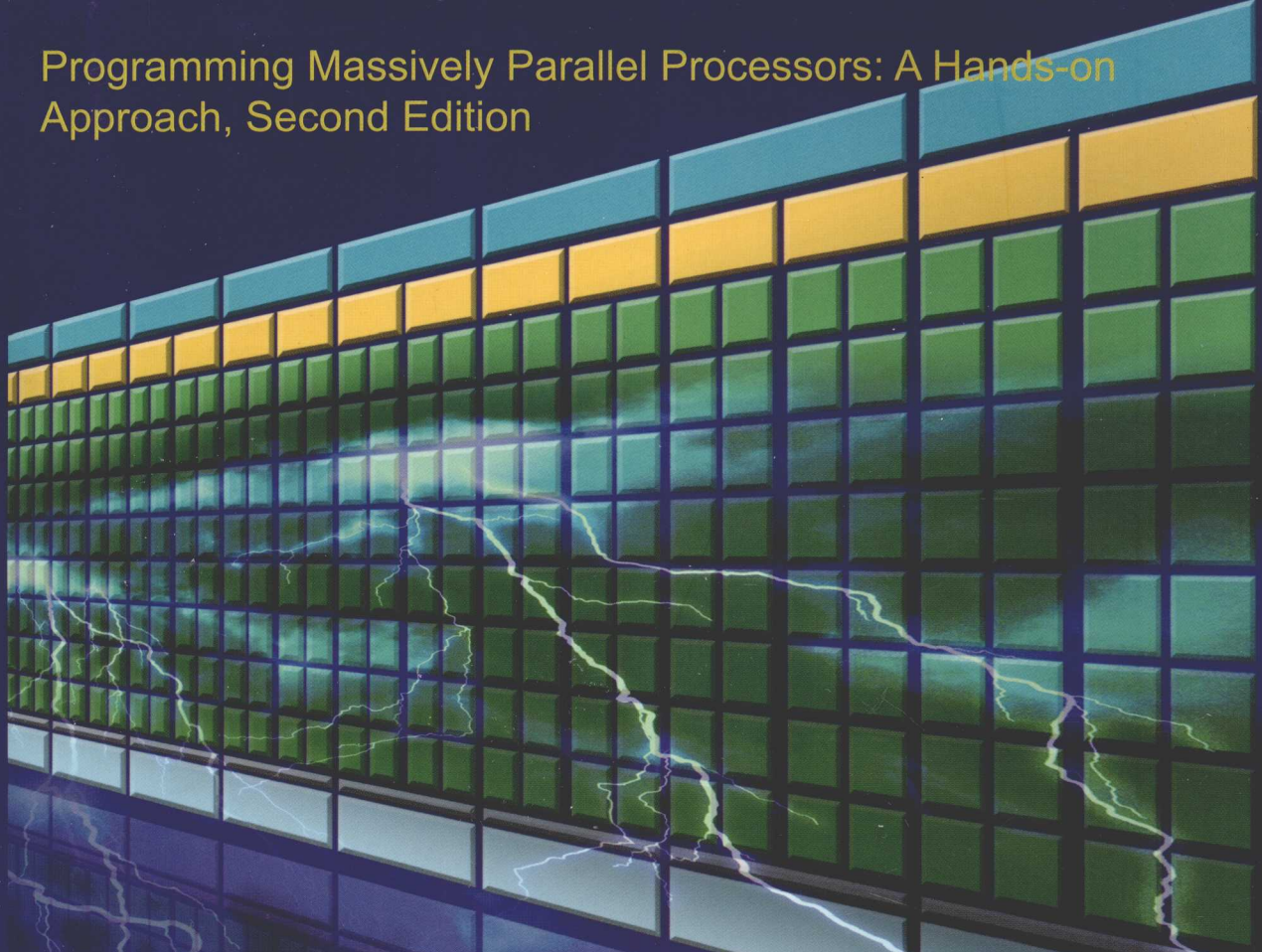
Programming Massively Parallel Processors: A Hands-on Approach, Second Edition



大规模并行处理器编程实战(第2版)

[美] David B. Kirk 著
Wen-mei W. Hwu
赵开勇 汪朝辉 程亦超 译

清华大学出版社

大规模并行处理器编程实战

(第2版)

