操作系统——算法题

第二章

- 进程的同步
  - 同步问题：给定前趋图，使用信号量实现
  - 互斥问题：生产者-消费者、哲学家进餐、读者-写者

第三章

- 银行家算法

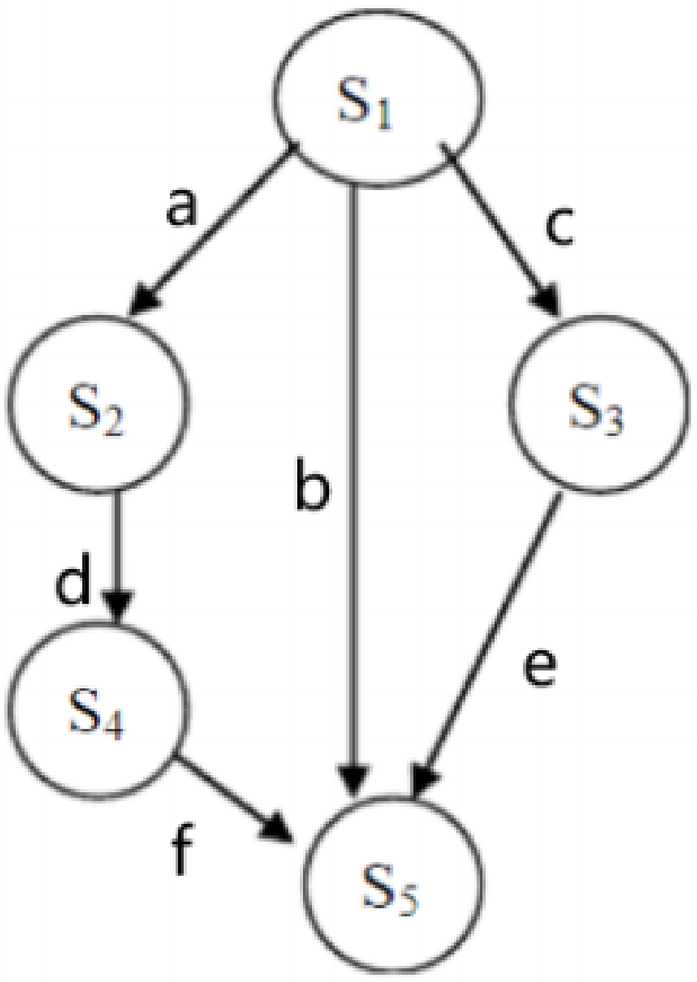
利用信号量实现前趋关系

- 可利用信号量来描述程序或语句之间的前趋关系。
  - 设有两个并发执行的进程P1和P2。P1中有语句S1；P2中有语句S2。我们希望在S1执行后再执行S2。

```
//P1和P2共享一个公用信号量S，并赋予其初值为S=0，
    在进程P1中用：        S1；signal(S)；
    在进程P2中用：        wait(S)；   S2；
```

同步问题：给定前趋图，使用信号量实现

- 请说明下图中的五个进程的前趋关系，并使用 PV 操作实现它们的前趋关系。


S1
a
b
c
S2
S3
d
e
S4
f
S5

生产者——消费者

解题思路

1. 分析各主体进程的同步、互斥关系

2. 确定各进程的P、V顺序

3. 设置信号量，并确定初值

4. 注：进入临界区之前，先实现同步P，再实现互斥P

在学校自选餐厅里工作人员向桌子上随机放一碗面条或米饭供学生拿走(如果桌上已有食物需要等待桌子空后再放)，学生从桌子上选择并取走一份主食，男生只取走面条(如果桌子上为空或是米饭需等待)，女生取走米饭(如果桌子上为空或是面条需等待)，桌子上只能容纳一份主食。请使用记录型信号量和 P，V 操作实现并写出工作人员、男生和女生三个进程同步过程的伪代码。

2. 购物问题。某超级市场，可容纳 100 个人同时购物，入口处备有篮子，每个购物者可持一个篮子入内购物。出口处结账，并归还篮子（出、入口仅容纳一人通过）。请用 P、V 操作完成购物同步算法。

