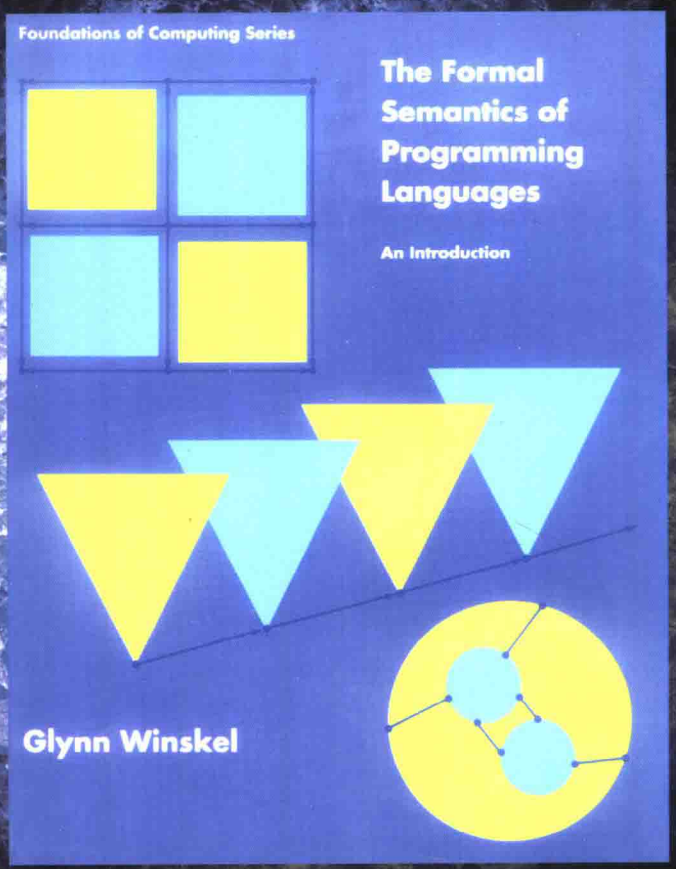
计 算 机 科 学 丛 书

程序设计语言的形式语义

Glynn Winskel 著 宋国新 邵志清 等译



The Formal Semantics of Programming Languages

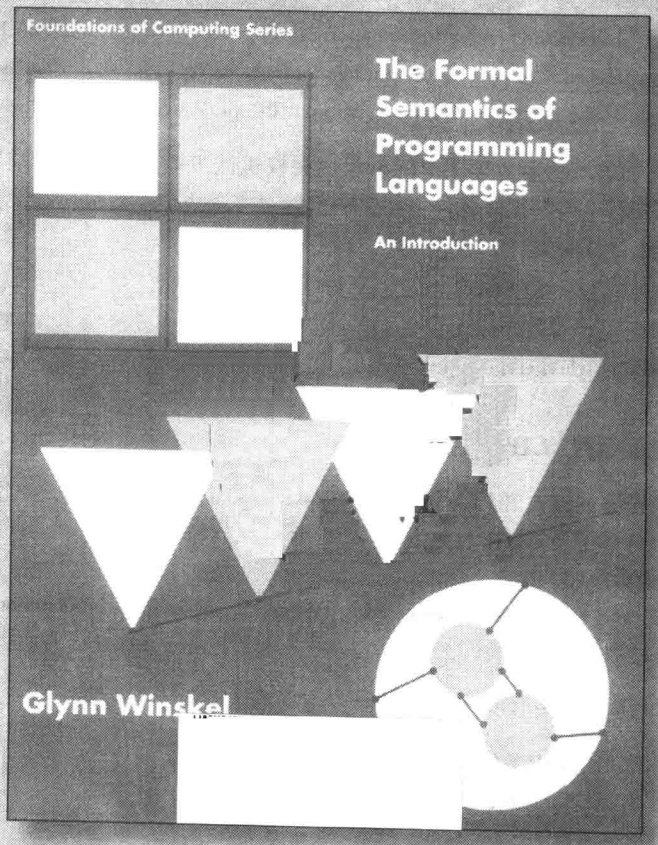
An Introduction

机械工业出版社 China Machine Press
中信出版社 CITIC PUBLISHING HOUSE

计 算 机 科 学 丛 书

程序设计语言的形式语义

Glynn Winskel 著　宋国新　邵志清　潘俊　孙霖　等译



The Formal Semantics of Programming Languages

An Introduction

机械工业出版社 China Machine Press
中信出版社 CITIC PUBLISHING HOUSE

本书是形式语义方面的一本经典之作，被国内外很多大学选作教材。书中包括集合论基础、指称语义、操作语义、公理语义、归纳原理、归纳定义、完备性、域论、递归方程、递归技术、高阶类型语言、信息系统、递归类型、不确定性和并行性、不完备性和不可判定性等内容。同时，本书始终强调指称语义和操作语义的联系，并给出它们的一致性证明。书中包含了丰富的练习。

本书以作者在剑桥大学和 Aarhus 大学的讲义为基础编写而成，可以作为计算机专业和数学专业的本科生和研究生形式语义课程的教材，也适用于软件开发人员参考。

Glynn Winskel: The Formal Semantics of Programming Languages: An Introduction (ISBN: 0-262-23169-7).

本书版权登记号：图字：01-2003-8856

图书在版编目(CIP)数据

程序设计语言的形式语义/温斯克尔(Winskel, G.)著；宋国新等译. -北京：机械工业出版社，2004.1

(计算机科学丛书)

书名原文：The Formal Semantics of Programming Languages: An Introduction

ISBN 7-111-13153-3

Ⅰ.程… Ⅱ.①温…②宋… Ⅲ.程序语言-形式语义学-高等学校-教材 Ⅳ.TP301.2

中国版本图书馆 CIP 数据核字(2003)第 092736 号

机械工业出版社(北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑：杨海玲

北京昌平奔腾印刷厂印刷·新华书店北京发行所发行

2004 年 1 月第 1 版第 1 次印刷

787mm×1092mm 1/16·18.75 印张

印数：0 001-4 000 册

定价：32.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

本社购书热线：(010) 68326294

