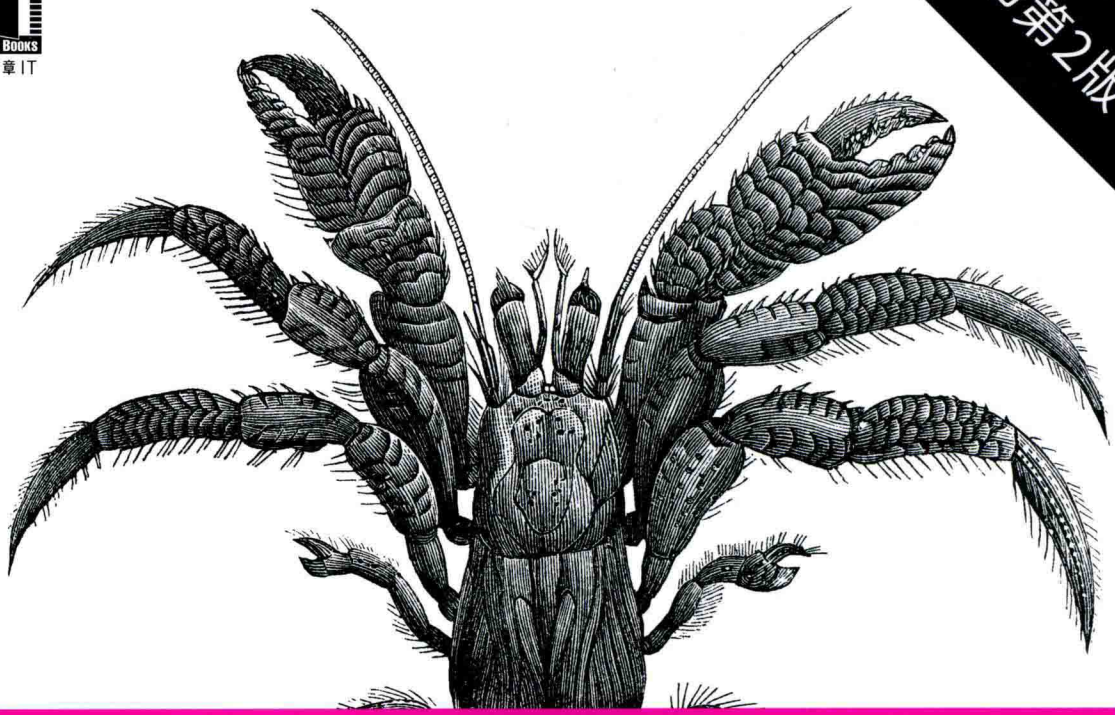
O'REILLY®

原书第2版

HZ BOOKS
华章IT



算法技术手册

Algorithms in a Nutshell, Second Edition

George T. Heineman
Gary Pollice Stanley Selkow 著
杨晨 曹如进 译

机械工业出版社
China Machine Press

原书第2版

算法技术手册

George T.Heineman　Gary Pollice
Stanley Selkow 著
杨晨　曹如进 译

Beijing • Boston • Farnham • Sebastopol • Tokyo　O'REILLY®
O'Reilly Media, Inc. 授权机械工业出版社出版

机械工业出版社

图书在版编目（CIP）数据

算法技术手册（原书第 2 版）/（美）乔治 T. 海涅曼（George T. Heineman）等著；杨晨，曹如进译 . －北京：机械工业出版社，2017.3

(O'Reilly 精品图书系列)

书名原文：Algorithms in a Nutshell, Second Edition

ISBN 978-7-111-56222-1

I. 算… II. ①乔… ②杨… ③曹… III. 电子计算机－算法理论－技术手册－高等学校－教材 IV. TP301.6-62

中国版本图书馆 CIP 数据核字（2017）第 040464 号

北京市版权局著作权合同登记

图字：01-2016-3780 号

书　　名 / 算法技术手册（原书第 2 版）

书　　号 / ISBN 978-7-111-56222-1

责任编辑 / 吴晋瑜

封面设计 / Randy Comer，张健

出版发行 / 机械工业出版社

地　　址 / 北京市西城区百万庄大街 22 号（邮政编码 100037）

印　　刷 / 三河市宏图印务有限公司

开　　本 / 178mm × 233mm　16 开本　22 印张

版　　次 / 2017 年 8 月第 1 版　2017 年 8 月第 1 次印刷

定　　价 / 89.00 元（册）

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

购书热线：(010)68326294；88379649；68995259

投稿热线：(010)88379604

读者信箱：hzit@hzbook.com

O'Reilly Media, Inc. 介绍

O'Reilly Media 通过图书、杂志、在线服务、调查研究和会议等方式传播创新知识。自 1978 年开始，O'Reilly 一直都是前沿发展的见证者和推动者。超级极客们正在开创着未来，而我们关注真正重要的技术趋势——通过放大那些“细微的信号”来刺激社会对新科技的应用。作为技术社区中活跃的参与者，O'Reilly 的发展充满了对创新的倡导、创造和发扬光大。

