O'REILLY®

TURING 图灵程序设计丛书

Java性能权威指南

Java Performance: The Definitive Guide

[美] Scott Oaks 著
柳飞 陆明刚 臧秀涛 译

中国工信出版集团 人民邮电出版社 POSTS & TELECOM PRESS

TURING 图灵程序设计丛书

Java 性能权威指南

Java Performance
The Definitive Guide

[美] Scott Oaks 著
柳飞 陆明刚 臧秀涛 译

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Tokyo
O'Reilly Media, Inc.授权人民邮电出版社出版

人民邮电出版社
北 京

图书在版编目（CIP）数据

Java性能权威指南 /（美）奥克斯（Oaks, S.）著；柳飞，陆明刚，臧秀涛译. -- 北京 : 人民邮电出版社，2016.3
（图灵程序设计丛书）
ISBN 978-7-115-41376-5

Ⅰ. ①J… Ⅱ. ①奥… ②柳… ③陆… ④臧… Ⅲ. ①JAVA语言－程序设计 Ⅳ. ①TP312

中国版本图书馆CIP数据核字(2016)第003957号

内 容 提 要

本书对Java 7和Java 8中影响性能的因素展开了全面深入的介绍，讲解传统上影响应用性能的JVM特征，包括即时编译器、垃圾收集、语言特征等。内容包括：用G1垃圾收集器最大化应用的吞吐量；使用Java飞行记录器查看性能细节，而不必借助专业的分析工具；堆内存与原生内存最佳实践；线程与同步的性能，以及数据库性能最佳实践等。

本书适合需要了解JVM调优的Java新手以及想要最优化应用性能的成熟开发人员阅读。

◆ 著　[美] Scott Oaks
译　柳 飞 陆明刚 臧秀涛
责任编辑　岳新欣
执行编辑　张 曼
责任印制　杨林杰

◆ 人民邮电出版社出版发行　北京市丰台区成寿寺路11号
邮编　100164　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市中晟雅豪印务有限公司印刷

◆ 开本：800×1000　1/16
印张：20.5
字数：487千字　2016年3月第1版
印数：1－4 000册　2016年3月河北第1次印刷
著作权合同登记号　图字：01-2014-6317号

定价：79.00元

读者服务热线：(010)51095186转600　印装质量热线：(010)81055316
反盗版热线：(010)81055315
广告经营许可证：京东工商广字第8052号

版权声明

O'Reilly Media, Inc.介绍

O'Reilly Media通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自1978年开始，O'Reilly一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了Make杂志，从而成为DIY革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版、在线服务或者面授课程，每一项O'Reilly的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar博客有口皆碑。”

——*Wired*

“O'Reilly凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——*Business 2.0*

“O'Reilly Conference是聚集关键思想领袖的绝对典范。”

——*CRN*

“一本O'Reilly的书就代表一个有用、有前途、需要学习的主题。”

——*Irish Times*

“Tim是位特立独行的商人，他不光放眼于最长远、最广阔的视野，并且切实地按照Yogi Berra的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去，Tim似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——*Linux Journal*

推荐序

可能很多 Java 程序员都会觉得解决性能问题是一件特别苦恼、特别让人抓狂的事情，因为系统的各个层面的问题都会导致性能问题。JVM 优化是个老生常谈的话题，也是程序员面试容易遇到的高频问题，貌似每个程序员或者面试官都知道那么一点儿，没有什么新意。市面上 JVM 性能方面的书也有几本，但是真正把这些知识整理成书，能够做到紧跟时代步伐的并不多。很多资料都忽略了 JVM 最近几年的发展和进步，这样我们就无法发挥 JVM 那些最激动人心的新特性，也会使自己像一个生活在 21 世纪的原始人。很多原来的优化方法都已失去了存在的意义，但还是被大家不断地讨论。片面、零散、落伍的知识在 JVM 领域大行其道，要命的是其中还有很多是错误的。当然，我们并不能怪那些多年前的作者，希望他们能够预知今天 JVM 的进展。但如果想再找一本能跟得上时代步伐的 JVM 调优的书的话，貌似当下只有这本书。这就是我推荐此书的理由：全面、实用、紧跟时代。本书很多章节都是我非常喜欢的，比如关于 JMC 的。相信很多有多年 JVM 调优经验的人也未必听说过 JMC，但不得不说，每个遇见 JMC 的人都如获至宝。

