CS & IT ENGINEERING

Operating System

Deadlock

Lecture -3

By- Vishvadeep Gothi sir

Recap of Previous Lecture

Topic Banker's Algorithm

Topic Deadlock Detection

Topics to be Covered

- Topic: Deadlock
- Topic: Memory Management

Topic : Banker's Algorithm

Process	Allocation	Max	Available	Need
	A B C	A B C	A B C	A B C
P_0	0 1 0	7 5 3	3 3 2	7 4 3
P_1	2 0 0	3 2 2		1 2 2
P_2	3 0 2	9 0 2		6 0 0
P_3	2 1 1	2 2 2		0 1 1
P_4	0 0 2	4 3 3		4 3 1

[NAT]

#Q. What will happen if process P0 requests one additional instance of resource type C ?

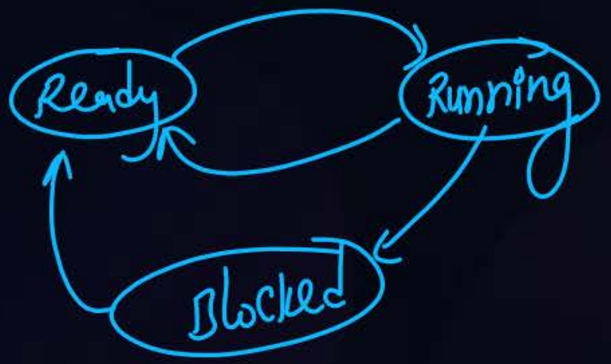
What will happen if process P3 requests one additional instance of resource type B and one instance of resource type C?

Both granted together

Topic : Types of Locks

1. Spinlock → busy waiting ⟹ while(true); ⇒ for single process
2. Livelock → deadlock but processes will be scheduled to run.
3. Deadlock ⇒ 2 processes are blocked due to each other
4. Semaphores
5. Reentrant Locks
6. Starvation ⇒ Indefinite wait

Ready ⇄ Running



→ A process or thread can take same lock again.

⇓

Single process can be deadlocked

⇓

Pi

lock() ✓

lock() ✓

ex:-

S1 = ~~1~~ 0

S2 = ~~1~~ 0

P1	P2
wait (S1)	wait (S2)
wait (S2)	wait (S1)
C.S.	C.S.
signal (S1)	signal (S2)
signal (S2)	signal (S1)

Solution 1

✓ Starvation

✗ Deadlock

```
Boolean lock=false;

while(true)
{
    while(lock);
    lock=true;
        //CS
    lock=false;
    RS;
}
```

```
while(true)
{
    while(lock);
    lock=true;
        //CS
    lock=false;
    RS;
}
```

Solution 2

✓ starvation
✗ deadlock

```
int turn=0;

while(true)
{
    while(turn!=0);
    CS
    turn=1;
    RS;
}
```

```
while(true)
{
    while(turn!=1);
    CS
    turn=0;
    RS;
}
```

Peterson's Solution

× starvation
× deadlock

```
Boolean Flag[2]={False, False};
int turn;

    while(true) {
      Flag[0]=true;
      turn=1;
      while(Flag[1] && turn==1);
          CS
      Flag[0]=False;
          RS;
    }
```

```
while(true){
    Flag[1]=true;
    turn=0;
    while(Flag[0] && turn==0);
        CS
    Flag[1]=False;
        RS;
}
```

Question 1

✓ starvation
× deadlock

```
turn=0;

while(true)
{
    while(turn);
    turn=1;
        //CS
    turn=0;
    RS;
}
```

```
while(true)
{
    while(turn);
turn lock=1;
        //CS
turn lock=0;
    RS;
}
```

Question 2

```
lock=False;

while(true)
{
    while(lock!=False);
    CS
    lock=True;
    RS;
}
```

```
while(true)
{
    while(lock!=True);
    CS
    lock=False;
    RS;
}
```

~ starvation
× deadlock

Question 3

```
lock=False;
while(true)
{
    while(lock ==False)
    {
        lock = True;
    }
    CS
    lock=False;
    RS;
}
```

P1

P2

✓ starvation

✗ deadlock

Question 4

```
Boolean lock= True;

while(true)
{
    while(lock)
    {
        CS
        lock = False ;
    }
    lock=True;
    RS;
}
```

✓ starvation

✗ deadlock

Topic : TestAndSet()

✓ starvation
× deadlock

```
Boolean Lock=False;

boolean TestAndSet(Boolean *trg){
boolean rv = *trg;
*trg = True;
Return  rv;
}
```

```
while(true)
{
  while(TestAndSet(&Lock));
        CS
  Lock=False;
}
```

Topic : Swap()

✓ starvation
× deadlock

```
Boolean Key;                    //Local
Boolean Lock=False;             //Shared

void Swap(Boolean *a, Boolean *b)
{
boolean temp = *a;
*a=*b;
*b=temp;
}
```

```
while(true){
  Key = True;
  while (key==True)
        Swap(&Lock, &Key);
     CS
  Lock=False;
}
```

Topic : Critical Section Solution

```
S = 1

while(True)
{
  wait(S)
       C.S.
   signal(s)
}
```

✓ Starvation
✗ deadlock

Ques)

```
lock = false

while (true)
{
   while (lock)
    { lock = true);
   }
 C.S.
 lock = true;
}
```

3 processes

P1, P2, P3

deadlock ?

⇓

yes

2 mins Summary

Topic Deadlock

Topic Memory Management

Happy Learning

THANK - YOU