# 第 12 章

CHAPTER 12

# 并发编程

正如我们在第 8 章学到的，如果逻辑控制流在时间上重叠，那么它们就是并发的(concurrent)。这种常见的现象称为并发(concurrency)，出现在计算机系统的许多不同层面上。硬件异常处理程序、进程和 Linux 信号处理程序都是大家很熟悉的例子。

到目前为止，我们主要将并发看做是一种操作系统内核用来运行多个应用程序的机制。但是，并发不仅仅局限于内核。它也可以在应用程序中扮演重要角色。例如，我们已经看到 Linux 信号处理程序如何允许应用响应异步事件，例如用户键入 Ctrl+C，或者程序访问虚拟内存的一个未定义的区域。应用级并发在其他情况下也是很有用的：

- *访问慢速 I/O 设备。*当一个应用正在等待来自慢速 I/O 设备(例如磁盘)的数据到达时，内核会运行其他进程，使 CPU 保持繁忙。每个应用都可以按照类似的方式，通过交替执行 I/O 请求和其他有用的工作来利用并发。
- *与人交互。*和计算机交互的人要求计算机有同时执行多个任务的能力。例如，他们在打印一个文档时，可能想要调整一个窗口的大小。现代视窗系统利用并发来提供这种能力。每次用户请求某种操作(比如通过单击鼠标)时，一个独立的并发逻辑流被创建来执行这个操作。
- *通过推迟工作以降低延迟。*有时，应用程序能够通过推迟其他操作和并发地执行它们，利用并发来降低某些操作的延迟。比如，一个动态内存分配器可以通过推迟合并，把它放到一个运行在较低优先级上的并发“合并”流中，在有空闲的 CPU 周期时充分利用这些空闲周期，从而降低单个 free 操作的延迟。
- *服务多个网络客户端。*我们在第 11 章中学习的迭代网络服务器是不现实的，因为它们一次只能为一个客户端提供服务。因此，一个慢速的客户端可能会导致服务器拒绝为所有其他客户端服务。对于一个真正的服务器来说，可能期望它每秒为成百上千的客户端提供服务，由于一个慢速客户端导致拒绝为其他客户端服务，这是不能接受的。一个更好的方法是创建一个*并发服务器*，它为每个客户端创建一个单独的逻辑流。这就允许服务器同时为多个客户端服务，并且也避免了慢速客户端独占服务器。
- *在多核机器上进行并行计算。*许多现代系统都配备多核处理器，多核处理器中包含有多个 CPU。被划分成并发流的应用程序通常在多核机器上比在单处理器机器上运行得快，因为这些流会并行执行，而不是交错执行。

使用应用级并发的应用程序称为*并发程序*(concurrent program)。现代操作系统提供了三种基本的构造并发程序的方法：

- *进程。*用这种方法，每个逻辑控制流都是一个进程，由内核来调度和维护。因为进程有独立的虚拟地址空间，想要和其他流通信，控制流必须使用某种显式的进程间通信(interprocess communication，IPC)机制。
- *I/O 多路复用。*在这种形式的并发编程中，应用程序在一个进程的上下文中显式地调度它们自己的逻辑流。逻辑流被模型化为状态机，数据到达文件描述符后，主程序显式地从一个状态转换到另一个状态。因为程序是一个单独的进程，所以所有的流都共享同一个地址空间。