

Intel Xeon Phi 协处理器高性能 编程指南

[美] Jim Jeffers James Reinders 著

并行科技 陈健 李慧 杨昆 李伯杨 李艳新 李津宇 译

英特尔 周嫻 孙相征 游亮

Intel Xeon Phi
Copro-processor
High-Performance
Programming

Intel Xeon Phi 协处理器高性能 编程指南

[美] Jim Jeffers James Reinders 著

并行科技 陈健 李慧 杨昆 李伯杨 李艳新 李津宇 译
英特尔 周嫻 孙相征 游亮

人民邮电出版社

北京

图书在版编目 (C I P) 数据

Intel Xeon Phi协处理器高性能编程指南 / (美) 杰
弗斯 (Jeffers, J.), (美) 仁达敬 (Reinders, J.) 著 ;
陈健等译. — 北京 : 人民邮电出版社, 2014. 4
ISBN 978-7-115-34784-8

I. ①I… II. ①杰… ②仁… ③陈… III. ①微处理
器—系统设计 IV. ①TP332

中国版本图书馆CIP数据核字 (2014) 第043130号

版权声明

Intel Xeon Phi Coprocessor High-Performance Programming

Jim Jeffers, James Reinders

ISBN: 978-0-12-410414-3

Copyright©2013 James R. Reinders and James L. Jeffers. Published by Elsevier Inc. All rights reserved.

Authorized Simplified Chinese translation edition published by the Proprietor. Copyright©2014 by Elsevier (Singapore)
Pte Ltd. All rights reserved.

Published in China by Posts and Telecom Press under special arrangement with Elsevier (Singapore) Pte Ltd.. This
edition is authorized for sale in China only, excluding Hong Kong SAR, Macau SAR and Taiwan. Unauthorized export
of this edition is a violation of the Copyright Act. Violation of this Law is subject to Civil and Criminal Penalties.

本书简体中文版由 Elsevier (Singapore) Pte Ltd. 授予人民邮电出版社在中国大陆地区 (不包括香港、澳门特别行
政区以及台湾地区) 出版与发行。未经许可之出口, 视为违反著作权法, 将受法律之制裁。

本书封底贴有 Elsevier 防伪标签, 无标签者不得销售。

-
- ◆ 著 [美] Jim Jeffers James Reinders
译 陈健 李慧 杨昆
并行科技 李伯杨 李艳新 李津宇
英特尔 周姗 孙相征 游亮

责任编辑 王峰松

责任印制 程彦红 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>
北京鑫正大印刷有限公司印刷

- ◆ 开本: 800×1000 1/16

印张: 25.75

字数: 533 千字

印数: 1—3 500 册

2014 年 4 月第 1 版

2014 年 4 月北京第 1 次印刷

著作权合同登记号 图字: 01-2013-2170 号

定价: 79.00 元

读者服务热线: (010)81055410 印装质量热线: (010)81055316

反盗版热线: (010)81055315

广告经营许可证: 京崇工商广字第 0021 号



内容提要

本书由英特尔的技术专家撰写，是目前为止最全面、最系统地讲解在英特尔至强处理器和至强融核协处理器上进行并行应用开发的专著。

本书从赛车与至强融核协处理器之间的相似性入手，抽丝剥茧、层层深入，探讨实际案例指导编程应用，并对协处理器向量化、协处理器分载模式、协处理器架构及 Linux 系统、数学库，以及协处理器采样、计时与优化等进行了详细的解析，是一本详尽的至强融核协处理器使用参考指南。特别是，本书所采用的方法兼容了对未来编程模式的展望和支持，书中所阐述的统一、标准和灵活的编程模式，对于未来作为独立处理器使用的 MIC 众核产品同样适用。

本书适用于想要系统学习英特尔至强融核协处理器编程的读者，无需借助其他参考书，即可循序渐进、全面掌握。

推荐序一

2013年6月17日，在德国莱比锡开幕的2013年国际超级计算机大会上，中国国防科技大学研制的天河二号超级计算机，以每秒33.86 Petaflops（千万亿次）的Linpack浮点运算速度拔得头筹，成为全球最快的超级计算机，几乎达到第二名Titan的两倍。这台超级计算机共使用了超过48 000块英特尔至强融核协处理器，配合超过32 000颗英特尔至强E5-2600 v2代号IvyBridge的12核处理器，落户于广州超级计算中心，为科研和社会服务提供强劲的动力。

