

<<算法之道(第2版)>>

图书基本信息

书名：<<算法之道(第2版)>>

13位ISBN编号：9787111370505

10位ISBN编号：7111370503

出版时间：2012-4-20

出版时间：机械工业出版社华章公司

作者：邹恒明

页数：343

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<算法之道(第2版)>>

前言

前言：起初神创造天地。

地是空虚混沌，渊面黑暗；神的灵运行在水面上。

神说：“要有光。

”就有了光。

神看光是好的，就把光与暗分开了。

神称光为昼，暗为夜。

有晚上，有早晨，这是头一日。

……神就照着自己的形象造人。

……神说：“看啊！

我将遍地上一切结种子的菜蔬和一切树上所结有核的果子，全赐给你们做食物。

至于地上的走兽和空中的飞鸟，以及各样在地上爬的有生命的物，我将青草赐给它们做食物。

”事就这样成了。

神看着一切所造的都甚好。

有晚上，有早晨，是第六日。

天地万物都造齐了。

6天圣经上写着：神6天创造天地万有，第7日安歇。

对于神创论者来说，这是不可怀疑的事实。

但对于进化论者来说，6天创造一切根本就不可能。

作为一本算法书，我们当然不打算加入神创论和进化论者的永无休止的争论当中去。

我们关心的是这么一个问题：圣经上为什么给出的是6天，而不是其他的时间长度。

不管是神创论者还是进化论者，弄清楚6这个数字的来历很可能对己方的观点有所帮助。

在这6天里，神将他的创作方程式重复了6次，每天1次。

对于全能的神来说，他完全可以在1天、1秒或者任何他所愿意的时间长度里创造天地万物，但却为什么是不多不少的6天呢？

而不管圣经上的“1天”是多长，这个问题都是值得讨论的。

我们知道，任何一个自然数的约数中都有1和它本身，而所有小于它本身的因数叫做这个自然数的真约数。

例如，6的所有真约数是1、2、3；数字8的真约数是1、2、4。

如果一个数的真约数之和等于这个自然数本身，则这个自然数就称为完全数，或者完美数。

例如， $6=1+2+3$ ，因此6是完美数；而 $8(1+2+4)$ ，因此8不是完美数。

因此，神6天创造世界，暗示着该创造是完美的！

以完美数来昭示创造的完美，似乎合情合理。

但问题是，完美数只有6这一个数吗？

如果不是，为什么不使用其他的完美数呢？

答案是，完美数虽然不只有6这一个，但确实数量稀少。

一直到现在(2009年6月)，数学家们探索了2600年，并且现代数学家们还借助了超级计算机的帮助，但也仅仅找到了47个完美数。

其中第1个完美数是6，接下来的4个完美数分别是：28、496、8128、33550336。

而第47个完美数有25956377个数位(注意，是数位，不是数值)，它的数值为： $243112608 \times (243112609^{?}1)$

完美数的稀少昭示着达到完美的难度，而神选择6天来创造天地万有也许是因为6是最小的完美数，即创造天地万有对于神来说是轻而易举的一件事情……完美与算法完美数由于其各种神秘属性(真约数之和等于自身只是其中的一种性质)而受到了特殊的关注。