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭橥了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短、从业人员较少的现状下，美国等发达国家在其计算机科学发展的几十年间积淀的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及庋藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专诚为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都步入一个新的阶段。为此，华章公司将加大引进教材的力度，在“华章教育”的总规划之下出版三个系列的计算机教材：除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”；同时，引进全美通行的教学辅导书“Schaum's Outlines”系列组成“全美经典学习指导系列”。为了保证这三套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这三套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业

的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证，但我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。教材的出版只是我们的后续服务的起点。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

电子邮件：hzedu@hzbook.com
联系电话：（010）68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周立柱	周克定	周傲英	孟小峰	岳丽华
范　明	郑国梁	施伯乐	钟玉琢	唐世渭
袁崇义	高传善	梅　宏	程　旭	程时端
谢希仁	裘宗燕	戴　葵		

秘　书　组

武卫东　　温莉芳　　刘　江　　杨海玲

译 者 序

“计算机科学与技术”这门学科，顾名思义，要求计算机工作者既具有坚实的计算机科学基础理论方面的涵养，又具备熟练解决实际问题的综合能力。几十年来，计算机科学理论方面的研究结果，为计算机技术与应用的发展提供了有力的支撑；同样，不断迅速发展的应用领域又为理论工作者提出了更多更新的课题。实践表明，正是理论与应用的紧密联系和相互渗透，才使计算机科学不断向前蓬勃发展。

程序设计语言是计算机科学的一个重要组成部分，是理论和应用紧密联系的典型。程序设计语言的每一次技术飞跃都极大地推动了整个计算机科学的发展。尽管程序设计语言品种繁多、层次不一，但总体上讲，与之关系密切的人员主要有三类：一类是程序设计语言的设计者，他们针对具体的应用定义语言的要素；另一类是程序设计语言的实现者，他们针对具体的机器提出语言的实现方案；再一类是程序设计语言的使用者，他们针对具体的问题进行程序设计以给出相应的解决方案。这样，人们可以从各种不同的角度来观察和刻画程序设计语言，从而导致三种主要的语义：指称语义、操作语义和公理语义。这三种语义的精确描述需要严密的数学工具。在长期寻找数学工具的过程中，逐渐形成了有关域论、推导规则、正确性断言等方面的理论和方法，奠定了形义语义学的基础。