我有幸参与了天河二号所使用的至强融核协处理器产品的技术支持，其中重要的是如何让更多、更大规模的应用程序扩展运行在Intel Xeon Phi协处理器上。这里借用天河二号在ISC13发布时创造的一个新概念，来概括Intel Xeon处理器与Intel Xeon Phi协处理器的编程模式，叫做“微异构计算”（Neo-heterogeneous Computing），就是指在一个计算节点内，采用统一的x86编程模式和标准编程语言实现主机和协处理器卡的协同并行计算。正如本书作者指出的，并行编程已经成为每一名程序员必须掌握的基本技能，这源于现代处理器的指令集并行和多核的特性。现在将程序进一步扩展到超过50核的协处理器上，有必要系统地学习英特尔集成众核（Many Integrated Core, MIC）协处理器的架构、指令集、编译器、操作系统以及性能优化等一系列知识。这本由英特尔技术专家Jim Jeffers与James Reinders撰写的《Intel Xeon Phi协处理器高性能编程指南》，是目前最为全面、最为系统地讲解在Intel Xeon和Intel Xeon Phi上进行并行应用开发的专著，中文版由并行科技和英特尔公司的工程师合作翻译。

本书的两位作者，一位是中国读者较为熟悉的仁达敬（James Reinders），他已有多部软件开发专著的中文版本在中国出版，包括大家熟知的《Vtune性能分析器基础》和《Intel Threading Building Blocks编程指南》。他是一位在英特尔软件集团热心技术写作和传播的“技术布道师”。另一位Jim Jeffers是MIC产品平台应用的首席工程师，也是我现在的经理。在本书计划写作之初，我们预计到由于Intel Xeon Phi协处理器采用x86统一编程模式，因此很多在至强处理器上并行编程需要掌握的知识对集成众核也是适用的，从理论上讲对熟悉在至强服务器上并行开发的读者而言，本书不应该太厚。然而想要系统学习Intel Xeon Phi协处理器编程的读者，也需要一本内容自足（Self-contained）的书籍，无需借助其他参考书就能够循序渐进全面掌握，因而本书又不能太过单薄。

两位作者在写作中巧妙地以赛车为例，由通用编程模式开始，从硬件到软件，从操作系统、编程实践、数学库到性能剖析和优化，层层展开，抽丝剥茧，写成了这本具有教科书品质的工

程专著。之所以说是工程专著，因为它秉承了英特尔公司“Engineer Write for Engineer Read”的传统，对于已经掌握了在至强平台上并行编程的读者，您可以直接跳到第 5 章至第 7 章，了解 Intel Xeon Phi 协处理器对向量化和多线程在编译器上的支持后，学习分载（offload）模式直接实践。而对 Intel Xeon Phi 协处理器微架构、系统软件栈及操作系统感兴趣的读者，可以直接从第 8 章阅读到第 10 章，在编程和优化的过程中，可反复验证这几章的知识点。第 11 章和第 12 章属于在 Intel Xeon Phi 协处理器上使用 MKL 数学库和进行 MPI 编程的进阶知识，第 13 章则进入了性能优化的领域。

本书的英文版由 Morgan Kaufmann 公司在美国出版，今年 4 月份英特尔公司在北京举行信息技术峰会（IDF）期间，两位作者慷慨地给我寄过来 30 本英文版，很快被很多用户和开发人员分享一空；我自己手头的一本，在上海交通大学举行的 ASC13 亚洲大学生超算大赛闭幕式上，被一位俄罗斯参赛队大学生要走。在 2011 年西雅图举行的 SC11 上，我支持的 Intel Xeon Phi 协处理器演示展台上，本书的预览版和我们国内合作厂商出版的 Intel Xeon Phi 协处理器样书“遭遇”了同样的命运，都被热心的读者“先睹为快”了。这些小故事，随着这本书中文版的发布，成为有趣的注脚。

Jim Jeffers 和 James Reinders（仁达敬）对本书的中文版出版很是热心，他们为此也写下了热情洋溢的推荐，原文翻译如下：

“对于这本关于并行编程基础及其在英特尔至强融核协处理器上应用的书，能够以中文版提供给高性能计算的开发人员，我们感到非常高兴和欢欣鼓舞。在 2013 年 6 月，天河二号使用了大量的英特尔至强融核协处理器，成为 Top 500 榜单中排名世界第一的超级计算机，为中国和全世界在未来创新的科学发现与洞见，提供了难以置信的强大计算资源。为了满足永无止境的未来计算需求，高性能计算行业的发展道路必将是不增长的硬件并行特性、开发工具和应用程序三者的结合。对开发者而言，拥有‘并行思维’以及努力开发高度并行的代码以获得在计算性能上的新突破，变得比任何时候都更加重要。希望我们的中国读者能够发现，本书不仅提供了通向并行计算的坚实基础，而且允许他们继续享受使用熟悉而标准的并行编程模式 and 操作环境所带来的好处。在此要特别感谢与我们密切合作的同事何万青博士对此书中文版翻译的支持，以及他在中国对天河二号和其他使用英特尔至强融核产品的超级计算机所做的重要支持。”