但到底哪些数是完美数则不是一件容易判断的事情。

显然，按照完美数的定义，判断一个数是否是完美数的不二法则是找出它的所有真约数，然后求和看

<<算法之道(第2版)>>

看其是否等于自身。

而这种方法效率太过低下，因为这意味着因式分解，而这是十分困难的(本书后面将会讨论到这个问题)。

如果判断一个数是否是完美数就已经非常困难，那么要找出所有的完美数则更是一个难上加难的任务。

因为这就意味着将所有的数进行上面描述的判断验证：因式分解。

这似乎是人类不可能完成的任务。

即使用世界上超快的计算机来进行计算，情况也不会有任何数量级的改善。

显然，我们需要新的解决方案，而不是发明或使用新的计算工具！

研究这样的解决方案就可以归结到算法的范畴里，因为如何高效地解决问题正是算法要研究的核心课题。

有意思的是，判断和搜索完美数是算法的研究范畴，而算法本身的追求却也是“完美”。

算法无处不在如果你觉得算法只是用来研究解决找出完美数之类的“漫无边际的问题”，那就大错特错了。

也许算法这个名词听上去很抽象，让人联想不到任何具体的物体。

也许你会觉得算法与自己的生活并无太多关系，它只不过存在于那些闲得无聊的数学家或计算机专业人士的脑海中。

但事实真是这样吗？

当然不是。

如果我们告诉你算法就是解决各种问题的方法，你就不会觉得它太抽象，与生活无关了吧。

事实是，算法无处不在。

每个人每天都在使用不同的算法来活出自己的人生，比如你去食堂买饭会选择一个较短的队列，而有人则可能选择一个推进速度更快的队列。

每天起床后，你可能先读一会儿书，再去吃早饭；另外一个人则可能先去吃早饭，然后再看书。

所有这些行为都是算法或算法一部分的体现。

也许运行这些算法并不在你的思想意识里，也许你并不知道算法在帮助自己的生活，但它确实存在。

这些算法也许没有经过精心设计，没有经过仔细分析，但它还是算法！

2009年7月23日下午，我在游览云南省大理市的蝴蝶泉时由于泉水边的石头很滑，在用泉水洗手时(导游金花说用该泉水洗手会带来好运)不慎滑落到蝴蝶泉水(见图3)里面全身湿透。

