

21世纪高等学校规划教材 | 计算机科学与技术



算法设计与分析

——以ACM大学生程序设计竞赛
在线题库为例

赵端阳 刘福庆 石洗凡 编著



清华大学出版社



算法设计与分析

——以ACM大学生程序设计竞赛 在线题库为例

赵端阳 刘福庆 石洗凡 编著

清华大学出版社
北京

内 容 简 介

本书介绍数据结构和标准模板库 STL、递归与分治策略、动态规划、贪心算法、回溯算法、分支限界算法、图论、数论和组合数学问题。本书包括大量实例,并在北京大学、浙江大学和杭州电子科技大学在线题库中精选原题,详细地分析解题的方法,深入浅出地讲解用到的算法,挑选在线题库中的典型题目作为每章后面的习题,供读者练习,以巩固所学的算法。本书内容基本上涵盖了目前大学生程序设计竞赛所要掌握的算法。

本书结构清晰,内容丰富,适合于作为计算机科学与技术、软件工程以及相关学科算法课程的教材和参考书,也特别适合有志于参加 ACM 大学生程序设计竞赛的读者学习和训练。

本书配备有电子教案和源代码,请到清华大学出版社网站上下载: www.tup.com.cn。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

版权所有,侵权必究。侵权举报电话: 010-62782989 13701121933

图书在版编目(CIP)数据

算法设计与分析: 以 ACM 大学生程序设计竞赛在线题库为例/赵端阳等编著. --北京: 清华大学出版社, 2015

21 世纪高等学校规划教材·计算机科学与技术

ISBN 978-7-302-40007-3

I. ①算… II. ①赵… III. ①电子计算机—算法设计—高等学校—题解 ②电子计算机—算法分析—高等学校—题解 IV. ①TP301.6-44

中国版本图书馆 CIP 数据核字(2015)第 086390 号

责任编辑: 闫红梅 薛 阳

封面设计: 傅瑞学

责任校对: 焦丽丽

责任印制: 李红英

出版发行: 清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址: 北京清华大学学研大厦 A 座 邮 编: 100084

社 总 机: 010-62770175 邮 购: 010-62786544

投稿与读者服务: 010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈: 010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 刷 者: 北京富博印刷有限公司

装 订 者: 北京市密云县京文制本装订厂

经 销: 全国新华书店

开 本: 185mm×260mm 印 张: 26 字 数: 636 千字

版 次: 2015 年 7 月第 1 版 印 次: 2015 年 7 月第 1 次印刷

印 数: 1~2000

定 价: 49.00 元

产品编号: 059655-01

出版说明

随着我国改革开放的进一步深化,高等教育也得到了快速发展,各地高校紧密结合地方经济建设发展需要,科学运用市场调节机制,加大了使用信息科学等现代科学技术提升、改造传统学科专业的投入力度,通过教育改革合理调整和配置了教育资源,优化了传统学科专业,积极为地方经济建设输送人才,为我国经济社会的快速、健康和可持续发展以及高等教育自身的改革发展做出了巨大贡献。但是,高等教育质量还需要进一步提高以适应经济社会发展的需要,不少高校的专业设置和结构不尽合理,教师队伍整体素质亟待提高,人才培养模式、教学内容和方法需要进一步转变,学生的实践能力和创新精神亟待加强。

教育部一直十分重视高等教育质量工作。2007年1月,教育部下发了《关于实施高等学校本科教学质量与教学改革工程的意见》,计划实施“高等学校本科教学质量与教学改革工程”(简称“质量工程”),通过专业结构调整、课程教材建设、实践教学改革、教学团队建设等多项内容,进一步深化高等学校教学改革,提高人才培养的能力和水平,更好地满足经济社会发展对高素质人才的需要。在贯彻和落实教育部“质量工程”的过程中,各地高校发挥师资力量强、办学经验丰富、教学资源充裕等优势,对其特色专业及特色课程(群)加以规划、整理和总结,更新教学内容、改革课程体系,建设了一大批内容新、体系新、方法新、手段新的特色课程。在此基础上,经教育部相关教学指导委员会专家的指导和建议,清华大学出版社在多个领域精选各高校的特色课程,分别规划出版系列教材,以配合“质量工程”的实施,满足各高校教学质量和教学改革的需要。

