程序设计的数学基础

〔美〕F.S. 贝克曼 著

科学出版社

程序设计的数学基础

〔美〕F. S. 贝克曼 著

曹德和 吴延佳 译

刘椿年 校

科学出版社

内 容 简 介

本书以描述性方式给出了与计算机程序设计和计算机设计有关的大量数学概念，揭示了计算过程的基本数学特征.

全书共分十二章，给出了能行性概念，函数、关系、布尔代数、命题演算、集合论、超穷数和图论等基本的数学概念，以及有穷自动机的概念；讨论了递归函数，图灵机和可计算性的关系，形式语言及其与自动机和程序设计语言的关系；介绍了计算复杂性理论；最后在第十二章对数学在计算中的影响进行了总结. 每章后均附有练习.

本书涉及面较广，叙述中着重各种概念的背景、基本思想和用途，不强调形式化描述. 本书可作为计算机科学系、应用数学系各有关专业的教材或参考书，同时对广大计算机软件工作者提高数学素质具有参考价值.

Frank S. Beckman

MATHEMATICAL FOUNDATIONS OF PROGRAMMING

Addison-Wesley Publishing Company, 1980

程序设计的数学基础

〔美〕F. S. 贝克曼 著

曹德和 吴延佳 译

刘椿年 校

责任编辑 刘晓融

科学出版社出版

北京东黄城根北街 16 号

邮政编码：100707

中国科学院印刷厂印刷

新华书店北京发行所发行 各地新华书店经售

*

1991 年 8 月第 一 版 开本：850×1168 1/32

1991 年 8 月第一次印刷 印张：14 7/8

印数：0001—3 400 字数：390 000

ISBN 7-03-002284-X/TP · 166

定价：14.80元

序　言

作者撰写本书的目的，是要用描述性的术语来概括与计算机程序设计，在某些情况下，也与计算机设计有关的一大部分数学内容。我们并不涉及与使用计算机有关的大量数学内容，而只限于论述那些与计算过程的基本性质有关的论题。

与此相应的学科范围——数理逻辑和基础、可计算性和递归函数理论、形式语言学以及自动机理论——是很广阔的；计算机的广泛应用，无疑促进了有关文献的规模迅速扩大。本书不是从数学的深度去探讨这些专题，它不适合在大学或研究生院对这些学科进行深入学习的数学系学生，那些地方可能需要更形式的和更完整的推导。但是，在某些情况下，这些学生最终可能发现，某一观点或某一重点同可能出现在这些专题的更为严格的论述里的观点或重点有所不同。

放宽对严格、简明和完整的要求，这就允许我们比高深的数学研究更具释义性。我们主要说明这些学科讲的是什么，而不是给它们作出完善的解释。

成千上万的从事计算机程序设计和正在学习这门专业的人员有着不同的数学基础。在大学普及正规的计算机科学教育之前进入这个领域的许多人，都是从文科或商业界转过来的，他们的数学基础大都比较薄弱。甚至一些具有大学自然科学或数学方面专业知识的人，对要在高层次上进行深奥数学论述也感到棘手。但是，那些将把自己的工作时间花在计算机上的人，对于数学能提供什么而使人们能够深入了解围绕计算机出现的各种情况（即使这种了解一时还没有明显的直接效用）似乎应该具有某种好奇心，并应在这方面有一定的认识。正是为了这些人（还有攻读计算机科学的大学生），才编写了这本关于这些学科的“聪明门外汉”的论著。然

而，在做这项工作时，存在着很大的危险性．一方面有肤浅、不准确及歪曲主题以致失去学术价值的危险；另一方面，又有不能适应本书所设想的读者的危险．我们希望能够顺利地从这两种危险的中间通过．

除了一些小的可以被忽略的例子之外，我们不要求读者具有微积分或大学程度的数学知识．然而，这决不是说本书在数学方面是很低级的．在大学的几个班级里讲授此书之后，笔者认为，学生所具有的数学成熟性，应当是通常学完一年基础微积分或有穷数学后所应有的程度．一定的数学能力是需要的，在本书中，我们有时把一部分数学解释附到一些难度不同的练习里．同时，读者应相当熟悉程序设计的技巧．

