南京大学
NANJING UNIVERSITY

计算机操作系统

6 并发程序设计 – 6.4 管程

6.4.3 霍尔管程的例

霍尔管程解决机票问题

霍尔管程解决读者写者问题

应用霍尔管程解决具体问题

哲学家问题

五个哲学家围坐在一个圆桌旁，桌中央有一盘通心面，每人面前有一只空盘子，每两人之间放一把叉子。每个哲学家思考，饥饿，然后吃通心面，为了吃面哲学家必须获得自己左边和右边的两把叉子

1、等待叉子的方案

2、等待盘子的方案

霍尔方法实现哲学家问题

```
TYPE dining_philosophers = MONITOR
    var state : array[0..4] of (thinking, hungry, eating);
          s : array[0..4] of semaphore;
          s_count : array[0..4] of integer;
    define pickup, putdown;
    use wait, signal;
procedure test(k : 0..4) {
    if state[(k-1) mod 5]<>eating and state[k]=hungry
          and state[(k+1) mod 5]<>eating then
            { state[k] := eating;    signal(s[k], s_count[k], IM); }
}
```

```
procedure pickup(i:0..4) {
    state[i] := hungry;
    test(i);
    if state[i]<>eating
    then wait(s[i], s_count[i],IM);
}
```

```
procedure putdown(i:0..4) {
    state[i] := thinking;
    test((i-1) mod 5);
    test((i+1) mod 5);
}
```

```
begin    for    i := 0 to 4 do state[i] := thinking;    end;
```

哲学家问题

```
Cobegin
 process philosopher_i() {
    L:thinking();
    P(IM.mutex);
    dining_philosopers.pickup(i);
    if IM.next_count>0 then V(IM.next);  else V(IM.mutex);
    eating();
    P(IM.mutex);
    dining_phi;osophers.putdown(i);
    if IM.next_count>0 then V(IM.next); else V(IM.mutex);
    goto L;
 }
coend
```

读者写者问题

有两组并发进程：读者与写者，共享一个文件，要求：1）允许多个读者同时执行读操作；2）任一写者出现后，新的读者与写者均被拒绝；3）写者在完成写操作之前不允许其他读者和写者工作；4）写者欲工作，要等待已存在的读者完成读操作

1、等待读的进程与计数；等待写的进程与计数

2、正在读的进程计数；正在与等待写的进程计数

霍尔方法解决读者写者问题

```
TYPE read-writer = MONITOR
  var Rc, Wc : integer; R, W : semaphore;   rc, wc: integer;
  define start_read, end_read, start_writer, end_writer;
  use wait, signal, check, release;
```

```
begin  rc:=0; wc:=0; Rc:=0; Wc:=0; R:=0; W:=0;  end;
```

```
procedure start_read;
begin
   if wc>0 then wait(R,Rc,IM);
   rc := rc + 1;
   signal(R, Rc, IM);
end;
```

连续释放读者

```
procedure end_read;
begin
   rc := rc - 1;
   if rc=0 then signal(W,Wc,IM);
end;
```

```
procedure start_write;
begin
   wc := wc + 1;
   if rc>0 or wc>1 then wait(W,Wc,IM);
end;
```

```
procedure end_write;
begin
   wc := wc - 1;
   if wc>0 then signal(W,Wc,IM);
           else signal(R, Rc,IM);
end;
```