2. 汽车司机与售票员之间必须协同工作，一方面，只有售票员把车门关好了,司机才能开车，因此，售票员关好车门应通知司机开车；另一方面，只有当汽车已经停下，售票员才能开门上下客，故司机停车后应通知售票员．假定某辆公共汽车上有一名司机与两名售票员．汽车当前正在始发站停车上客.试设必要的信号量并赋初值，用 P、V 操作写出他们的同步算法。

哲学家进餐（有限资源的竞争问题）

问题：如果5个哲学家同时进餐，且在0号哲学家拿起左边筷子时，发生进程切换到1号哲学家拿起左边筷子，又发生进程切换...最终5个哲学家都拿起左边的筷子，每个哲学家都在等待其他哲学家放下筷子，自己不放下筷子的死锁现象

解决方法一：对哲学家进程添加一些限制条件，比如最多运行四个哲学家同时进餐，这样就可以保证至少有一个哲学家可以进餐，进餐完成后放下两根筷子，激活旁边堵塞的哲学家进程。。。所有哲学家都可以进餐

解决方法二：要求奇数号的哲学家先拿左边的筷子，然后再拿右边的筷子，二偶数号哲学家刚好相反。用这种方法可以保证如果相邻两个奇偶号哲学家都想吃饭，那么自会有其中一个哲学家拿起第一只筷子，另外一个没有拿起筷子的哲学家进程会直接堵塞，这就避免了哲学家占有一只筷子后等待另一只的情况、

解决方法三：仅当一个哲学家左右两边筷子都可以使用时才允许其进餐。

读者—写者问题

- 设置互斥信号量Wmutex，实现Reader与Writer进程间在读或写时的互斥。
- 设置整型变量readcount，表示正在读的进程数目。
- 设置互斥信号量Rmutex，实现多个Reader进程对临界资源readcount的访问。
- 说明：若有一个Reader进程在读，就不允许Writer进程去写。因此，仅当无Reader进程在读时(**readcount=0**)，Reader进程才需要执行**Wait(Wmutex)**操作。若**Wait(Wmutex)**操作成功，Reader进程便可去读，并相应做**readcount加1**操作。同理，**仅当Reader进程执行readcount减1操作后值为0时，才须执行signal(Wmutex) 操作，以便让Writer进程写。**

Summary:

1. 确定写n个进程
2. 分析进程之间的关系

① 抢用-临界资源 → 互斥关系　A B

wait (Rmutex)
signal (Rmutex) 〉成对出现　设置一个信号量 mutex 就够了

② 生产者-消费者 → 协作关系　A → ↗ B

分别为 A 期望的(不满)和 B期望发生的(不空)事件设置信号量
P.V 操作可能出现在不同的进程里)

③ 实现前驱关系　A → B　(设置一个信号量 mutex=0，初始值为0)
谁结束谁开始　　一开始 B 在执行不了 P操作
P.V 操作交叉在不同进程里，但总要对等!　mutex++　signal释放 → wait

读者—写者问题可描述如下:

```
        semaphore Rmutex, Wmutex= 1 , 1;
        int  readcount =0;

void Reader()  {                    控制对readcount的互斥操作
                  while(True){
                      wait(Rmutex);
                      if Readcount==0   wait(Wmutex);
                      Readcount=Readcount+1;
                      signal(Rmutex);
                        ...
                      perform read operation;
                        ...

                    wait(Rmutex);
                    readcount=readcount-1;
                    if readcount==0  signal(Wmutex);
                    signal(Rmutex);
                  }
             }
void writer() {
                repeat
                    wait(Wmutex);
                    perform write operation;
                    signal(Wmutex);
                until false;
             }
```

3. （1）独木桥问题。某条河上只有一座独木桥，以便行人过河。现在河的两边都有人要过桥，按照下面的规则过桥。为了保证过桥安全，请用 P、V 操作分别实现正确的管理。过桥的规则是：同一方向的可连续过桥，某方向有人过桥时另一方向的人要等待。