我们用不同的方式强调了那些与计算机程序设计有关的数学概念和结论．一些基本思想和技巧照例应是每位程序设计者“武库”中的一部分；一些论题在一定程度上为程序设计技巧提供了“文化背景”．随着程序设计由一种技巧发展成为一门科学，一些论题正在程序系统及编译程序的设计中获得应用；一些论题目前虽然还没有明显的效用，但它们却与某些理论的形成有关，而这些理论将来可能导致重要的应用．

我们建议本书作为大学计算机科学专业或计算数学专业一学年的教材．对于许多学生来说，本课程可以被认为是取代了大学目前开设的一系列课程，比如离散结构、可计算理论、形式语言和自动机以及顺序机等．而笔者认为，用这种方式处理这门学科的内容，将能在计算机科学的这些数学领域中，把大学生和研究生的课程更为恰当而又明确地区别开来．

第一章着重讲述可以在没有时间或存储空间限制的数字计算机上执行的能行过程的基本特性．在附录中，简略地提到了一些关系到数学、物理学中的能行性作用的有争议的问题．对于多数班级(但不是全部)来讲，在这门课程刚开始时最好不讨论这个附录，可以把它略去，也可以在课程快结束时再加以考虑．第二、三章涉及与程序设计有关的多方面的论题．这些都是作为数学基础

中的论题，相当松散地结合在一起的．它们包括函数概念及其推广和定义方法，布尔代数和它的一些模式(其中包含命题演算)，数学和计算理论中的"原生"方法的使用，形式系统，算术化技术，集合理论的一些基本概念，超穷数的一个简要介绍，以及作为附录的图论的基本原理的论述．

在第四章中，简略地提到递归(或可计算)函数理论的基本思想及其对于过程定义和结构程序设计的某些含义．在第五章中，通过使用图灵机，对可计算函数概念作了详细阐述．特别是图灵的"短码"概念，有些人把它看作是使用高级程序设计语言的一个先声．在第六章中，我们把通用图灵机作为数字计算机的一种理论模式加以强调，指出图灵机理论对于"实际"计算世界的一些附加含义，并注意到计算过程的其它模式．

第七章包括对自动机的一般叙述，着重强调"状态"和状态转移，并包含自动机的一般分类．第八章专门讲述有穷状态自动机这一非常重要的类别，这是在编译和程序系统设计理论中的一个很重要的领域．第九章讲述符号运算系统，并对形式语言学作了介绍，其中包括语法和语义的讨论、Chomsky 定义文法的方法，以及这些概念对定义程序设计语言和开发其编译程序的技术所具有的意义．

第十章包括形式语言和自动机之间，形式语言和程序设计系统之间的一些重要关系．简略地提到了关于在程序验证和定义程序设计语言语义的方法方面所进行的一些研究．作为一个附加部分，我们举出与上面的论述有一定关系的某些有争议的研究，这就是设计一种为了与遥远星系中的文明世界进行交流的形式语言．第十一章简述新发展起来的（并有些混乱的）计算复杂性理论．为了使读者对这个领域中所研究的问题有一个正确的估价，还简略地描述了至今已取得的许多不同成果．第十二章对最近数十年来的一些关键性思想和新领域的发展进行了总结性评述，并讨论了最近几年引进的若干令人注目的现代代数学的统一性概念，以及这些思想对这些学科未来的发展可能产生的影响．这本书

几乎全都与数学对计算的贡献有关，但它同样注意到数学对计算的欠债(目前尚不能对此作出充分估计)。第十二章的附录打算给读者一些在研究自动机、语言和机器的一些问题时使用抽象代数方法的体验。

我得到了许多人士的帮助，对此向他们表示衷心感谢。(以下略)