本书完整地介绍了 JVM 调优需要的方方面面，而不仅仅限于那些诡异参数的简单说明，非常具有实用性和系统性。值得一提的是，几位译者都是在这个领域非常资深的专家，翻译水平上乘。我觉得每个对 JVM 感兴趣的程序员都应该看看这本书。

程显峰

Kage 技术咨询合伙人

这是一部关于 Java 性能调优的卓越作品。该书涉及性能测试、性能分析、性能调优的原理、方法、工具等诸多方面，书中最新的 JVM 和体系结构的相关知识可以帮助我们更好地理解 Java，同时该书又包含了许多非常工程性的经验，比如多线程、数据库、序列化以及 Java API 等，这些经验不仅对 Java 工程师很有帮助，也为其他开发人员及性能调优人员提供了问题解决思路和方法上的启迪。借助这本书，我们可以从 Java 纷繁复杂的性能调优参数中解脱出来，看到背后的动机和缘由，从而获得对性能的不一样的理解。

邹飞

Google 资深软件工程师，技术经理

前言

起初 O'Reilly 公司让我写一本关于 Java 性能调优的书时，我还不确定是否值得写。我在想，难道 Java 性能调优我们做得还不够吗？事实上，虽然我日常的基本工作是 Java（和其他）应用程序的性能调优，但我宁愿将大多数时间都花在提高应用程序的算法效率以及处理外部系统的性能瓶颈上，而不是直接进行 Java 自身性能的调优。

但转念一想，我不禁哑然失笑（像往常一样）。的确，我的大量时间都花在了端到端的系统性能调优上，有时会发现那些原本可以用 $O(\log N)$ 却用了 $O(n^2)$ 算法的代码。不过这也说明，我每天考虑的都是 GC 调优、JVM 编译器的性能调优，或者是如何使 Java EE API 的性能发挥到极致。

说这些并不是想要抹杀过去 15 年里 Java 和 JVM 在性能上取得的巨大进步。1990 年代晚期，我在 Sun 公司担当 Java 布道师，当时仅有的真正意义上的“基准测试”工具来自 Pendragon 软件的 CaffeineMark 2.0。由于种种原因，该基准测试工具设计上的不足很快就限制了它的价值；然而在那个年代，我们总喜欢告诉所有人，依据这个基准测试，Java 1.1.8 的性能比 Java 1.0 快八倍。这并非耸人听闻——Java 1.1.8 已经有了真正的即时编译器，而 Java 1.0 差不多完全是解释型的。

之后，Java 标准委员会开始制定更严谨的基准测试，Java 的性能测试开始围绕这些基准测试展开。最终，JVM 的所有领域——垃圾收集、编译器和 API 都获得了长足的进步。这个过程一直延续到今天，而关于性能的一个重要事实是，调优正变得越来越艰难。引入即时编译器后所获得的八倍性能提升，只是一个简单的工程问题，即使编译器持续改进，我们也无法再次看到如此巨大的改进了。并行化垃圾收集也曾极大地提升过性能，但最近的变化则是渐进式的。

这是典型的应用发展过程（JVM 本身也是另外一个应用）：在项目初期，很容易找到架构上的改进点（或代码缺陷），一旦找到就能极大改善性能。而在成熟应用中，要找到这样的性能改进点则很罕见。

起初我觉得，从很大程度上说，Java 性能调优都已经工程化了，但有几件事情让我相信我错了。首先，我每天在特定环境下运行 JVM 时，都会遇到许多这样或那样的问题。新工程师源源不断地进入 Java 领域。在特定的领域，JVM 的行为仍然相当复杂，因此有份描