本书的英文版有些章节我读过不止一遍，其中既有技术阅读者的乐趣所在，也从另一个侧面印证了此书所具备的教科书品质。特别地，这本书的原理兼容了对未来编程模式的展望和支持，比如对未来作为独立处理器使用的 MIC 众核产品，本书所阐述的统一、标准和灵活的编程模式同样适用。在本书中文版出版之际，我代表两位作者对付出辛勤劳动的各位译者，表示衷心的感谢！

何万青 博士

英特尔技术计算集团平台应用架构师

2013 年 7 月 10 日于香港返京 CA102 航班

推荐序二

迟迟才拿到这本书，翻开扉页，心中一下子五味杂陈。不能忘记那一段值得珍藏的旅程。

2005 年，当时的资深首席工程师道格·卡米恩（Doug Carmean）在与客户的交流中设想出一种新的体系结构。其时整个英特尔都在向多核（multi-core）转弯，而道格的视线落在了更为激进的众核（many-core）上，这是一种配置 32 个以上 x86 核的怪兽级架构。即使在英特尔，这仍是令人啧啧称奇的想法。那么高的并行性，用来做什么呢？设计团队想到了图形处理。于是，一个定位于高端独立显卡市场的新架构问世了，内部叫它 Larrabee，名字来源于一个公园，其吉祥物是一只叫 Larry 的蜜蜂。道格在世界各地招募了一些研发人员，我在北京的团队也在机缘巧合下“共襄盛举”。我们从模拟器做起，从一个早期奔腾处理器的设计改过来，调试分布式的缓存一致性协议和纹理采样器（texture sampler），提交第一个支持新向量指令集的编译器，用小的 kernel 程序验证向量指令集，基于 Eclipse 实现集成开发环境，试验着色程序（shader）编译器，还有就是研究 Larrabee 之上的并行编程环境。我记得在连续几次的内部 Larrabee 峰会上，中国这边的演讲占到了整个议程的四分之一。这一切仍历历在目！

2009 年，在内外极高的期望下，Larrabee 失败了。是非功过，自有评述。我们北京的团队也转向新的领域，但无法不去想它。天可怜见，作为图形处理器的 Larrabee 不存在了，但它的很多设计脱胎换骨，成为英特尔集成众核（Many Integrated Core, MIC）协处理器的基础。原先的硬件经过大刀阔斧地修改演变为第一代 MIC。早期 Larrabee 的基于 BSD 的操作系统也进化成为更为大众化的、以 Linux 为核心的众核操作系统。幸运的是，原有的编程工具基本得到了传承。北京团队原来的一些工作仍然得到了保留，如大家在本书中可以看到 MYO 异构共享内存编程模型。我们在数据并行编程语言上的探索也影响了 Cilk plus 数据段的设计和实现。

第一代 MIC 虽然不是正式产品，但作为软件开发平台（Software Development Vehicle）交付于部分客户，在与客户的携手探索中不断改进。这是我们学到的重要一课，硬件不能脱离软件和客户的应用而独立发展。如今，Xeon Phi 虽是第一代协处理器产品，其实架构上已经是第二代，其中有生态系统中共同旅行者的贡献。

Xeon Phi 与中国还有更多的缘分。“天河二号”自然不需多说。推荐序一的作者何万青博士一直在耕耘中国的生态系统，是具有很高造诣的应用架构师。译者陈健博士在英特尔公司时辅佐何万青博士，作为并行科技的领军人物，是屈指可数的技术、商业两栖“并行达人”。中国这边还有数支软件团队默默支撑着本地高性能计算和互联网客户在 Xeon Phi 上的探索。必须要提

一下学术界，中国的学术界积极参与了“众核研究社区（Many-Core Research Community）”，不仅是在 Xeon Phi 上，在英特尔公司另一款 48 核实验芯片“单芯片云（Single-Chip Cloud）处理器”上也做出了令人激赏的研究。还有正在读书的年轻一代极客，他们在刚刚过去的 2013 年中国高性能计算年会“并行应用程序优化大赛”中展现出来的爆发力，让我肃然起敬。