为了深入贯彻落实教育部《关于加强高等学校本科教学工作,提高教学质量的若干意见》精神,紧密配合教育部已经启动的“高等学校教学质量与教学改革工程精品课程建设工作”,在有关专家、教授的倡议和有关部门的大力支持下,我们组织并成立了“清华大学出版社教材编审委员会”(以下简称“编委会”),旨在配合教育部制定精品课程教材的出版规划,讨论并实施精品课程教材的编写与出版工作。“编委会”成员皆来自全国各类高等学校教学与科研第一线的骨干教师,其中许多教师为各校相关院、系主管教学的院长或系主任。

按照教育部的要求,“编委会”一致认为,精品课程的建设工作从开始就要坚持高标准、严要求,处于一个比较高的起点上。精品课程教材应该能够反映各高校教学改革与课程建设的需要,要有特色风格、有创新性(新体系、新内容、新手段、新思路,教材的内容体系有较高的科学创新、技术创新和理念创新的含量)、先进性(对原有的学科体系有实质性的改革和发展,顺应并符合21世纪教学发展的规律,代表并引领课程发展的趋势和方向)、示范性(教材所体现的课程体系具有较广泛的辐射性和示范性)和一定的前瞻性。教材由个人申报或各校推荐(通过所在高校的“编委会”成员推荐),经“编委会”认真评审,最后由清华大学出版

社审定出版。

目前,针对计算机类和电子信息类相关专业成立了两个“编委会”,即“清华大学出版社计算机教材编审委员会”和“清华大学出版社电子信息教材编审委员会”。推出的特色精品教材包括:

(1) 21 世纪高等学校规划教材·计算机应用——高等学校各类专业,特别是非计算机专业的计算机应用类教材。

(2) 21 世纪高等学校规划教材·计算机科学与技术——高等学校计算机相关专业的教材。

(3) 21 世纪高等学校规划教材·电子信息——高等学校电子信息相关专业的教材。

(4) 21 世纪高等学校规划教材·软件工程——高等学校软件工程相关专业的教材。

(5) 21 世纪高等学校规划教材·信息管理与信息系统。

(6) 21 世纪高等学校规划教材·财经管理与应用。

(7) 21 世纪高等学校规划教材·电子商务。

(8) 21 世纪高等学校规划教材·物联网。

清华大学出版社经过三十多年的努力,在教材尤其是计算机和电子信息类专业教材出版方面树立了权威品牌,为我国的高等教育事业做出了重要贡献。清华版教材形成了技术准确、内容严谨的独特风格,这种风格将延续并反映在特色精品教材的建设中。

清华大学出版社教材编审委员会

联系人:魏江江

E-mail: weijj@tup.tsinghua.edu.cn

前言

“算法分析与设计”是一门理论性与实践性结合很强的课程。在信息技术高速发展的今天,计算机技术已经应用到了很多科学领域。从理论上来说,算法研究已经被公认为是计算机科学的基石。David Harel 在他的《算法学:计算精髓》一书中说道:“算法不仅是计算机科学的一个分支,它更是计算机科学的核心。可以毫不夸张地说,它和绝大多数的科学、商业和技术都是相关的。”

在 ACM 国际大学生程序设计竞赛中,在线裁判系统是开展竞赛的核心,它是一个在线的程序与算法设计的练习和竞赛平台。系统可以提供大量的关于程序和算法设计的题目供学生练习或竞赛,学生可以使用自己熟悉的语言提交相关题目的程序代码,系统编译提交代码,如果没有错误,则生成可执行文件。利用系统的测试用例来测试,如果输出结果正确,则返回程序消耗的内存空间和时间。对于竞赛题目,系统可以从程序正确性、运行总时间、消耗内存空间、返回结果等方面来考察学生提交的代码。系统可以实现在指定的时间段举行竞赛的功能,根据学生解题数目和时间进行排名,也可以批量导出学生代码,进行分析。

