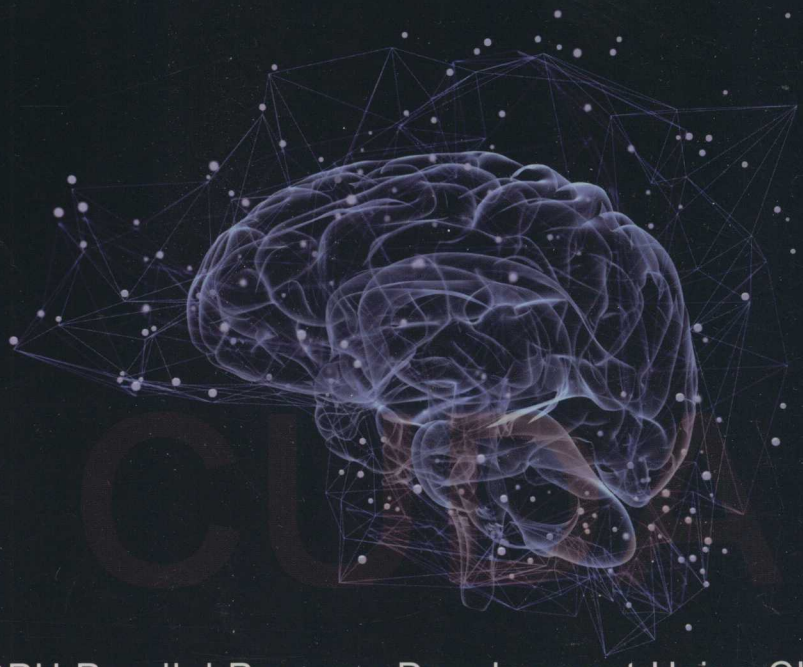
HZ BOOKS
华章IT

作者结合其多年CUDA教学经验以及与Nvidia公司多年合作积累的工程经验精心撰写
紧扣深度学习热点，包含cuDNN技术
通俗易懂、循序渐进，是学习CUDA编程的最佳选择

高性能计算技术丛书

CRC Press
Taylor & Francis Group



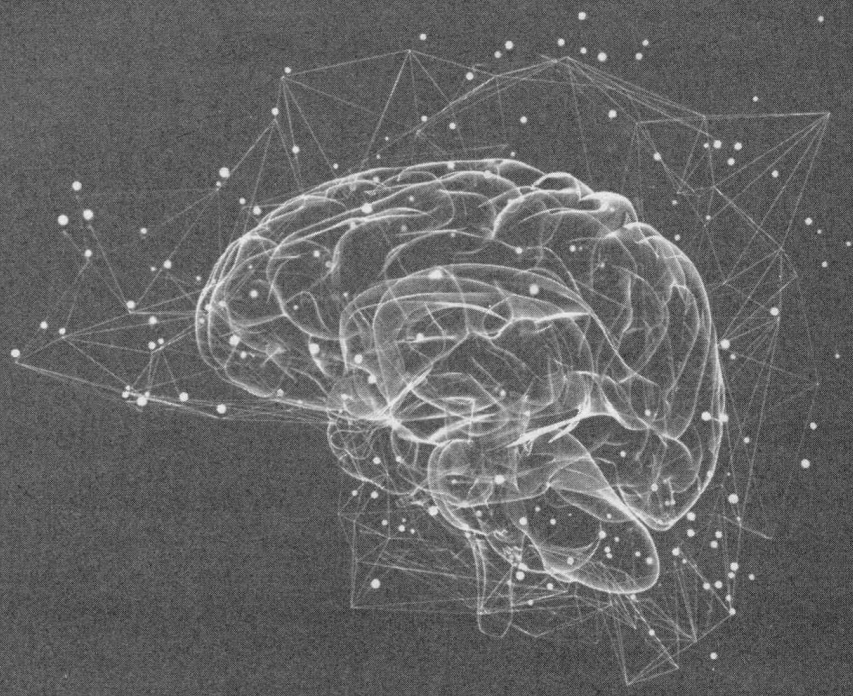
GPU Parallel Program Development Using CUDA

