TURING 图灵程序设计丛书

Packt

Python High Performance
Second Edition

Python 高性能

（第2版）

[加] 加布丽埃勒·拉纳诺 著　袁国忠 译

利用并发和分布式处理技术构建高性能、可伸缩的Python应用程序

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

Python High Performance
Second Edition

Python高性能

（第2版）

[加] 加布丽埃勒·拉纳诺 著
袁国忠 译

人民邮电出版社
北 京

图书在版编目（CIP）数据

Python高性能 : 第2版 / (加) 加布丽埃勒·拉纳诺 (Gabriele Lanaro) 著 ; 袁国忠译. -- 北京 : 人民邮电出版社, 2018.8
（图灵程序设计丛书）
ISBN 978-7-115-48877-0

Ⅰ. ①P… Ⅱ. ①加… ②袁… Ⅲ. ①软件工具－程序设计 Ⅳ. ①TP311.561

中国版本图书馆CIP数据核字(2018)第155441号

内 容 提 要

本书主要介绍如何让 Python 程序发挥强大性能，内容涵盖针对数值计算和科学代码的优化，以及用于提高 Web 服务和应用响应速度的策略。具体内容有：基准测试与剖析、纯粹的 Python 优化、基于 NumPy 和 Pandas 的快速数组操作、使用 Cython 获得 C 语言性能、编译器探索、实现并发性、并行处理、分布式处理、高性能设计等。

本书适合 Python 开发人员阅读。

◆ 著 [加] 加布丽埃勒·拉纳诺
译 袁国忠
责任编辑 岳新欣
责任印制 周昇亮

◆ 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
三河市君旺印务有限公司印刷

◆ 开本：800×1000 1/16
印张：12.25
字数：280千字 2018年8月第1版
印数：1－3 000册 2018年8月河北第1次印刷
著作权合同登记号 图字：01-2017-8618号

定价：59.00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：京东工商广登字 20170147 号

加布丽埃勒·拉纳诺

(Gabriele Lanaro)

数据科学家、软件工程师，对机器学习、信息检索、数值计算可视化、Web开发、计算机图形学和系统管理有浓厚的兴趣。开源软件包chemlab和chemview的开发者。现就职于Tableau软件公司。

微信连接

回复“Python”查看相关书单

微博连接

关注@图灵教育 每日分享IT好书

QQ连接

图灵读者官方群I：218139230
图灵读者官方群II：164939616

图灵社区
iTuring.cn

在线出版，电子书，《码农》杂志，图灵访谈

版权声明

前　言

最近几年，Python 编程语言的人气急剧上升，其直观而有趣的语法及大量质量上乘的第三方库居功至伟。很多大学的编程入门和进阶课程，以及科学和工程等数值密集型领域，都选择将 Python 作为编程语言，它还被用于编写机器学习应用程序、系统脚本和 Web 应用程序。

大家普遍认为，Python 解释器参考版 CPython 比 C、C++和 Fortran 等低级语言效率低下。CPython 之所以性能糟糕，是因为程序指令没有编译成高效的机器码，而是由解释器处理。虽然使用解释器有些优点，如可移植性以及可省略编译步骤，但在程序和机器之间增加了一个间接层，降低了执行效率。

