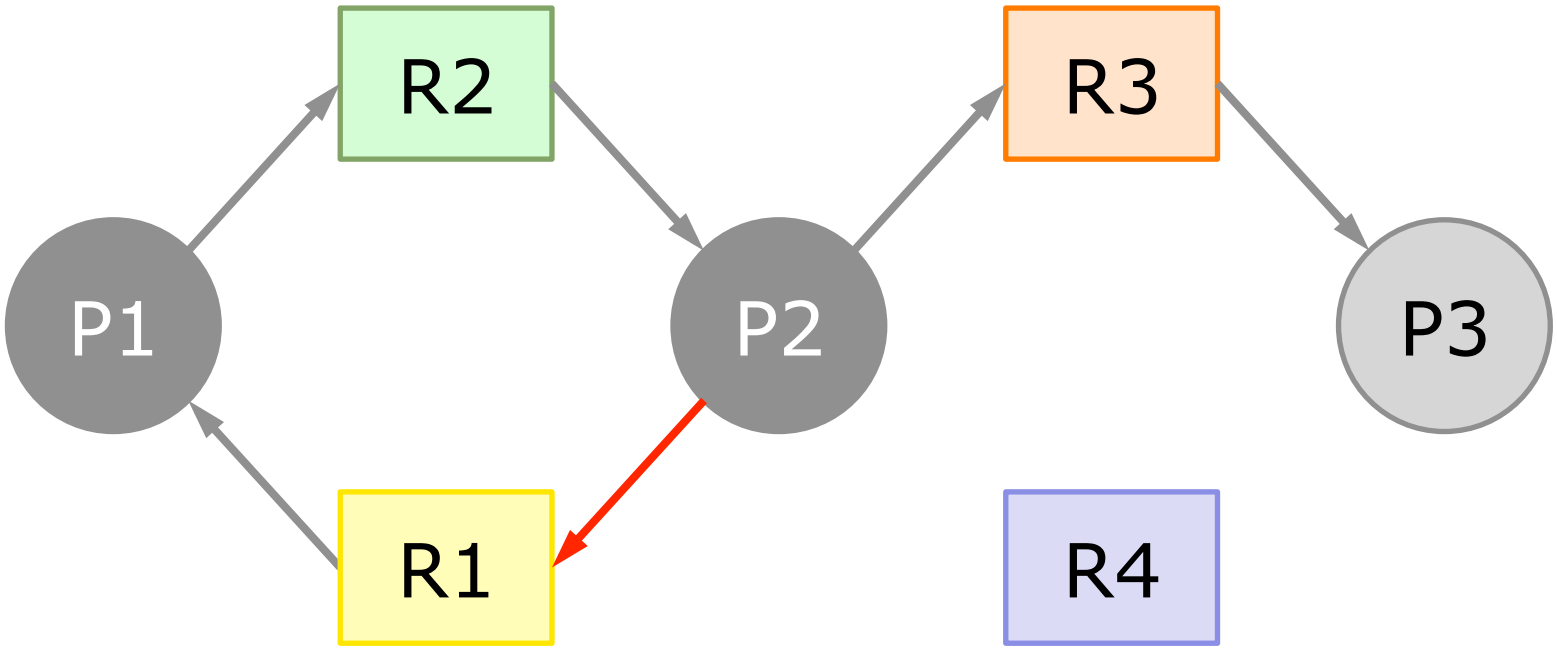
Modelo del interbloqueo

- Otro ejemplo:



Modelo del interbloqueo

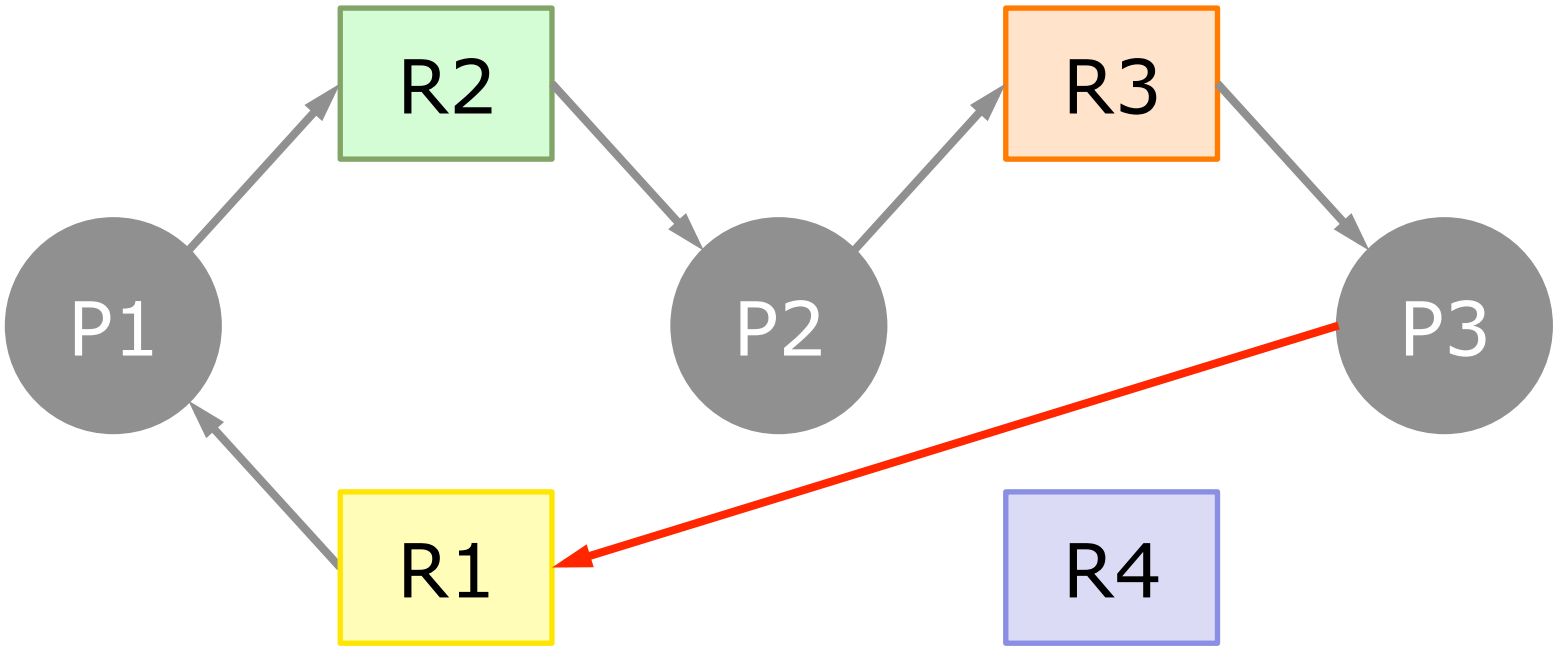
- Otro ejemplo:



Si P2 *solicita* R1 hay interbloqueo entre P1 y P2

Modelo del interbloqueo

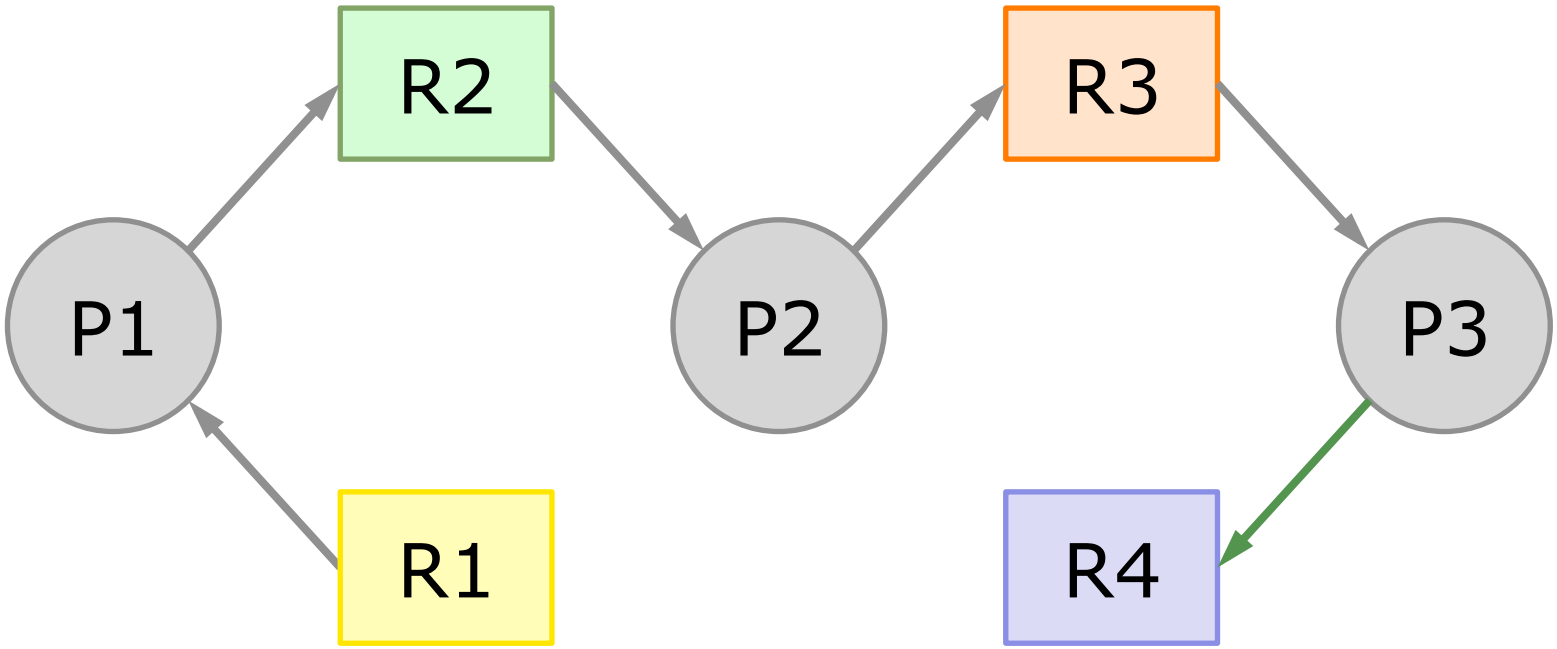
- Otro ejemplo:



…o si P3 *solicita* R1 hay interbloqueo entre P1, P2 y P3

Modelo del interbloqueo

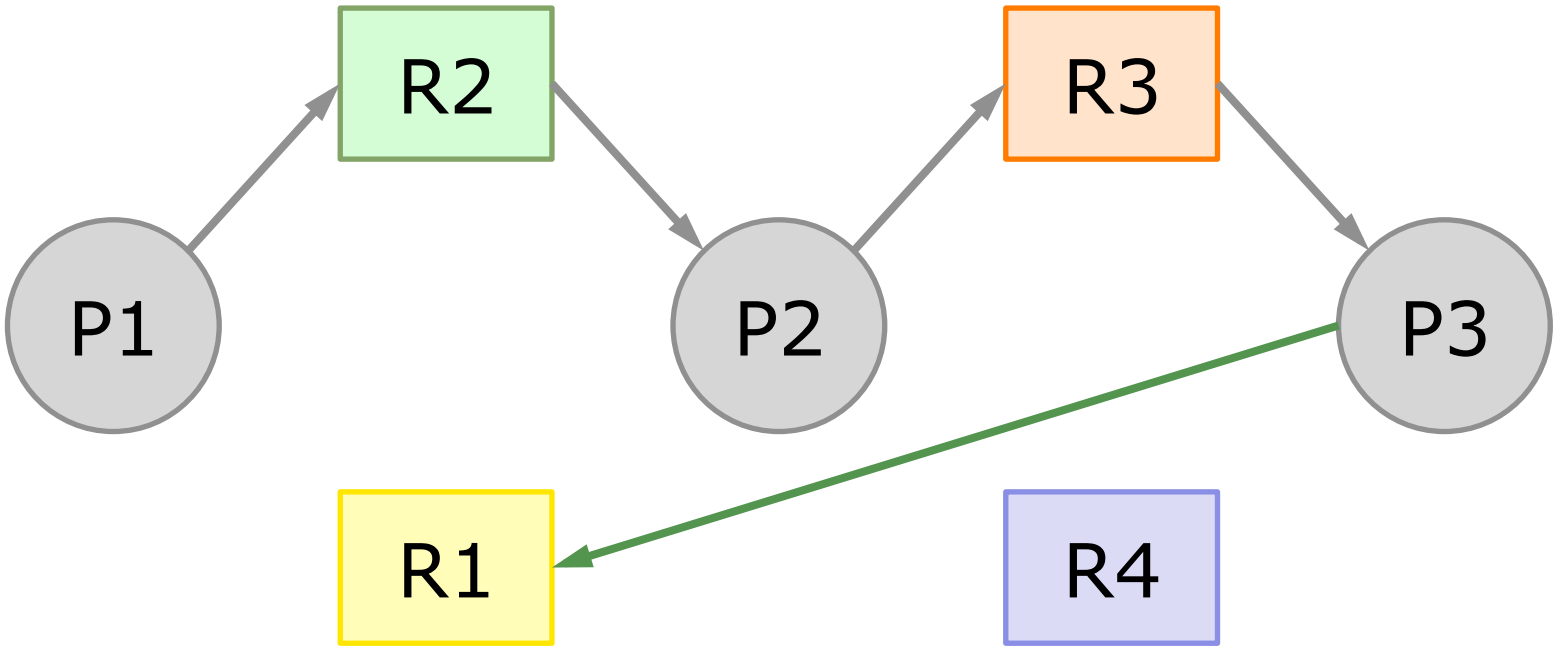
- Otro ejemplo:



…pero no lo habría si P3 solicita R4

Modelo del interbloqueo

- Otro ejemplo:



…o si P3 solicita R1 una vez que P1 lo libere

UPV / EHU

3. Soluciones al interbloqueo

OCW OpenCourseWare

Soluciones al interbloqueo

UPV / EHU

- Dos enfoques:
  - Políticas *paliativas:*
    - El Algoritmo del Avestruz
    - Detección y eliminación
  - Políticas *preventivas:*
    - Prevención de interbloqueos
    - Predicción de interbloqueos

OCW OpenCourseWare

Soluciones al interbloqueo

- El *Algoritmo del Avestruz*
  - No hacer nada
    - …se reinicia el ordenador cuando suceda.
    - Esta solución resulta práctica si el ordenador no se usa para tareas crítica (p.ej., un PC doméstico o un móvil).
- Detección y eliminación
  - Detectar cuando ocurra un interbloqueo
    - Manteniendo un grafo de asignación o mediante otros indicios.
  - Una vez detectado, tratar de eliminarlo
    - Eliminando alguno de los procesos involucrados.

Soluciones al interbloqueo

UPV / EHU

- Mejor prevenir que curar…
  - Las políticas preventivas evitan los interbloqueos impidiendo las *condiciones* para que ocurran o, sin impedirlas, *prediciendo* la posibilidad de un interbloqueo para actuar en consecuencia.

Soluciones al interbloqueo
Condiciones para el interbloqueo

- Un interbloqueo solo puede producirse si en el sistema se cumplen todas y cada una de las condiciones siguientes:

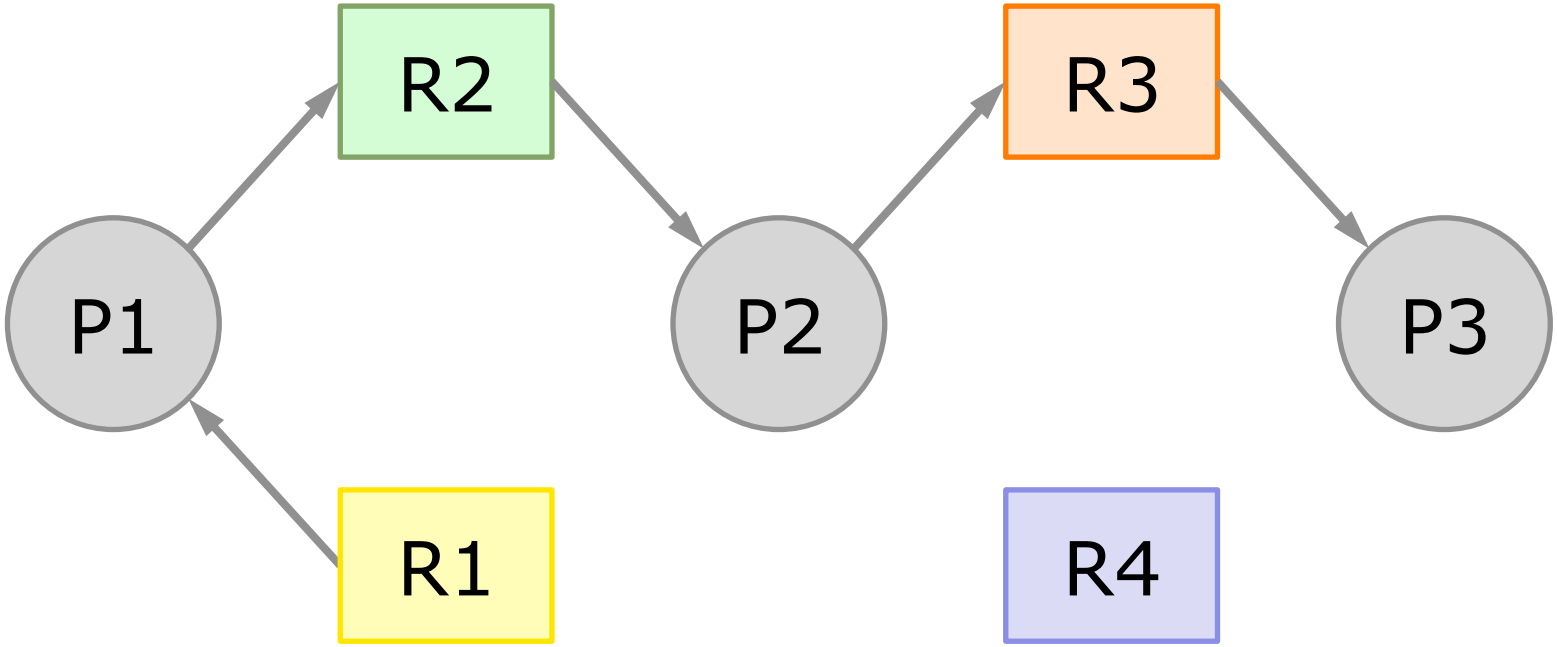
1. *Exclusión mutua*. En todo momento, cada recurso, o está asignado a un proceso, o está libre.
2. *Retención y espera*. Un proceso que está utilizando un recurso r_i puede solicitar otro recurso r_j antes de liberar r_i.
3. *No expulsión*. Un proceso no puede ser forzado a liberar un recurso.
4. *Espera circular*. Existe un conjunto de procesos P_D entre los que se define un círculo de espera por recursos que están siendo usados

Soluciones al interbloqueo
Prevención de interbloqueos

- Si el sistema impide una sola de las condiciones para el interbloqueo, ¡estos no se producirán!
  - Impedir las Condiciones 1 y 3 resulta muy restrictivo o imposible.
  - Impedir la Condición 2:
    - Asignando recursos por conjuntos.
    - Restringe la concurrencia.
  - Impedir la Condición 4:
    - Asignar recursos en un orden determinado:
      - Si P está usando R_i, puede solicitar R_k solo si k>i

Soluciones al interbloqueo
Prevención de interbloqueos

- ¡No pueden producirse ciclos si los recursos se asignan en orden creciente!



Soluciones al interbloqueo
Predicción de interbloqueos

UPV / EHU

- Se definen estados *seguros* e *inseguros*.
  - Los estados inseguros son los que pueden conducir a interbloqueos y se evitan denegando el acceso a recursos libres.
  - Se usa el *Algoritmo del banquero*
    - Requiere un número fijo de procesos.
    - Muy restrictivo y computacionalmente complejo.

Soluciones al interbloqueo
El Algoritmo del Banquero

- Con dos procesos y dos recursos se puede representar en el plano.
- Objetivo: conseguir una trayectoria en la ejecución que haga que todos los procesos terminen.