

CS241 Operating Systems Deadlock Issues (1)

Klara Nahrstedt

Lecture 16

2/24/2006

Content

- Deadlock Concept
- Deadlock Conditions

Administrative Notes

- MP2 on scheduling is running
- Quiz 4 today
- Read Tanenbaum Chapter 3.3.1 and 3.3.2

Resource (1)

- A **resource** is a commodity needed by a process.
- Resources can be either:
 - **serially reusable**: e.g., CPU, memory, disk space, I/O devices, files.
acquire → use → release
 - **consumable**: produced by a process, needed by a process; e.g., messages, buffers of information, interrupts.
create → acquire → use
Resource ceases to exist after it has been used, so it is not released.

Resource (2)

- Resources can also be either:
 - **preemptible**: e.g., CPU, central memory or
 - **non-preemptible**: e.g., tape drives.
- And resources can be either:
 - **shared** among several processes or
 - **dedicated** exclusively to a single process.

Using Semaphore to Share Resource

	Process P();	0	→	Process Q();
2	→ { A.Down();	6	→	{ A.Down();
3	→ B.Down();			B.Down();
	use both resource			use both resource
4	→ B.Up();			B.Up();
5	→ A.Up(); }			A.Up(); }

External Semaphore A(1), B(1);

2 External Semaphore A(0), B(1);

3 External Semaphore A(0), B(0);

4 External Semaphore A(0), B(1);

5 External Semaphore A(1), B(1);

But Deadlock can Happen!

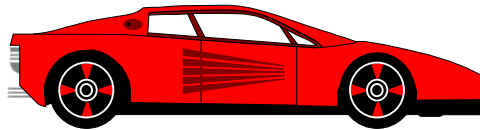
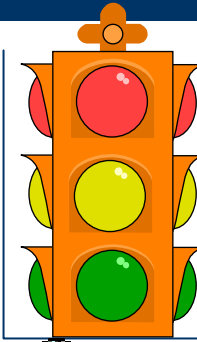
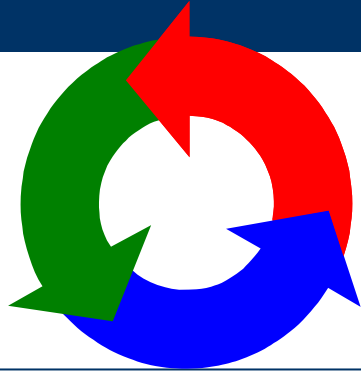
```
1 → Process P();  
2 → { A.Down();  
      B.Down();  
      use both resources  
      B.Up();  
      A.Up();
```

```
1 → Process Q();  
3 → { B.Down();  
      A.Down();  
      use both resources  
      A.Up();  
      B.Up(); }
```

DEADLOCK

- 1 External Semaphore A(1), B(1);
- 2 External Semaphore A(0), B(1);
- 3 External Semaphore A(0), B(0);

Deadlock



Mechanism for Deadlock Control

Deadlock Definition

- What is a deadlock?
 - A process is **deadlocked** if it is waiting for an event that will never occur.
 - Typically, but not necessarily, more than one process will be involved together in a deadlock (the *deadly embrace*).
- Is deadlock the same as starvation (or indefinitely postponed)?
 - A process is **indefinitely postponed** if it is delayed repeatedly over a *long* period of time while the attention of the system is given to other processes. I.e., logically the process may proceed but the system never gives it the CPU.

Conditions for Deadlock

- What conditions should exist in order to lead to a deadlock
- Real life analogy such as
 - “You take the monitor, I grab the keyboard”
 - Cars in intersection

Necessary and Sufficient Conditions for Deadlock

- **Mutual exclusion**
 - Processes claim **exclusive** control of the resources they require
- **Wait-for condition**
 - Processes hold resources already allocated to them while waiting for additional resources
- **No preemption condition**
 - Resources cannot be removed from the processes holding them until used to completion
- **Circular wait condition**
 - A circular chain of processes exists in which each process holds one or more resources that are requested by the next process in the chain

Mutual Exclusion

- Processes claim **exclusive** control of the resources they require



Wait-for Condition

- Processes hold resources already allocated to them while waiting for additional resources