本书的 14 章内容，凝聚了英特尔公司成百上千的工程师和研究人员 7 年的智慧和汗水，由仁达敬（James Reinders）和 Jim Jeffers 娓娓道来，具有极高的写作水准和技术水准。对于中国的更多有志于并行计算的年轻人，这本书是最佳的理论兼实践指南。而对已经或即将采用 Xeon Phi 的企业和研究机构来说，本书也是一种承诺。书中描绘了 Xeon Phi 未来的发展路线，下一代众核产品在软件兼容性上能让您在“众核时代”品尝免费的午餐（无需修改应用自动获得性能提升），而作为独立处理器（而非协处理器），它将开拓更多的应用场景。我本人已经转到大数据的研究，非常看好下一代 Xeon Phi 跃出高性能计算的象牙塔，在技术计算（Technical computing）和大数据领域大展宏图。

推动任何一款新的计算架构，高水平的硬件、软件、用户教育三者缺一不可。至此，Xeon Phi 已经三者兼具，不再有憾。

吴甘沙

英特尔中国研究院 首席工程师

2013 年 12 月 18 日于北京

译者序

北京时间 2012 年 11 月 13 日，英特尔公司在美国盐湖城举办的超级计算机（SC12）大会上，宣布发布新的至强融核协处理器以适应当前热门的高性能计算市场需求，同期也在北京举办了发布活动。随着英特尔至强融核协处理器在市场上的广泛使用，对于其高性能计算编程应用的需求也日趋高涨，英文专著 *Intel Xeon Phi Coprocessor High-Performance Programming* 就在此种情形下应运而生。同时，为了顺应中国读者的需求，北京并行科技有限公司联合英特尔公司合作翻译了此书籍。由于至强融核协处理器沿袭了至强融核 x86 的 CPU 架构，读者仅需要参考中文版书籍的编程指南，熟悉在集成众核协处理器上的并行开发及并行编程，重新编译代码即可运行，进而确保在软件上的投资得到最大程度的保护。

本书英文原版的两位作者，一位是中国读者较为熟悉的英特尔公司首席技术布道师兼软件产品总监仁达敬（James Reinders），热衷于技术写作与技术传播。另一位 Jim Jeffers 则是 MIC 产品平台应用的首席工程师，对至强融核协处理器有着较为深入与全面的研究。本书的英文版由 Morgan Kaufmann 公司在美国出版，已经有大量的读者群。今年 4 月英特尔公司在北京举行信息技术峰会（IDF）期间，由北京并行科技有限公司提供的中文试读本，很快被与会人员索取一空。读者对此书极高的热情使得此书中文版的出版势在必行，同时也促使译者更加认真地翻译此书。

至强融核协处理器广受欢迎，有其独到的优势，它完全支持 Fortran、C 和 C++ 编程语言，并支持通用的编程方法，例如 OpenMP、MPI、Intel TBB 等，同时也支持 Coarray Fortran、Intel Cilk Plus、OpenCL 等此类广泛使用的新方法。另外，至强融核协处理器的优化扩展性，向量使用及内存使用，对运行在至强处理器上的应用程序同样有益。应用程序在至强融核协处理器上通过改进达到最佳性能，将其运行于至强处理器上时会得到同样的效果，此即为本书提到的“移植与优化的双重优势”，也是至强融核协处理器颇受用户欢迎的根本原因，保证了用户在软件上的编程投资得到最大程度的保护。

本书的写作风格较为独特，从一个形象的类比——赛车与至强融核协处理器之间的相似性入手，层层深入，前 4 章探讨具有实际意义的案例来指导编程，后续内容则是对细节内容更为深入的解析，如协处理器向量化，协处理器分载模式，协处理器架构及 Linux 系统，数学库，协处理器采样、计时与优化，各类编程方法等。毋庸置疑，本书是一本详尽的至强融核协处理器使用参考指南。

本书讲解的是一种方法论，涉及的技术对于处理器和协处理器均有效，即本书强调“移植与优化的双重优势”，这使得用户的编程投资转换到其他任何通用处理器平台上同样能够得到良好的回报，即使是未来的系统，也同样适用，这也被称为面向未来的可扩展性（forward scaling）。