述它如何运作的指南很有必要。其次，现在的计算环境发生了变化，已经影响到了工程师们所面临的性能问题。

在过去几年中，性能关注点有了分歧。一方面，有大量内存堆可运行 JVM 的大机器现在已经很普遍了。为了应对这样的变化，JVM 也有了新的垃圾收集器（G1）——这项新技术相比传统的收集器更需要手工调优。同时，云计算又提升了单 CPU 的小机器的重要性：你可以从 Oracle、Amazon 或其他公司以非常便宜的价格租用单 CPU 机器，运行小的应用服务器。（你获得的并不是真的单 CPU 机器，而是一台巨大机器上的一个虚拟 OS 镜像，但虚拟 OS 被限制为使用单个 CPU。从 Java 角度看，它和单 CPU 的机器相同。）在这些环境中，正确管理小量内存变得非常重要。

Java 平台也在持续演变。Java 的每个新版本都会提供新的语言特性和新的 API，这些特性和 API 并不总是为了提高应用性能，也是为了改善开发人员的生产率。语言特性运用得好，应用的运行就会变得轻快，反之则缓慢笨重。另外，平台的演化也带来了一些重要的性能课题：毫无疑问，程序间用 JSON 交换信息要比用高度优化的私有协议更容易。节约开发人员的时间就是巨大的收益——但真正的目的是确保生产率提升的同时，性能也能提升（至少是两者之间取得平衡）。

读者对象

本书适合那些渴望深入了解 JVM 和 Java API 性能各个方面的性能调优工程师和开发者。

假如你想快速修复性能问题，比如网站周一早上要上线，而现在已经是周日深夜了，那么本书可能不适合你。

如果你是性能分析的新手，正要开始进行 Java 的性能分析，那么本书会对你有所帮助。我的目的主要是为新工程师提供足够多的信息和上下文，以便使他们明白如何将基本的性能调优原则运用到 Java 应用中去。然而，系统分析的范畴非常广阔，已经有大量的优秀资源（那些原则当然也适用于 Java），从这个意义上来说，希望本书也能成为它们的有益补充。

不过从根本上说，想让 Java 运行得飞快，就得深入理解 JVM（以及 Java API）的实际工作原理。有数以百计的 Java 性能调优参数，而 JVM 调优并不像瞎猫碰死耗子那样，调一下再看看是否奏效。与之相反，我的目的是提供更为详尽的 JVM 和 API 工作原理，以期在你理解它们如何工作的原理之后，能理解应用的某些行为为何糟糕。理解了这些之后，排除那些性能低下、令人不快的行为就会变成简单（至少是比以前更简单）的任务。

Java 性能调优工作还有一个有趣的方面，就是开发人员的背景和性能调优或 QA 组人员的背景常常有很大差别。我认识一些开发人员，他们可以记住成千上万个令人费解的很少使用的 Java API 方法签名，但他们对 `-Xmn` 的含义却没有什么概念。我也认识一些测试工程师，他们可以通过设置垃圾收集器的各种标志来榨取最后一滴性能，但他们却很少有人能用 Java 写出像样的“Hello, World”。

Java 性能调优覆盖这两个领域：编译器和垃圾收集器等的调优参数，以及 API 的最佳实践。所以，我假定你对如何编写 Java 程序有很好的理解。即便你主要的兴趣不是在 Java 编程，我仍然会花一点时间讨论编程，包括例子中包含大量数据的示例程序。

然而，如果你的主要兴趣是 JVM 自身的性能调优——意思是不用更改任何代码而改变 JVM 的行为，那么本书的大量章节都对你有用。可以随意跳过代码部分，而关注你所感兴趣的领域。也许你会顺便为 Java 应用如何影响 JVM 性能提出一些真知灼见，并向开发人员提出更改建议，以便让你的性能调优测试工作更加如鱼得水。

排版约定

本书使用的排版约定如下。

- 楷体
  表示新术语。
- 等宽字体（`constant width`）
  表示程序片段，以及正文中出现的变量、函数名、数据库、数据类型、环境变量、语句和关键字等。
- 等宽粗体（**`constant width bold`**）
  表示应该由用户输入的命令或其他文本。
- 等宽斜体（*`constant width italic`*）
  表示应该由用户输入的值或根据上下文确定的值替换的文本。

