的内存必须在所有路径上释放，以及设定许多其他的条件。

13.4 性能

所有事情都是平等的，一个快速的操作系统比一个慢速的操作系统好。然而，一个快速而不可靠的操作系统还不如一个慢速但可靠的操作系统。由于复杂的优化经常会导致程序错误，有节制地使用它们是很重要的。尽管如此，在性能是至关重要的地方进行优化还是值得的。在下面几节我们将看一些一般的技术，这些技术在特定的地方可以用来改进性能。

13.4.1 操作系统为什么运行缓慢

在讨论优化技术之前，值得指出的是许多操作系统运行缓慢在很大程度上是操作系统自身造成的。例如，古老的操作系统，如MS-DOS和UNIX版本7在几秒钟内就可以启动。现代UNIX系统和Windows Vista尽管运行在快1000倍的硬件上，可能要花费几分钟才能启动。原因是它们要做更多的事情，有用的或无用的。看一个相关的案例。即插即用使得安装一个新的硬件设备相当容易，但是付出的代价是在每次启动时，操作系统都必须要检查所有的硬件以了解是否存在新的设备。这一总线扫描是要花时间的。

一种替代的（并且依作者看来是更好的）方法是完全抛弃即插即用，并且在屏幕上包含一个图标标明“安装新硬件”。当安装一个新的硬件设备时，用户可以点击图标开始总线扫描，而不是在每次启动的时候做这件事情。当然，当今的系统设计人员是完全知道这一选择的。但是他们拒绝这一选择，主要是因为他们假设用户太过愚笨而不能正确地做这件事情（尽管他们使用了更加友好的措辞）。这只是一个例子，但是还存在更多的事例，期望让系统“用户友好”（或者“防傻瓜”，取决于你的看法）却使系统始终对所有用户是缓慢的。

或许系统设计人员为改进性能可以做的最大的一件事情，是对于添加新的功能特性更加具有选择性。要问的问题不是“用户会喜欢吗？”而是“这一功能特性按照代码大小、速度、复杂性和可靠性值得不计代价吗？”只有当优点明显地超过缺点的时候，它才应该被包括。程序员倾向于假设代码大小和程序错误计数为0并且速度为无穷大。经验表明这种观点有些过于乐观。

另一个重要因素是产品的市场销售。到某件产品的第4或第5版上市的时候，真正有用的所有功能特性或许已经全部包括了，并且需要该产品的大多数人已经拥有它了。为了保持销售，许多生产商仍然继续生产新的版本，具有更多的功能特性，正是这样才可以向现有的顾客出售升级版。只是为了添加新的功能特性而添加新的功能特性可能有助于销售，但是很少会有助于性能。

13.4.2 什么应该优化

作为一般的规则，系统的第一版应该尽可能简单明了。惟一的优化应该是那些显而易见要成为不可避免的问题的事情。为文件系统提供块高速缓存就是这样的一个例子。一旦系统引导起来并运行，就应该仔细地测量以了解时间真正花在了什么地方。基于这些数字，应该在最有帮助的地方做出优化。

这里有一个关于优化不但不好反而更坏的真实故事。作者的一名学生编写了MINIX的mkfs程序。该程序在一个新格式化的磁盘上布下一个新的文件系统。这名学生花了大约6个月的时间对其进行优化，包括放入磁盘高速缓存。当他上交该程序时，它不能工作，需要另外几个月进行调试。在计算机的生命周期中，当系统安装时，该程序典型地在硬盘上运行一次。它还对每块做格式化的软盘运行一次。每次运行大约耗时2秒。即使未优化的版本耗时1分钟，花费如此多的时间优化一个很少使用的程序也是相当不值的。

对于性能优化，一条相当适用的口号是：

足够好就够好了。

通过这条口号我们要表达的意思是：性能一旦达到一个合理的水平，榨出最后一点百分比的努力和复杂性或许并不值得。如果调度算法相当公平并且在90%的时间保持CPU忙碌，它就尽到了自己的职责。发明一个改进了5%但是要复杂得多的算法或许是一个坏主意。类似地，如果缺页率足够低到不是瓶颈，克服重重难关以获得优化的性能通常并不值得。避免灾难比获得优化的性能要重要得多，特别是针对一种负载的优化对于另一种负载可能并非优化的情况。