针对本书的篇章结构，以下是供读者参考的阅读指导。

第一部分：绪论（第1章）。阐述对 Intel Xeon Phi 协处理器的持续需求，通过优化可以在 Intel Xeon 处理器和 Intel Xeon Phi 协处理器上同步提升应用程序性能，进而引出 Intel Xeon Phi 协处理器编程极具回报性和保护编程投资的关键因素，即“移植与优化的双重优势”。同时，引出第二部分 Intel Xeon Phi 协处理器与赛车之间的类比。

第二部分：案例探讨与编程要点学习（第2~4章）。比较 Intel Xeon Phi 协处理器和赛车之间的相似性，为了达到最佳计算性能，需要确保代码向量化及计算能力的规模扩展。通过案例探讨，引导读者掌握 Intel Xeon Phi 协处理器并行编程要领。

第三部分：实现性能提升的方法（第5~7章）。详细阐述 Intel Xeon Phi 协处理器向量化的方法，编写可向量化代码需要具备的核心内容，以及为了保证充分的任务级并行进行规模扩展的方法，同时针对向量化与规模扩展探讨以分载模式应用这些技术。

第四部分：协处理器支撑环境（第8~10章）。深入了解 Intel Xeon Phi 协处理器的硬件架构，其在标准环境中运行所需要的软件架构和组件，以及协处理器在 Linux 系统上的实现，为用户在协处理器上高效运行各种应用程序，提供了必要的基础知识。

第五部分：开发高性能并行应用程序的各种函数库和工具（第11~13章）。包括英特尔数学核心库（Intel MKL）、MPI 及其他一些运行分析工具，综合使用这些工具可以带来应用程序性能分析的突破性进展。

至此，本书已全面展示 Intel Xeon Phi 协处理器高性能编程的完整内容，书中的相关术语可参考最后的术语表，对于一些专有名词本书提供了较为详尽的解析。

本书由北京并行科技有限公司与英特尔（中国）有限公司的相关技术专家合作翻译。整个翻译团队的所有成员，专注于国内高性能计算应用在英特尔平台上的并行开发和性能优化，积累了丰富的工程实践经验，这一点在本书的行文风格上体现得淋漓尽致。具体地，本书的第1、5章由李慧翻译，第2、3、11章由李伯杨翻译，第4章由游亮翻译，第6章由李艳新翻译，第7章由李艳新、李津宇共同翻译，第8章由周姗姗翻译，第9、10章由杨昆翻译，第12章由孙相征翻译，第13、14章由陈健翻译，序和致谢由李艳新翻译，术语表由李慧和李津宇共同负责翻译。另外，英特尔的何万青博士在百忙之中抽出宝贵的时间翻译了本书英文版的序，并仔细校对了英文版前言和致谢的翻译文字。本书的代码测试及运行方法的整理工作由李伯杨完成，翻译工作由贺玲全程指导安排，全书的统稿校对由李艳新负责。

同时，陈健在初稿定稿后进行了认真细致的审阅，并提出需要再次确认内容的详细问题。英特尔的乔楠、王哲对本书所有未确定复杂问题提供了周密解答，使得本书进一步完善、精确。

因为各位译者的共同努力与探索，对技术精益求精的追求，使得本书可以如期诞生。在此对团队成员及支持他们的家人表示衷心的感谢！

技术的精进永无止境，此次翻译过程影响着对整个翻译团队在未来工作中一丝不苟、严谨求实的态度。由于译者的时间和水平有限，翻译中的疏漏之处在所难免，还望读者和同行不吝指正。苏轼有言：“博观而约取，厚积而薄发。”传递的理念即是勤于积累和精于应用。追求技术的精湛，是整个社会未来发展的不竭源泉。

译者

2013年9月25日

原书序

自 20 世纪 80 年代中期引入大规模并行计算机以来，比如英特尔 iPSC 和 nCUBE，以及随后的 IBM SP 和 Beowulf 机群，我难以想象还有比这更为激动人心（或者说异彩纷呈）的高性能计算时代。目前，技术革命不仅助益于那些使用 MPI 并行化或其他分布式内存模型的高性能应用程序，更是逐步蔓延到单个节点内而惠及众生，无论你是使用笔记本电脑还是数千万亿次级别（multi-petaFLOPS/s）的超级计算机。与过去 10 年冲击我们集体想象力的 GPGPU 形成鲜明对比的是，Intel Xeon Phi 系列产品（其第一代产品已经达到 1teraFLOP/s 的双精度峰值速度！）将超级计算机的性能带入了每个人的办公室，其使用完全兼容于办公桌面环境的标准编程工具，包含完整的数值软件和 GNU/Linux 软件栈。基于从多核英特尔架构（Intel Architecture）处理器到众核 Intel Xeon Phi 产品所具备的架构和软件的一致性，我们针对可移植、高性能计算系统，其设计基于常见且完整线程化和向量化的编程模型，首次推出了完整的方案。

