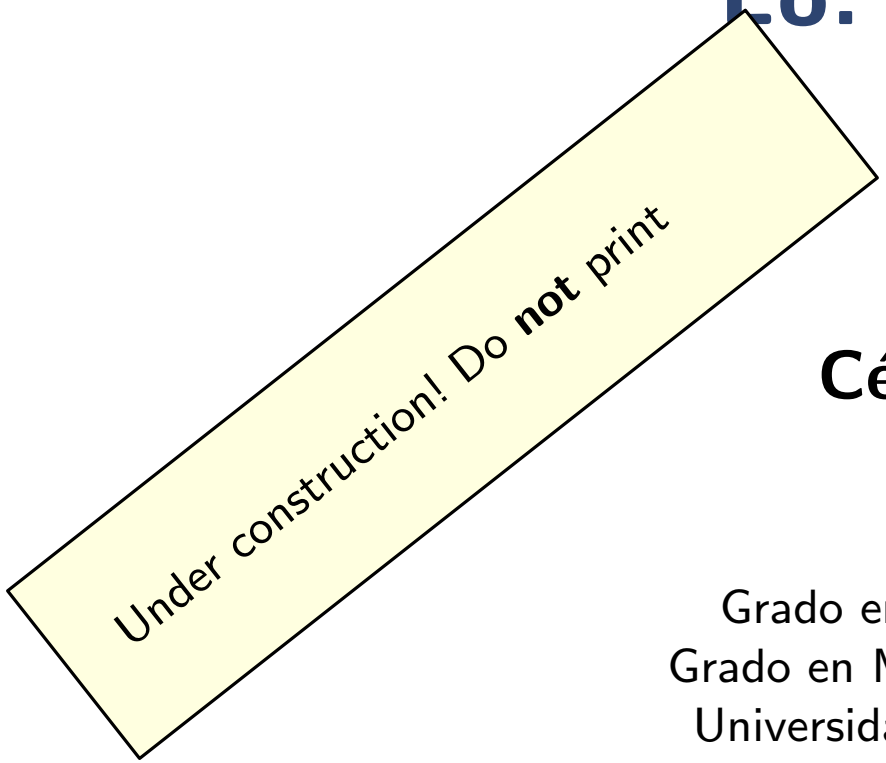
L8: Deadlocks



César Sánchez

Grado en Ingeniería Informática
Grado en Matemáticas e Informática
Universidad Politécnica de Madrid

Fri, 27 Feb 2015

Este texto se distribuye bajo los términos de la Creative Commons License

Mapa Conceptual

$$\text{Concurrency} = \text{Simultaneous} + \text{Nondeterminism} + \text{Interaction}$$

$$\text{Interaction} = \text{Communication} \mid \text{Synchronization}$$

$$\text{Synchronization} = \text{Mutual Exclusion} \mid \begin{array}{l}\text{Conditional} \\ \text{Synchronization}\end{array}$$

- Terminology:

atomic	race condition
interleaving	busy-wait
mutual exclusion	critical section
deadlock	livelock
liveness	

Deadlock

Deadlock:

> *a deadlock is a situation in which two or more competing actions are each waiting for the other to finish, and thus neither ever does.*

Deadlock

Deadlock:

> *a deadlock is a situation in which two or more competing actions are each waiting for the other to finish, and thus neither ever does.*

"When two trains approach each other at a crossing, both shall come to a full stop and neither shall start up again until the other has gone."

Statute passed by the Kansas State Legislature, early in the 20th century.

Livelock

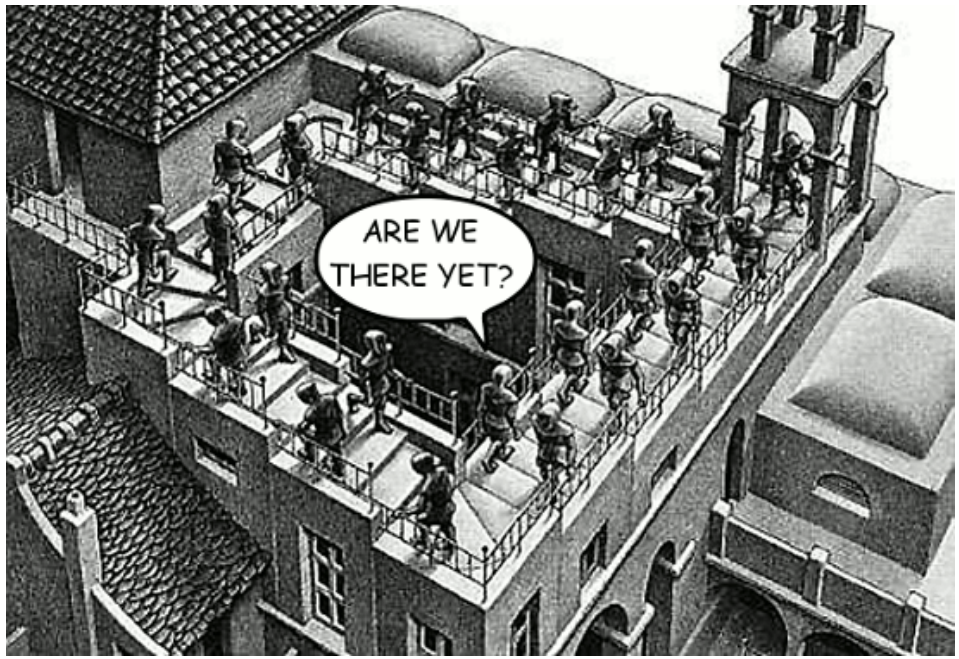
Livelock:

A livelock is similar to a deadlock, except that the states of the processes involved in the livelock constantly change with regard to one another, none progressing.

Livelock

Livelock:

A livelock is similar to a deadlock, except that the states of the processes involved in the livelock constantly change with regard to one another, none progressing.



Conditions for Deadlock (necessary and sufficient)

1. Mutual Exclusion

   The resource is only accessed by one thread at a time.

2. Hold and wait

   The resource is kept until the job is finished.

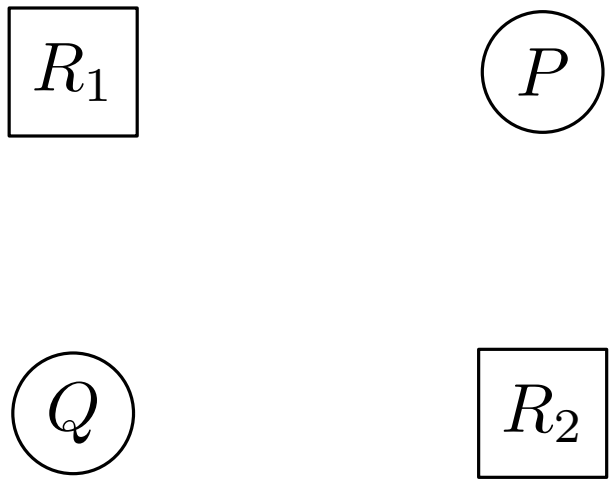
3. No preemption

   A resource is not stolen.

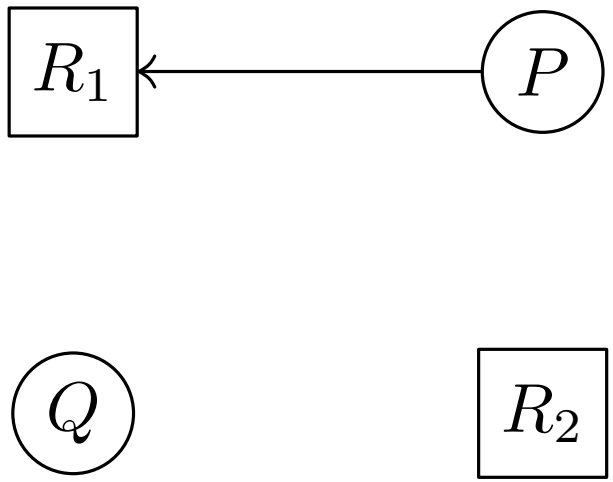
4. Circular wait

   There is a circular chain of processes waiting for resources held by the next process in the chain.