多年来，已制定出很多克服 CPython 性能缺点的策略。本书旨在填补这方面的空白，介绍如何让 Python 程序的性能始终强劲。

本书介绍如何优化数值计算和科学代码，还涵盖了缩短 Web 服务和应用程序响应时间的策略，这些对很多读者都极具吸引力。

本书可按顺序从头到尾地阅读，但其中的每章也自成一体，所以如果你已熟悉前面的主题，可直接跳到感兴趣的部分。

涵盖的内容

第 1 章介绍如何评估 Python 程序的性能，以及找出并隔离速度缓慢代码的实用策略。

第 2 章讨论如何使用 Python 标准库和第三方 Python 模块提供的高效数据结构和算法来缩短程序的执行时间。

第 3 章提供了 NumPy 和 Pandas 包的使用指南。掌握这些包后，你就可使用简洁而富有表达力的接口来实现快速的数值算法。

第 4 章是一个 Cython 教程，这种语言使用与 Python 兼容的语法来生成高效的 C 语言代码。

第 5 章介绍可用来将 Python 代码编译成高效机器码的工具。在该章中，你将学习如何使用

Numba 和 PyPy，其中前者是一个 Python 函数优化编译器，而后者是一个能够动态地执行并优化 Python 程序的解释器。

第 6 章提供了异步编程和响应式编程指南。你将学习重要的术语和概念，以及如何使用框架 `asyncio` 和 RxPy 编写整洁的并发代码。

第 7 章简要地介绍多核处理器和 GPU 并行编程。在该章中，你将学习如何使用模块 `multiprocessing` 以及 Theano 和 Tensorflow 来实现并行性。

第 8 章是前一章内容的延伸，专注于在分布式系统上运行并行算法来解决大型问题和大数据处理问题。该章还介绍了 Dask、PySpark 和 `mpi4py` 库。

第 9 章讨论通用的优化策略，以及开发、测试和部署高性能 Python 应用程序的最佳实践。

需要什么

本书的示例代码都在 Ubuntu 16.04 系统中使用 Python 3.5 进行了测试，但这些示例大都能够在 Windows 和 Mac OS X 操作系统上运行。

推荐使用 Anaconda 发行包来安装 Python 和相关的库，这个发行包有用于 Linux、Windows 和 Mac OS X 的版本，可从 https://www.continuum.io/downloads 下载。

为谁而写

本书适合想要改善应用程序的性能并掌握了 Python 基本知识的 Python 程序员阅读。

排版约定

为将不同类型的信息区分开来，本书使用了很多文本样式。下面列出其中一些样式及其含义。

正文中的代码、数据库表名、用户输入，使用如下样式："总之，我们将实现一个名为 `ParticleSimulator.evolve_numpy` 的方法，并使用基准测试将其同纯粹的 Python 版本（更名为 `ParticleSimulator.evolve_python`）进行比较。"

代码块使用如下样式：

```
def square(x):
return x * x

inputs = [0, 1, 2, 3, 4]
outputs = pool.map(square, inputs)
```

要让你注意代码块的特定部分时，相关的代码行用粗体表示：

```
def square(x):
return x * x

inputs = [0, 1, 2, 3, 4]
outputs = pool.map(square, inputs)
```

命令行输入或输出使用如下样式：

```
$ time python -c 'import pi; pi.pi_serial()'
real 0m0.734s
user 0m0.731s
sys 0m0.004s
```

新术语和重要词语使用黑体字。

此图标表示警告或重要的注意事项。

此图标表示提示和技巧。

读者反馈

欢迎提供反馈，请将你对本书的看法告诉我们：哪些方面是你喜欢的，哪些方面你不喜欢。读者的反馈对我们来说很重要，因为这可帮助我们推出可最大限度发挥其功效的著作。

要给我们提供反馈，只需向 feedback@packtpub.com 发送电子邮件，并在主题中注明书名。

如果你有擅长的主题，并有志于写书或撰稿，请参阅 www.packtpub.com/authors 的撰稿指南。

客户支持

购买英文版图书后，你将获得各种帮助，让你购买的图书最大限度地发挥其功效。

下载示例代码

你可访问 http://www.packtpub.com 并使用你的账户下载本书的代码示例文件。如果你是在其他地方购买的本书，可访问 http://www.packtpub.com/support 并注册，以便我们将文件通过电子邮件发送给你。

要下载代码文件，可采取如下步骤。

(1) 访问我们的网站，使用电子邮件地址和密码注册并登录。
(2) 将鼠标指向页面顶部的标签 SUPPORT。
(3) 单击 Code Downloads & Errata。
(4) 在搜索框中输入书名。
(5) 选择要下载哪本书的代码文件。
(6) 从下拉列表中选择该书是在哪里购买的。
(7) 单击 Code Download。

下载文件后，使用下列软件的最新版解压缩：

- WinRAR / 7-Zip（Windows）；
- Zipeg / iZip / UnRarX（Mac）；
- 7-Zip / PeaZip（Linux）。

本书的示例代码还托管在 GitHub 上（https://github.com/PacktPublishing/Python-High-Performance-Second-Edition）。我们还在 https://github.com/PacktPublishing/提供了众多图书的示例代码以及视频，敬请访问！

