PV问题

2025年6月28日 星期六 17:12

例题：有一个仓库，可以存放 A 与 B 两种产品，仓库的存储空间足够大，但要求：

①每次只能存入一种产品（A 或 B）；

②-N<A 产品数量 - B 产品数量<M；

其中，N 和 M 是正整数。

试用"存放 A"和"存放 B"和 wait、signal 描述产品 A 与产品 B 的入库过程。

解析：每次存放 A 以前都要检查 A-B 是否已经超过 M-1 了，存放 A 结束以后告知 B-A 这个数需要减一，也就是如果 B 原本已经到达上限，那么现在就又可以继续放 B 了。B 的存放同理。现在将 A-B 和 B-A 抽象为两个信号量。

```
semorphore
mutex=1
sa=M-1
sb=N-1;
 Process Input _A:
     while (true){
        Get a product A;
        wait(sa);
        wait(mutex);
        // put product A into the
        // depository;
        signal(mutex);
        signal(sb); }
```

```
Process Input _B :
     while (1) {
         Get a product B;
         wait(sb);
         wait(mutex);
         // put product B into the
         // depository;
         signal(mutex);
         signal(sa);
     }
```


3. 信号量问题。假设操场上共有 22 个名额，有两个体育活动 A 和 B，规定当在操场上的人数不大于 22 时可以参与活动，否则需要等待。如果 A 的人数比 B 的人数多 5 人以上，参加 A 活动的需要等待；同理，如果 B 的人数比 A 的人数多 5 人以上，参加 B 活动的需要等待。参加 A 和 B 活动的可随时退出。根据“参与 A”“退出 A”“参与 B”“退出 B”和相应的信号量写出伪代码。

与上一个题的不同之处：本题对进有差值限制，但退无差值限制

因此不能用 AB 和 BA 的有差作为临界量

```
int numA=0, numB=0 , total=0
semaphore mutex=1 , cordA , cordB
参与A(){
  while(1){
     P(mutex)
     if(total>=22) or (numA>numB+5){
         V(mutex)
         P(cordA)
         continue
     }
     numA+=1
     total+=1
     V(mutex)
     break;
  }
}
```

```
退出A(){
   P(mutex)
   numA-=1
   total-=1
   V(cordA)
   V(cordB)
   V(mutex)
}
```

七、从前有座山，山上有座庙，山下有口井。庙里小和尚需要挑水。有人舞担，有人拿桶，有人诵挑水秘诀。挑水时，三个和尚必须一人持担，一人拿桶、一人诵挑水秘诀(同时进行)后方能挑水。每个和尚都是先喜欢诵诀，其次持担、其次持桶。请写出信号量和相关伪代码。

```
semaphore pole=1, bucket=1, mantra=1
semaphore ready=0, mutex=1
int count=0
Monk(){
   role=""
   P(mutex)
   if mantra>0
      P(mantra)
      V(mutex)
      role="诵读"
   else if pole>0
      P(pole)
      V(mutex)
      role="持担"
   else
      P(bucket)
      V(mutex)
      role="拿桶"
   P(mutex)
   count+=1
   if count==3
      V(ready)
      V(ready)
      V(ready)
      count=0
   V(mutex)
   P(ready)
   挑水;
   if role=="诵读"
      signal(mantra)
   else if role=="持担"
      signal(pole)
   else role=="拿桶"
      signal(bucket)
}
```

5. 信号量机制实现进程同步，一个发送消息，n个接收消息，缓冲区大小为1，必须n个进程均接收消息（一个进程只能接收1次）后，缓冲区才能清空。

```
Semaphore empty = 1       // 缓冲区空
Semaphore mutex = 1       // 保护 count
Semaphore slots[n] = {0, 0, ..., 0}  // 每个收进程单独一个 slot
int count = 0
```

```
Process Sender():
    while(1):
        P(empty)

        # 写消息到缓冲区
        buffer = new message

        # 重置接收标记
        for i = 1 to n:
            V(slots[i])     # 唤醒 n 个 receiver

        P(mutex)
        count = n
        V(mutex)
```

```
Process Receiver(i):
    while(1):
        P(slots[i])          # 等待自己的 slot 被 signal

        # 读取消息
        read buffer

        P(mutex)
        count -= 1
        if count == 0:
            V(empty)
        V(mutex)
```

问题描述：

在一个盒子里，混装了数量相等的黑白围棋子·现在用自动分拣系统把黑子、白子分开，设分拣系统有二个进程P1 和P2，其中P1 拣白子；P2 拣黑子。规定每个进程每次拣一子；当一个进程在拣时，不允许另一个进程去拣；当一个进程拣了一子时，必须让另一个进程去拣。试写出两进程P1 和P2 能并发正确执行的程序。

问题分析：

大家熟悉了生产-消费问题(PC)，这个问题很简单。题目较为新颖，但是本质非常简单即：生产-消费问题的简化或者说是两个进程的简单同步问题。答案如下：

设信号量s1 和s2 分别表示可拣白子和黑子；

不失一般性，若令先拣白子。

```
var S1 , S2 : semaphore;
S1 : = 1; S2 : =0;
cobegin
    process P1          process P2
     begin              begin
     repeat             repeat
     P(S1);             p(S2);
     pick The white;    pick the black;
     V(S2);             v(s1);
     until false ;      until false;
     end                end
coend
```