这个图标代表提示或建议。

这个图标代表重要说明。

这个图标代表警告或提醒。

使用代码示例

可以在这里下载本书随附的资料（代码示例、练习题等）：https://github.com/ScottOaks/JavaPerformanceTuning。

让本书助你一臂之力。也许你需要在自己的程序或文档中用到本书中的代码。除非大段大段地使用，否则不必与我们联系取得授权。例如，无需请求许可，就可以用本书中的几段代码写成一个程序。但是销售或者发布 O'Reilly 图书中代码的光盘则必须事先获得授权。引用书中的代码来回答问题也无需授权。将大段的示例代码整合到你自己的产品文档中则必须经过许可。

使用我们的代码时，希望你能标明它的出处，但不强求。出处一般包括书名、作者、出版商和 ISBN，例如：*Java Performance: The Definitive Guide* by Scott Oaks (O'Reilly, 2014). Copyright by Scott Oaks, 978-1-449-35845-7。

如果还有关于使用代码的未尽事宜，可以随时与我们联系：permissions@oreilly.com。

Safari® Books Online

Safari Books Online（http://www.safaribooksonline.com）是应需而变的数字图书馆。它同时以图书和视频的形式出版世界顶级技术和商务作家的专业作品。

Safari Books Online 是技术专家、软件开发人员、Web 设计师、商务人士和创意人士开展调研、解决问题、学习和认证培训的第一手资料。

对于组织团体、政府机构和个人，Safari Books Online 提供各种产品组合和灵活的定价策略。用户可通过一个功能完备的数据库检索系统访问 O'Reilly Media、Prentice Hall Professional、Addison-Wesley Professional、Microsoft Press、Sams、Que、Peachpit Press、Focal Press、Cisco Press、John Wiley & Sons、Syngress、Morgan Kaufmann、IBM Redbooks、Packt、Adobe Press、FT Press、Apress、Manning、New Riders、McGraw-Hill、Jones & Bartlett、Course Technology 以及其他几十家出版社的上千种图书、培训视频和正式出版之前的书稿。要了解 Safari Books Online 的更多信息，我们网上见。

联系我们

请把对本书的评价和问题发给出版社。

美国：

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472

中国：

北京市西城区西直门南大街 2 号成铭大厦 C 座 807 室（100035）
奥莱利技术咨询（北京）有限公司

O'Reilly 的每一本书都有专属网页，你可以在那儿找到本书的相关信息，包括勘误表、示例代码以及其他信息。本书的网站地址是：

http://oreil.ly/java-performance-tdg。

对于本书的评论和技术性问题，请发送电子邮件到：bookquestions@oreilly.com

要了解更多 O'Reilly 图书、培训课程、会议和新闻的信息，请访问以下网站：http://www.oreilly.com

我们在 Facebook 的地址如下：http://facebook.com/oreilly

请关注我们的 Twitter 动态：http://twitter.com/oreillymedia

我们的 YouTube 视频地址如下：http://www.youtube.com/oreillymedia

致谢

感谢在我写这本书时所有帮助过我的人。从各方面来看，这本书汇集了我过去 15 年来在 Sun 公司和 Oracle 公司的 Java 性能调优小组中的所学所知。如果将为这本书提出真知灼见的人都列出来的话，会是一份长长的列表。感谢在那段时间与我一同工作的工程师们，特别是在那些年里耐心回答我铺天盖地的问题的人们。

我要特别感谢 Stanley Guan、Azeem Jiva、Kim LiChong、Deep Singh、Martijn Verburg 和 Edward Yue Shung Wong，感谢他们拨冗审阅本书的初稿，感谢他们提供的宝贵意见。虽然本书因为他们的建议而完善了许多，但我相信错误仍然在所难免。

O'Reilly 的工作人员总是那么乐于助人，感谢我的编辑 Meg Blanchette，感谢你在整个写作过程中的鼓励。最后，衷心感谢我的丈夫 James 给予我在为书而抓狂的漫漫长夜里的悉心陪伴，以及周末晚餐。

目录