勘误

我们万分小心，力图让图书的内容准确无误，即便如此，错误也在所难免。如果你在出版的图书中发现错误（无论是正文还是代码中的错误），请告诉我们，我们将感激不尽。这样做将让其他读者免遭同样的挫折，还可帮助我们改进该书的后续版本。无论你发现什么错误，都请告诉我们。为此，你可访问 http://www.packtpub.com/submit-errata，输入书名，单击链接 Errata Submission Form，再输入你发现的错误的详情。[①]你提交的勘误得到确认后，将被上传到我们的网站或添加到既有的勘误列表中。

要查看已提交的勘误，请访问 https://www.packtpub.com/books/content/support，并在搜索框中输入书名，Errata 栏将列出你搜索的信息。

打击盗版

在网上发布盗版材料是个屡禁不绝的问题。在保护版权和许可方面，本社的态度非常严肃，如果你在网上看到本社作品的非法复制品，请马上把网址或网站名告诉我们，以便我们采取补救措施。

① 中文版可访问图灵社区本书主页 www.ituring.com.cn/book/2006 提交勘误。——编者注

请通过 copyright@packtpub.com 与我们取得联系，并提供你怀疑的盗版材料的链接。

对于你为保护我们的作者和提供有价值内容的能力提供的帮助，我们感激不尽。

问题

无论你有什么与本书相关的问题，都可通过 questions@packtpub.com 与我们联系，我们将竭尽全力去解决。

电子书

扫描如下二维码，即可购买本书电子版。

致　谢

感谢Packt出版社Vikas Tiwari等编辑的支持；感谢我的女朋友Harani忍受我长时间挑灯写作；感谢朋友们自始至终的陪伴和支持；还要感谢父母给我机会追求自己的理想。

最后，感谢百怡咖啡赋予我写作本书的动力。

目　录

第1章 基准测试与剖析

就提高代码速度而言，最重要的是找出程序中速度缓慢的部分。所幸在大多数情况下，导致应用程序速度缓慢的代码都只占程序的很小一部分。确定这些关键部分后，就可专注于需要改进的部分，避免将时间浪费于微优化。

通过**剖析**（profiling），可确定应用程序的哪些部分消耗的资源最多。**剖析器**（profiler）是这样一种程序：运行应用程序并监控各个函数的执行时间，以确定应用程序中哪些函数占用的时间最多。

Python提供了多个工具，可帮助找出瓶颈并度量重要的性能指标。本章将介绍如何使用标准模块`cProfile`和第三方包`line_profiler`，还将介绍如何使用工具`memory_profiler`剖析应用程序的内存占用情况。本章还将介绍另一个很有用的工具——KCachegrind，使用它能以图形化方式显示各种剖析器生成的数据。

基准测试程序（benchmark）是用于评估应用程序总体执行时间的小型脚本。本章将介绍如何编写基准测试程序以及如何准确地测量程序的执行时间。

本章介绍如下主题：

- 通用的高性能编程原则；
- 编写测试和基准测试程序；
- Unix命令`time`；
- Python模块`timeit`；
- 使用`pytest`进行测试和基准测试；
- 剖析应用程序；
- 标准工具`cProfile`；
- 使用KCachegrind解读剖析结果；
- 工具`line_profiler`和`memory_profiler`；
- 使用模块`dis`对Python代码进行反汇编。

1.1　设计应用程序

就设计高性能程序而言，最重要的是在编写代码期间不进行细微的优化。

“过早优化是万恶之源。”

——高德纳

在开发过程的早期阶段，程序的设计可能瞬息万变，你可能需要大规模地改写和重新组织代码。在此阶段，你需要对不同的原型进行测试，而不进行优化，这样可自由地分配时间和精力，确保程序能够得到正确的结果，同时具有灵活的设计。归根结底，谁都不想要一个运行速度很快但结果却不正确的应用程序。

