第2章 进程与线程

从本章开始我们将深入考察操作系统是如何设计和构造的。操作系统中最核心的概念是进程：这是对正在运行程序的一个抽象。操作系统的其他所有内容都是围绕着进程的概念展开的，所以，让操作系统的设计者（及学生）尽早并透彻地理解进程是非常重要的。

进程是操作系统提供的最古老的也是最重要的抽象概念之一。即使可以利用的CPU只有一个，但它们也支持（伪）并发操作的能力。它们将一个单独的CPU变换成多个虚拟的CPU。没有进程的抽象，现代计算将不复存在。在本章里我们会通过大量的细节去探究进程，以及它们的第一个亲戚——线程。

2.1 进程

所有现代的计算机经常会在同一时间做许多件事。习惯于在个人计算机上工作的人们也许不会十分注意这个事实，因此列举一些例子可以更清楚地说明这一问题。先考虑一个网络服务器。从各处进入一些网页请求。当一个请求进入时，服务器检查是否其需要的网页在缓存中。如果是，则把网页发送回去；如果不是，则启动一个磁盘请求以获取网页。然而，从CPU的角度来看，磁盘请求需要漫长的时间。当等待磁盘请求完成时，其他更多的请求将会进入。如果有多个磁盘存在，会在满足第一个请求之前就接二连三地对其他的磁盘发出一些或所有的请求。很明显，需要一些方法去模拟并控制这种并发。进程（特别是线程）在这里就可以产生作用。

现在考虑只有一个用户的PC。一般用户不知道，当启动系统时，会秘密启动许多进程。例如，启动一个进程用来等待进入的电子邮件，或者启动另一个防病毒进程周期性地检查是否有新的有效的病毒定义。另外，某个用户进程也许会在所有用户上网的时候打印文件以及烧录CD-ROM。所有的这些活动需要管理，于是一个支持多进程的多道程序系统在这里就显得很有用了。

在任何多道程序设计系统中，CPU由一个进程快速切换至另一个进程，使每个进程各运行几十或几百个毫秒。严格地说，在某一个瞬间，CPU只能运行一个进程。但在1秒钟期间，它可能运行多个进程，这样就产生并行的错觉。有时人们所说的*伪并行*就是指这种情形，以此来区分多处理器系统（该系统有两个或多个CPU共享同一个物理内存）的真正硬件并行。人们很难对多个并行活动进行跟踪。因此，经过多年的努力，操作系统的设计者发展了用于描述并行的一种概念模型（顺序进程），使得并行更容易处理。有关该模型、它的使用以及它的影响正是本章的主题。

2.1.1 进程模型

在进程模型中，计算机上所有可运行的软件，通常也包括操作系统，被组织成若干*顺序进程*（sequential process），简称*进程*（process）。一个进程就是一个正在执行程序的实例，包括程序计数器、寄存器和变量的当前值。从概念上说，每个进程拥有它自己的虚拟CPU。当然，实际上真正的CPU在各进程之间来回切换。但为了理解这种系统，考虑在（伪）并行情况下运行的进程集，要比我们试图跟踪CPU如何在程序间来回切换简单得多。正如在第1章所看到的，这种快速的切换称作*多道程序设计*。

在图2-1a中我们看到，在一台多道程序计算机的内存中有4道程序。在图2-1b中，这4道程序被抽象为4个各自拥有自己控制流程（即每个程序自己的逻辑程序计数器）的进程，并且每个程序都独立地运行。当然，实际上只有一个物理程序计数器，所以在每个程序运行时，它的逻辑程序计数器被装入实际的程序计数器中。当该程序执行结束（或暂停执行）时，物理程序计数器被保存在内存中该进程的逻辑程序计数器中。在图2-1c中我们看到，在观察足够长的一段时间后，所有的进程都运行了，但在任何一个给定的瞬间仅有一个进程真正在运行。