**操作系统第二章习题解答**

2.1 情况（a）和情况（b）具有相同的答案。

假设处理器的操作不能重叠，但I/O操作可以。

1job：时间周期=NT

处理器利用率=50%；

2jobs：时间周期=NT

处理器利用率=100%；

4jobs：时间周期=（2N-1)NT

处理器利用率=100%

2.2 I/O限制程序只用相对较少的处理时间，

因此，受到短期调度算法的偏爱。然而，如果

一个处理器限制程序在一段很长的时间内被处

理器时间拒绝，那同样的这个短期调度算法则

会允许处理机去处理过去一段时间一直没有使

用处理机的程序，所以，并不是永远不受理处

理器限制程序所需的处理器时间。

2.3 关于分时系统，我们所关注的是周转时间。

首选的是时间片，因为它在一个很短的时间给

所有的程序一个访问权限去使用处理器。在批

处理系统，我们所关注的是吞吐量和更少量的上

下文转换，对于进程来说获得了更多的处理时

间。因此，最小化上下文转换的处理是有优势

的。

2.4 应用程序运用系统调用去调用操作系统所

提供的功能。关键的是，系统调用导致转换到

进入内核模式的系统程序。

**操作系统第三章习题解答**

3.1 系统和用户进程的创建和删除：在系统中进程对于信息共享，加速计算，模块性 和便利性都能并发执行。并发的执行需要进程的创建和删除机制。进程所需要的资源在进程被创建时获得或者在其运行的时候分配。当进程结束时，操作系统需要收回任何可重用资源。

进程的挂起和恢复：在进程调度中，当进程在等待某些资源时，操作系统需要把进程状态改变成等待或者就绪状态。当进程所要求的资源可用时，操作系统需要把它的状态变为运行状态恢复它的执行。

进程同步机制：协调进程分享数据。 并发访问使用共享数据可能导致数据不一致性，操作系统不得不为其提供一种进程同步机制用来确保协作进程有序的实行，从而保证数据的一致性。

进程通信机制 ：在操作系统下执行的进程要么是独立的进程要么是协作的进程。 协作进程必须使用某些方法来实现进程间的通信。

死锁处理机制：在一个多道程序设计环境里，一些进程可能因为有限数量的资源而产生竞争。 如果一个死锁发生，全部等待的进程都不会从等待状态改变成运行状态，那么资源被浪费，工作不会被完成。

3.4 对处于就绪/挂起状态的所有进程通过一

个固定的优先级层次来划分，如分成一到两

个优先级，只有当就绪/挂起状态的进程优先

级高于所有就绪状态进程的优先级时，才把

处理机分配给它。

3.6

a）采用4种模式的优点在于：系统能够提高对存储器的访问使用的灵活性，同时对内存储器的运行起到很好的保护作用。缺点：复杂度和处理开销。例如，处理器运行在不同的访问模式需要分离可访问的堆栈。

b）原则上，模式越多，灵活性适应性越大，但系统越复杂，举出一 种有4种以上模式的情况较难。

3.7 a） 当j<i时，一个在Di中运行的进程被阻止访问Dj中的对象。因此，如果Dj中包含的信息比Di优先权更高或者比Di更安全，这个限制是适当的。然而，这个安全政策可以用下面的方法更简单的获得。一个在Dj中运行的进程可以从Dj中读取数据，并且可以把数据复制到Di中，随后，在Di中运行的进程便可读取这些信息。

b）一个近似的解决这个问题的方法就是大家都知道的可信系统。在以后的章节会详细解释。

3.8 a）一个应用程序可能正处理从另一个进程收到的数据并且把结果储存在磁盘上。如果有等待取自其它进程的数据，应用程序可能进入下一个进程取出数据并且处理它。 如果一个先前的磁盘写操作已经完成并且有处理的数据写出，应用程序会将其写入下一个磁盘。需要考虑的一点就是，进程等待输入进程的额外数据和磁盘的可用性。

b）有几种处理的方式。 或者一种特定类型的队列来处理，或者进程可能被放进两个单独的队列。 无论哪种情况，操作系统必须处理细节，提醒进程注意双方事件一个接一个的发生。

3.9 这技术基于一个假设——中断进程A响应中断后将会继续进行。

但是， 通常， 一个中断将引起基本监督程序抢占进程A有利于另一个进程B。有必要在描叙进程A相关进程中断的位置复制进程A的执行状态，机器最好第一时间把它们储存在那里，以方便后续操作的进行。

3.10 因为存在进程不能被抢占的情况 (例如正在内核模式里执行的进程)，因此操作系统不能快速回复实时需求。

**操作系统第四章习题解答**

4.1 是的。因为更多的状态信息必须保留下来用于一个程序到另一个程序的转换。

4.2 因为，关于用户级线程，一个进程的线程结构对于操作系统来讲是不可视的，它仅仅是基于进程调度的一个基本单位。

进程和线程的区别在于：

简而言之,一个程序至少有一个进程,一个进程至少有一个线程. 线程的划分尺度小于进程，使得多线程程序的并发性高。

另外，进程在执行过程中拥有独立的内存单元，而多个线程共享内存，从而极大地提高了程序的运行效率。

线程在执行过程中与进程还是有区别的。

每个独立的线程有一个程序运行的入口、顺序执行序列和程序的出口。但是线程不能够独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制。

从逻辑角度来看，多线程的意义在于一个应用程序中，有多个执行部分可以同时执行。但操作系统并没有将多个线程看做多个独立的应用，来实现进程的调度和管理以及资源分配。这就是进程和线程的重要区别。

（续）

进程是具有一定独立功能的程序关于某个数据集合上的一次运行活动,进程是系统进行资源分配和调度的一个独立单位.

线程是进程的一个实体,是CPU调度和分派的基本单位,它是比进程更小的能独立运行的基本单位.线程自己基本上不拥有系统资源,只拥有一点在运行中必不可少的资源(如程序计数器,一组寄存器和栈),但是它可与同属一个进程的其他的线程共享进程所拥有的全部资源.

一个线程可以创建和撤销另一个线程;同一个进程中的多个线程之间可以并发执行.

4.4 这里的这个问题是机器花费了它工作中大

量时间去等待I/O的完成。在一个多线程程序

中，一个内核级线程使系统调用阻塞，而其它

内核级线程可以继续运行，而对于一个单独的

处理器，一个进程只有使所有调用阻塞，才能

使其它线程继续。

4.5 不会。当一个进程退出，它将带走关于它

的所有东西，内核级线程、进程结构、存储空

间，也包括线程。

4.6 尽可能多的关于地址空间的信息能够和其它地址空间进行交换，从而保存到主存储器中。

4.7 a）如果采取保守策略，那么最多有20/4=5个作业同时执行。因为分配给各自进程的设备中有一个设备在大多数时间里都是空闲的，在同一时间，最多有5个设备空闲，最好的情况，没有设备空闲，全部都在工作状态。

b）为了提高设备的利用率，最初每个作业分配3个磁带设备，第4个则要按需求分配。根据这个策略，至多有[20/3]=6个作业能被同时激活，最小空闲设备数是0，最大空闲设备数是2。

4.8 每一个调用都可能改变一个线程的优先级或者一个可运行的、具有更高优先级的线程也可以调用调度程序，而且它依次抢占低优先级的活跃线程。因此，可运行的、高优先级线程不具备题目所述。