[美] David B. Kirk
Wen-mei W. Hwu 著

赵开勇 汪朝辉 程亦超 译

清华大学出版社

图书在版编目(CIP)数据

大规模并行处理器编程实战(第 2 版)/(美)柯克(Kirk, D.B.)，(美)胡(Hwu, W.W.) 著；赵开勇，汪朝辉，程亦超 译. —北京：清华大学出版社，2013

书名原文：Programming Massively Parallel Processors: A Hands-on Approach, Second Edition
ISBN 978-7-302-34272-4

Ⅰ. ①大… Ⅱ. ①柯… ②胡… ③赵… ④汪… ⑤程… Ⅲ. ①并行程序—程序设计 Ⅳ. ①TP311.11

中国版本图书馆 CIP 数据核字(2013)第 249417 号

责任编辑：王　军　刘伟琴
装帧设计：牛静敏
责任校对：曹　阳
责任印制：宋　林

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦 A 座　　**邮　　编**：100084
社 总 机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 刷 者：清华大学印刷厂
装 订 者：三河市新茂装订有限公司
经　　销：全国新华书店
开　　本：185mm×230mm　　**印　　张**：27　　**字　　数**：554 千字
版　　次：2013 年 11 月第 1 版　　**印　　次**：2013 年 11 月第1 次印刷
印　　数：1～4000
定　　价：59.80 元

产品编号：049320-01

译 者 序

本书是一本由浅入深地介绍异构高性能编程的优秀教材。

2007 年，NVIDIA 公司推出了 CUDA(Compute Unified Device Architecture，统一计算设备架构)这一编程模型的 1.0 版本。作为译者，我有幸于 2007 年底就开始接触 CUDA。这几年间，异构高性能计算不断发展，支持异构计算的架构不断更新和涌现，越来越多的人开始关注异构高性能计算。在大数据时代，异构高性能计算是处理大数据的一把利器。传统并行计算算法的研究在 20 世纪 60 年代和 70 年代非常活跃，现在的很多并行算法实际上在那个时代都已出现。技术的不断发展，芯片的高度集成，使得以前需要用大规模计算集群完成的工作，现在可在一个小芯片上完成，从而使高性能并行计算得以普及。高性能计算小芯片(包括多核心 CPU、GPU、众核核心协处理器等)使万亿次计算走向了单个节点。除了高性能计算集群外，个人高性能计算机和移动平台都会用到高性能编程技术。高性能编程技术在个人计算机和移动平台将成为未来的一个发展方向。针对移动平台的高性能计算，将来也会是一个重要领域。且计算和传输本就是一体的，为在降低传输带宽的同时达到高质量的传输，越来越多的新协议需要大量的计算。单节点的高性能计算能力，可以很好地解决计算部分的问题。

硬件的变革要求我们紧跟国际步伐，在这几年的 TOP 500 排名中，中国多次排名前列。但是，对于高性能计算而言，光有硬件架构往往是不够的。近几年硬件的变化基本是 18 个月左右革新一次。如果没有优秀软件的支持，再好的硬件也很快就会过时，投入的大量人力物力就会在几年之内迅速贬值。对于新的硬件架构必须开发新的软件，全世界基本是在同一条起跑线上。我们希望更多的开发者加入到异构高性能开发中来。高性能异构计算也是解决大数据的一个强有力手段。

本书首先讲述硬件的体系结构，以 GPU 为例讲解异构计算芯片的架构，然后结合实际开发示例，讲解软件系统结构，以及异构高性能计算软件开发的思想和实践。异构高性能计算芯片的发展近几年非常活跃，作为译者，我们希望，通过翻译本书，能架起知识传播的桥梁，将国外的最新技术传播给国内的读者。

感谢清华大学出版社对我们的信任，把这本重要的书交给我们翻译。感谢参与校对和审稿的同仁们：武汉某研究所的周学谦，香港中文大学的王若薇，武汉大学的方留杨，香港科技大学的李佳俐，北京师范大学的赵国兴，中国科技大学近代力学系的冉伟，香港浸会大学异构计算实验室的梅辛欣，中科院超算中心的黄潘育，华大基因的张帆。没有你们