基于程序设计竞赛的教学模式具有以下优势。

(1) 提供一个开放的、自主学习的实验环境。在线评测系统通过网络使用,学生可以随时随地提交程序代码;在丰富的算法设计题库中寻找适合自己的题目,训练程序设计能力。

(2) 有效地训练学生程序设计能力,培养创新型 IT 人才。本课程的学习难点在于如何将常见的算法策略应用到实际的应用环境中。通过在线评测系统的实践训练,让学生熟练掌握常见的算法设计策略,训练学生的创新思维,加深学生对各种算法设计策略的认识,理解算法的意义及精髓,达到学以致用。

(3) 形成了良好的学习氛围,加强了学生之间的交流。使用在线评测系统进行课程考核并举办程序与算法设计竞赛,以团队方式参与,可以形成良好的校园竞争和交流的学习氛围;学生有了在课余时间自主进行本学科知识钻研的机会和环境;也让学生体验到团队协作的重要性,为软件项目团队化的合作要求做好准备。

算法分析与设计是面向设计的核心课程,主要通过介绍常见的算法设计策略及复杂性分析方法,培养学生分析问题和解决问题的能力,为开发高效的软件系统及相关领域的研究工作奠定坚实的基础。该课程理论与实践并重,内容具有综合性、广泛性和系统性,是一门集应用性、创造性及实践性为一体的综合性极强的课程。

目前,该课程的教学方法还是以传统的讲解为主,通常只是将已有的经典算法在已有的数学模型和数据结构上解释给学生;在实践环节只是盲目地验证算法,而对该算法的运行效率、测试数据规模以及实际的应用场景则很少考虑。学生的学习则主要以理解和记忆的继承式学习为主,虽然记住了大量的算法理论,但没有“理解”和“消化”,不能灵活运用算法;在实践环节学生代码抄袭严重,很难达到训练的效果。在这种教学模式下,学生缺乏问题抽象能力,在遇到实际问题时无从下手,思维创新能力和实践能力难以得到有效的提高,很难

培养出高水平的程序员。

本书利用程序设计竞赛模式和在线评测系统的特点,结合课程特点和实际教学,弥补课程教学中存在的不足,以此探讨算法分析与设计的课程教学改革,培养高水平的编程人才。

本书共分为 11 章:

第 1 章 算法概述,主要介绍算法的基本概念、算法的复杂性、大学生程序设计竞赛概述和程序设计在线题库的基本情况。

第 2 章 数据结构和标准模板库 STL,主要介绍栈(Stack)、向量(Vector)、映射(Map)、列表(List)、集合(Set)、队列(Queue)和优先队列(Priority Queue),典型例题有 ZOJ1004-Anagrams by Stack、ZOJ1094-Matrix Chain Multiplication、ZOJ1062-Trees Made to Order 和 ZOJ1944-Tree Recovery 等。

第 3 章 递归与分治策略,主要介绍递归算法和分治策略,典型例题有循环赛日程表、棋盘覆盖问题、选择问题、整数因子分解和 ZOJ1633-Big String 等。

第 4 章 动态规划,主要介绍动态规划算法的基本要素,典型例题有矩阵连乘积、最长公共子序列、最大子段和、0-1 背包问题和最长单调递增子序列等,并求解 ZOJ1027-Human Gene Functions、ZOJ1074-To the Max、ZOJ1163-The Staircases 和 ZOJ1196-Fast Food 等。

第 5 章 贪心算法,主要介绍贪心算法的理论基础。典型例题有活动安排问题、背包问题、最优装载问题、单源最短路径和最小生成树等,并求解 ZOJ1012-Mainframe、ZOJ1025-Wooden Sticks、ZOJ1076-Gene Assembly 和 ZOJ2109-FatMouse' Trade 等。

第 6 章 回溯算法,主要介绍回溯算法的理论基础。典型例题有装载问题、0-1 背包问题、图的 m 着色问题、 n 皇后问题、旅行商问题和流水作业调度问题等。

第 7 章 分支限界算法,主要介绍分支限界算法的基本理论,并对回溯算法与分支限界算法进行比较。典型例题有单源最短路径问题、装载问题、0-1 背包问题和旅行商问题,并求解 ZOJ1136-Multiple。

