Deadlock

Daniel Zappala

CS 360 Internet Programming Brigham Young University

Deadlock Definition and Conditions

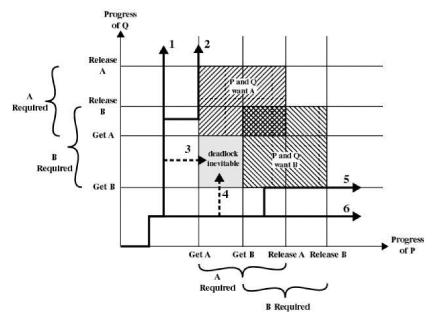
- permanent blocking of a set of processes or threads that either compete for system resources or communicate with each other
- conditions
 - 1 mutual exclusion: only one thread may use a resource at a time
 - 2 hold-and-wait: a thread keeps one resource while waiting for another
 - 3 no preemption: a thread can't be forced to release a resource
 - 4 circular wait: a cycle of threads waiting for each other
- if first three conditions hold, then deadlock is possible if circular wait occurs
- · depends on execution order!

Example

1	Thread P	1	Thread Q
2		2	• • •
3	Get A	3	Get B
4		4	• • •
5	Get B	5	Get A
6		6	• • •
7	Release A	7	Release B
8		8	• • •
9	Release B	9	Release A

• is deadlock possible?

Deadlock Possibilities



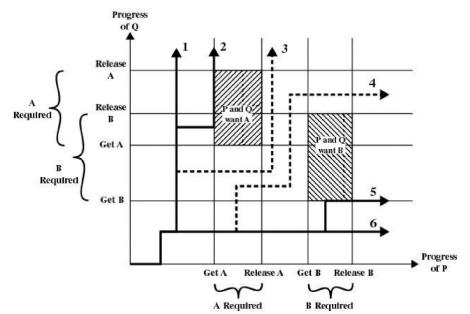
Revised Sample Code

```
Thread P
...
Get A
...
Release A
...
Get B
...
Release B
```

```
Thread Q
...
Get B
...
Get A
...
Release B
...
Release A
```

• is deadlock possible?

Deadlock Avoided



Deadlock Avoidance and Detection

avoidance

- monitor system and avoid granting a request to a lock (even if no other thread has the lock) if it might lead to deadlock
- requires tracking allocation of all locks and determining safe vs unsafe states

detection

- allow deadlock to occur, but detect and resolve
- requires tracking allocation of all locks and determining when deadlock has occurred
- requires pre-empting a thread that owns a lock

Simple Deadlock Prevention

- prevent one of the conditions from happening
- simplest to prevent is hold-and-wait: hold only one resource at a time
- can also prevent circular wait: impose ordering on resources