哈工大操作系统-L16进程同步与信号量

多进程同步如何合理有序?

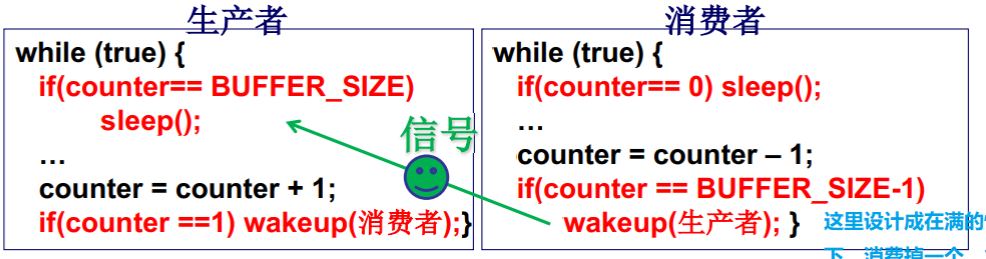
1.等待--实现进程同步的重要一步

- 需要让"进程走走停停"来保证多进程合作的合理有序，这就是进程同步
- 分析什么时候停和什么时候走是非常关键的。

2.生产者消费者问题

2.1初步设想--通过发信号解决问题

只发信号还不能解决全部问题



这里设计成在满的情况下，消费掉一个，才唤醒生产者。
那如果满了之后，连续消费掉两个，且有两个生产者在等待。
那么只有一个生产者能被唤醒。

- (1) 缓冲区满以后生产者P_1生产一个item放入，会sleep
- (2) 又一个生产者P_2生产一个item放入，会sleep
- (3) 消费者C执行1次循环，counter==BUFFER_SIZE-1，发信号给P_1，P_1 wakeup
- (4) 消费者C再执行1次循环，counter==BUFFER_SIZE-2，P_2不能被唤醒

问题：怎么办?

- 单纯依靠发信号是不足以解决问题的。
- **还需要多一个量来判断是否需要发信号，由此引出信号量**

2.2引出信号量

从信号到信号量

- 不只是等待信号、发信号？ 对应睡眠和唤醒!
- 还应该能记录一些信息
  - 能记录有“2个进程等待”就可以了…
  - *(1) 缓冲区满，P_1执行，P_1 sleep，记录下1个进程等待*
  - *(2) P_2执行， P_2 sleep，记录下2个进程等待*
  - *(3) C执行1次循环，发现2个进程等待，wakeup 1个*
  - *(4) C再执行1次循环，发现?个进程等待，再?*
  - *(5) C再执行1次循环，怎么办? 此时再来P_3怎么办?*

  什么是信号量？记录一些信息(量)，并根据这个信息决定睡眠还是唤醒(信号)。

信号量开始工作…

信号量解决了单纯发信号P2永远不会唤醒的问题

初始时 sem = ?

- *(1) 缓冲区满，P_1执行，P_1 sleep* sem = -1 有一个生产者在等待。负数表示缺一个坑位 什么含义?
- *(2) P_2执行， P_2 sleep* sem = -2 有两个生产者在等待。负数表示缺两个坑位
- *(3) C执行1次循环， wakeup P_1* sem = -1
- *(4) C再执行1次循环， wakeup P_2* sem = 0 正数表示坑位有多一个，下一个生产者来了可以直接生产
- *(5) C再执行1次循环，* sem = 1 什么含义?
- *(6) P_3执行， P_2 继续执行* sem = 0

- 总结一下：这个整数什么时候-1?什么时候+1?什么时候睡眠?什么时候唤醒?

- 信号量--记录一些信息的量，由这些信息决定是睡眠还是唤醒。
- 上述感觉反了，生产的资源应该永远不会超过可用资源的个数。
  - 下面是我对这个模型的理解
  - 如果我们用生产者消费者构建访问共享资源的模型。
  - 可用的资源即为共享的资源
  - 消费者跟生产者是相对的，每个进程都能执行消费和生产的动作
  - 当进程想要访问资源的时候，就为消费者，如果有可用的资源才可消费(访问)
  - 当进程访问资源结束的时候，释放资源，即为生产者(因此，生产者不会生产多于可用资源个数的资源)

2.3信号量的定义

什么是信号量? 信号量的定义...

- 信号量: 1965年，由荷兰学者Dijkstra提出的一种特殊整型变量，量用来记录，信号用来sleep和wakeup

```
struct semaphore
{
   int value; //记录资源个数
   PCB *queue;
    //记录等待在该信号量上的进程
}
P(semaphore s);    //消费资源
V(semaphore s);    //产生资源
```

```
P(semaphore s)
{
  s.value--;
  if(s.value < 0) {
    sleep(s.queue); }
}
```

```
V(semaphore s )
{
  s.value++;
  if(s.value<=0) //加完之后等于x说明原来有 -x+1个人在等待
  {
    wakeup(s.queue);
  }
}
```

问题：写出V(s)的代码?

P的名称来源于荷兰语的proberen，即test
V的名称也来源于荷兰语verhoogen(increment)

- semaphore 英 ['seməfɔ:(r)]
  - n. 信号标; 旗语; 信号量
  - v. 打旗语
- 信号量由两部分构成 value 和queue。
  - queue是一个等待队列
  - value为负，当前无可用资源，且有多个进程在等待
  - value为0，当前无可用资源，但无进程等待资源
  - value为正，说明有可用资源，且无进程在等待资源
- 上面这个代码传参数应该用引用或者指针才行...

用信号量解生产者–消费者问题

```
int fd = open("buffer.txt");
write(fd, 0, sizeof(int)); //写in
write(fd, 0, sizeof(int)); //写out
```

用文件定义共享缓冲区

```
semaphore full = 0;
semaphore empty = BUFFER_SIZE;
semaphore mutex = 1;
```

信号量的定义和初始化

```
Producer(item) {
   P(empty);
   P(mutex);
   读入in;将item写入到in的位置上;
   V(mutex);
   V(full); }
```

```
Consumer() {
   P(full);
   P(mutex);
   读入out;从文件中的out位置读出到item;打印item;
   V(mutex);
   V(empty); }
```

互斥信号量

这个互斥信号量，讲的不够全面，还是看电子科大的课好了