第4-6章

1.线程

（1）进程中的**进程控制块和用户地址空间**被一个进程中的所有线程共享

（2）用户级线程相较于内核级线程的三个优点：

(1)线程切换不需要内核态特权，节省了两次状态转换的开销。
(2)调度可以是应用程序相关的。
(3)用户级线程可以在任何操作系统中运行，不需要对底层内核进行修改以支持用户级线程。

（3）用户级线程相较于内核级线程的两个缺点：

(1)用户级线程执行一个系统调用时，进程中的所有线程都会被阻塞。
(2)不能利用多处理技术。

2.信号量

（1）定义：用于进程间传递信号的一个整数值。在信号量上只可以进行三种操作，即初始化、递减和增加，这三种操作都是原子操作。递减操作作用于阻塞一个进程，递增作用于解除一个进程的阻塞。信号量也称为计数信号量或一般信号量。

（2）信号量原语

```
struct semaphore {
     int count;
     queueType queue;
};
void semWait(semaphore s)
{
     s.count--;
     if (s.count < 0) {
          /* place this process in s.queue */;
          /* block this process */;
     }
}
void semSignal(semaphore s)
{
     s.count++;
     if (s.count <= 0) {
          /* remove a process P from s.queue */;
          /* place process P on ready list */;
     }
}
```

Figure 5.3 A Definition of Semaphore Primitives

（3）信号量实现互斥伪代码

```
/* program mutualexclusion */
const int n = /* number of processes  */;
semaphore s = 1;
void P(int i)
{
     while (true) {
          semWait(s);
          /* critical section   */;
          semSignal(s);
          /* remainder   */;
     }
}
void main()
{
     parbegin (P(1), P(2), . . ., P(n));
}
```

Figure 5.6 Mutual Exclusion Using Semaphores

5.【简答题】（10分）

假设一个阅览室有100个座位，没有座位时读者在阅览室外等待；每个读者进入阅览室时都必须在阅览室门口的一个登记本上登记座位号和姓名，然后阅览，离开阅览室时要去掉登记项。每次只允许一个人登记或去掉登记。用信号量操作描述读者的行为。

参考答案

```
数据结构：
struct{char name[10];
        int number;
  }A[100];
semaphore mutex, seatcount;
mutex：用来互斥，初值为1
seatcount：对空座位计数，初值为100
for(int i=0;i<100;i++)
    {A[i].number=i; A[i].name=null;}

process readeri(char readername[]){
   semWait(seatcount);
   semWait (mutex);
   for(int i=0;i<100;i++)
    if(A[i].name==null) A[i].name=readername;//跳出
   reader get the seat number=i;
   semSignal (mutex);
    进入阅览室，在座位号i处坐下读书;
   semWait (mutex);
   A[i].name=null;
      semSignal(mutex);
   semSignal(seatcount);
       离开阅览室；}
```

6.在公共汽车上，司机和售票员的活动如下：1）司机：启动汽车，正常行车，到站停车； 2）售票员：关车门，售票，开门上下客。信号量操作描述两者的同步

参考答案

```
数据结构:
semaphore  s1=0;  /*表示是否允许司机启动汽车*/
semaphore  s2=0;  /*表示是否允许售票员开门*/
driver:
  while(true)
{
  semWait(s1);
   启动车辆;
   正常行车;
   到站停车;
  semSignal(s2);
}
busman:
  while(true)
{
  关车门;
  semSignal(s1);
  售票;
  semWait(s2);
  开车门;
  上下乘客;
}
```

7.独木桥问题

```
数据结构
semaphore wait, mutex1, mutex2;
mutex1=mutex2=wait=1;
int count1, count2;
count1=count2=0;

 process P 东(){
    semWait(mutex1);
    count1++;
    if(count1==1)   semWait(wait);
    semSignal(mutex1);
     过独木桥;
    semWait(mutex1);
    count1--;
    if(count1==0)   semSignal(wait);
    semSignal(mutex1);
 }

 process P 西(){
    semWait(mutex2);
    count2++;
    if(count2==1)   semWait(wait);
    semSignal(mutex2);
     过独木桥;
    semWait(mutex2);
    count2--;
    if(count2==0)   semSignal(wait);
    semSignal(mutex2);
 }
```

3.死锁

1.

产生死锁的四个条件是什么？ 死锁避免、检测和预防之间的区别是什么？

（1）产生死锁的四个条件：互斥、占有且等待、不可抢占、循环等待；
（2）死锁避免通过限制进程启动或资源分配，预防通过运用某种策略来消除产生死锁的四个条件之一，来保证不让死锁状态出现；而检测允许死锁出现，定期检测死锁的存在并从死锁中恢复出来。

第7-9章

1.抖动：抖动是虚存管理方案中可能出现的一种现象，处理器花费在交换上的时间多于执行指令的时间

2.转换检测缓冲区的目的：

TLB是一个高速缓冲存储器，包含最近一段时间频繁使用到的页表项，从而能够减少数据访问需要的时间。

3.驻留集是指进程执行的任何时候都在内存里的进程的部分页集，而工作集是指进程在过去一段时间被访问到的页集。

4.

抢占：当前正在运行的进程可能被中断，并转移到就绪状态； 非抢占：一旦进程处于运行状态，除非阻塞，会一直运行到终止状态。

5.

操作系统创建一个新进程所执行的步骤是什么?

我的作答:

1.给进程分配一个唯一的进程识别号。
2.给进程分配空间。
3.初始化进程控制块
4.设置正确的连接
5.创建或扩充其他的数据结构