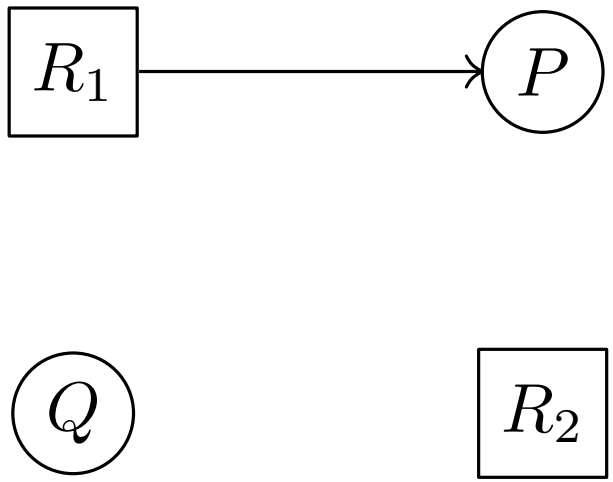
Resource allocation graph



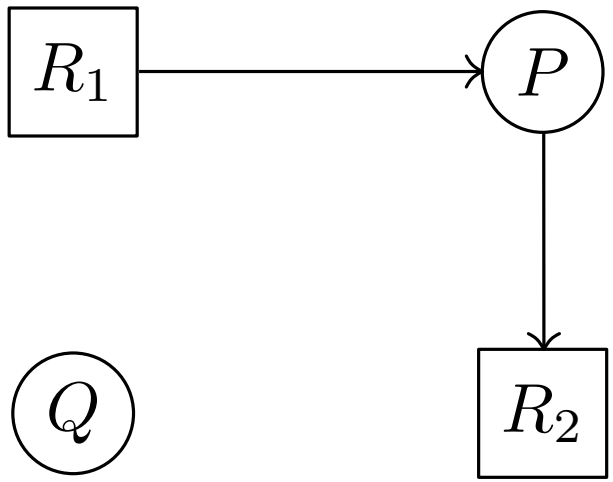
Resource allocation graph



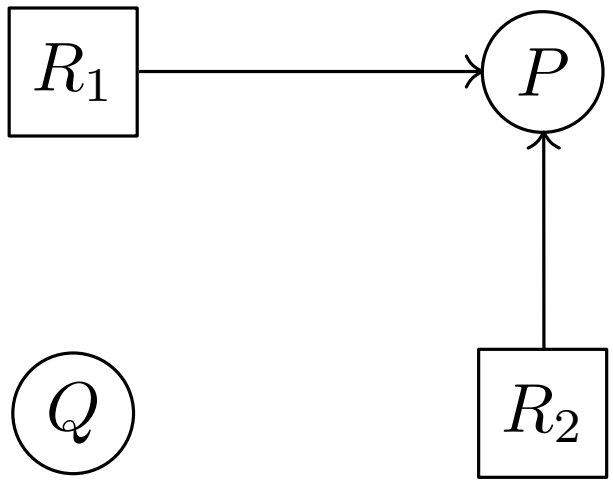
Resource allocation graph



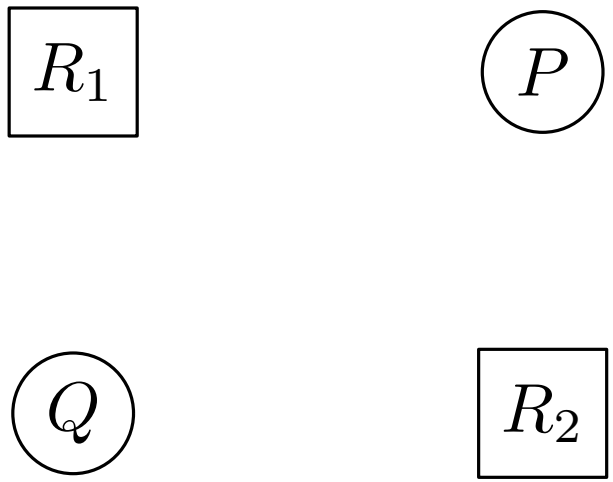
Resource allocation graph



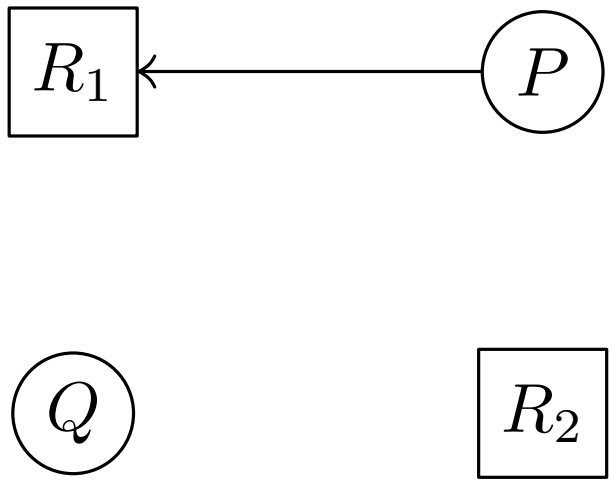
Resource allocation graph



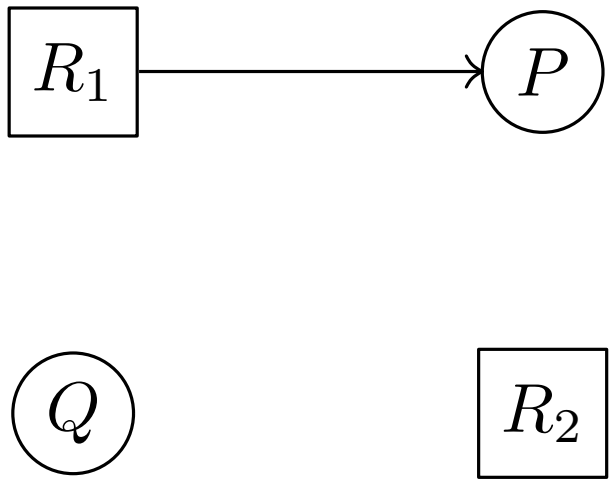
Resource allocation graph (Deadlock Example)