本书涉及的程序设计语言既包括命令式语言又包括函数式语言，既包括顺序语言又包括并行语言，既包括确定性语言又包括不确定性语言。书中介绍了支撑这些语言的形式语义的数学理论、方法和概念。读者掌握了这些必要的数学知识，就可以运用它们去创造规则、刻画规则和证明规则，从而可以描述和推导各类程序设计语言的各种成分和性质。

本书第 1 章到第 10 章由潘俊和宋国新翻译，第 11 章到第 14 章由孙霖和宋国新翻译，附录由丁志义翻译，全书由宋国新和邵志清负责整理和统稿。

本书结构合理、内容丰富、蔚为大观，是一本难得的形式语义学的经典著作和教材。译者在翻译过程中获益良多，但限于译者水平，译本中错误和疏漏之处在所难免，读者如果发现译本中的错误，请与译者联系：gxsong@ ecust. edu. cn 或 zshao@ ecust. edu. cn。

译者于华东理工大学

2003 年 11 月 28 日

译者简介

宋国新，教授，博士生导师。1945 年生，1968 年本科毕业于华东水利学院水港系，1983 年毕业于上海交通大学计算机软件专业，获工学硕士学位。1983 ~ 1987 年在上海交通大学计算机系任教。1987 年至今在华东理工大学信息学院计算机系任教，1991 ~ 1993 年在美国休斯顿大学计算机系作访问学者。曾任华东理工大学计算机系主任兼计算中心主任，现任信息学院副院长，计算机研究所所长。主要研究程序验证、软件测试、软件开发方法、嵌入式系统软硬件协同设计等方向。主持“并行程序的验证”等多项国家自然基金和 863 项目的研究工作。已在国内外主要学术刊物上发表论文 30 多篇。获教育部科技进步二等奖两项，国家教学改革成果二等奖一项，上海市教学改革成果一等奖一项。

邵志清，教授、博士生导师，现任华东理工大学计算机科学与工程系系主任。1966 年生，1986 年毕业于南京大学数学系，获理学学士学位；1989 年毕业于中国科学院软件研究所，获理学硕士学位；1998 年毕业于上海交通大学计算机科学与工程系，获工学博士学位；2001 ~ 2002 年以国家杰出访问学者身份赴美国 Rensselaer Polytechnic Institute 计算机系进行合作研究。主持国家自然科学基金项目等多项课题，发表学术论文 60 多篇，获教育部科技进步二等奖；出版教材《离散数学》，受聘为教育部高等学校本科教学工作水平评估中青年专家，获国务院政府特殊津贴、霍英东教育基金会高等院校青年教师奖、上海高校优秀青年教师等奖励。

前　言

在给出程序设计语言的形式语义时，我们着重于建立一个数学模型。目的是使其作为理解程序以及对程序的行为进行论证的基础。数学模型不仅对于各种各样的分析和验证是有用的，同时在更为基础的层次上，应认识到数学模型的重要性在于通过它来精确地刻画程序构造的含义时更能显示出各种精妙细微之处。本书主要介绍支撑形式语义的数学理论、技术和概念。

由于历史的原因，人们通常认为程序设计语言的语义由三条主线构成：

- 操作语义通过规定程序设计语言在抽象机器上的执行过程来描述程序设计语言的含义。我们侧重于“结构化操作语义”，这是 Gordon Plotkin 在 Aarhus 大学的讲座中提倡的方法，由规则规定求值和执行关系，它是语法制导的。
- 指称语义是由 Christopher Strachey 首先提出的一种定义程序设计语言含义的技术，Dana Scott 奠定了它的数学基础。由于使用了完全偏序、连续函数和最小不动点等更为抽象的数学概念，曾一度被称为“数学语义”。
- 公理语义试图通过在程序逻辑的范围内给出证明规则来确定程序设计构造的含义，该方法的代表人物是 R. W. Floyd 和 C. A. R. Hoare。因此，从一开始，公理语义就强调正确性的证明。

