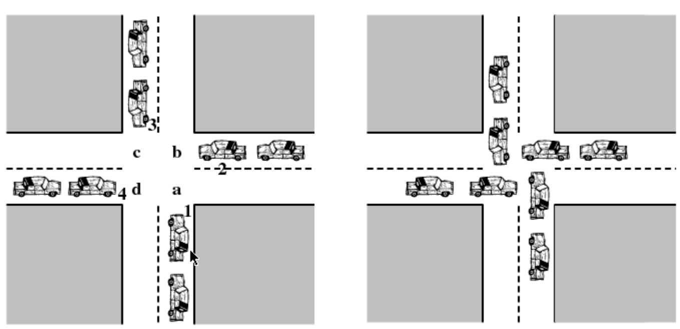
Deadlock

- Permanent blocking of a set of processes that either compete for system resources or communicate with each other
- No efficient solution
- Involve conflicting needs for resources by two or more processes

1

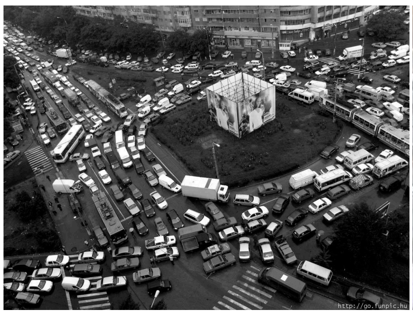


(a) Deadlock possible (b) Deadlock

Figure 6.1 Illustration of Deadlock

2

Better illustrations ☺



Reusable Resources

- Used by only one process at a time and not depleted by that use
- Processes obtain resources that they later release for reuse by other processes
  - E.g. Processors, I/O channels, main and secondary memory, devices, and data structures such as files, databases, and semaphores
- Deadlock occurs if each process holds one resource and requests the other

4

Example of Deadlock

Process P

Step	Action
p_0	Request (D)
p_1	Lock (D)
p_2	Request (T)
p_3	Lock (T)
p_4	Perform function
p_5	Unlock (D)
p_6	Unlock (T)

Process Q

Step	Action
q_0	Request (T)
q_1	Lock (T)
q_2	Request (D)
q_3	Lock (D)
q_4	Perform function
q_5	Unlock (T)
q_6	Unlock (D)

Figure 6.4 Example of Two Processes Competing for Reusable Resources

Now consider the following sequence:
$p_0\ p_1\ q_0\ q_1\ p_2\ q_2$

Another Example of Deadlock

- Space is available for allocation of 200Kbytes, and the following sequence of events occur

P1	P2
. . .	. . .
Request 80 Kbytes;	**Request 70 Kbytes;**
. . .	. . .
Request 60 Kbytes;	**Request 80 Kbytes;**

- Deadlock occurs if both processes progress to their second request

Consumable Resources

- Created (produced) and destroyed (consumed)
- Interrupts, signals, messages, and information in I/O buffers
- Deadlock may occur if a Receive message is blocking
- May take a rare combination of events to cause deadlock

Example of Deadlock

- Deadlock occurs if receive is blocking

P1

```
. . .
Receive(P2);
. . .
Send(P2, M1);
```

P2

```
. . .
Receive(P1);
. . .
Send(P1, M2);
```

Resource Allocation Graphs

- Directed graph that depicts a state of the system of resources and processes

9

Resource Allocation Graphs

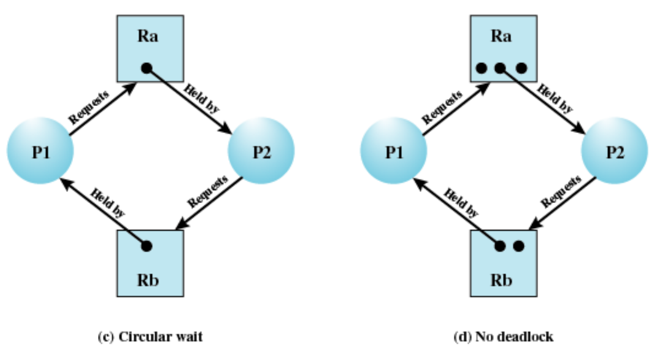


Figure 6.5 Examples of Resource Allocation Graphs

10

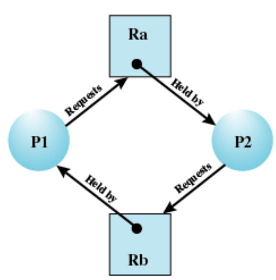
Conditions for Deadlock

- Mutual exclusion
  – Only one process may use a resource at a time
- Hold-and-wait
  – A process may hold allocated resources while awaiting assignment of others
- No preemption
  – No resource can be forcibly removed from a process holding it

11

Conditions for Deadlock

- Circular wait
  – A closed chain of processes exists, such that each process holds at least one resource needed by the next process in the chain



(c) Circular wait

12