F.S.贝克曼
纽约市立大学
布鲁克林学院
1980年1月

目　　录

第一章 能 行 的

本书的若干评论者都感到这一章的标题很不顺眼，认为作者采用这样一种别扭的形式是故弄玄虚，因此也许有必要稍微解释一下。作为一个能引起人们注意的标志，选择这样一个不合语法的标题是为了强调能行(或可计算)过程的广泛出现和它们的基本重要性——这些过程能够用数值术语完整地表达出来，并能在存储量足够大的数字计算机上，在有限的时间内完成。这样的过程可以不太严格地描述为：它们以符号演算为基础，而这种符号演算使用若干简单明确的，长度与数量均有限的规则。在某种意义上，"能行的"和"完全精确的"是相同的，而我们并不知道计算机反映客观世界的极限是什么。如果我们简单地把这样的论述称作"能行过程"(这意味着我们正在考察在某种一般分类中的一个特定过程类)，这也许就没有把这种过程所起的重要作用表达出来。在这些可计算过程中，我们强调依靠有限方法和构造结果——"可以找到一个数，使得……"而不仅是"存在一个数，使得……"——是与数学、物理学及哲学的基本结构有关的。许多数学问题已按这些(可计算的)条件给出，同时一些人认为所有的数学问题都应具有这种构造倾向。在本章的附录里，我们通过一个有关构造数学的简单讨论，给出基础微积分中的一种在某些数学家看来是应避免的数学论证的例子。这个例子对我们提出的意想中的读者——聪明的门外汉来说可能太专了，但这样的例子在全书中也只是偶然出现。

1.1 能行性定义

在计算机科学中，过程是"能行的"这一概念有着十分明确的

含义。它引伸了像“产生预期的和确定的作用”这种对该词的一般定义。它经常用的不太严格的等价术语有“机械的”、“构造的”、“限定的”和“算法的”。在数字计算机程序设计中，我们总是同上述过程打交道——它们提供了具体的、总能用数值形式表达的结果。这类过程是用很明确的方式产生这些结果的。如果不加改变地重复它们，就会产生同样的结果。这就是我们感兴趣的实际结果；我们不满足于仅仅知道这些结果的存在，而是希望能把它们显示出来。这种过程是由下列特性定义的：

1. 它们是确定的——其含义由这样一个事实说明，即我们预期由相同的初始条件，得到相同的结果。

2. 它们能在有限时间内，在有限设施上执行。但我们一般假设，在过程执行当中，如需增加设施，就像增加计算稿纸一样，是可以办到的。这就是说，所需要的“中间存储”量不必事先知道。

这里我们只要求资源有限，而对这些资源的大小不加限定，这就存在着相当不现实的因素。例如，这可能允许一些过程花费的时间比所谓的宇宙年龄还要长；或者它们需要的存储容量可能超过有限宇宙的大小。但在下面一些章节中，我们将考虑那些限制其作用域大小的过程或机器设备。

3. 每一个这种过程的执行都是“机械的”或“构造的”，且可以被精确地描述，使得另一个智能体，或者也许是一个设备，能够接受这种描述，并用它实施该过程而得出同样的结果。

4. 这些过程可以用数值术语编写——这可能是（3）的推论。它们含有能够由自然数（即正整数 1，2，3，…）表示的对象。同时我们总能把这种过程内的运算解释为算术运算。所得到的数值结果就是在我们的应用中很重要的那些东西可能采取的值。进一步讲，甚至这些过程的语句也是有限的，且自身能被表示为自然数（后一点将在第五章中详细说明）。

对这类过程的强调可能是那种构成数字计算基础的理论研究的最突出的性质。对这些过程的广泛研究，与数学、物理学和哲学都有着重要的联系。在能行过程的某些方面的讨论中，我们将简

要提到与数学的基本性质有关的几个具有深刻特点的问题，这些问题至今还没有一个能使所有在这个领域里工作的人都感到满意的解答．我们的目的不是给出这些问题的完整的描述，而只是说明，我们在数字计算上所涉及的事物，形成了在精神上密切相联的智力活动的壮观图景的一部分．