O'Reilly 为软件开发人员带来革命性的“动物书”；创建第一个商业网站（GNN）；组织了影响深远的开放源代码峰会，以至于开源软件运动以此命名；创立了 Make 杂志，从而成为 DIY 革命的主要先锋；公司一如既往地通过多种形式缔结信息与人的纽带。O'Reilly 的会议和峰会集聚了众多超级极客和高瞻远瞩的商业领袖，共同描绘出开创新产业的革命性思想。作为技术人士获取信息的选择，O'Reilly 现在还将先锋专家的知识传递给普通的计算机用户。无论是通过书籍出版，在线服务或者面授课程，每一项O'Reilly的产品都反映了公司不可动摇的理念——信息是激发创新的力量。

业界评论

“O'Reilly Radar 博客有口皆碑。”

——Wired

“O'Reilly 凭借一系列（真希望当初我也想到了）非凡想法建立了数百万美元的业务。”

——Business 2.0

“O'Reilly Conference 是聚集关键思想领袖的绝对典范。”

——CRN

“一本 O'Reilly 的书就代表一个有用、有前途、需要学习的主题。”

——Irish Times

“Tim 是位特立独行的商人，他不光放眼于最长远、最广阔的视野并且切实地按照 Yogi Berra 的建议去做了：‘如果你在路上遇到岔路口，走小路（岔路）。’回顾过去 Tim 似乎每一次都选择了小路，而且有几次都是一闪即逝的机会，尽管大路也不错。”

——Linux Journal

译者序

算法，神秘而晦涩的词汇。算法是计算机科学中最重要同时也是最基础的一环。在学习计算机之初，我们就深知算法是整个计算机科学的核心。然而，直至我们工作数年后，能够真正学好算法的却依旧是凤毛麟角。这并不是计算机教育的错，也不是计算机从业人员的错，更不是算法的错。长久以来，算法就像古老的咒语般难懂，其背后高深的数学知识更让人望而生畏。其实，我们始终没有找到一条从理论走向实践的路。

在这里，我们很高兴能向大家介绍本书。它是一本不可多得的好书，能够帮助你学好算法。

本书的三位作者是美国伍斯特理工学院的教授，其中 George T. Heineman 毕业于美国达特茅斯学院和哥伦比亚大学，曾经获得过 GE、IBM 和 AT&T 的研究奖金，在软件工程方面有独到的见解。而 Gary Pollice 曾经供职于 Rational Software、Sun 等多家巨头公司，在工业界有着丰富的经验，擅长将学术和工业结合起来。Stanley Selkow 毕业于美国卡内基梅隆大学和宾夕法尼亚大学，擅长图论和算法设计。

本书由伍斯特理工学院这三位计算机理论专家合著，其中展示了工业界和学术界对算法的不同看法，以及如何高效地将理论和实践结合起来。

本书可供本科生以及程序设计人员参考使用，也适用于产品和项目管理人员。由于译者的知识和经验有限，翻译中难免有疏漏或不足之处，敬请广大读者批评指正。

杨晨 曹如进

目录

前言

修订一本书向来都是一项艰巨的任务。我们既希望保留第1版（于2009年出版）中的精华，也希望弥补其中的一些不足并增加一些新的篇幅。在第2版中，我们延续了第1版中列出的原则，包括：

- 使用实际代码而非伪代码来描述算法。
- 将算法独立于解决的问题之外。
- 恰到好处地介绍数学知识。
- 以经验主导支撑数学分析。

在更新修订过程中，我们精简了文字描述，简化了一些布局，从而有助于补充新的算法和其他内容。我们相信，从*概括*的角度介绍计算机科学的一个重要领域，会对实用软件系统有着深远影响。