第 8 章 图的搜索算法,主要介绍图的深度和广度优先搜索遍历算法。

第 9 章 图论,主要介绍网络流问题和二分图匹配问题,分析剩余网络的增广路、Ford-Fulkerson 算法和 Edmonds-Karp 算法;二分图最大匹配的匈牙利算法、Hopcroft-Karp 算法和 Kuhn Munkres 算法。求解 ZOJ1734-Power Network、ZOJ1137-Girls and Boys 等。

第 10 章 数论,主要介绍扩展欧几里得算法、欧拉函数、中国剩余定理和一元线性同余方程组,并求解 ZOJ1096 Relatives、ZOJ1160 Biorhythms 和 POJ2480 Longge's Problem。

第 11 章 组合数学,主要介绍母函数、Stirling 数、Catalan 数、容斥原理与鸽巢原理。

冯志林、刘均、袁鹤、吴艳、金海溶、方蕾和曹平参加了本书的编写工作。

本教材获得浙江工业大学重点教材建设项目资助。

由于作者水平所限,书中难免有不足之处,恳请广大读者批评指正。作者电子邮件地址: 727946579@qq.com。参考网站: www.acmbook.org。

编 者

目 录

第 1 章 算法概述	1
1.1 引言	1
1.1.1 算法的描述	1
1.1.2 算法的设计	3
1.2 算法的复杂性	6
1.2.1 时间复杂性	6
1.2.2 空间复杂性	9
1.3 大学生程序设计竞赛概述	10
1.4 程序设计在线测试题库	11
第 2 章 数据结构和标准模板库 STL	13
2.1 栈	13
2.2 向量	14
2.3 映射	17
2.4 列表	19
2.5 集合	21
2.6 队列	22
2.7 优先队列	23
2.8 ZOJ1004-Anagrams by Stack	25
2.9 ZOJ1094-Matrix Chain Multiplication	28
2.10 ZOJ1011-NTA	31
2.11 ZOJ1062-Trees Made to Order	36
2.12 ZOJ1097-Code the Tree	40
2.13 ZOJ1156-Unscrambling Images	43
2.14 ZOJ1167-Trees on the Level	47
2.15 ZOJ1016-Parencodings	50
2.16 ZOJ1944-Tree Recovery	53
2.17 ZOJ2104-Let the Balloon Rise	55
上机练习题	56
第 3 章 递归与分治策略	59
3.1 递归算法	59

3.1.1	Fibonacci 数列	60
3.1.2	集合的全排列问题	61
3.1.3	整数划分问题	62
3.2	分治策略	63
3.2.1	分治法的基本步骤	63
3.2.2	分治法的适用条件	64
3.2.3	二分搜索技术	64
3.2.4	循环赛日程表	65
3.2.5	棋盘覆盖问题	67
3.2.6	选择问题	70
3.2.7	输油管道问题	72
3.2.8	半数集问题	73
3.2.9	整数因子分解	75
3.2.10	取余运算	76
3.3	ZOJ1633-Big String	78
	上机练习题	79
第 4 章	动态规划	81
4.1	矩阵连乘积问题	82
4.1.1	分析最优解的结构	84
4.1.2	建立递归关系	85
4.1.3	计算最优值	85
4.1.4	构造最优解	87
4.2	动态规划算法的基本要素	88
4.2.1	最优子结构	88
4.2.2	重叠子问题	89
4.2.3	备忘录方法	89
4.3	最长公共子序列	91
4.3.1	最长公共子序列的结构	91
4.3.2	子问题的递归结构	92
4.3.3	计算最优值	92
4.3.4	构造最长公共子序列	94
4.4	最大子段和	94
4.5	0-1 背包问题	96
4.5.1	递归关系分析	97
4.5.2	算法实现	97
4.6	最长单调递增子序列	99
4.7	数字三角形问题	100
4.8	ZOJ1027-Human Gene Functions	101