并且，英特尔集成众核（Intel MIC）架构的愿景将会带领我们从 2012 年的早期千万亿次（petaFLOP）时代步入 2020 年的百万亿亿次（exaFLOP）时代，事实上，基于 Intel MIC 架构的 Intel Xeon Phi 协处理器足以窥见未来。

那么有什么内幕？实际上并没有新消息——串行（更具体的是指单线程、非向量化）计算甚至在台式机上已经死了。死很久了，多年前被流水线功能单元、多指令发射、SIMD 指令扩展以及多核架构杀死了。但是，如果你有一个属于 99% 的尚未多线程和向量化的程序，那么在带有 AVX SIMD 单元的多核 Intel Xeon 处理器上，你会有错失性能加速 100 倍的可能，而在高度线程化的英特尔集成众核架构上，则意味着错过加速 1000 倍的可能。

是的，上述数据你没有读错。一个标量单线程应用，取决于限制其性能的因素，可能会在单 CPU 上遗留多达数个数量级的性能空间。所有的现代处理器，无论是 CPU 抑或是 GPGPU，都依赖大量的并行性以实现高性能。使用 Intel Xeon Phi 协处理器的一个好消息是，感谢标准编程环境支持的开箱即用的应用（再次对比 GPGPU 需要重新编码才能运行），你可以使用熟悉的工具分析性能，并拥有一个稳健的适应增量式代码转变的优化路径，而该优化可直接带入到主流处理器。但是如何优化代码呢？哪些算法、数据结构、数值表达、循环语句、编程语言、编译器等更适于 Intel Xeon Phi 产品？如何做到这一切，使其不仅局限于当前的英特尔集成众核架

构而可扩展至未来，甚至非英特尔架构上？本书周到而慷慨地回答了上述问题并将更多知识倾囊相授。

一个关键的要点是，早期“杀手级微处理器”革命使得专用向量处理器被商品化通用 CPU 所取代，应用开发者停止了开发向量化算法的努力，因为前几代的 CPU 确实在操作密集型的串行应用上获得相当好的峰值性能。时间快进 20 年，这种情况已经不切实际，时光荏苒，在我们的共同记忆中，那些在 Cray 和其他向量机上如此成功的向量化算法的智慧和传说被逐渐淡忘。然而，那个时代的成就赋予我们极大的信心，多线程、具备丰富向量指令的标量-向量编程模式被证明适用于非常广泛的算法和学科。毋庸讳言，新的挑战总是存在的，包括更深和更复杂的现代处理器的存储层级，数量级增长的线程数，缺少硬件级别的 Gather/Scatter 操作，以及要赶上（或重新发现？）25 年前水平的编译器。

2010 年 10 月，在休斯敦一个由 John Mellor-Crummey 组织的关于编程语言工具的非常棒的研讨会上，我做了题为“DSL, Vectors 和健忘症”的讲话。而“健忘症”指的就是上面所提到的缺失的向量化能力。我以理性思考早期 GPGPU 平台宣称取得令人惊异的加速比为例，来判断主流编程的成功与否。在 GPGPU 实现了峰值内更高性能的情况下，两个平台上对应用的不一致优化是一个简单解释。但为何差别如此，关于性能可移植代码的编写，从中又可学到什么？性能差异的底层原因有两个。首先，OpenCL 和 Nvidia 的 CUDA 这类数据并行编程语言，强制编程人员编写大规模数据并行代码，通过良好的编译器，以及对向量长度和数据布局进行优化来完美匹配底层硬件，从而实现高性能。其次，所宣布的性能通常是对比一个执行标量、单线程计算的 x86 代码，目前看来是远未优化的实现。因此，通用的方案应该是将 GPGPU 代码逆向移植到多核 CPU，并根据核数和向量/缓存规模重新优化——事实上如果足够小心并且运气不错的话，相同的代码基础可以运行在两个平台上（如果使用 OpenCL 编程，必然可以实现）。所有针对于局部性、带宽、并发和向量化的优化被直接运用，同时，编译器更容易分析更清晰、简化、设备无关的代码。因此有充分理由相信，几乎所有在当前 GPGPU 上运行良好的算法，只需做最小的重构即可在 Intel Xeon Phi 协处理器上同样运行，而那些需要细粒度并发控制的算法，在协处理器上实现要远易于 GPGPU。

