《操作系统》第4章作业 2023.10.19

大数据2101班 李嘉鹏 U202115652

4-3

进程就是一个程序在给定空间和初始环境下，在一个处理机上的执行过程。

进程与程序的主要区别主要有3点：

①程序是指令的有序集合，是一个静态概念，其本身没有任何运行的含义。而进程是程序在处理机上的一次执行过程，是一个动态概念。程序可以作为一种软件资料长期保存，而进程则是有一定生命期的，它能够动态地产生和消亡，也就是说进程可以由“创建”而产生，由调度而执行，因得不到资源而暂停，以致最后由“撤销”而消亡。

②进程是一个能独立运行的单位，能与其它进程并行活动，是竞争计算机系统有限资源的基本单位，也是进行处理机调度的基本单位，而程序则不是。

③进程与程序之间没有一一对应的关系。一个程序可以对应多个进程，一个进程至少包含一个程序。

4-9

（1）变迁2：时间片耗尽；变迁3：服务请求（如I/O）；变迁4：服务完成/事件来到。

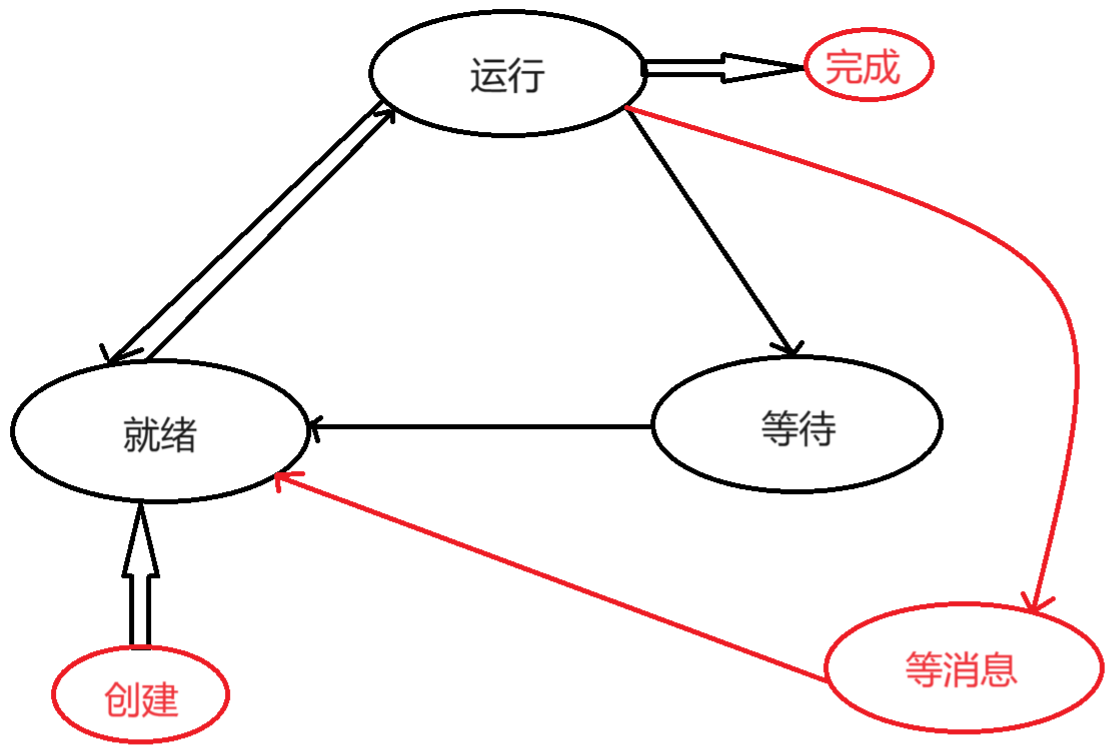
（2）一个进程如果需要处理输入输出，则会发生变迁3，该进程状态由“运行”变为“等待”。此时CPU资源空闲，如果此时就绪队列非空，那么下一个就绪的进程即可发生变迁1，状态由“就绪”变为“运行”。