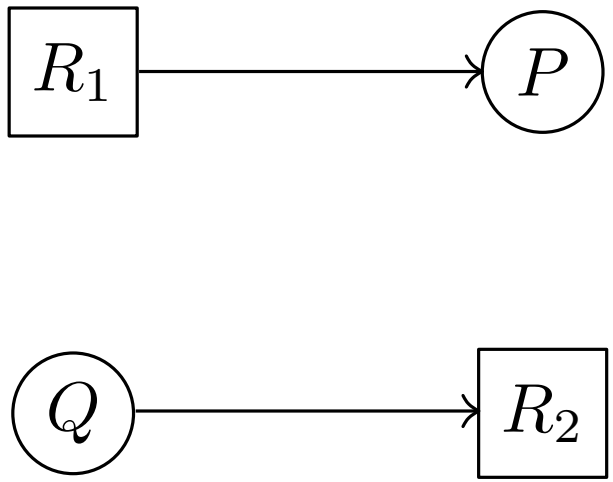
Resource allocation graph (Deadlock Example)



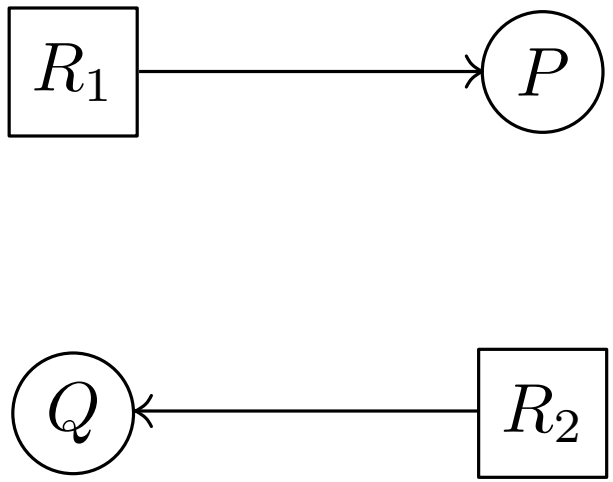
Resource allocation graph (Deadlock Example)



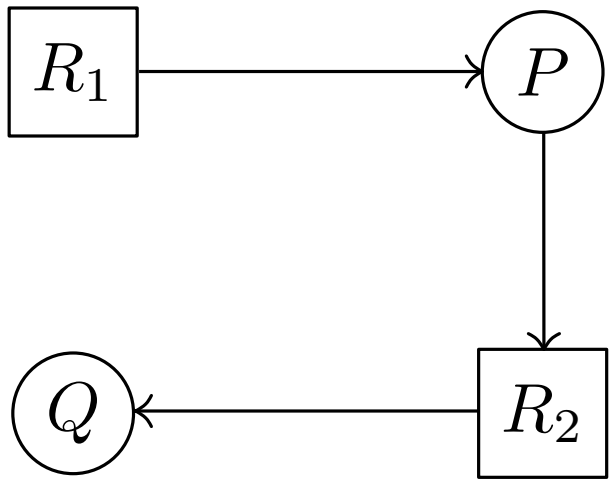
Resource allocation graph (Deadlock Example)



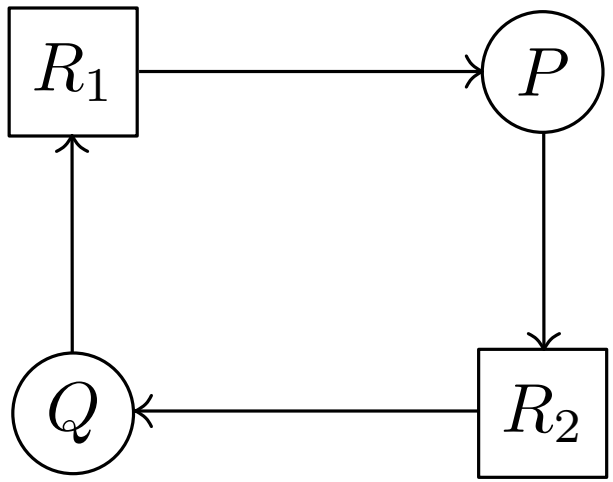
Resource allocation graph (Deadlock Example)



Resource allocation graph (Deadlock Example)



Resource allocation graph (Deadlock Example)



Deadlock.java

```
public void run () {
   for (int n = 1; n <= Dining.ROUNDS; ++ n) {
      //think();
      left.take();
      System.out.println(me+" takes fork "+left.me);
      right.take();
      System.out.println(me+" takes fork "+right.me);
      eat();
      right.put_back();
      left.put_back();
   }
   System.out.println(me+" leaves");
}
```

Starvation

Starvation:

> *Starvation describes a situation where a thread is unable to gain regular access to shared resources and is unable to make progress. This happens when shared resources are made unavailable for long periods by "greedy" threads.*