4.9	ZOJ1074-To the Max	105
4.10	ZOJ1093-Monkey and Banana	106
4.11	ZOJ1107-FatMouse and Cheese	111
4.12	ZOJ1108-FatMouse's Speed	114
4.13	ZOJ1147-Formatting Text	118
4.14	ZOJ1149-Dividing	123
4.15	ZOJ1163-The Staircases	127
4.16	ZOJ1183-Scheduling Lectures	130
4.17	ZOJ1196-Fast Food	133
4.18	ZOJ1206-Win the Bonus	137
4.19	ZOJ1227-Free Candies	140
4.20	ZOJ1234-Chopsticks	144
	上机练习题	147
第 5 章	贪心算法	151
5.1	活动安排问题	151
5.2	贪心算法的理论基础	153
5.2.1	贪心选择性质	154
5.2.2	最优子结构性质	154
5.2.3	贪心算法的求解过程	154
5.3	背包问题	155
5.4	最优装载问题	158
5.5	单源最短路径	159
5.6	最小生成树	163
5.6.1	最小生成树的性质	163
5.6.2	Prim 算法	164
5.6.3	Kruskal 算法	166
5.7	删数问题	169
5.7.1	问题的贪心选择性质	170
5.7.2	问题的最优子结构性质	171
5.8	多处最优服务次序问题	171
5.8.1	问题的贪心选择性质	173
5.8.2	问题的最优子结构性质	173
5.9	ZOJ1012-Mainframe	174
5.10	ZOJ1025-Wooden Sticks	178
5.11	ZOJ1029-Moving Tables	181
5.12	ZOJ1076-Gene Assembly	183
5.13	ZOJ1161-Gone Fishing	185
5.14	ZOJ1171-Sorting the Photos	189

5.15	ZOJ2109-FatMouse' Trade	190
	上机练习题	192
第6章	回溯算法	194
6.1	回溯算法的理论基础	194
6.1.1	问题的解空间	194
6.1.2	回溯法的基本思想	195
6.1.3	子集树与排列树	197
6.2	装载问题	198
6.3	0-1 背包问题	201
6.4	图的 m 着色问题	203
6.5	n 皇后问题	206
6.6	旅行商问题	208
6.7	流水作业调度问题	210
6.8	子集和问题	213
6.9	ZOJ1145-Dreisam Equations	215
6.10	ZOJ1157-A Plug for UNIX	220
6.11	ZOJ1166-Anagram Checker	226
6.12	ZOJ1213-Lumber Cutting	230
	上机练习题	234
第7章	分支限界算法	235
7.1	分支限界算法的基本理论	235
7.1.1	分支限界算法策略	235
7.1.2	分支结点的选择	236
7.1.3	提高分支限界算法的效率	236
7.1.4	限界函数	237
7.2	单源最短路径问题	237
7.3	装载问题	242
7.4	0-1 背包问题	245
7.5	旅行商问题	251
7.6	ZOJ1136-Multiple	254
7.7	回溯算法与分支限界算法的比较	258
	上机练习题	258
第8章	图的搜索算法	259
8.1	图的深度优先搜索遍历	259
8.2	ZOJ1002-Fire Net	260
8.3	ZOJ1008-Gnome Tetravex	262

8.4	ZOJ1047-Image Perimeters	267
8.5	ZOJ1084-Channel Allocation	271
8.6	ZOJ1142-Maze	274
8.7	ZOJ1190-Optimal Programs	278
8.8	ZOJ1191-The Die Is Cast	284
8.9	ZOJ1204-Additive Equations	287
8.10	ZOJ1245-Triangles	290
8.11	ZOJ2100-Seeding	293
8.12	图的广度优先搜索遍历	296
8.13	ZOJ1079-Robotic Jigsaw	297
8.14	ZOJ1085-Alien Security	301
8.15	ZOJ1103-Hike on a Graph	305
8.16	ZOJ1148-The Game	308
8.17	ZOJ1217-Eight	311
8.18	ZOJ1091-Knight Moves	317
	上机练习题	321
第 9 章	图论	323
9.1	网络流问题	323
9.1.1	流和割的概念	323
9.1.2	剩余网络和增广路	324
9.1.3	Ford-Fulkerson 算法	325
9.1.4	Edmonds-Karp 算法	326
9.1.5	ZOJ1734-Power Network——Edmonds-Karp 算法	327
9.1.6	ISAP 算法	330
9.1.7	ZOJ1734-Power Network——ISAP 算法	332
9.1.8	Dinic 算法	334
9.1.9	ZOJ1734-Power Network——Dinic 算法	335
9.1.10	最小费用流——SPFA 算法	337
9.1.11	ZOJ2404-Going Home——SPFA 算法	338
9.2	二分图匹配问题	341
9.2.1	匹配问题	341
9.2.2	二分图最大匹配——匈牙利算法	343
9.2.3	ZOJ1137-Girls and Boys	344
9.2.4	ZOJ1140-Courses——匈牙利算法	346
9.2.5	PJU1247-The Perfect Stall——匈牙利算法	349
9.2.6	Hopcroft-Karp 算法	352
9.2.7	ZOJ1140-Courses——Hopcroft-Karp 算法	353
9.2.8	PJU1247-The Perfect Stall——Hopcroft-Karp 算法	355