（3）a.2→1是可能的（一定会发生），例如一个“运行”的进程用尽时间片后转为“就绪”，下一个“就绪”的进程即可开始运行；

b.3→2是不可能的；

c.4→1是可能的，对于某一个进程，一旦其从“等待”成功转变为“就绪”且当前CPU空闲、就绪队列为空，则可以直接从“就绪”转变为“运行”。

4-10



增加新状态后，进程状态变迁图如上图所示。每一种变迁的原因分别是：

①“创建”→“就绪”：很显然，一个进程被创建出来之后就可以获得除了CPU之外的所有资源（理想情况下）。

②“就绪”→“运行”：当CPU转为空闲时，通过进程调度，进程就可以立即开始运行。

③“运行”→“等待”：服务请求（例如请求I/O等）。

④“运行”→“就绪”：时间片到。

⑤“等待”→“就绪”：服务完成/事件来到。

⑥“运行”→“等消息”：当前进程需要等待其它进程发送必要的消息或数据，这涉及到不同进程之间的通信。

⑦“等消息”→“就绪”：进程获得必要的信息后，即可转变为就绪状态，一旦CPU空闲即可继续执行。

⑧“运行”→“完成”：进程执行完成。

4-12

设信号灯为（s, q），其中s是一个具有非负初值（此题中取初值为1）的整型变量，q为一个初始状态为空的PCB（进程控制块）的队列。

用信号灯实现n个进程互斥时的程序描述如下：

main() {

int s=1; //信号灯初值为1

cobegin

p1(); //进程1

p2(); //进程2

……

pn(); //进程n

coend

}

p1() {

…

P(s); //信号灯-1

临界段

V(s); //信号灯+1

}

……（后面每个p进程的程序结构均与p1类似，分别为p2到pn）

信号灯值的取值范围：s∈[1-n, 1]且s为整数。

每个取值的物理意义：s=1时，代表公共变量Q可用，且当前没有任何一个进程在使用Q。

s=0时，代表公共变量Q正在被某一个进程使用，且其它所有进程此时并没有尝试使用Q。

s＜0时，代表公共变量Q正在被某一个进程使用，且此时有|s|个进程正在等待使用Q。

4-13

（a）设置三个信号灯，分别为s2、s3、s4

用信号灯的P、V操作实现进程之间的同步，程序描述如下：

main() {

int s2=0;

int s3=0;

int s4=0;

cobegin

p1();

p2();

p3();

p4();

coend

}

p1() {

…

V(s2);

V(s3);

V(s4);

}

p2() {

P(s2);

…

}

p3() {

P(s3);

…

}

p4() {

P(s4);

…

}

（b）设置两个信号灯，分别为s1、s2

用信号灯的P、V操作实现进程之间的同步，程序描述如下：

main() {

int s1=0;

int s2=0;

cobegin

p1();

p2();

p3();

coend

}

p1() {

…

V(s1);

}

p2() {

…

V(s2);

}

p3() {

P(s1);

P(s2);

…

}

4-16

进程的互斥是指：在操作系统中，当某一进程正在访问某一存储区域时，不允许其他进程来读出或者修改该存储区的内容，否则就会发生后果无法估计的错误。进程之间的这种相互制约关系称为互斥。

进程的同步是指：并发进程在一些关键点上可能需要互相等待与互通消息，这种相互制约的等待与互通信息称为进程同步。

同步与互斥这两个概念的联系与区别：二者都属于同步的范畴，描述了并发进程之间的约束关系。同步的概念强调的是保证进程之间操作的先后次序的约束，而互斥更强调在某一点、某个时刻只允许一个进程执行。

4-22

线程是比进程更小的活动单位，是进程中的一个执行路径。一个进程可能有多个执行路径，即线程。

线程与进程的区别在于：

①线程是进程的一个组成部分，一个进程可以有多个线程，并且有至少一个可执行的线程；

②进程是资源分配的基本单位，它拥有自身的地址空间和各种资源；而线程是处理机调度的基本单位，只能共享父进程的资源和地址空间；

③进程的多个线程都在进程的地址空间内活动。在以线程为单位进行处理机调度和切换时，处理时间较短；而以进程为单位进行处理机调度和切换时，由于涉及到资源转移及现场保护等问题，处理时间较长；

④线程不能将程序和数据交换到外存去，而进程可以；

⑤进程的调度和控制大多数由操作系统的内核完成，而线程的控制既可以由操作系统的内核完成，也可以由用户控制完成。

4-29

设置一个信号灯（count, q）。count初值为5，代表长凳有5个空位。同时设置一个信号灯为mutex，代表访问当前count的互斥信号灯。

使用长凳的过程可以被抽象成一个进程：

main() {

int count=5;

int mutex=1;

cobegin

guesti; //i=1,2,3…

coend

}

guesti() { //i=1,2,3…

P(mutex);

if(count==0) {

V(mutex);

return;

}

count--;

V(mutex);

（休息）

P(putex);

count++;

V(mutex);

}

或者也可以只使用一个信号灯（s, q）。s初值为5，代表长凳有5个空位。

main() {

int s=5;

cobegin

guesti; //i=1,2,3…

coend

}

guesti() { //i=1,2,3…

P(s);

（休息）

V(s);

}