第8章 并发代码设计

本章主要内容

- 线程间划分数据的技术
- 影响并发代码性能的因素
- 性能因素是如何影响数据结构的设计
- 多线程代码中的异常安全
- 可扩展性
- 并行算法的实现

之前章节着重于介绍使用 c++ 11中的新工具来写并发代码。在第6、7章中我们了解到，如何使用这些工具来设计可并发访问的基本数据结构。这就好比一个木匠，其不仅要知道如何做一个合页，一个组合柜，或一个桌子；并发的代码的使用，要比使用/设计基本数据结构频繁的多。要将眼界放宽，就需要构建更大的结构，进行高效的工作。我将使用多线程化的 c++ 标准库算法作为例子，不过这里的原则也适用于对其他应用程序的扩展。

认真思考如何进行并发化设计，对于每个编程项目来说都很重要。不过，写多线程代码的时候，需要考虑的因素比写序列化代码多得多。不仅包括一般性因素，例如：封装，耦合和聚合(这些在很多软件设计书籍中有很详细的介绍)，还要考虑哪些数据需要共享，如何同步访问数据，哪些线程需要等待哪些线程，等等。

本章将会关注这些问题，从高层(但也是基本的)考虑，如何使用线程，哪些代码应该在哪些线程上执行；以及，这将如何影响代码的清晰度，并从底层细节上了解，如何构建共享数据来优化性能。

那么就先来看一下，如何在线程间划分工作。