优化代码时，必须牢记如下箴言。

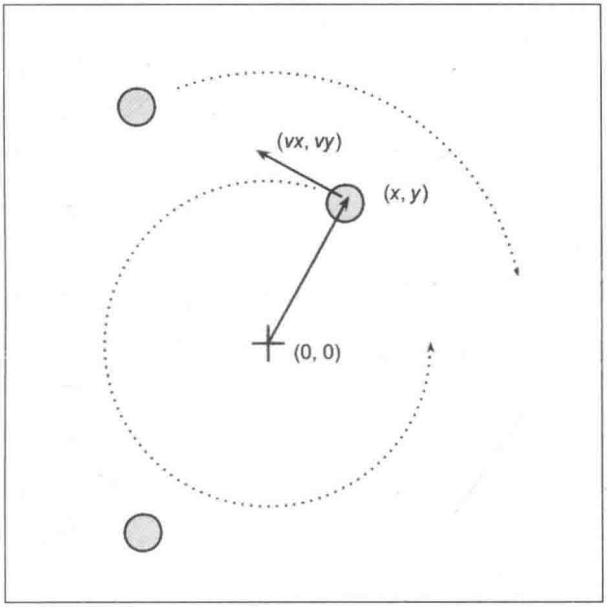
- **让它能够运行**：必须让软件能够运行，并确保它生成的结果是正确的。这个探索阶段让你能够对应用程序有更深入的认识，并在早期发现重大设计问题。
- **确保设计正确**：必须确保程序的设计是可靠的。进行任何性能优化前务必先重构，这可帮助你将应用程序划分成独立而内聚且易于维护的单元。
- **提高运行速度**：确保程序能够运行且结构优良后，就可专注于性能优化了。例如，如果内存消耗是个问题，你可能想对此进行优化。

在本节中，我们将编写一个**粒子模拟器**测试应用程序并对其进行剖析。这个**模拟器**程序接受一些粒子，并根据我们指定的规则模拟这些粒子随时间流逝的运动情况。这些粒子可能是抽象实体，也可能是真实的物体，如运动的桌球、气体中的分子、在太空中移动的星球、烟雾颗粒、液体等。

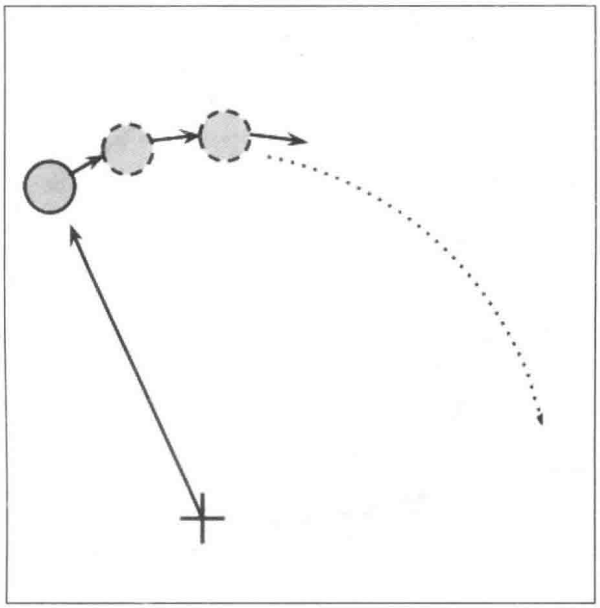
在物理、化学、天文学等众多学科中，计算机模拟都很有用。对用于模拟系统的应用程序来说，性能非常重要，因此科学家和工程师会花费大量时间来优化其代码。为了研究真实的系统，通常必须模拟大量的实体，因此即便是细微的性能提升也价值不菲。

在这个模拟系统示例中，包含的粒子以不同的速度绕中心点不断地旋转，就像钟表的指针一样。

为了模拟这种系统，需要如下信息：粒子的起始位置、速度和旋转方向。我们必须根据这些信息计算粒子在下一个时刻的位置。下图说明了这个系统，其中原点为(0, 0)，位置用向量 *x* 和 *y* 表示，而速度用向量 *vx* 和 *vy* 表示。



圆周运动的基本特征是，粒子的运动方向始终与其当前位置到中心点的线段垂直。要移动粒子，只需采取一系列非常小的步骤（对应于系统在很短时间内的变化），并在每个步骤中都根据粒子的运动方向修改其位置，如下图所示。



我们将以面向对象的方式设计这个应用程序。根据这个应用程序的需求，显然需要设计一个