然而，将这三种不同类型的语义彼此对立起来是错误的。它们各有各的用处。一个清晰的操作语义有助于语言的实现。公理语义对于特定类型的语言可以给出十分优美的证明系统，对开发和验证程序是有用的。在强大的数学理论支持下，指称语义提供了最有深度和最为广泛的可应用技术。事实上，这些不同的语义类型是相互高度依赖的。例如，要证明公理语义的证明规则是正确的，必须依赖于基本的指称语义或操作语义。要证明语言的实现是正确的，正是要依据指称语义去判断，要求证明操作语义和指称语义是一致的。同时，在操作语义的论证中使用指称语义有巨大的帮助，指称语义往往有远离不重要的实现细节的优势，并给出用来理解计算行为的高层概念。近几年的研究使这几个不同的方法有望得到统一，我们希望能看到指称语义、操作语义和程序逻辑携手共同发展。本书的一个目标是指出操作语义和指称语义是如何取得一致的。

语义学中使用的技术非常是依赖于数理逻辑。如果没有一个好的逻辑知识背景，这些技术并不总能很容易地被计算机科学专业或数学专业的学生所接受。在这里，我们试图以一种完整但尽量基本的方式提出语义。例如，在对操作语义进行介绍之后，引出了归纳定义的论述和推导操作语义的技术，从而使我们处于一个很好的位置，进一步学习抽象的指称语义的基础——完全偏序和连续函数。从操作语义的有限规则到集合上的连续算子，再到连续函数，这种过渡安排有助于理解为什么可计算函数要求有连续性。关于各种归纳原理，包括更一般的良基递归原理的论述，对于在良基关系的集合上定义函数是很重要的。在对递归类型语言的更为深入的研究中，利用信息系统不仅可以给出求解递归域方程的一个基本方法，同时还产生

了能结合操作语义和指称语义的技术。

本书简介

本书以作者在剑桥大学和 Aarhus 大学的讲座内容为基础，主要针对计算机科学专业和数学专业的本科生和研究生而编写，可作为开始学习形式化和推导程序设计语言的方法的教材。本书介绍了必要的数学背景知识，读者可以运用它们去创造、形式化和证明一些规则，使用这些规则可推导各种各样的程序设计语言。本书的内容是基础的，但其中有一些主题来自于最近的研究。书中包含了丰富的从简单到复杂的练习。

本书首先介绍集合论基础，接着是结构化操作语义，并将其作为定义程序设计语言含义的一种方式，同时也介绍了一些基本的证明技术。对指称语义和公理语义是以一个简单的 while 程序语言为例进行说明的，并给出了操作语义和指称语义之间等价的完整证明，以及公理语义的可靠性和相对完备性，也包括哥德尔不完备性定理的一个证明。该定理强调公理语义不可能达到绝对的完备性，这一结论可以从附录中得到支持，附录基于 while 程序介绍了可计算性理论。在域论之后，介绍了指称语义的基础，论述了几种函数式语言的语义和证明方法。最简单的函数式语言是既可以传值调用也可以传名调用求值的递归方程。这些研究工作可以进一步扩展到含有高阶类型和递归类型的语言，其中包括对活性和惰性 λ 演算的论述。本书始终强调指称语义和操作语义的联系，并给出它们的一致性证明。本书较高深的部分之一是递归类型的论述，它要利用信息系统来表示域。在最后一章里介绍了并行程序设计语言，并讨论了不确定性和并行程序的验证方法。

