分析和衡量绩效

定义一个目标：如果你有一个明确的、量化的目标，就更容易知道如何优化以及何时停止优化

测量：一旦我们知道要测量什么以及限制是什么，我们就可以测量应用程序现在的性能。从第一步开始，我们应该很清楚我们是否对平均时间、峰值、负载等感兴趣。在这一步中，我们只关心测量我们设置的目标。根据应用的不同，测量可以是从使用秒表到使用高度复杂的性能分析工具的任何东西。

找到瓶颈：接下来，我们需要找到应用程序的瓶颈--那些太慢而使应用程序变得无用的部分。在这一点上不要相信你的直觉！也许您通过在第2步中的各个点测量代码获得了一些见解-这很好，但您通常需要进一步分析代码，以便找到最重要的热点。

.做一个有根据的猜测：提出一个关于如何提高性能的假设。可以使用查找表吗？我们是否可以缓存数据以获得整体吞吐量？我们可以修改代码，让编译器可以向量化它吗？我们能否通过重用内存来减少临界区中的分配数量？如果您知道这些想法只是有根据的猜测，那么提出这些想法通常并不难。错了没关系你以后会发现它们是否有影响。

优化：让我们实现我们在步骤4中概述的假设。不要花太多时间在这一步上，在你知道它实际上有效果之前，把它做得完美。准备好拒绝这种优化。它可能不会达到预期的效果。

评价：再次测量。执行与步骤2中完全相同的测试并比较结果。我们得到了什么？如果我们没有得到任何东西，拒绝代码并返回到步骤4。如果优化实际上有积极的效果，你需要问自己是否足够好，花更多的时间。值得这么做吗？这是一个一般的性能增益，还是高度特定于特定的案例/平台？它是maintenance吗？我们可以封装它吗？还是它会散布在整个代码库中？如果你不能激励优化，回到第4步，否则继续到最后一步。

.重构：如果您遵循了步骤5中的说明，并且没有花太多时间编写完美的代码，那么现在是时候重构优化以使其更清晰。优化几乎总是需要一些注释来解释为什么我们以不寻常的方式做事。