信号量机制在 P—V操作中的应用

乔良才 (徐州工程学院信电工程学院, 江苏徐州, 221008)

[摘 要] 详细介绍和分析信号量机制的实现, 利用 PV操作实现进程的互斥与同步, 通过两个实例的进一步分析, 求得类似问题的求解规律。

[关键词] 进程的同步与互斥; PV操作; 信号量
[中图分类号] TP301 [文献标识码] A

[作者简介] 乔良才 (1977—), 男, 江苏连云港人, 讲师, 研究方向为计算机软件理论。

多道程序系统中进程是并发执行的, 这些进程之间存在着不同的相互制约关系, 为了协调进程之间的相互制约关系, 就需要实现进程的同步, 互斥是同步的一种特殊情况。本文通过两个实例的详细分析, 从而寻求该类问题规律性的求解方案。

一、 P—V操作的描述

(一) 进程同步与互斥

进程同步, 是指对多个相关进程在执行次序上的协调, 这些进程相互合作, 在一些关键点上可能需要互相等待或互通消息。在操作系统中, 当一个进程进入临界区使用临界资源时, 另一个进程必须等待, 当占用临界资源的进程退出临界区后, 另一进程才被允许去访问此临界资源, 我们称进程之间的这种相互制约关系为互斥。

(二) 进程同步机制应遵循的原则

为了实现进程的同步与互斥, 可利用软件方法, 也可以在系统中设置专门的同步机制来协调诸进程, 但所有的同步机制都必须遵循以下四条原则:[1]

①空闲让进。当没有进程处于临界区时, 临界资源处于空闲状态, 可以允许一个请求进入临界区的进程进入自己的临界区, 有效地使用临界资源。

②忙则等待。当已经有进程进入自己的临界区时, 意味着临界资源正被访问, 因而其它试图进入临界区的进程必须等待, 以保证进程互斥的使用临界资源。

③有限等待。对要求访问临界资源的进程, 应保证进程在有效的时间内进入自己的临界区, 以免陷入“死等”状态。

④让权等待。当进程不能进入自己的临界区时, 应立即释放处理器, 以免进程陷入“忙等”状态。

(三) P—V操作描述

1965年, 荷兰学者 Dijkstra提出的信号量 (Semaphores) 机制是一种卓有成效的进程同步工具。信号量是一个确定的二元组 (S, Q), 其中 S是一个具有非负初值的整形变量, Q是一个初始状态为空的队列, 整形变量 S表示系统中某类资源的数目, 当其值大于 0时, 表示系统中当前可用资源的数目, 当其值小于 0时, 其绝对值表示系统中因请求该类资源而被阻塞的进程数目。除信号量的初值外, 信号量的值仅能有 P操作和 V操作改变。

一个信号量的建立必须经过说明, 即应该准确说明 S的意义和初值, 每个信号量都有相应的一个队列, 在建立信号量时, 队列为空。

P, V操作的定义如下:

P操作记为 P(S), 它执行时主要完成下述动作:

S=S-1;

若 S≥0, 则该进程继续运行。

若 S<0, 则该进程被阻塞, 并将它插入该信号量的等待队列中, 即 block(S, Q)。

V操作记为 V(S), 它执行时主要完成下述动作:

S=S+1;

若 S>0, 则进程继续执行。

若 S≤0, 则从信号量的等待队列中移出第一个等待进程, 使其变为就绪状态, 然后再返回原进程继续执行, 即 wakenup(S, Q)。

二、利用 P—V操作实现进程互斥

例: 生产者——消费者 (producer—consumer) 问题是进程同步与互斥的经典模型, 描述的是有一组生产者向一组消费者提供产品, 它们共享一个有界缓冲区, 生产者向其中投放产品, 消费者从中取得产品, 它们之间必须保持同步, 即不允许消费者进程到一个空缓冲区去取产品, 也不允许生产者进程向一个已装满产品且尚未被取走的缓冲区中投放产品。

分析: 我们可以通过一个有界缓冲区把一群生产者和一群消费者联系起来, 假定这些生产者和消费者是相互等效的, 设置信号量 avail初值为 n表示可以利用的缓冲区数目, full初值为 0表示计算产品数目, 只要缓冲区未满, 生产者在执行了 P(avail) 操作后 avail≥0 就可以把产品送入缓冲区; 类似的, 只要缓冲区未空, 消费者在执行 P(full) 操作后 full≥0 就可以从缓冲区取走产品并消耗它, 仅当缓冲区满时, 生产者在由于在执行 P(avail) 操作后 avail<0 因而被阻塞; 类似的, 缓冲区空时消费者进程执行了 P(full) 操作后 full<0 因而被阻塞, 为了实现上述进程的互斥进入临界区, 我们设置了两个私用信号量和一个公用信号量如下:

①公用信号量 mutex, 初值为 1。

②私用信号量 full, 初值为 0

③私用信号量 avail 初值为 n。

生产者——消费者问题的流程如下：

```
BEGIN integer mutex, avail, full;
mutex := 1;
avail := n;
full := 0;
COBEGIN

Producer: BEGIN                    Consumer: BEGIN
L1: producer next product;         L2: P(full);
P(avail);                          P(mutex);
P(mutex);                          Take from buffer;
Add to buffer;                     V(avail);
V(full);                           V(mutex);
V(mutex);                          Consume the product;
Goto L1;                           Goto L2;
END                                END;
                                   COEND
                                   END
```

在这里应特别注意的一点是进行 P操作的顺序，无论是生产者还是消费者进程，V操作的顺序都无关紧要，因为 V操作不会阻塞进程，但 P操作的次序不能颠倒，否则将可能造成死锁。

三、利用 P—V操作实现进程同步

例：有三个进程 PA、PB和 PC合作解决文件打印问题：PA将文件记录从磁盘读入主存的缓冲区 1，每执行一次读一个记录；PB将缓冲区 1的内容复制到缓冲区 2，每执行一次复制一个记录；PC将缓冲区 2的内容打印出来，每执行一次打印一个记录。缓冲区的大小等于一个记录大小，请用 P、V操作来保证文件的正确打印[2]。

分析：在本题中，这是一个明显的同步问题，也称为生产者和消费者问题。确定进程 PA、PB、PC之间的关系为：PA与 PB共用一个缓冲区，而 PB又与 PC共用一个单缓冲区，其合作方式如图所示：当缓冲区 1为空时，进程 PA可将一个记录读入其中；若缓冲区 1中有数据且缓冲区 2为空，则进程 PB可将记录从缓冲区 1中复制到缓冲区 2中；若缓冲区 2中有数据，则进程 PC可以打印记录。在其他条件下，相应进程必须等待。事实上，这是一个生产者－消费者问题。相对于缓冲区 1，PA是生产者；相对于缓冲区 2，PC是消费者；PB身份特殊，相对于缓冲区 1，PB是消费者，相对于缓冲区 2，PB是生产者。

为遵循这一同步规则。应设置 4个信号量 empty1、empty2、full1、full2，信号量 empty1及 empty2分别表示缓冲区 1及缓冲区 2是否为空，其初值为 1；信号量 full1及 full2分别表示缓冲区 1及缓冲区 2是否有记录可供处理，其初值为 0。

其同步描述如下：

```
int empty1 = 1;
int empty2 = 1;
int full1 = 0;
int full2 = 0;

main()
{
cobegin
PA();
PB();
PC();
coend
}
PA()
{
while(1)
{ 从磁盘读一个记录;
p(empty1);
将记录存入缓冲区 1;
v(full1);
}
}
PB()
{
while(1)
{ p(full1);
从缓冲区 1中取出记录;
v(empty1);
p(empty2);
将记录存入缓冲区 2;
v(full2);
}
}
PC()
{
while(1)
{ p(full2);
从缓冲区 2中取出记录;
v(empty2);
打印记录;
}
}
```

四、结 语

由以上两个例子可以得出 P—V操作的使用规则：（1）分清哪些是互斥问题（互斥访问临界资源的），哪些是同步问题（具有前后执行顺序要求的，（2）对互斥问题要设置互斥信号量，不管有互斥关系的进程有几个或几类，通常只设置一个互斥信号量，且初值为 1，代表一次只允许一个进程对临界资源进行访问。（3）对同步问题要设置同步信号量，通常同步信号量的个数与参与同步的进程种类有关，即同步关系涉及几类进程，就有几个同步信号量。同步信号量表示该进程是否可以开始或该进程是否已经结束。（4）在每个进程中用于实现互斥的 P—V操作必须成对出现；用于实现同步的 P—V操作也必须成对出现，但可以分别出现在不同的进程中；在某个进程中如果同时存在互斥与同步的 P操作，则其顺序不能颠倒，必须先执行对同步信号量的 P操作，在执行对互斥信号量的 P操作，但 V操作的顺序没有严格要求。

说明：同步或互斥问题的解题步骤如下：（1）确定进程。包括进程的数量和进程的工作内容，可以用流程图描述。（2）确定同步或互斥关系。根据使用的是临界资源，还是处理的前后关系，来确定同步与互斥，然后确定信号量的个数、含义，以及对信号量的 P—V操作。（3）用类 C语言描述同步或互斥算法。

读者－写者问题是进程同步问题中的经典实例，若能熟练掌握，对于常见的同类进程多次访问，而不同类的进程必须互斥访问资源的控制问题也就不难解决了。

参考文献：

[1] 汤子瀛. 计算机操作系统[M]. 西安：西安电子科技大学出版社，2006.

[2] 前沿考试研究室. 操作系统与编译原理分册[M]. 北京：民邮电出版社，2000.