(据说一天至多只会有一人滑落到泉里，可见我的运气不错！

看来“蝴蝶泉边好梳妆”的歌词也许应该改为“蝴蝶泉里好冲凉”。

)泉水冰冷透凉，而大理的气温又低。

这样，我就面临一个是否更换全身衣服的选择。

问题是，旅游团需要马上赶去登游船游览洱海风光，而若找地方或者回旅店换衣服就将赶不上游船。

如何处理这件事情就是一个算法问题：是先上游船再在船上找地方换衣服，还是找个地方换衣服而放弃游览苍山洱海。

显然不同的算法有着不同的收益和代价。

如果能够在游船上找到合适的地方更换衣服，则采用先上游船再换衣服的算法为佳；否则就是放弃游览的算法更好，因为如果冻病了显然就不划算了。

最后，我选择了在游船上更换衣服的算法：在游船上找到了一个贵宾室更衣。

算法由问题驱动算法的发现总是由相关的问题驱动的。

拿排序来说，因为生活中到处都充满次序，每个人都要接受自己在某个次序里的位置。

比如，各种排名、评优、民意调查等，最后的结果都体现为一个次序！

看来，“没有次序无以成方圆”并不是空穴来风！

而谈到排序用的方法，人们很自然地想到了插入法。

因为这种朴素的算法和人的思维方式非常类似：它就是人们打牌时整理手中扑克牌的算法。

但是随着数据量的增多，插入排序的效率缺陷迅速变为人们无法容忍的缺点。

<<算法之道(第2版)>>

于是人们发明了归并排序、堆排序、快速排序等，这些排序的方法大大改善了速度，但是人们却并不满足于此，因此又发明了效率更高的线性排序。

表1给出的是各种排序算法平均情况下的效率比较：最上面一行的数字代表输入的规模，如10表示一共有10个数据项，1M表示一共有100万个数据项。

其他格子里面的数据为相应算法在相应输入规模下完成排序所需要的时间，单位为毫秒。

所有输入数据为随机产生。

一个个新的算法都是为了解决前面算法遗留的问题而产生的。

从表1里的数字可以看出，一般来说，随着新的算法出现，排序效率在不断提高。

不过，虽然每个算法似乎解决了前面算法的遗留问题，但新的问题也会被有意或无意地引入。

例如，线性排序虽然将排序的时间复杂性降低到线性级，但各种前提条件极大地限制了其应用范围。

也许这就是算法永远也不能或不会停止发展的一个原因吧。

算法是计算机的灵魂因为人不是全能的，一个时刻只能做一件事情，所以做事情就要有一个步骤。

由于算法要满足人的这种特性，因此它通常表示为一个做事情的行为序列。

因此，从一般意义上说，算法就是求解问题的步骤。

由于计算机的计算操作完全是一步一步进行，因此算法的上述性质用于计算机是再合适不过了，可以说算法弥漫在计算机的一切行为上。

如果说操作系统是计算机的心智，那么算法就是计算机的灵魂。

理解灵魂当然不是一件容易的事情，由于它高度抽象与简洁，许多学生都望而却步。

先看一个纸牌魔术：1)任选一位观众将一副扑克牌充分洗好。

2)背对观众，请观众随机抽出一张牌，记住牌面，然后将这张牌放回整副牌的最上面。

3)接过牌后，洗牌几次。

洗牌的时候保持最上面一张牌不动。

4)对观众说：“我来教你魔法，只要吹一口气，就能把刚才你抽的牌吹到任意位置上”。

5)请观众说出一个数字，比如说10，然后一边吹气，一边想着刚才说的数字10。

6)在吹完气后，请观众一张一张地将上面的牌取出放在桌上。

7)到第10张时，将牌翻开，发现并不是其原来抽的牌。

8)接回整副牌，并把上一个步骤里取出堆放在桌上的牌收起，仍放在整副牌的最上面。

9)然后洗牌几次，洗牌的时候保持上面放回来的那堆牌不动。

10)从观众手上拿回刚才翻开的那张牌，插入最上面9个位置中的任意一个。

11)对观众说：“你刚才不是在想着那个数字的时候吹的气，而是在吹气的时候想着那个数字，而这是完全不同的两回事。

我现在演示如何吹气。

”对着牌吹一口气。

12)请观众从上到下数牌，到第10张时翻开。

13)这张翻开的牌就是观众一开始抽的那张牌。

读者看明白了上面的这个魔术了吗？

这里面隐藏着一个算法。

如果看懂了就可以在朋友面前一显身手了。

当然，如果没有看懂也没有什么关系。

算法本来就不是轻易让人看懂的嘛。

如果这些问题读者都能回答，那么恭喜你。

看来算法分析对于读者来说将是件很容易的事情，不过可能也不一定。

如果你回答不出这些问题，不用担心，因为回答诸如此类的问题就是本书的目的。

当然了，本书回答的远不止这么几个简单问题，而是会阐述更重要的算法精髓：算法思想、战略和分析！

<<算法之道(第2版)>>

内容概要

本书追求的目标是算法背后的逻辑，是一本启示书，而不是一本包罗万象的算法大全。因此，本书甄选了那些最能展现算法思想、战略和精华，并能够有效训练算法思维的内容。本书将算法的讨论分为五篇：算法基础篇、算法设计篇、算法分析篇、经典算法篇、难解与无解篇。每篇分别讨论算法的一个方面：基础、设计、分析、经典和难解问题。第2版还对进程调度问题、跳转表问题、概率分析应用、遗传算法等方面进行了论述。

本书既可以作为大学本科或研究生的算法教材或参考书，也可以作为对算法有兴趣的读者提升认知深度的读物。

<<算法之道(第2版)>>

作者简介

邹恒明，美国密歇根大学（University of Michigan-Ann Arbor）计算机科学与工程博士、中国科学院计算技术研究所硕士、华中科技大学计算机科学与技术学士。
曾先后在美国IBM、美国国家数据公司、美国朗讯和美国EMC公司任职8年多。
现为上海交通大学教授。

<<算法之道(第2版)>>

书籍目录

前言

第一篇 算法基础篇

第1章 从无有到无穷

- 1.1 意念与现实
- 1.2 什么是算法
- 1.3 算法的表示
- 1.4 算法之魂
- 1.5 如何比较速度
- 1.6 算法与计算机的关系
- 1.7 算法的范畴
- 1.8 为什么学习算法

思考题

第2章 计数与渐近

- 2.1 算法的分析
 - 2.1.1 正确性分析
 - 2.1.2 时空效率分析
 - 2.1.3 时空特性分析
- 2.2 计数：算法分析的核心
- 2.3 算法设计
- 2.4 算法效率表示
- 2.5 渐近分析
- 2.6 表示
- 2.7 最好、最坏、平均
- 2.8 另一类定义
- 2.9 性质
- 2.10 要更快的计算机还是要更快的算法

思考题

第3章 分治与递归

- 3.1 分而治之为上策
- 3.2 分治策略
- 3.3 递归表达式求解
 - 3.3.1 递归树法
 - 3.3.2 替换解法
 - 3.3.3 大师解法
- 3.4 分治策略举例1：乘方运算
- 3.5 生命中不能承受之重：矩阵乘法
- 3.6 魔鬼序列：斐波那契序列
 - 3.6.1 由底至上
 - 3.6.2 使用通式
 - 3.6.3 使用矩阵乘方
- 3.7 VLSI 布线
- 3.8 多项式乘法
- 3.9 分治就在潜意识

思考题

第二篇 算法设计篇

<<算法之道(第2版)>>

第4章 动态规划思想

- 4.1 什么是动态规划
- 4.2 流水线问题
- 4.3 最长公共子序列
 - 4.3.1 第一种解法：蛮力策略
 - 4.3.2 第二种解法：动态规划
- 4.4 最长公共子序列变种
- 4.5 记忆递归法
- 4.6 空间效率改善
- 4.7 最优二叉搜索树
 - 4.7.1 递归解法
 - 4.7.2 计算最优答案
- 4.8 最优子结构与重叠子问题
 - 4.8.1 最优子结构
 - 4.8.2 重叠子问题
- 4.9 动态规划与静态规划的关系
- 4.10 动态规划与静态规划的相互转换