本书的使用方法

下面标明了各章之间的依赖关系，希望有助于阅读、参考和设计讲课内容。例如，“逻辑和计算”课程的内容可以只用第 1 ~ 7 章以及附录。附录包含了第 7 章要用到的概念“可计算性”，如果读者已学习过这方面的知识，则可以跳过附录。此外，像“语义学导论”这样的课程只用第 1 ~ 5 章的内容就可以了，也可以加上第 14 章。第 8、10、12 章的内容可作为“域论”课程的基础内容，有时可能要简单地引用一下第 5 章的内容。第 8 ~ 13 章可以作为“函数式程序设计的数学基础”课程的内容。第 14 章大体上自成体系，只加上第 2 章就可以作为“不确定性和并行性”课程的概论，只是关于模型检查的讨论要用到克纳斯特-塔尔斯基定理，它的证明在第 5 章。

有一些练习包含了一些小的实现任务。在 Aarhus 的课上，我们发现使用诸如 Prolog 之类的语言很有帮助，可以使前面的操作语义的讲述更为生动。在讲述后面几章的语言时，用标准 ML 语言或 Miranda 语言可能更合适一些。

致谢

首先，我要感谢 Dana Scott 和 Gordon Plotkin 所做的基础性工作，他们从根本上影响了本书。在阅读过程中，读者会发现本书受到 Gordon Plotkin 的著作，尤其是他在爱丁堡大学关于完全偏序和指称语义的讲义的极大影响。

在剑桥大学，Tom Melham、Ken Moody、Larry Paulson 和 Andy Pitts 的评论意见对我很有帮助（特别是 Andy 的讲义和对 Eugenio Moggi 工作的评论意见都已写进了域论一章中）。在 Aar-

hus 大学，Mogens Nielsen 提供了有价值的反馈和鼓励，他用最早的草稿开设了课程。Erik Meineche Schmidt 的建议改进了相对完备性定理和哥德尔定理的证明。在 Aarhus 大学，众多的学生提供了改正和建议，我要特别感谢 Henrik Reif Andersen、Torben Brauner、Christian Clausen、Allan Cheng、Urban Engberg、Torben Amtoft Hansen、Ole Hougaard 和 Jakob Seligman。附带要感谢 Bettina Blaaberg Sφrensen，在准备本书的各个阶段，她都快速地阅读并提出建议。感谢 Douglas Gurr 对有关域论的几章提出了诚恳的批评意见。Kim Guldstrand Larsen 对不确定性和并行性一章提出了改进意见。

1991 年秋，Albert Meyer 以本书为基础开设了课程，他和讲师 A. Lent、M. Sheldon 和 C. Yoder 非常友善地对打印错误和证明结构的调整给出了很多宝贵的意见。另外，Albert 慷慨地提供了可计算理论的讲义，作为附录的基础。我感谢他们，但愿他们对最后的结果不感到失望。

感谢 Karen Mφller 在文字录入方面提供的帮助。最后，对 MIT 出版社，特别是 Terry Ehling 的耐心表示感谢。

各章之间的关系

目　录

第1章　集合论基础

本章介绍非形式的逻辑记号、集合记号及其基本概念，后面的论证在此基础上展开。它来源于人们的日常语言，用于讨论类似于集合这样的数学对象；但它与我们稍后讨论的程序设计语言的形式语言或形式断言又有不同，二者不应该混淆。

本章是对基础知识的总结，并作为后续章节的参考。建议读者先快速通读一遍，不要求全部理解吸收。

1.1　逻辑记号

我们将使用非形式的逻辑记号来简洁地表示数学命题。例如，对于命题（断言）A 和 B，我们通常使用以下形式的缩写：

- $A \,\&\, B$（A 与 B），即 A 和 B 的合取。
- $A \Rightarrow B$（A 蕴涵 B），即如果 A 则 B。
- $A \Leftrightarrow B$（A 当且仅当 B，简写为 A iff B），即 A 等价于 B。

用析取运算表示命题“A 或 B”，记作 $A \vee B$；用否定运算表示“非 A”，记作 $\neg A$，其值为真当且仅当 A 为假。命题“7 不小于 5”通常不记作 $\neg(7<5)$，而是记作 $7 \not< 5$，以符合我们的表达习惯。