9.2.9	二分图最佳匹配——Kuhn Munkres 算法	357
9.2.10	ZOJ2404-Going Home——Kuhn Munkres 算法	358
	上机练习题	360
第 10 章	数论	362
10.1	扩展欧几里得算法	362
10.2	PJU2115-C Looooops	363
10.3	欧拉函数	365
10.4	ZOJ1906-Relatives	366
10.5	PJU2480-Longge's Problem	367
10.6	PJU3696-The Luckiest Number	369
10.7	中国剩余定理	372
10.8	ZOJ1160-Biorhythms	372
10.9	一元线性同余方程组	374
10.10	PJU2891-Strange Way to Express Integers	375
10.11	HDU1572-X 问题	377
	上机练习题	378
第 11 章	组合数学	381
11.1	母函数	381
11.1.1	普通型母函数	381
11.1.2	指数型母函数	383
11.1.3	Stirling 数	384
11.1.4	Catalan 数	386
11.2	HDU2082-找单词	387
11.3	HDU1521-排列组合	388
11.4	HDU2065-“红色病毒”问题	389
11.5	HDU3625-Examining the Rooms	391
11.6	POJ2084-Game of Connections	393
11.7	容斥原理与鸽巢原理	394
11.7.1	容斥原理	394
11.7.2	错排问题	395
11.7.3	鸽巢原理	396
11.8	HDU2048-神、上帝以及老天爷	397
11.9	PJU2356-Find a Multiple	398
11.10	ZOJ2836-Number Puzzle	400
	上机练习题	401
	参考文献	404

第1章

算法概述

算法(Algorithm)是一系列解决问题的清晰指令,代表着用系统的方法描述解决问题的策略机制。算法能够依据一定的规范的输入,在有限时间内获得所要求的输出。如果一个算法有缺陷,或不适合于某个问题,执行这个算法将不会解决这个问题。不同的算法可能用不同的时间、空间或效率来完成同样的任务。一个算法的优劣可以用空间复杂性与时间复杂性来衡量。算法可以使用自然语言、伪代码、流程图等多种不同的方法来描述。计算机系统中的操作系统、语言编译系统、数据库管理系统以及各种各样的计算机应用系统中的软件,都必须使用具体的算法来实现。算法设计与分析是计算机科学与技术的一个核心问题。

1.1 引言

“算法”即演算法,中文名称出自《周髀算经》;而英文名称 Algorithm 来自于 9 世纪波斯数学家 al-Khwārizmī,他在数学上提出了“算法”这个概念。“算法”原为 algorism,意思是阿拉伯数字的运算法则,在 18 世纪演变为 algorithm。欧几里得算法被人们认为是史上第一个算法。第一次编写程序是 Ada Byron 于 1842 年为巴贝奇分析机编写求解伯努利方程的程序,因此 Ada Byron 被大多数人认为是世界上第一位程序员。但是由于查尔斯·巴贝奇(Charles Babbage)未能完成他的巴贝奇分析机,这个算法未能在巴贝奇分析机上执行。

一本早期的德文数学词典 Vollständiges Mathematisches Lexicon《数学大全辞典》,给出了 Algorithmus(算法)一词的如下定义:“在这个名称之下,组合了 4 种类型的算术计算的概念,即加法、乘法、减法、除法。”拉丁短语 algorithmus infinitesimalis(无限小方法),在当时就用来表示 Leibnitz(莱布尼兹)所发明的以无限小量进行计算的微积分方法。

1950 年左右,algorithm 一词经常同欧几里得算法(Euclid's algorithm)联系在一起。欧几里得算法就是在欧几里得的《几何原本》(Euclid's Elements,第Ⅶ卷,命题Ⅰ和Ⅱ)中所阐述的求两个数的最大公约数的过程(即辗转相除法)。