第2版的变动

在修订过程中，我们遵循以下原则：

挑选新的算法

在第1版出版之后，我们常常会收到一些留言，比如，“为什么漏掉了**归并排序**？”或“为什么没有介绍**快速傅里叶变换**（Fast Fourier Transform，FFT）？”虽然我们无法满足所有这些要求，但第2版还是增添了一些新的算法，包括：

- **Fortune算法**，它用于计算点集的Voronoi图。
- 归并排序，既包括针对内存数据的内部排序，也包括外部文件的外部排序。
- 多线程**快速排序**。

- AVL 平衡二叉树实现。
- 新的章节（第 10 章）用于介绍*空间算法*，包括**R 树**（R-Tree）和**四叉树**（Quadtree）。

总的来说，本书差不多介绍了 40 种核心算法。

简化描述

为了方便新增内容，我们几乎对第 1 版的所有内容进行了修订，简化了算法描述框架，并且减少了一些附带描述。

增加 Python 实现

我们并没有使用 Python 重新实现已有的算法，而是特意为大部分新增的算法提供了 Python 实现。

管理代码资源

第 1 版中的代码是通过 ZIP 压缩包文件的方式提供的。之后，我们就迁移到了 GitHub 代码库。这些年里，我们不仅提高了代码质量和增加了相关文档，还加入了在第 1 版出版后撰写的一些博客文章。代码库中不仅拥有超过 500 个的单元测试用例，还使用代码覆盖工具以确保 99% 的 Java 代码都被覆盖。目前代码库中的代码行数总计超过 11 万行。

目标读者

我们期望这本书能够成为读者的一本主要参考书，方便查阅如何实现和使用某些算法。书中了介绍了一系列用于解决问题的已有算法，并遵循以下的一些原则：

- 在介绍每种算法时，我们会使用一种固定格式的模板。这种模板可以帮助恰当地设计每一次的讨论和解释每种算法的要点。
- 我们使用了不同的语言实现了每种算法（包括 C、C++、Java 和 Python）。得益于此，我们能够使用读者熟悉的编程语言对算法进行详细的讨论。
- 我们描述了每种算法的预期性能，并根据经验加以证明。

我们希望这本书对软件工程师、程序员以及设计师有所帮助。为了实现目标，你需要使用大量关于实际解决方案和算法的资源，才能解决手头的实际问题。你已经知道了如何使用多种语言编写一个程序，也了解了计算机科学中的关键数据结构（例如数组、链表、栈、队列、散列表、二叉树以及有向图和无向图），但是你并不需要亲自实现这些数据结构，因为可以在代码库中找到它们。

我们希望，读者能从这本书中学习到如何选择和测试解决方案来快速高效地解决问题，同时也能学习到一些高级数据结构和使用标准数据结构的新方法来提高程序的性能。而解决问题的能力高低就取决于所选择算法的效率高低。

本书体例

在印刷上的一些例行惯例：

代码（Code）

所有代码示例都使用这种字体。

这些代码都是直接取自代码库，是现实中使用的代码。此外，书中所有代码清单都进行了“美化处理”以强调对应程序设计语言的语法。

斜体（Italic）

斜体用于表示描述算法和数据结构的关键术语，也会用于指代示例伪代码描述中的变量。

等宽字体（Constant Width）

等宽字体用于表示程序实现中的实际软件元素，例如Java 类、C 语言实现中的数组名以及常量（如 true 或 false）。

在本书中，我们引用了大量的书籍、文章和网址。这些引用都用括号标注出来，例如（Cormen 等，2009），并且在每章末尾都会列出本章所使用的参考文献。若参考引用紧跟在作者姓名之后，则不会重复其姓名。例如，当提到 Donald Knuth 的《Art of Computer Programming》一书时，括号中仅附带出版年份。

本书中的所有 URL 于 2016 年 1 月验证过，并且我们所选用的 URL 在近期均可用。除此之外，我们也用到了较短的 URL，如 *http://www.oreilly.com*，此类 URL 会直接出现在正文中，也会出现在脚注和每章末尾的参考文献中。

代码使用说明

补充资料（代码示例、练习题等）可以从 *https://github.com/heineman/algorithms-nutshell-2ed* 下载。

本书的目的是帮助读者更好地完成工作。通常来说，你可以在自己的程序和文档中直接使用本书提供的示例代码。除非需要大量复制这些代码，否则你不需要联系我们获得许可。例如，如果你所编写的程序使用了本书中几段代码，则无须获取许可。但是如果作销售或者发行光盘之用，则需要许可。引用本书和使用书中的样例代码回答问题也无须获取许可。但是如果你在自己的产品文档大量地使用本书中的代码，则需要许可。