思考题

第5章 贪婪选择思想

- 5.1 仅有动态规划是不够的
- 5.2 什么是贪婪
- 5.3 背包问题
- 5.4 贪婪选择属性
- 5.5 教室规划问题
- 5.6 最小生成树
 - 5.6.1 Kruskal算法的正确性
 - 5.6.2 Kruskal算法的时间分析
- 5.7 Prim算法
- 5.8 霍夫曼树和霍夫曼编码
 - 5.8.1 霍夫曼树
 - 5.8.2 霍夫曼编码
 - 5.8.3 霍夫曼编码的无前缀编码性质
- 5.9 进程调度问题
- 5.10 贪婪选择属性
- 5.11 标准分治、动态规划和贪婪选择的比较

思考题

第6章 随机化思想

- 6.1 为什么要随机化
- 6.2 随机的平方
- 6.3 什么是随机化算法
- 6.4 拉斯维加斯算法
- 6.5 蒙特卡罗算法
- 6.6 素性测试
- 6.7 矩阵乘积验证器
- 6.8 随机化最小生成树算法
 - 6.8.1 Karger-Klein-Tarjan算法
 - 6.8.2 结点降低算法

<<算法之道(第2版)>>

6.8.3 线性时间最小生成树算法

6.8.4 线性时间最小生成树算法的时间成本分析

6.9 随机数的生成

6.10 随机化算法的应用

思考题

第三篇 算法分析篇

第7章 概率分析

7.1 一切都在概率中

7.2 什么是概率分析

7.3 梦幻情人的代价

7.3.1 直接分析

7.3.2 最坏情况分析

7.3.3 最好情况分析

7.3.4 平均情况分析

7.3.5 平均情况下成本的概率分析

7.3.6 概率分析结果的有效性

7.3.7 正确概率分析的保障

7.4 梦幻情人的概率

7.5 随机排列问题

7.6 跳转表问题

7.6.1 跳转表插入操作

7.6.2 随机化跳转表构建算法

7.7 南柯一梦：从无穷到无有

7.8 概率分析的其他应用

思考题

第8章 摊销分析

8.1 什么是摊销分析

8.2 摊销分析与数据结构

8.3 摊销分析的几种方法

8.4 聚类分析

8.4.1 栈操作的聚类分析

8.4.2 二进制计数器的聚类分析

8.5 会计分析

8.6 势能分析

8.6.1 栈操作的势能分析

8.6.2 二进制计数器的势能分析

8.7 摊销分析应用：表格扩展的代价

8.7.1 动态表插入操作的聚类分析

8.7.2 动态表插入操作的会计分析

8.7.3 动态表插入操作的势能分析

8.8 运气不好就摊销

思考题

第9章 竞争分析

9.1 什么是竞争分析

9.2 在线算法和离线算法

9.3 竞争力

9.4 健忘对手和优良对手

<<算法之道(第2版)>>

- 9.5 线性表更新问题
- 9.6 前置移动算法的竞争分析
- 9.7 聚类问题
 - 9.7.1 聚类问题的次优解算法
 - 9.7.2 CLUSTERING-ALGORITHM算法的竞争分析
- 9.8 竞争分析与普通算法分析

思考题

第四篇 经典算法篇

第10章 排序与次序

- 10.1 排序无处不在
- 10.2 插入排序
 - 10.2.1 插入排序的效率分析
 - 10.2.2 折半插入排序
- 10.3 归并排序
- 10.4 快速排序
 - 10.4.1 快速排序的过程
 - 10.4.2 快速排序的时间复杂性分析
 - 10.4.3 最坏情况分析
 - 10.4.4 最好情况分析
 - 10.4.5 平均情况分析
- 10.5 随机化快速排序
- 10.6 排序的下限
- 10.7 线性排序
- 10.8 计数排序
- 10.9 基数排序
 - 10.9.1 基数排序的正确性
 - 10.9.2 基数排序的时间效率分析
- 10.10 桶排序
 - 10.10.1 桶排序的定义
 - 10.10.2 桶排序的正确性
 - 10.10.3 桶排序的时间复杂性分析
- 10.11 次序选择
- 10.12 快速次序选择算法
- 10.13 随机快速次序选择算法
- 10.14 最坏情况下的线性选择算法
 - 10.14.1 杠杆点好坏分析
 - 10.14.2 算法时间复杂性分析

思考题

第11章 搜索与散列

- 11.1 搜索问题
- 11.2 顺序搜索
- 11.3 折半搜索
- 11.4 常数搜索
- 11.5 散列搜索
- 11.6 散列函数选择
 - 11.6.1 直接散列
 - 11.6.2 除法(模除法)散列

