Operating Systems (A) (Honor Track)

Semaphore Implementation

Yao Guo (郭耀)

Peking University

Implementing General Semaphores

How to implement a counting semaphore using binary semaphore(s)?

Solution #1 (incorrect)

- BinarySemaphore mutex = 1; // ensures mutual exclusion
- BinarySemaphore delay = 0; // allows process to sleep/block
- Integer count = N; // initial value of counting semaphore

```
P(Semaphore S)
{
    PB(mutex);
    S.count--;
    if(S.count<0) {
        VB(mutex);
        PB(delay);
    }
    else
        VB(mutex);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count<=0) {
        VB(delay);
    }
    VB(mutex);
}
```

Solution #1: What's Wrong?

- Line A1: Suppose multiple threads waiting here.
- Multiple VB(delay) called consecutively => Only one PB(delay) can continue
  - Recall: properties of binary semaphore

```
P(Semaphore S)
{
    PB(mutex);
    S.count--;
    if(S.count<0) {
        VB(mutex);
        PB(delay);  // A1
    }
    else
        VB(mutex);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count<=0) {
        VB(delay);
    }
    VB(mutex);
}
```

Solution #2: Fixing Solution #1

- BinarySemaphore mutex = 1; // ensures mutual exclusion
- BinarySemaphore delay = 0; // allows process to sleep/block
- Integer count = N; // initial value of counting semaphore

```
P(Semaphore S)
{
    PB(mutex);
    S.count--;
    if(S.count<0) {
        VB(mutex);
        PB(delay);
    }
    VB(mutex);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count<=0) {
        VB(delay);
    }
    else
        VB(mutex);
}
```

D. Hemmendinger, "A correct implementation of general semaphores", Operating Systems Review, vol. 22, no. 3 (July, 1988), pp. 42-44.

Solution #2: Inefficient

- Issue: Once processes start to wait on the internal variable delay, it forces a lock step behavior between the processes invoking the P and V operations,

```
P(Semaphore S)
{
    PB(mutex);
    S.count--;
    if(S.count<0) {
        VB(mutex);
        PB(delay);
    }
    VB(mutex);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count<=0) {
        VB(delay);
    }
    else
        VB(mutex);
}
```

D. Hemmendinger, "A correct implementation of general semaphores", Operating Systems Review, vol. 22, no. 3 (July, 1988), pp. 42-44.

Solution #3: Barz's Solution

- BinarySemaphore mutex = 1; // ensures mutual exclusion
- BinarySemaphore delay = min(1, N); // 0 if N==0, otherwise 1
- Integer count = N; // initial value of counting semaphore

```
P(Semaphore S)
{
    PB(delay);
    PB(mutex);
    S.count--;
    if(S.count>0) {
        VB(delay);
    }
    VB(mutex);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count==1) {
        VB(delay);
    }
    VB(mutex);
}
```

H. W. Barz, Implementing semaphores by binary semaphores. SIGPLAN Notices, volume 18, number 2, (February, 1983), pp 39-45.

Solution #4b: Hsieh's Solution

- BinarySemaphore mutex = 1; // ensures mutual exclusion
- BinarySemaphore delay = 0; // controls process block/sleep
- Integer count = 0; // initial value of counting semaphore

```
P(Semaphore S)
{
    PB(delay);
    PB(mutex);
    S.count--;
    if(S.count>0) {
        VB(delay);
    }
    VB(mutex);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count==1) {
        VB(delay);
    }
    VB(mutex);
}
```

C. Samuel Hsieh, "Further comments on implementation of general semaphores", Operating Systems Review, vol. 23, no. 1 (January, 1989), pp. 9-10.

Solution #4c: another similar solution

- BinarySemaphore mutex, barrier = 1; // ensures mutual exclusion
- BinarySemaphore delay = 0; // controls process block/sleep
- Integer count = 0; // initial value of counting semaphore

```
P(Semaphore S)
{
    PB(barrier);
    PB(mutex);
    S.count--;
    if(S.count<0) {
        VB(mutex);
        PB(delay);
    } else
        VB(mutex);
    VB(barrier);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count==0) {
        VB(delay);
    }
    VB(mutex);
}
```

Solution #5: Kearn's Solution

- BinarySemaphore mutex = 1; // ensures mutual exclusion
- BinarySemaphore delay = 0; // controls process block/sleep
- Integer count = 0; // initial value of counting semaphore
- Integer wakecount = 0; // # of blocked processed needed to wake up

```
P(Semaphore S)
{
    PB(mutex);
    S.count--;
    if(S.count<0) {
        VB(mutex);
        PB(delay);
        wakecount--;
        if(wakecount>0)
            VB(delay);
    }
    VB(mutex);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count<=0) {
        wakecount++;
        VB(delay);
    }
    VB(mutex);
}
```

Solution #5: Problems with Kearn's

- Too many signal operations
  - V operation signals delay *and* the P operation may also signal delay, i.e. when wakecount > 0.

```
P(Semaphore S)
{
    PB(mutex);
    S.count--;
    if(S.count<0) {
        VB(mutex);
        PB(delay);
        wakecount--;
        if(wakecount>0)
            VB(delay);
    }
    VB(mutex);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count<=0) {
        wakecount++;
        VB(delay);
    }
    VB(mutex);
}
```

Solution #5: Fixing Kearn's

- Remove extra signal operations

```
P(Semaphore S)
{
    PB(mutex);
    S.count--;
    if(S.count<0) {
        VB(mutex);
        PB(delay);
        wakecount--;
        if(wakecount>0)
            VB(delay);
    }
    VB(mutex);
}
```

```
V(Semaphore S)
{
    PB(mutex);
    S.count++;
    if(S.count<=0) {
        wakecount++;
        if(wakecount==1)
            VB(delay);
    }
    VB(mutex);
}
```

D. Hemmendinger, "Comments on "A correct and unrestrictive implementation of general semaphores"", Operating Systems Review, vol. 23, no. 1 (January, 1989), pp. 7- 8.

References

1. H. W. Barz, Implementing semaphores by binary semaphores. SIGPLAN Notices, volume 18, number 2, (February, 1983), pp 39-45.
2. D. Hemmendinger, “A correct implementation of general semaphores”, Operating Systems Review, vol. 22, no. 3 (July, 1988), pp. 42-44.
3. D. Hemmendinger, “Comments on “A correct and unrestrictive implementation of general semaphores””, Operating Systems Review, vol. 23, no. 1 (January, 1989), pp. 7- 8.
4. C. Samuel Hsieh, “Further comments on implementation of general semaphores”, Operating Systems Review, vol. 23, no. 1 (January, 1989), pp. 9-10.
5. P. Kearns, “A correct and unrestrictive implementation of general semaphores”, Operating Systems Review, vol. 22, no. 4 (October, 1988), pp. 46-48.
6. Trono, John A. and Taylor, William E.. "Further comments on "A correct and unrestrictive implementation of general semaphores".." Operating Systems Review 34 , no. 3 (2000): 5-10.