基于CUDA的GPU并行程序开发指南

[美] 托尔加·索亚塔(Tolga Soyata) 著
唐杰 译

机械工业出版社
China Machine Press

高性能计算技术丛书



GPU Parallel Program Development Using CUDA

基于CUDA的GPU并行程序开发指南

[美] 托尔加·索亚塔(Tolga Soyata) 著
唐杰 译

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

基于 CUDA 的 GPU 并行程序开发指南 /（美）托尔加·索亚塔（Tolga Soyata）著；唐杰译 . —北京：机械工业出版社，2019.6
（高性能计算技术丛书）
书名原文：GPU Parallel Program Development Using CUDA

ISBN 978-7-111-63061-6

I. 基… II. ①托… ②唐… III. 图像处理 - 程序设计 - 指南 IV. TP391.413-62

中国版本图书馆 CIP 数据核字（2019）第 126626 号

本书版权登记号：图字 01-2018-2746

基于 CUDA 的 GPU 并行程序开发指南

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码：100037）
责任编辑：赵 静　　责任校对：殷 虹
印 刷：中国电影出版社印刷厂　　版 次：2019 年 7 月第 1 版第 1 次印刷
开 本：186mm × 240mm 1/16　　印 张：27.75
书 号：ISBN 978-7-111-63061-6　　定 价：179.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88379426 88361066　　投稿热线：（010）88379604
购书热线：（010）68326294　　读者信箱：hzit@hzbook.com

内容简介

本书旨在帮助读者了解与基于CUDA的GPU并行编程技术有关的基本概念，并掌握使用C语言进行GPU高性能编程的相关技巧。本书共分为三部分，第一部分通过CPU多线程编程解释了并行计算，使得没有太多并行计算基础的读者也能毫无阻碍地入门CUDA；第二部分重点介绍了基于CUDA的GPU大规模并行程序的开发与实现，并通过大量的性能分析帮助读者理解如何开发一个好的GPU并行程序以及GPU架构对程序性能的影响；第三部分介绍了一些常用的CUDA库、OpenCL编程语言、其他GPU编程语言和API以及深度学习库cuDNN。

本书内容翔实、实例丰富，可作为高等院校相关专业高年级本科生和研究生课程的教材，也可作为计算机相关技术人员的参考书。

原书封面

译者简介

唐杰博士

南京大学计算机科学与技术系副教授，主要研究高性能计算与并行处理技术，主持和参与国家科技重大专项、国家自然科学基金等十余项课题，在国内外学术会议和期刊上发表了50多篇论文，还参与编写了多部教材。

The Translator's Words 译 者 序

近 10 年来，随着大数据、深度学习等相关领域的发展，对计算能力的需求呈几何级数增长。与此同时，大规模集成电路的发展却受到功耗、散热、晶体管尺寸等客观因素的限制，难以继续维持摩尔定律。因此，人们逐渐把目光转向了并行系统。GPU 自诞生之日起就是为计算机的图形图像渲染等大规模并行处理任务而服务的，因而越来越受到研究界和企业界的关注。随着 CUDA 等计算架构模型的出现，这一趋势更加明显。

CUDA（Compute Unified Device Architecture，统一计算设备架构）是 Nvidia（英伟达）提出的并行计算架构，它可以结合 CPU 和 GPU 的优点，处理大规模的计算密集型任务。同时，它采用了基于 C 语言风格的语法，又将 CPU 端和 GPU 端的开发有效地集成到了同一环境中，对于大多数 C 程序员来说，使用十分方便，因而一经推出就迅速占领了 GPU 开发环境的市场。

然而，会写 CUDA 程序与会写好的 CUDA 程序相差甚远！

阻碍 CUDA 程序获得高性能的原因有很多。首先，GPU 属于单指令多数据类型的并行计算，因而任务切分方式非常关键，既要充分挖掘线程级的并行性，也要充分利用流来实现任务级的并行。其次，GPU 的存储类型和访问模式比 CPU 的要丰富得多，一个成功的 CUDA 程序要能充分利用不同类型的存储。再次，Nvidia GPU 的架构还处于高速发展期，新一代 GPU 所推出的新功能也能够有效地提升计算效率。最后，万丈高楼平地起并不是 CUDA 开发的最佳方式，Nvidia 和一些第三方机构都开发了很多基于 CUDA 的支撑库，利用好这些第三方库可以让你的开发过程事半功倍。

Tolga Soyata 结合他 10 多年的 CUDA 教学经验以及与 Nvidia 多年合作的经历精心撰写了本书，针对上述问题进行了详细而生动的阐述。本书最独特的地方是它在第一部分中通过 CPU 多线程解释并行计算，使没有太多并行计算基础的读者也能毫无阻碍地进入 CUDA 天