的支持与帮助，本书将无法顺利付梓。在此，还要由衷地感谢我的导师褚晓文老师，感谢您对我的支持与鼓励。

异构高性能计算不断发展，翻译本书旨在为传播知识做一点绵薄之力。书中所涉及的领域比较广泛，翻译中如有不当之处，敬请广大读者批评指正。

再次向参与、支持和关切本书出版的同仁表示诚挚的谢意！

赵开勇

2013 年 7 月 4 日

于香港浸会大学计算机系异构计算实验室

致　谢

有很多人为我们出版第 2 版提供了帮助。首先需要感谢本书新章节的作者。Yuan Lin 和 Vinod Grover 撰写了 OpenACC 这一章的初稿。Nathan Bell 和 Jared Hoberock 在 Chris Rodrigues 的基础上撰写了 Thrust 这章的初稿。Greg Ruetsch 和 Massimiliano Fatica 撰写了 CUDA FORTRAN 这章的初稿。David Callahan 撰写了 C++ AMP 这一章。Isaac Gelado 撰写了 MPI-CUDA 这章的初稿。Brent Oster 为 Kepler 章节提供了原始资料和代码实例。如果没有这些人的帮助，我们就不能深入地将这些内容提供给读者。

非常感谢 Izzat E1 Hajj，他不厌其烦地帮我们验证代码示例并改善插图和练习题的质量。

也要特别感谢 CUDA 之父 Ian Buck 和 Tesla GPU 计算架构的首席架构师 John Nickolls。他们的团队为这个课程提供了优秀的基础设施。在我们撰写第 2 版本的时候，John 与世长辞。我们非常怀念他。Nadeem Mohammad 组织 NVIDIA 修订和提供了附录 B。Bill Bean、Simon Green、Mark Harris、Nadeem Mohammad、Brent Oster、Peter Shirley、Eric Young 和 Cyril Zeller 对手稿进行了修订和校对。Calisa Cole 帮助设计了封面。Nadeem 的无私分享为这本书的完成起到了关键性的作用。

尤其要感谢 Jensen Huang 提供大量人力物力开设这门课程，为本书的出版奠定了基础。Tony Tamasi 的团队为本书提供了大量的评论和修订建议。Jensen 还花费了大量时间阅读早期的草稿，为我们提供了很有价值的反馈。David Luebke 为这个课程提供了 GPU 计算资源。Jonah Alben 提供了颇有价值的见解。Michael Shebanow 和 Michael Garland 给予了友情讲座和教学材料。

伊利诺伊的 John Stone 和 Sam Stone 为案例研究和 OpenCL 章节提供了大量的基础材料。John Stratton 和 Chris Rodrigues 为编程思维这一章提供了素材。I-Jui“Ray”Sung、John Stratton、Xiao-Long Wu 和 Nady Obeid 在他们的研究之外自愿担当助教，提供了实验素材并帮助改进课程材料。Jeremy Enos 孜孜不倦地工作，使学生有一个稳定的、用户友好的 GPU 计算集群用于实验和项目研究。

感谢 Dick Blahut 让我们在伊利诺伊大学有挑战性地开展了这个课程。他不断提醒我们需要写这一本书来帮助我们不断前行。Beth Katsinas 安排了一次与 Dick Blahut 和 NVIDIA 的副总 Dan Vivoli 的会议。通过这次会议，向 David 介绍了 Blahut，也促使 David 和 Wen-mei 在伊利诺伊大学开设这门课程。

也要感谢伊利诺伊大学的 Thom Dunning 和密歇根大学的 Sharon Glotzer，作为计算机科学和工程虚拟大学的副执行长，他们盛情举办了暑期课程。Trish Barker、Scott Lathrop、Umesh Thakkar、Tom Scavo、Andrew Schuh 和 Beth McKown 都参与组织了暑期课程。Robert Brunner、Klaus Schulten、Pratap Vanka、Brad Sutton、John Stone、Keith Thulborn、Michael Garland、Vlad Kindratenko、Naga Govindaraju、Yan Xu、Arron Shinn 和 Justin Haldar 为暑期课程提供了教材和课程计划。