在另外一些场合，我们经常使用一些术语、概念或者进行一些分析、分类，此时，我们对自己正在探讨的问题并无十分确切的理解．可能用到的概念是含糊的并多少有些混乱，而且不同的人使用就可能有不同的意义．像"精神"、"幸福"、"美丽"和"爱"这样一些主观概念就提供了这方面的例证．一般讲，这种模糊性是不可避免的．但是，只要听到这些词的人能在他(她)自己的头脑中把这些词翻译成大体上恰当的概念，我们就认为可以了．即使对于一些看起来是客观的修饰语，我们往往也缺乏把能行过程建立在这些概念基础上所需要的那种严格性．作为一个很普通的例子，我们来看一看确定一个人是否"秃顶"的过程．一般说来，确定这一事实需要某种主观判断，很难形成一个"能行的"过程．但我们却可以很容易地定义一个肯定不大合理的能行过程．比如，我们可能说，如果他的头"顶"上（什么叫头顶也必须讲明）的头发少于15000根，他就是(秃顶)，否则不是．我们可以按这种方式构想出实现这种判别的自动机构．请注意，我们可能修改这个过程并允许对他的秃顶问题作出是、不是、不确定三种反应．但为了使过程对同样的对象总是产生同样的答案(即它是确定的)，我们仍必须给出一个近乎荒谬的定义．当然，在我们谈到计算机时，这就是所需要的那种完全的精确性．

但是，我们有关能行过程的隐含定义中有一部分很容易受到严厉的批评．例如，我们讲过程是机械的或构造的，是什么意思？根据什么说是构造的？正如Rosser[10]所说的，古希腊人在考虑初等平面几何的结构问题时，头脑中有一个非常严格的关于几何可构造性概念．当他们考虑角的三等分，或者说把一个任意角分成三个相等部分的问题时，允许使用的工具只有直尺和圆规．在

这种限制下(当然,学过高中几何的人都知道),这个解了两千多年的著名问题,最终在 19 世纪被证明在一般情况下是不可能解决的. 但是,借助一些很简单的辅助工具,如一张纸条,这个问题就迎刃而解了(参见图 1.1). 类似地,在任何构造的或有限可完成的过程中,都隐含着一张由可允许的基本步骤构成的表.

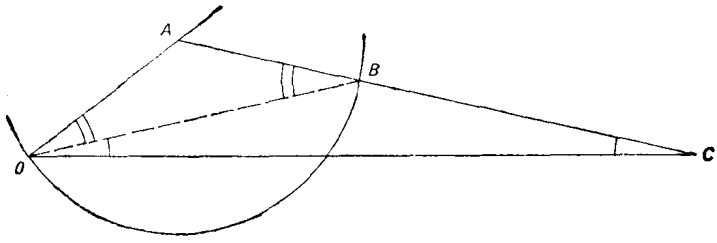

图 1.1 三等分角 AOC. 在一边上任选一点 A 并以它为圆心过 O 点作圆,用直尺,或者说直边过 A 点转动,直到 $BC = OB$,于是我们有 $\angle ACO = \angle BOC$, $\angle AOB = \angle ABO = 2\angle BOC$, 所以 $\angle BOC = (1/3)\angle AOC$

鉴于前面所述定义的那些缺点,我们可以求助以下两点:第一,我们允许对有限可完成的操作概念不加定义而依靠直观理解来确定这种操作要表达的意思,这也许把我们限制在一定的标准不加以说明的算术操作上.这种处理方法看起来似乎不大严密,但是,如果我们回想一下,在任何逻辑推演的问题中,(为了能够开头)用一些未加定义的元素作为这种推演的开始是十分必要的,这种处理方法并非没有道理,或许还是正确的. 这就像采用整数作为(打个比方说)数学的"基本建筑构件"而不打算用更基本的概念来定义它一样. 这种对基本的有限可执行操作不加说明的处理方法,在 E. Bishop[2] 给出的结构数学分析的讲解中被采用.

另一方面,我们可以引进一些设备或设备种类,同时定义能在这些设备上执行的过程为"机械的"或"构造的"过程. 类似地,我们可以描述一定的符号串处理,其中可能包括数字串上的算术操作,然后根据这些串处理定义允许的基本操作. 于是我们就可以只考虑适合这个机构的那些操作. 在下面的章节里,我们要详细讨论用图灵机实现第一种处理方法. 同时我们也要考虑等效的符