20 世纪的英国数学家图灵提出了著名的图灵论题,并提出一种假想的计算机的抽象模型,这个模型被称为图灵机。图灵机的出现解决了算法定义的难题,图灵的思想对算法的发展起到了重要作用。

1.1.1 算法的描述

在 20 世纪 50 年代,欧几里得描述了求两个数的最大公因子算法的过程,被称为欧几里

得算法。欧几里得算法又称辗转相除法,用于计算两个正整数 m, n 的最大公约数。

步骤 1: 如果 $m < n$, 则交换 m 和 n 。

步骤 2: 令 r 是 m/n 的余数。

步骤 3: 如果 $r = 0$, 则输出 m ; 否则令 $m = n, n = r$ 并转向步骤 2。

其计算原理依赖于下面的定理:

定理: $\gcd(m, n) = \gcd(n, m \bmod n)$ ($m > n$ 且 $m \bmod n$ 不为 0)

算法 1.1 欧几里得算法。

```
int gcd(int m, int n)
```

```
{
```

```
    if (m < n) gcd(n, m);
```

```
    int r;
```

```
    do {
```

```
        r = m % n;
```

```
        m = n;
```

```
        n = r;
```

```
    } while(r);
```

```
    return m;
```

```
}
```

欧几里得算法的递归实现:

```
int gcd(int m, int n)
```

```
{
```

```
    if (m < n) gcd(n, m);
```

```
    if (n == 0) return m;
```

```
    else return gcd(n, m % n);
```

```
}
```

算法设计的先驱者唐纳德·E. 克努特(Donald E. Knuth)对算法的特征做了如下描述。

1. 有穷性

算法在执行有限步之后必须终止。

在算法 1.1 中,对输入的任意正整数 m, n ($m > n$),令 r 是 m/n 的余数,经过辗转相除,从而使 m 和 n 变小。如此往复进行,最终使 r 为 0,算法终止。

2. 确定性

算法的每一步骤必须有确切的定义。

算法的每一个步骤,都有精确的定义。要执行的每一个动作都是清晰的、无歧义的。在算法 1.1 中,如果 m 和 n 是无理数,那么 m 除以 n 的余数是什么,就没有一个明确的界定。欧几里得算法规定了 m 和 n 都是正整数,从而保证了算法能够确定地执行。

3. 输入

一个算法有 0 个或多个输入,作为算法开始执行前的初始值或初始状态。0 个输入是指算法本身定出了初始条件。

算法 1.1 中有两个输入 m 和 n , 都是正整数。

4. 输出项

一个算法有一个或多个输出, 以反映对输入数据加工后的结果。没有输出的算法是毫无意义的。算法 1.1 中的输出是 m 和 n 的最大公约数。

5. 可行性

在有限时间内完成计算过程。

算法 1.1 用一个正整数来除另一个正整数, 判断一个整数是否为 0 以及整数赋值等, 这些运算都是可行的。因为整数可以用有限的方式表示, 所以至少存在一种方法来完成一个整数除以另一个整数的运算。

必须注意到, 在实际应用中, 有限性的限制是不够的。一个实用的算法, 不仅要求步骤有限, 同时要求运行这些步骤所花费的时间是人们可以接受的。如果一个算法需要执行数万亿计数的运算步骤, 从理论上说, 它是有限的, 最终可以结束。但是, 以当代计算机每秒数亿次的运算速度, 也必须运行数百年以上, 这是人们所无法接受的, 因而不实用的算法。

1.1.2 算法的设计

算法设计的整个过程, 可以包含对问题需求的说明、数学模型的拟制、算法的详细设计、算法的正确性验证、算法的实现、算法分析、程序测试和文档资料的编制。

计算机科学家尼克劳斯·沃思曾写过一本著名的书《数据结构+算法=程序》, 可见算法在计算机科学界与计算机应用界的地位。

同一问题可用不同算法解决, 而一个算法的质量优劣将影响到算法乃至程序的效率。算法分析的目的在于选择合适算法和改进算法。一个算法的评价主要从时间复杂性和空间复杂性两方面来考虑。