Nicolas Pinto 在 MIT 的课程中测试了早期版本的前几章，然后提供了非常有价值的反馈和修改建议。Steve Lumetta 和 Sanjay Patel 讲授了不同版本的课程，为我们提供了有价值的建议。John Owens 非常友善地为我们提供了他的幻灯片。Tor Aamodt、Dan Connors、Tom Conte、Michael Giles、Nacho Navarro 和全世界许许多多的教员以及他们的学生为我们提供了非常有价值的反馈信息。

尤其要感谢我们的同事 Kurt Akeley、Al Aho、Arvind、Dick Blahut、Randy Bryant、Bob Colwell、Ed Davidson、Mike Flynn、John Hennessy、Pat Hanrahan、Nick Holonyak、Dick Karp、Kurt Keutzer、Dave Liu、Dave Kuck、Yale Patt、David Patterson、Bob Rao、Burton Smith、Jim Smith 和 Mateo Valero。他们抽出时间来与我们一起分享积累多年的知识。

感谢所有为本书和相关课程做出贡献的人们，他们的慷慨和热情令我们感动。

David B. Kirk 和 Wen-mei W.Hwu

前　言

我们很荣幸向大家介绍这本《大规模并行处理器编程实战(第2版)》。大众市场的计算系统，结合多核计算机处理单元(CPU)和多线程GPU计算，带来了万亿次计算能力的笔记本电脑和千万亿次的计算机集群。如此强大的计算能力，为我们在科研、工程、卫生和商业等领域使用计算机带来新的曙光。许多领域通过计算机系统将实验的规模、准确性、可控性和客观性等方面提升到前所未有的水平，将会在各自的领域实现突破。本书为实现这个愿景提供了一个关键因素：教授数以百万计的研究生和本科生学习并行编程，使他们拥有并行计算思维和并行编程技巧，从而应对无处不在的计算需求。

自从2010年第1版出版以来，我们收到了读者和老师的很多建议。许多人提出他们重视的一些现有功能。另一些人给我们提出了如何丰富内容的建议，从而使本书更有价值。此外，异构并行计算的硬件和软件已经越来越先进。在硬件领域，自从此书第1版出版以后，至少两款通用图像显示卡(GPU)计算架构——Fermi和Kepler已发布。在软件领域，CUDA 4.0和CUDA 5.0也可以让程序员使用Fermi和Kepler的新特性。因此，我们增加了8章新的内容并几乎重写了5章已有的内容。

总的来说，我们在保留第1版很多有用内容的基础上，增加了3部分主要的内容。第1个增加的部分是更系统地讲解了并行编程。具体内容包括：(1)新增了第8、9和10章来介绍常用的基本并行算法模式；(2)在第3、4、5和6章中增加了更多的背景材料；(3)在第7章中增加了数值稳定性的处理。这些增加的内容皆在消除“学生已经熟悉基本并行编程概念”的假设。我们还通过更多的例子来帮助读者理解。

第2个增加的部分是在异构并行计算系统中采用MPI-CUDA模型的并行编程实践。这一部分内容是大量读者希望增加的部分。鉴于GPU的成本效益明显以及单位瓦特的高吞吐量，现在许多高性能计算系统在每个节点中都使用了GPU。新增加的第19章在概念上解释了这些系统编程接口背后的框架。

第3个增加的部分是介绍了新的并行编程接口和工具，这些都可以用来加快数据并行编程。新增加的第15、16、17和18章介绍了OpenACC、Thrust、CUDA FORTRAN和C++ AMP。与介绍这些工具的用户使用手册不同，我们更注重于从概念上理解使用上面这些工具解决的编程问题。

虽然我们做了这些改进，但也保留了第1版的特性，这似乎有助于它的受欢迎程度。

首先，我们尽可能保证本书简洁。虽然我们非常想增加更多有用的材料，但还是希望尽量减少页数。第二，我们尽量直观地解释问题。虽然一些概念是非常形式化的，特别是介绍基本并行算法时，但我们尽量使讲解直观而实用。

