涵盖并行程序设计基础、思路、方法和实战
内容丰富，实例典型，实用性强

Broadview®
www.broadview.com.cn

实战Java高并发程序设计

（第2版）

葛一鸣 著

中国工信出版集团
電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

非外借

实战 Java 高并发程序设计

（第 2 版）

葛一鸣　著

電子工業出版社
Publishing House of Electronics Industry

内 容 简 介

在单核 CPU 时代，单任务在一个时间点只能执行单一程序，随着多核 CPU 的发展，并行程序开发变得尤为重要。

本书主要介绍基于 Java 的并行程序设计基础、思路、方法和实战。第一，立足于并发程序基础，详细介绍 Java 进行并行程序设计的基本方法。第二，进一步详细介绍了 JDK 对并行程序的强大支持，帮助读者快速、稳健地进行并行程序开发。第三，详细讨论了“锁”的优化和提高并行程序性能级别的方法和思路。第四，介绍了并行的基本设计模式，以及 Java 8/9/10 对并行程序的支持和改进。第五，介绍了高并发框架 Akka 的使用方法。第六，详细介绍了并行程序的调试方法。第七，分析 Jetty 代码并给出一些其在高并发优化方面的例子。

本书内容丰富，实例典型，实用性强，适合有一定 Java 基础的技术开发人员阅读。

图书在版编目（CIP）数据

实战 Java 高并发程序设计 / 葛一鸣著. —2 版. —北京：电子工业出版社，2018.9
ISBN 978-7-121-35003-0

Ⅰ. ①实… Ⅱ. ①葛… Ⅲ. ①JAVA 语言—程序设计 Ⅳ. ①TP312.8

中国版本图书馆 CIP 数据核字（2018）第 207736 号

策划编辑：董　英
责任编辑：汪达文
印　　刷：三河市良远印务有限公司
装　　订：三河市良远印务有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　　邮编：100036
开　　本：787×980　1/16　　印张：26　　字数：525 千字
版　　次：2015 年 10 月第 1 版
　　2018 年 9 月第 2 版
印　　次：2018 年 9 月第 1 次印刷
印　　数：3000 册　　定价：89.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888，88258888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

本书咨询联系方式：010-51260888-819，faq@phei.com.cn。

前　言

关于 Java 与并行

由于单核 CPU 的主频逐步逼近极限，多核 CPU 架构成了一种必然的技术趋势，因此多线程并行程序便显得越来越重要。并行计算的一个重要应用场景就是服务端编程。目前服务端 CPU 的核心数已经轻松超越 10 个，而 Java 显然已经成为当下最流行的服务端编程语言，且已经更新到 JDK 10，因此熟悉和了解基于 Java 的并行程序开发有着重要的实用价值。

本书的体系结构

本书立足于实际开发，又不缺乏理论介绍，力求通俗易懂、循序渐进。本书共分为 9 章。

第 1 章主要介绍了并行计算中相关的一些基本概念，树立读者对并行计算的基本认识，介绍了两个重要的并行性能评估定律，以及 Java 内存模型 JMM。

第 2 章介绍了 Java 并行程序开发的基础，包括 Java 中 Thread 的基本使用方法等，也详细介绍了并行程序容易引发的一些错误，以及容易出现的误用。

第 3 章介绍了 JDK 内部对并行程序开发的支持，主要介绍 juc（java.util.concurrent）中一些工具的使用方法、各自的特点及它们的内部实现原理。

第 4 章介绍了在开发过程中可以进行的对锁的优化，也进一步简要描述了 Java 虚拟机层面对并行程序的优化支持。此外，还花费一定的篇幅介绍了无锁的计算。

第 5 章介绍了并行程序设计中常见的一些设计模式，以及一些典型的并行算法和使用方法，其中包括重要的 Java NIO 和 AIO 的介绍。

第 6 章介绍了 Java 8/9/10 为并行计算做的改进，包括并行流、CompletableFuture、StampedLock、LongAdder，以及发布和订阅模式等。