算法可大致分为基本算法、数据结构的算法、数论与代数算法、计算几何的算法、图论的算法、动态规划以及数值分析、加密算法、排序算法、检索算法、随机化算法、并行算法。算法大致分为以下三类。

(1) 有限的、确定性算法。这类算法在有限的一段时间内终止。它们可能要花很长时间来执行指定的任务, 但仍将在一定的时间内终止。这类算法得出的结果常取决于输入值。

(2) 有限的、非确定算法。这类算法在有限的时间内终止。然而, 对于一个(或一些)给定的数值, 算法的结果并不是唯一的或确定的。

(3) 无限的算法。指那些由于没有定义终止定义条件, 或定义的条件无法由输入的数据满足而不终止运行的算法。通常, 无限算法的产生是由于未能确定的定义终止条件。

经典的算法有很多, 这里主要列举以下算法。

1. 穷举搜索法

穷举搜索法(Exhaustive Search Algorithm)是对可能是解的众多候选解按某种顺序进行逐一枚举和检验, 并从中找出那些符合要求的候选解作为问题的解。

穷举算法的特点是算法简单, 但运行时所花费的时间量大。有些问题所列举出来的情

况数目会大得惊人,就是用高速计算机运行,其等待运行结果的时间也将使人无法忍受。在用穷举算法解决问题时,应尽可能将明显不符合条件的情况排除在外,以尽快取得问题的解。

2. 迭代算法

迭代算法(Iterative Algorithm)是数值分析中通过从一个初始估计出发寻找一系列近似解来解决问题(一般是解方程或者方程组)的过程,为实现这一过程所使用的方法统称为迭代法。

迭代法是用于求方程或方程组近似根的一种常用的算法设计方法。设方程为 $f(x)=0$, 用某种数学方法导出等价的形式 $x=g(x)$, 然后按以下步骤执行。

(1) 选一个方程的近似根, 赋给变量 x_0 ;

(2) 将 x_0 的值保存于变量 x_1 , 然后计算 $g(x_1)$, 并将结果存于变量 x_0 ;

(3) 当 x_0 与 x_1 的差的绝对值还小于指定的精度要求时, 重复步骤(2)的计算。

若方程有根, 并且用上述方法计算出来的近似根序列收敛, 则按上述方法求得的 x_0 就认为是方程的根。

3. 递推算法

递推算法(Recursive Algorithm)是利用问题本身所具有的一种递推关系求问题解的一种方法。它把问题分成若干步, 找出相邻几步的关系, 从而达到目的, 称为递推法。

设要求问题规模为 n 的解, 当 $n=0$ 或 1 时, 解或为已知, 或能非常方便地得到解。能采用递推法构造算法的问题有重要的递推性质, 即当得到问题规模为 $i-1$ 的解后, 由问题的递推性质, 能从已求得的规模为 $1, 2, \dots, i-1$ 的一系列解, 构造出问题规模为 i 的解。这样, 程序可从 $i=0$ 或 1 出发, 重复地, 由已知至 $i-1$ 规模的解, 通过递推, 获得规模为 i 的解, 直至得到规模为 n 的解。

4. 递归算法

递归算法(Recursive Algorithm)是一种直接或者间接地调用自身的算法。在计算机编写程序中, 递归算法对解决一大类问题是十分有效的, 它往往使算法的描述简洁而且易于理解。

能采用递归描述的算法通常有这样的特征: 为求解规模为 n 的问题, 设法将它分解成规模较小的问题, 然后从这些小问题的解方便地构造出大问题的解, 并且这些规模较小的问题也能采用同样的分解和综合方法, 分解成规模更小的问题, 并从这些更小问题的解构造出规模较大问题的解。特别地, 当规模 $n=0$ 或 1 时, 能直接得解。

用递归算法解决问题具有以下特点。

(1) 递归就是在过程或函数里调用自身;

(2) 在使用递归策略时, 必须有一个明确的递归结束条件, 称为递归出口;

(3) 递归算法解题通常显得很简洁, 但递归算法解题的运行效率较低;

(4) 在递归调用的过程当中系统为每一层的返回点、局部变量等开辟堆栈来存储。递归次数过多容易造成堆栈溢出等。