地。第二部分重点介绍了基于CUDA的GPU大规模并行程序的开发与实现。与现有的同类书籍相比，本书的特点是在多个Nvidia GPU平台（Fermi、Kepler、Maxwell和Pascal）上并行化，并进行性能分析，帮助读者理解GPU架构对程序性能的影响。第三部分介绍了一些重要的CUDA库，比如cuBLAS、cuFFT、NPP和Thrust（第12章）；OpenCL编程语言（第13章）；使用其他编程语言和API库进行GPU编程，包括Python、Metal、Swift、OpenGL、OpenGL ES、OpenCV和微软HLSL（第14章）；当下流行的深度学习库cuDNN（第15章）。

本书通过生动的类比、大量的代码和详细的解释向读者循序渐进地介绍了基于CUDA编程开发的GPU并行计算方法，内容丰富翔实，适合所有具备基本的C语言知识的程序员阅读，也适合作为GPU并行计算相关课程的教材。

在本书的翻译过程中得到了机械工业出版社华章公司朱捷先生的大力支持，在此表示由衷的感谢！

限于水平，翻译中难免有错误或不妥之处，真诚希望各位读者批评指正。

唐　杰

2019年1月

Preface 前　言

我经历过在 IBM 大型机上编写汇编语言来开发高性能程序的日子。用穿孔卡片编写程序，编译需要一天时间；你要留下在穿孔卡片上编写的程序，第二天再来拿结果。如果出现错误，你需要重复这些操作。在那些日子里，一位优秀的程序员必须理解底层的机器硬件才能编写出好的代码。当我看到现在的计算机科学专业的学生只学习抽象层次较高的内容以及像 Ruby 这样的语言时，我总会感到有些焦虑。尽管抽象是一件好事，因为它可以避免由于不必要的细节而使程序开发陷入困境，但当你尝试开发高性能代码时，抽象就变成了一件坏事。

自第一个 CPU 出现以来，计算机架构师在 CPU 硬件中添加了令人难以置信的功能来“容忍”糟糕的编程技巧。20 年前，你必须手动设置机器指令的执行顺序，而如今在硬件中 CPU 会为你做这些（例如，乱序执行）。在 GPU 世界中也能清晰地看到类似的趋势。由于 GPU 架构师正在改进硬件功能，5 年前我们在 GPU 编程中学习的大多数性能提升技术（例如，线程发散、共享存储体冲突以及减少原子操作的使用）正变得与改进的 GPU 架构越来越不相关，甚至 5 ～ 10 年后，即使是一名非常马虎的程序员，这些因素也会变得无关紧要。当然，这只是一个猜测。GPU 架构师可以做的事取决于晶体管总数及客户需求。当说晶体管总数时，是指 GPU 制造商可以将多少个晶体管封装到集成电路（IC）即“芯片”中。当说客户需求时，是指即使 GPU 架构师能够实现某个功能，但如果客户使用的应用程序不能从中受益，就意味着浪费了部分的晶体管数量。

从编写教科书的角度出发，我考虑了所有的因素，逐渐明确讲授 GPU 编程的最佳方式是说明不同系列 GPU（如 Fermi、Kepler、Maxwell 和 Pascal）之间的不同并指明发展趋势，这可以让读者准备好迎接即将到来的下一代 GPU，再下一代，……我会重点强调那些相对来说会长期存在的概念，同时也关注那些与平台相关的概念。也就是说，GPU 编程完全关乎性能，

如果你了解程序运行的平台架构，编写出了与平台相关的代码，就可以获得更高的性能。所以，提供平台相关的解释与通用的 GPU 概念一样有价值。本书内容的设计方式是，越靠后的章节，内容越具有平台特定性。

我认为本书最独特的地方就是通过第一部分中的 CPU 多线程来解释并行。第二部分介绍了 GPU 的大规模并行（与 CPU 的并行不同）。由于第一部分解释了 CPU 并行的方式，因此读者在第二部分中可以较为容易地理解 GPU 的并行。在过去的 6 年中，我设计了这种方法来讲授 GPU 编程，认识到从未学过并行编程课程的学生并不是很清楚大规模并行的概念。与 GPU 相比，“并行化任务”的概念在 CPU 架构中更容易理解。