第 7 章主要介绍了高并发框架 Akka 的基本使用方法，并使用 Akka 框架实现了一个简单的粒子群算法，模拟超高并发的场景。

第 8 章介绍了使用 Eclipse 进行多线程调试的方法，并演示了通过 Eclipse 进行多线程调试重现 ArrayList 的线程不安全问题。

第 9 章介绍了 Jetty，并分析了 Jetty 的一些关键代码，主要展示它在高并发优化中所做的一些努力，也为读者学会并深入理解高并发带来一些提示和思考。

本书特色

本书的主要特色如下。

1. **结构清晰**。本书一共 9 章，总体上循序渐进，逐步提升。每一章都有鲜明的侧重点，有利于读者快速抓住重点。
2. **理论结合实战**。本书注重实战，书中重要的知识点都安排了代码实例，帮助读者理解。同时也不忘对系统的内部实现原理进行深度剖析。
3. **通俗易懂**。本书尽量避免采用过于理论化的描述方式，简单的白话文风格贯穿全书，配图基本上为手工绘制，降低了理解难度，并尽量做到读者在阅读过程中少盲点、无盲点。

适合阅读人群

虽然本书力求通俗，但是要通读本书并取得良好的学习效果，要求读者具备基本的 Java

知识或者一定的编程经验。因此，本书适合以下读者。

- 拥有一定开发经验的 Java 平台开发人员（Java、Scala、JRuby 等）。
- 软件设计师、架构师。
- 系统调优人员。
- 有一定的 Java 编程基础并希望进一步加深对并行程序的理解的研发人员。

本书的约定和更新

本书在叙述过程中，有如下约定。

- 本书所述的 JDK 1.5、JDK 1.6、JDK 1.7、JDK 1.8、JDK 1.9、JDK 1.10 分别等同于 JDK 5、JDK 6、JDK 7、JDK 8、JDK 9、JDK 10。
- 如无特殊说明，本书的程序、示例均在 JDK 1.8 以上环境中运行。

相较前一版，本书的主要更新如下。

1. 第 3 章增加的内容如下。

- 使用 JMH 进行性能测试。
- CopyOnWriteArrayList ConcurrentLinkedQueue 性能测试。
- 使用 Guava 的 RateLimiter 限流。
- Guava 中对线程池的扩展。
- 介绍 JDK 中 ArrayBlockingQueue 的算法。

2. 第 5 章增加的内容如下。

- Guava 对 Future 模式的支持。

3. 第 6 章增加的内容如下。

- 支持 timeout 的 CompletableFuture。
- ConcurrentHashMap 在新版本 JDK 中的增强。
- 发布和订阅模式。

4. 增加第 9 章，主要介绍 Jetty 多线程优化。从细节入手分析 Jetty 在多线程并发方面做出的努力和优化，对实践应用具有极强的参考价值。

联系作者

本书的写作过程远比我想象得艰辛，为了让全书能够更清楚、更准确地表达和论述，我经历了很多个不眠之夜，即使现在回想起来，我也忍不住会打个寒战。由于写作水平的限制，书中难免会有不妥之处，望读者谅解。

为此，如果读者有任何疑问或者建议，非常欢迎大家加入 QQ 群 254693571，一起探讨学习中的困难、分享学习经验，我期待与大家一起交流、共同进步。同时，大家也可以关注我的博客 http://www.uucode.net/。欢迎大家去博文视点社区下载本书推荐的参考文献。

感谢

本书能够面世，是因为得到了众人的支持。首先，要感谢我的妻子，她始终不辞辛劳、毫无怨言地对我照顾有加，才让我得以腾出大量时间安心工作。其次，要感谢所有编辑为我一次又一次地审稿改错，批评指正，促进本书逐步完善。最后，感谢我的母亲三十年如一日对我的体贴和关心。