<<算法之道(第2版)>>

- 11.6.3 乘法散列
- 11.6.4 乘法散列的赌徒原理
- 11.6.5 乘方取中法
- 11.7 散列算法的碰撞问题
 - 11.7.1 开放寻址散列
 - 11.7.2 开放寻址散列的时间成本
 - 11.7.3 开放寻址下成功搜索的时间成本
 - 11.7.4 封闭寻址散列
 - 11.7.5 探寻序列的设计
 - 11.7.6 封闭寻址散列的效率分析
 - 11.7.7 搜索不成功的时间成本
 - 11.7.8 成功搜索的效率分析
- 11.8 散列表元素删除
- 11.9 随机化散列
- 11.10 全域散列
- 11.11 完美散列
- 思考题
- 第12章 最短路径
 - 12.1 剑指罗马
 - 12.2 最短路径问题
 - 12.3 单源单点最短路径问题
 - 12.3.1 深度优先与广度优先搜索
 - 12.3.2 深度优先解法
 - 12.4 单源多点最短路径问题
 - 12.4.1 最短路径的性质
 - 12.4.2 Dijkstra最短路径算法
 - 12.4.3 Dijkstra算法举例
 - 12.4.4 Dijkstra算法与洪水泛滥
 - 12.4.5 Dijkstra算法的正确性
 - 12.4.6 Dijkstra算法的时间复杂性
 - 12.5 Bellman-Ford算法
 - 12.5.1 负权重的应对方式
 - 12.5.2 Bellman-Ford算法的正确性
 - 12.5.3 负循环检查问题
 - 12.5.4 Bellman-Ford算法的时间复杂性
 - 12.6 多源多点最短路径问题
 - 12.6.1 多源多点最短路径问题解决思路
 - 12.6.2 直接动态规划解法
 - 12.6.3 矩阵乘法解法
 - 12.6.4 Floyd-Warshall算法
 - 12.6.5 Johnson算法
 - 12.6.6 Johnson等效变换
 - 12.6.7 差限问题解决
 - 12.7 天意难违
- 思考题
- 第五篇 难解与无解篇
 - 第13章 易解与难解

<<算法之道(第2版)>>

- 13.1 我们战无不胜吗
- 13.2 易解与难解
- 13.3 决策问题和优化问题
- 13.4 决策问题
- 13.5 P类问题
- 13.6 NP类问题
- 13.7 (确定性)图灵机
- 13.8 非确定性图灵机
- 13.9 非确定性算法
- 13.10 回到NP类问题
- 13.11 P和NP
- 13.12 搜索问题、决策问题和优化问题
- 13.13 有没有解和是否可决定

思考题

第14章 NP完全问题

- 14.1 玉龙雪山下的审判
- 14.2 NP完全问题的定义
- 14.3 NP完全的重要性
- 14.4 多项式时间规约
- 14.5 如何证明一个问题S是NP完全问题
- 14.6 第1个NP完全问题的证明
- 14.7 库克定理
- 14.8 3-SAT问题
- 14.9 证明NP难的技巧
- 14.10 整数规划
- 14.11 独立集问题
- 14.12 哈密尔顿回路问题
- 14.13 讨论：弱NP完全、强NP完全和中NP完全

思考题

第15章 无解与近似

- 15.1 难解问题
- 15.2 不可决定问题
- 15.3 程序终结的判断
- 15.4 难解之题的求解
- 15.5 智能穷举、近似算法和本地搜索
- 15.6 智能穷举之回溯策略
- 15.7 智能穷举之分支限界
- 15.8 贪婪近似策略
- 15.9 启发式搜索策略
- 15.10 模拟退火算法
 - 15.10.1 模拟退火算法的思想
 - 15.10.2 模拟退火算法的基本循环
 - 15.10.3 退火算法描述
- 15.11 基因/遗传算法
 - 15.11.1 生物进化与遗传
 - 15.11.2 遗传算法的基本要义
 - 15.11.3 遗传算法的实现

<<算法之道(第2版)>>

15.11.4 遗传算法的基本运算过程

15.11.5 遗传算法的现状

15.12 概率尽在一切中

思考题

结语 算法之道

附录 算法随想

参考文献

<<算法之道(第2版)>>

编辑推荐

《算法之道(第2版)》既可以作为大学本科或研究生的算法教材或参考书，也可以作为对算法有兴趣的读者提升认知深度的读物。

<<算法之道(第2版)>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介, 请支持正版图书。

更多资源请访问:<http://www.tushu007.com>