Process 1

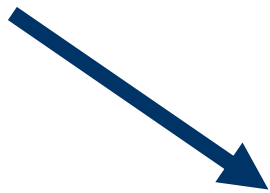
P(s1) Blocked
P(s2)
Work
V(s2)
V(s1)

Process 2

P(s2)
P(s3)
Work
V(s3)
V(s2)

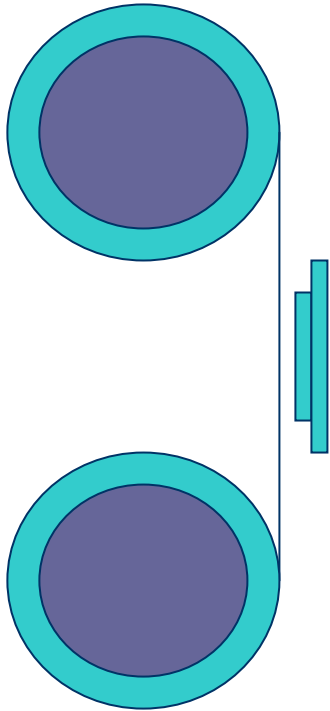
Process 3

P(s3)
P(s1)
Work
V(s1)
V(s3)



No Preemption Condition

- Resources cannot be removed from the processes holding them until used to completion



Its difficult to take a tape drive away
from a process that is busy writing a
tape



Circular Wait Condition

- A circular chain of processes exists in which each process holds one or more resources that are requested by the next process in the chain

Process 1

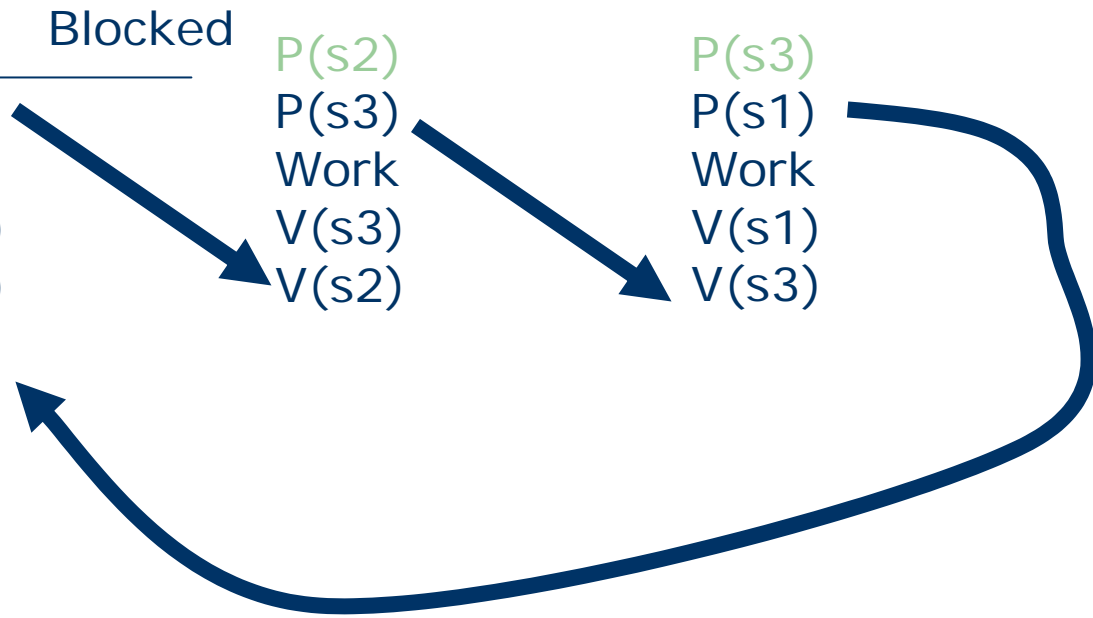
P(s1) Blocked
P(s2)
Work
V(s2)
V(s1)

Process 2

P(s2)
P(s3)
Work
V(s3)
V(s2)

Process 3

P(s3)
P(s1)
Work
V(s1)
V(s3)



Reminder

- MP2 is running, deadline Feb 27
- Read Tanenbaum 3.3.1 and 3.3.2