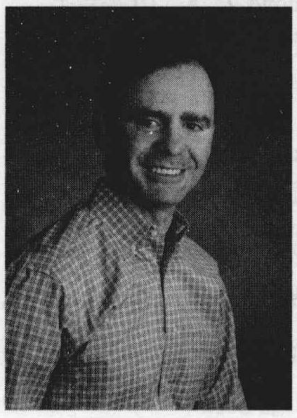
本书的组织如下。第一部分（第 1 章至第 5 章）使用一些简单的程序来演示如何将大任务分成多个并行的子任务并将它们映射到 CPU 线程，分析了同一任务的多种并行实现方式，并根据计算核心和存储单元操作来研究这些方法的优缺点。本书的第二部分（第 6 章至第 11 章）将同一个程序在多个 Nvidia GPU 平台（Fermi、Kepler、Maxwell 和 Pascal）上并行化，并进行性能分析。由于 CPU 和 GPU 的核心和内存结构不同，分析结果的差异有时很有趣，有时与直觉相反。本书指出了这些结果的不同之处，并讨论了如何让 GPU 代码运行得更快。本书的最终目标是让程序员了解所有的做法，这样他们就可以应用好的做法，并避免将不好的做法应用到项目中。

尽管第一部分和第二部分已经完全涵盖了编写一个好的 CUDA 程序需要的所有内容，但总会有更多需要了解的东西。本书的第三部分为希望拓宽视野的读者指明了方向。第三部分并不是相关主题的详细参考文档，只是给出了一些入门介绍，读者可以从中获得学习这些内容的动力。这部分主要介绍了一些流行的 CUDA 库，比如 cuBLAS、cuFFT、Nvidia Performance Primitives 和 Thrust（第 12 章）；OpenCL 编程语言（第 13 章）；使用其他编程语言和 API 库进行 GPU 编程，包括 Python、Metal、Swift、OpenGL、OpenGL ES、OpenCV 和微软 HLSL（第 14 章）；深度学习库 cuDNN（第 15 章）。

书中代码的下载地址为：https://www.crcpress.com/GPU-Parallel-ProgramDevelopment-Using-CUDA /Soyata/p/book/9781498750752。

Tolga Soyata

About the Author 关于作者



Tolga Soyata 于 1988 年在伊斯坦布尔技术大学电子与通信工程系获得学士学位，1992 年在美国马里兰州巴尔的摩的约翰·霍普金斯大学电气与计算机工程系（ECE）获得硕士学位，2000 年在罗切斯特大学电气与计算机工程系获得博士学位。2000 年至 2015 年间，他成立了一家 IT 外包和复印机销售 / 服务公司。在运营公司的同时，他重返学术界，在罗切斯特大学电气与计算机工程系担任研究员。之后，他成为助理教授，并一直担任电气与计算机工程系教职研究人员至 2016 年。在罗切斯特大学电气与计算机工程系任职期间，他指导了三名博士研究生。其中两人在他的指导下获得博士学位，另一位在他 2016 年加入纽约州立大学奥尔巴尼分校担任电气与计算机工程系副教授时留在了罗切斯特大学。Soyata 的教学课程包括大规模集成电路、模拟电路以及使用 FPGA 和 GPU 进行并行编程。他的研究兴趣包括信息物理系统、数字健康和高性能医疗移动云计算系统等。

Tolga Soyata 从 2009 年开始从事 GPU 编程的教学，当时他联系 Nvidia 将罗切斯特大学认证为 CUDA 教学中心（CTC）。在 Nvidia 将罗切斯特大学认证为教学中心后，他成为主要负责人。之后，Nvidia 还将罗切斯特大学认证为 CUDA 研究中心（CRC），他也成为项目负责人。Tolga Soyata 在罗切斯特大学担任这些计划的负责人直到他于 2016 年加入纽约州立大学奥尔巴尼分校。这些计划后来被 Nvidia 命名为 GPU 教育中心和 GPU 研究中心。在罗切斯特大学期间，他讲授了 5 年 GPU 编程和高级 GPU 项目开发课程，这些课程同时被列入电气与计算机工程系以及计算机科学与技术系的课程体系。自 2016 年加入纽约州立大学奥尔巴尼分校以来，他一直在讲授类似的课程。本书是他在两所大学讲授 GPU 课程的经验结晶。

目　录 *Contents*