阅读本书你会逐渐认识作者。通过参与英特尔集成众核架构先期客户评估计划，我曾和作者之一的 Jim Jeffers 共事，他是一位思路清晰、为人亲切谦和的专家，值得您付出时间与精力研读本书。而且，他与另外一些 Intel Xeon Phi 产品开发项目领导深刻地意识到，他们所做的重要且富变革意义，将会塑造计算机行业的未来，值得为其奉献出自己的职业生涯和个人生活。

这本书属于每一个 HPC 从业者的书架。它不仅成功地教会我们在英特尔集成众核架构上如何使用并获得高性能，还涵盖了更多丰富的内容。它将读者带回到高性能计算通用原理的学习，

包括如何思考、推理以将算法的性能映射到现有架构之上，同时还教给读者强大的工具，多年后仍将受用不尽。

Robert J. Harrison

先进计算科学研究院

石溪大学

2012 年 10 月

英特尔技术计算集团平台应用架构师何万青博士

于 2013 年 12 月 1 日译

致谢

非常感谢用户和英特尔工程师的帮助，此书才得以如期诞生。

感谢 Scott McMillan 编写的第 12 章 MPI 内容。非常欣赏 Scott McMillan 的真知灼见与写作技巧。

感谢石溪大学的先进计算科学研究院 (Institute for Advanced Computational Science) 主任 Robert Harrison，感谢他的支持、专业和热情，以及在前言中分享的独到见解。

我们希望感谢几个重要人物，他们提供了所写文献或是白皮书，作为编写本书的参考资料。参考资料的品质和数量在编写此书之初解决了我们的燃眉之急：Ron Te 和 Mike Casscles 的《系统软件开发指南》(*System Software Developers Guide*)；Clayton Craft 的《指令集架构》(*Instruction Set Architecture*) 文档；Robert Reed 和 Shannon Cepeda 提供的第 4 章中的模板代码 (stencil code)；Shannon Cepeda 和 Wendy Doerner 提供的向量化培训和相关文献；Martyn Corden 提供的向量化文章；Shannon Cepeda、Larry Meadows 和 David Mackay 提供的优化文献；Johnnie Peters 和 Alexandra Yates 提供了第 10 章的 Linux 配置信息；Rakesh Krishnaiyer 关于预取给出相关建议；Tylor Kidd 提供了无数的培训文档，突出强调本书关键点和一些需要特别注意的事项；Raymundo Vazquez 提供了计时器的信息；Michael Hebenstreit、Romain Dolbeau、Jeremy C. Siadal、Andreas Hirstius、Andres More、Clem Cole、Matias Cabral 和 Werner Krotz-Vogel 提供的机群配置信息。

我们想对英特尔 SSG Information Development 团队所做的伟大工作表达谢意，他们编写出了杰出的产品文档，对我们来说既具有指导意义，又能用作参考资料。如果我们没能提到做出贡献的所有人，我们诚挚表示抱歉：感谢整个团队，包括 Mitch Bodart、Ivan De Cesaris、Kevin Davis、Rajiv Deodhar、Karen Dickinson、Wendy Doerner、Stephen Goodman、Ronald Green、Elena Indrupskaya、Irina Kosareva、Rakesh Krishnaiyer、Lorri Menard、Sabrina Mesters-Woell、Ravi Narayanaswamy、Sunil Pandey、Christopher Raleigh、Premanand M Rao、Amanda Sharp、Michael Sturm、Michael Toguchi、Miwako Tokugawa、Stan Whitlock 和 Anatoly Zvezdin 付出的辛勤努力。

感谢 Knights Corner 项目的参与者，他们投入了很多精力，并且针对第 1 章内容提供了极具参考价值的反馈，这些人包括：Mani Anandan、Andrew Brownsword、George Chrysos、William Clifford、Kim Colosimo、Charles Congdon、Joe Curley、Ervin Dehmlow、Pradeep Dubey、Gus Esponosa、Robert Geva、Milind Girkar、Ron Green、Michael Greenfield、John Hengeveld、Scott

Huck、Jim Jeffers、Mike Julier、Rakesh Krishnaiyer、Belinda Liverio、David Mackay、Bill Magro、Larry Meadows、Chris Newburn、Brian Nickerson、Beth Reinders、Frances Roth、Bill Savage、Elizabeth Schneider、Gregg Skinner、Sanjiv Shah、Lance Shuler、Dancil Strickland、Brock Taylor、Philippe Thierry、Xinmin Tian、Matt Walsh 和 Joe Wolf。