读者对象

本书的读者对象是研究生和本科生，他们来自各种科学学科和工程学科，他们的学科中需要用到计算思维能力和并行编程技巧，通过用已经普及的万亿次运算硬件能使他们在自己的领域中有所突破。我们假设读者至少已经具备了基本的 C 语言编程经验，因此不管是在计算机科学领域内还是领域外，他们都是比较高级的程序员。我们专门面向计算领域内的科学家，如机械工程、土木工程、电气工程、生物工程、物理、化学、天文学和地理学，这些科学家将通过计算在他们各自的领域中做进一步的研究。正因为这样，这些科学家既是他们领域内的专家，也是高级程序员。本书采用基本的 C 语言编程技巧，教大家用 C 语言进行并行编程。我们使用 CUDA C 并行编程环境(NVIDIA 公司开发的 GPU 上支持它，并可以在 CPU 上进行模拟)。大约有 3.75 亿左右的处理器被用户和专业人士在使用，其中超过 120 000 的程序员在积极地使用 CUDA。读者作为学习经验的一部分而开发的应用程序将会被一个庞大的用户社区使用。

如何使用本书

我们愿意提供在本书教学过程中的一些经验。从 2006 年开始，我们讲了多个类型的课程：采用为期一学期的模式和为期一周的密集模式。原来的 ECE498AL 课程在伊利诺伊大学香槟分校已成为一个永久的课程，称为 ECE408 或 CS483。当我们开设 ECE498AL 这门课的时候，开始了一些早期章节的撰写。前面 4 章在 2009 年的春天由 Nicolas Pinto 在 MIT 课程中进行了测试。从那时起，我们已经在众多的学校(包括 VSCSE 和 PUMPS 暑期学校)使用这本书。

分 3 个阶段的学习方法

为了能同时兼顾 ECE498AL 的课程和编程作业，我们把学习过程分成 3 个阶段：

第 1 个阶段——基于第 3 章的一次课致力于教会学生 CUDA 基本的存储器和线程模型、CUDA 对 C 语言的扩展和基本的编程与调试工具。在这次课后，学生要能在几个小时

内编出简单矢量相加的代码。随后的4个课时，让学生从概念上理解CUDA的内存模型、CUDA线程执行模型、GPU硬件特性和现代计算机体系架构。这些课时基于第4、5和6章的内容。

第2个阶段——一系列的课时涵盖并行计算中的浮点数问题和在高性能并行应用中需要的数据并行编程模式。这些课时基于第7～第10章。经过这个阶段的学习，矩阵乘法的代码性能提高了近10倍。在这一阶段，学生还应该完成关于卷积、向量归约和前缀扫描的学习。

第3个阶段——一旦学生掌握了CUDA基本的编程技巧，剩下的课程内容涵盖计算思想、各种并行执行模型和并行编程规则。这些课的内容对应于第11～第20章(这些课程的录音和视频可以从网上下载(http://courses.engr.illinois.edu/ece408/))。

把所有内容连贯起来：最终项目

当读者通过这门课、实验和本书的章节掌握了基础知识后，最终项目把所学到的经验联系起来。最终项目对这门课十分重要，它在这门课中的地位很显眼，大概需要花费两个月的时间。它采用了5个创新点：指导、专题讨论、检查、最终的报告和专题讨论会(在ECE408的网站上可以下载最终项目的更多信息，我们想提供对这些设计因素的深入思考)。

我们鼓励学生选取目前研究领域中一些具有挑战性的问题作为他们的最终项目。在这个过程中，指导老师会成立计算科学研究小组来提出问题，这些研究小组担当导师的角色。我们要求导师提供一两页的项目规格说明表，简单地描述应用程序的重要性，导师想要学生组在应用程序中完成的工作，了解与开发应用程序所需要具有的技术和技能(特定类型的数学、物理或化学课程)，以及一个网站和传统资源列表。学生可以利用这些表中对应的内容了解技术背景、总体信息构建模块，以及特定的实现方式和代码示例对应的具体的URL或FTP路径。这些项目说明表也应该为学生提供学习经验，在以后的工作当中可以定义他们自己的研究项目(在ECE408课程的网站上可以下载几个这种示例)。

我们也鼓励读者在项目的选择过程中与导师不断交流。只要学生和导师都同意了某个项目，他们就会密切地联系、频繁地切磋并不断地报告项目的进展情况。我们试图为学生和导师之间的协作关系提供便利，不仅是为学生也是为导师增长宝贵的经验。