我们希望但是不强制要求标明归属。一个归属说明通常包括标题、作者、出版商和 ISBN 。例如，“George T. Heineman、Gary Pollice 和 Stanley Selkow 编写的《Algorithms

in a Nutshell, Second Edition》。C2016 George Heineman、Gary Pollice and Stanley Selkow，978-1-4919-4892-7。”

如果你觉得使用代码的情况超出了我们所允许的范围，请与我们联系：*permissions@oreilly.com*。

联系我们

对于本书，如果有任何意见或疑问，请按照以下地址联系本书出版商。

美国：

O'Reilly Media，Inc.
1005 Gravenstein Highway North
Sebastopol，CA 95472

中国：

北京市西城区西直门南大街 2 号成铭大厦 C 座 807 室（100035）
奥莱利技术咨询（北京）有限公司

我们为本书提供了网页，该网页上面列出了勘误表、范例和任何其他附加的信息。您可以访问如下网页获得：

http://bit.ly/algorithms_nutshell_2e

要询问技术问题或对本书提出建议，请发送电子邮件至：

bookquestions@oreilly.com

要获得更多关于我们的书籍、会议、资源中心和O'Reilly网络的信息，请参见我们的网站：

http://www.oreilly.com
http://www.oreilly.com.cn

致谢

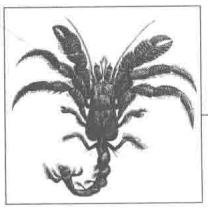
在此，我们真诚地感谢本书的审阅人员，感谢他们对于本书细节的关注以及提出的宝贵建议，这些建议有助于提高本书的质量。特别感谢在第 1 版中给予我们帮助的 Alan Davidson、Scot Drysdale、Krzysztof Duleba、Gene Hughes、Murali Mani、Jeffrey

Yasskin 和 Daniel Yoo，以及第 2 版中给予我们帮助的 Alan Solis、Robert P. J. Day 和 Scot Drysdale。

George Heineman 想要感谢那些带领他走入算法领域的人，包括 Scot Drysdale 教授（美国达特茅斯学院）和 Zvi Galil 教授（美国哥伦比亚大学，现任美国佐治亚理工学院计算学院系主任）。一如既往，George 也非常感谢他的妻子 Jennifer 和他的孩子们——Nicholas（他现在已经开始学习编程了）和 Alexander（他喜欢用这个版本的打印稿折纸玩）。

Gary Pollice 想要感谢他的妻子 Vikki 陪伴他走过 46 年的美好时光。他同样感谢 WPI 计算机科学系提供给他的优质工作和幽雅环境。

Stanley Selkow 想要感谢他的妻子 Deb。这本书是在他们相伴走过的时光中的又一个结晶。



第 1 章
用算法的眼光去看问题

算法真的是非常重要！算法的选择会对软件的性能产生巨大影响。本书将指导你学习一系列的算法，例如搜索和排序，还会介绍一些算法经常采用的策略，例如分治或贪心。希望从此之后，你能够灵活运用这些算法来改善软件的性能。

自“计算”出现之后，数据结构便一直与算法紧密相关。本书将介绍一些基本的数据结构，并展示如何使用这些数据结构来高效地处理数据。

“在选择算法时，你需要做什么？”我们将在接下来的章节中探究这个问题的答案。

1.1 理解问题

设计算法的第一步就是理解将要解决的问题。以一个计算几何领域的简单问题为例：给定一个二维点集 P（见图 1-1），想象着用一个橡皮圈把这些点围起来，松开之后，这个橡皮圈的形状就是*凸包*（能够把点集 P 所有的点全部包围起来的最小凸形）。那么，现在的任务就是编写一个算法，针对一个给定的二维点集，计算它的凸包。

仔细观察点集 P 的凸包就会发现，连接 P 中任何两点的线段都位于这个凸包内。假设，我们把凸包中的点按顺时针排好，不难发现，这个凸包其实是由 h 个顺时针排列的点 L_0, L_1, ..., L_{h-1} 组成，如图 1-2 所示。每三个相邻的点 L_i, L_{i+1}, L_{i+2} 组成一个向右拐的线条。