葛一鸣

读者服务

轻松注册成为博文视点社区用户（www.broadview.com.cn），您即可享受以下服务：

- **下载资源**：本书如提供示例代码及资源文件，均可在 下载资源 处下载。
- **提交勘误**：您对书中内容的修改意见可在 提交勘误 处提交，若被采纳，将获赠博文视点社区积分（在您购买电子书时，积分可用来抵扣相应金额）。
- **与作者交流**：在页面下方 读者评论 处留下您的疑问或观点，与作者和其他读者一同学习交流。

页面入口：http://www.broadview.com.cn/35003

目　录

1

第 1 章
走入并行世界

当你打开本书，也许你正试图将你的应用改造成并行模式运行，也许你只是单纯地对并行程序感兴趣。无论出于何种原因，你正对并行计算充满好奇、疑问和求知欲。如果是这样，那就对了，带着你的好奇和疑问，让我们一起遨游并行程序的世界，深入了解它们究竟是如何工作的吧！

不过首先，我想要公布一条令人沮丧的消息。就在大伙儿都认为并行计算必然成为未来的大趋势时，2014 年底，在 Avoiding ping pong 论坛上，伟大的 Linus Torvalds 提出了一个截然不同的观点，他说：“忘掉那该死的并行吧！”（原文：Give it up. The whole "parallel computing is the future" is a bunch of crock.）

1.1 何去何从的并行计算

我们到底该如何选择呢？本节的目的就是拨云见日。

1.1.1 忘掉那该死的并行

Linus Torvalds 是一个传奇式的人物（图 1.1），是他给出了 Linux 的原型，并一直致力于推广和发展 Linux 系统。1991 年，他首先在网络上发布了 Linux 源码，从此 Linux 迅速崛起壮大，成为目前使用最广泛的操作系统之一。

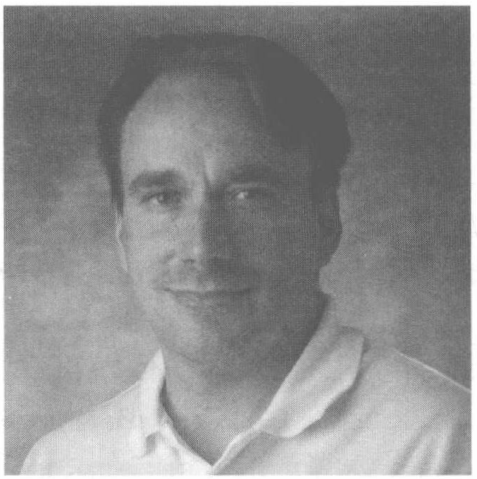

图 1.1 传奇的 Linus Torvalds

自 2002 年起，Linus 就决定使用 BitKeeper 作为 Linux 内核开发的版本控制工具，以此来维护 Linux 的内核源码。BitKeeper 是一套分布式版本控制软件，它是一套商用系统，由 BitMover 公司开发。2005 年，BitKeeper 宣称发现 Linux 内核开发人员使用逆向工程来试图解析 BitKeeper 内部协议。因此，决定向 Linus 收回 BitKeeper 授权。虽然 Linux 核心团队与 BitMover 公司进行了协商，但是仍无法解决他们之间的分歧。因此，Linus 决定自行研发版本控制工具来代替 BitKeeper。于是，Git 诞生了。

如果你正在使用 Git，那么我相信你一定会为 Git 的魅力所征服；如果你还没有了解过 Git，那么我强烈建议你关注一下这款优秀的产品。

而正是这位传奇人物，给目前红红火火的并行计算泼了一大盆冷水。那么，并行计算究竟应该何去何从呢？

在 Linus 的发言中这么说道：

Where the hell do you envision that those magical parallel algorithms would be used?

The only place where parallelism matters is in graphics or on the server side, where we already largely have it. Pushing it anywhere else is just pointless.