项目专题讨论

整个班级对彼此的最终项目的意见主要通过专题讨论进行交流。我们通常会安排6个报告厅来进行专题讨论。专题讨论是为学生设计的。例如，如果一个学生已经确定了项目，

就可以把专题讨论作为一个提出初步想法、获取反馈意见和招募队友的场所。如果一个学生还没有确定自己的项目，他可以仅参加演讲和讨论并加入其中的一个项目组。在项目专题讨论期间，学生没有等级之分，这么做是为了营造一种和谐融洽的气氛，所以学生可以集中精力与导师、教学助理以及班上其他同学进行有意义的谈话。

有了专题讨论日程表后，导师和教学助理就可以花些时间给项目组提供反馈意见，以便学生可以提出问题。演讲的时间限制在 10 分钟内，因此在上课期间还有反馈和提问的时间。假设每次课 90 分钟，班级的大小最好限制在 36 人左右。为了严格控制时间并使反馈时间最大化，所有演示文稿预先加载到 PC 中。由于在专题讨论中并不是每个学生都会出席，因此每个班大概能容纳 50 个学生，可以根据需要提供额外的专题讨论时间。

导师和教学助理必须参加所有演讲，并给出有用的反馈意见。学生经常在下列问题上寻求帮助。第一，对于给定的时间，项目的规模是太大还是太小？第二，在这个领域中是否存在有利于这个项目的现有工作？第三，为并行执行而设立的目标计算是否适合 CUDA 编程模型？

设计文档

一旦学生决定选择某个项目并形成了团队后，我们就要求学生提交一个关于项目的设计文档。这有助于他们在正式进入项目前，思考整个项目的步骤。制定这种计划的能力对于他们将来事业的成功很重要。设计文档应该讨论项目的背景与目的，应用程序级目标和可能带来的影响，应用程序的主要特点，设计的概况，实现计划，它们的性能目标，经过论证的计划和可接受的测试，以及项目进度表。

在班级专题讨论会的前一周，教学助理对最终项目组进行一次项目检查。这个检查过程有助于确保学生没有偏离轨道，并且他们已经发现了设计过程早期存在的障碍。要求学生组带上他们最初的草案来检查，草案包括其应用程序的如下 2 个版本：(1)性能最优的 CPU 串行代码，采用 SSE2 和其他优化方案构建一系列健壮的基本代码，便于比较它们的速度；(2)性能最优的 CUDA 并行代码——这个版本是项目的主要成果。学生可以用这个版本认识到并行算法涉及额外的计算量时带来的系统开销。

学生组要准备好讨论每个版本代码的主要思想，任何浮点精度问题，任何针对应用程序之前结果的比较，以及显著地提高运行速度会对这个领域产生的影响。依我们的经验来看，在班级专题研讨会前一周进行检查，这是最佳的时间安排。如果时间太早，可能会把时间浪费在对不太成熟的项目和没有太大意义的会议上。如果时间太晚，学生就没有足够的时间根据反馈意见修订他们自己的项目。

项目报告

学生应该提交一份项目报告，这个报告讨论的是他们所在团队的主要发现。把 6 次课的时间放在一天，举行班级专题讨论会。在专题讨论会期间，学生所用的演讲时间正比于团队的规模。在演讲期间，学生要强调项目报告中做得最好的部分，这样也有益于整个班集体。演讲过程在学生的成绩中占的比例很高。每个学生必须单独回答一个直接向他提出的问题，因此同一个团队中每个学生的成绩也不相同。我们记录了之前的一些参与学习的情况，在 ECE408 网站上可以查看。专题研讨会为学生提供了制作简洁的演示文稿的大好机会，简洁的演示文稿能激发其他人阅读整篇文档。演讲结束后，学生要提交一份最终项目的完整报告。

在线补充

如果教师用本书作为课程的教材，就可以使用我们的实验安排、最终项目指南和示例项目说明。虽然本书提供了这些课程的知识性内容，但要想实现全部教学目标还应该查阅其他资料。我们也想邀请大家充分利用与本书相关的在线资料，这些资料可以在 ELSEVIER 出版社的网站 www.elsevierdirect.com 上下载。

最后，我们鼓励读者提供反馈意见。如果读者有关于改进本书的任何想法或者对在线资料的补充，希望能反馈给我们。当然，我们也想知道读者喜欢本书的哪一方面。

目　录