有了这样的信息，也许你可以手工画出任意点集的凸包，但是否能写出一个算法呢（给出一系列循序渐进的指令，并且能够高效地计算出任意点集的凸包）？

有趣之处在于，这个凸包问题貌似并不属于任何众所周知的算法范畴。而且，似乎也并没有任何线性时间的排序算法能够按照从左到右顺时针对点进行排序。与此类似，要找到凸包上的一条线段并不难，因为这条线段一定会使得平面上剩余的 n-2 个点都位于其“右侧”，但也没有显而易见的搜索算法可用。

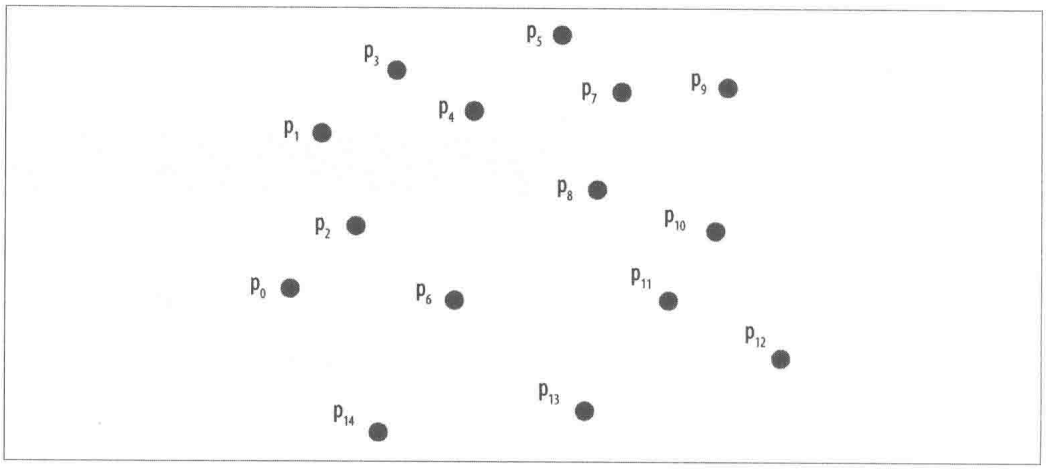


图 1-1：平面上由 15 个点组成的示例点集

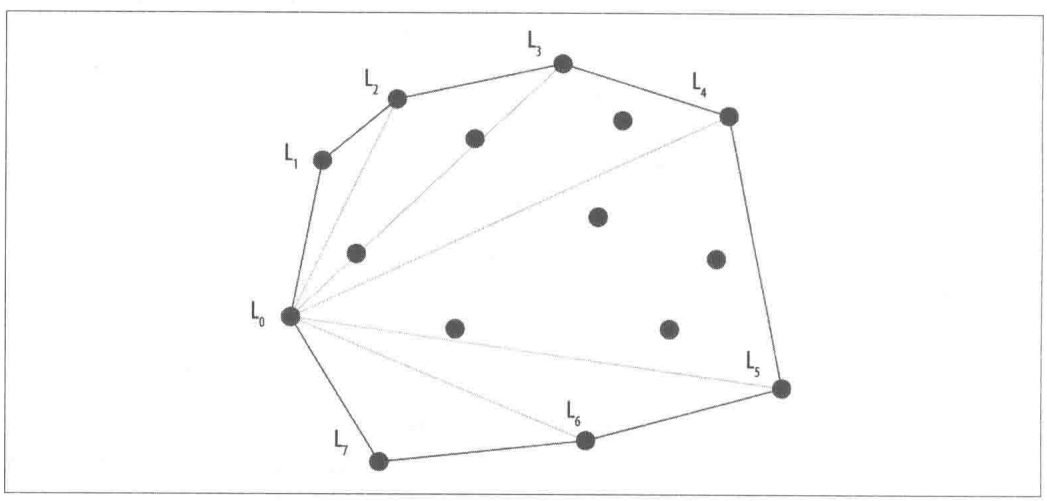


图 1-2：图 1-1 中点的凸包

1.2 简单解法

很显然，任意包含三个或三个以上点的点集都肯定有凸包。但是如何构建一个凸包呢？可以这么考虑，从上述集合中选择任意三个点组成一个三角形，如果剩余的 $n-3$ 个点中的任意一个位于该三角形之内，那么这些位于内部的点是不可能成为凸包的一部分的。我们可以用伪代码描述大致流程。本书其他章节也会采用类似的伪代码来描述算法。