还有许多人校对了此书，非常感谢他们以及那些在百忙之中愿意奉献时间改进此书的人们，其中包括北卡罗利纳大学复兴计算研究所（RENCI/University of North Carolina）的 Rob Fowler，瑞士国家超级计算中心（CSCS/Swiss National Supercomputing Centre）的 Neil Stringfellow 和 Jeff Poznanovic，克雷公司（Cray Inc.）的 Kevin Thomas 和 James L. Schwarzmeier，英特尔的 Clayton Craft、Jeanette Feldhousen、Michael Hebenstreit、Elena Indrupskaya、Hans Pabst、Steve Lionel、Susan Meredith、Xinmin Tian、Zhang Zhang 和 Georg Zitzlsberger，以及桑地亚国家实验室的 James A. Ang。

Heinrich Bockhorst、Kathy Carver、Jim Cownie、Loc Nguyen、Klaus-Dieter Oertel、Dmitry Sivkov、Alexander Supalov 和 James Tullos 提供了针对第 12 章的宝贵反馈。

科尔法公司（Colfax Corporation）Andrey Vladimirov 的真知灼见和培训材料给予了我们编写此书的灵感，他提供了宝贵的反馈，包括第 7 章中使用的某些代码示例。

感谢得克萨斯高级计算中心（Texas Advanced Computing Center, TACC）团队提供关于本书和英特尔众核项目的宝贵反馈意见，以及第 3 章中模板算法的原理，包括 Kent Milfeld、Dan Stanzone、Bill Barth、Karl Schultz、Tommy Minyard、Jim Brown、John McCalpin、Lars Koesterke 和得克萨斯高级计算中心主任 Jay Boisseau。

Glenn Brook、Ryan Braby、Ryan Hulguin、Vince Betro 和许多其他在国家计算科学研究所（National Institute of Computational Science, NICS）的同仁对本书提供了反馈，以及他们将协处理器用于科研的杰出工作和宝贵意见。

James Reinders 要特别感谢他的妻子 Elizabeth 和女儿 Katie，她们在本书漫长的写作过程中给予了无私的支持与鼓励。Beth Reinders 的显著才能对第 1 章绪论居功至伟，她不但使该章言之成理，而且对于更广大读者所需要的简练要点，她给出了对于本书助益颇多的深刻见解，特别是对我们在第 1 章和第 8 章所面临的挑战。最后，要感谢我的合作执笔者和益友 Jim Jeffers，他在教导、理解和答疑解惑方面的天赋印证了他是编写本书的最佳搭档，他始终以一种幽默诙谐的方式努力工作着，即使是在客户危机时期和主打产品发布之际（这就是为什么我们会同意在业余时间编写此书！）。

Jim Jeffers 要特别感谢他的妻子 Laura 和儿女 Tim、Patrick、Colleen、Sarah 和 Jon 一直以来的支持、鼓励和照顾，理解包容他在交流和编写此书过程中没能有足够的时间陪伴她们。同时，感谢我的挚友兼同事 Joe Curley 带我回到英特尔，为了这个气势宏伟和振奋人心的项目贡献个人力量。感谢 Nathan Schultz 的支持与鼓励，尤其是在本书完成之际的帮助。最后，感谢 James Reinders 深厚的友谊、专业的指导和清晰的思路，推动着此书走向完整。

我们要感谢 George Chrysos、Joe Curley、Stuart Douglas、Herb Hinstorff、Michael Julier、Michael McCool、Chris Newburn、Elizabeth Reinders、Nathan Schultz 和 Arch Robison 付出了宝贵的时间，感谢他们的大力支持与专业意见。

感谢英特尔出版社团队在此书完成之际给予的极大帮助，特别是 David Clark 和 Stuart Douglas。感谢 Dan Mandish 提供的精美插图。非常感谢 Morgan Kaufmann 团队的辛勤工作，包括与我们直接协作的三位：Todd Green、Lindsay Lawrence 和 Mohanambal Natarajan。

很多同事提供了资料、建议和展望。许多听课的学生提供了反馈信息，进一步提高了我们培训资料的质量，这些信息同时也添加到了本书中。当然还有许多直接或间接帮助我们的人们没有提到。感谢所有帮助我们的人们，对于那些帮助我们却未被提及的人们，在此我们表示诚挚的歉意。

感谢所有人！

James L. Jeffers & James R. Reinders