命题中常常会包含变量（未知的或占位用的），例如

$$(x \leqslant 3) \,\&\, (y \leqslant 7)$$

这个命题在变量 x 和 y 分别取小于等于 3 和 7 时为真，否则为假。像 $P(x,y)$ 这样含有变量 x 和 y 的命题称为谓词（或性质、关系、条件），它在 x 和 y 取某一特定值的时候为真或假。

逻辑量词 $\exists$ 读作“存在”，量词 $\forall$ 读作“对于所有的”。断言

$$\exists x.\, P(x)$$

读作“对于某些 x，$P(x)$”或者“存在 x，使得 $P(x)$”，断言

$$\forall x.\, P(x)$$

读作“对于所有的 x，$P(x)$”或者“对于任何 x，$P(x)$”。断言

$$\exists x \exists y \cdots \exists z.\, P(x,y,\cdots,z)$$

简记为

$$\exists x,y,\cdots,z.\, P(x,y,\cdots,z)$$

并且，断言

$$\forall x \forall y \cdots \forall z.\, P(x,y,\cdots,z)$$

简记为

$$\forall x,y,\cdots,z.\, P(x,y,\cdots,z)$$

我们经常希望确定量词对集合 X 的作用范围。我们把 $\forall x.\, x \in X \Rightarrow P(x)$ 记作 $\forall x \in X.\, P(x)$，把 $\exists x.\, x \in X \,\&\, P(x)$ 记作 $\exists x \in X.\, P(x)$。

还有一个与量词有关的符号，有时我们可能想表达：存在满足 $P(x)$ 性质的惟一 x。约定把

$$(\exists x.\, P(x)) \,\&\, (\forall y,z.\, P(y) \,\&\, P(z) \Rightarrow y = z)$$

简记为

$$\exists !x.\, P(x)$$

其含义是存在满足性质 P 的 x，且如果任意的 y,z 都满足性质 P，则必有 $y=z$，即存在满足 $P(x)$ 的惟一 x。

1.2 集合

直观上讲，集合就是一组无序的对象，称这些对象为集合的*元素*或者*成员*。我们用 $a \in X$ 表示 a 是集合 X 的一个元素，用 $\{a,b,c,\cdots\}$ 表示由元素 $a,b,c,\cdots$ 组成的集合。

称集合 X 是集合 Y 的*子集*，当且仅当 X 中的所有元素也都是 Y 中的元素，记作 $X \subseteq Y$，即

$$X \subseteq Y \Longleftrightarrow \forall z \in X.\, z \in Y$$

集合由它的元素惟一确定，称两个集合相等当且仅当它们含有相同的元素。因此，集合 X 和 Y 相等，记作 $X=Y$，当且仅当集合 X 中的每一个元素都是集合 Y 中的一个元素，反之亦然。这
2 里提供了一种方法来证明两个集合 X 和 Y 相等，等价于证明 $X \subseteq Y$ 且 $Y \subseteq X$。

1.2.1 集合与性质

集合还可以由性质确定，这时集合的元素恰好满足这个性质，记作：

$$X = \{x \mid P(x)\}$$

表示集合 X 由所有满足性质 $P(x)$ 的 x 组成。

在集合论建立时，人们曾认为任何性质 $P(x)$ 都确定了一个集合

$$\{x \mid P(x)\}$$

但是罗素（Bertrand Russell）发现，这种集合的表示方法会推出矛盾，该结果令人震惊。

罗素悖论表明，按照上述不加限制的方法构造集合，会推出矛盾。由以下步骤开始：考察性质

$$x \notin x$$

它表示“x 不是 x 的元素”。如果性质可以确定集合，则根据该性质的描述，我们就建立了集合

$$R = \{x \mid x \notin x\}$$

这时或者 $R \in R$;或者 $R \notin R$。如果 $R \in R$,那么 R 就是 R 的一个元素,由 R 的定义可以推出 $R \notin R$; 如果 $R \notin R$,由 R 的定义又可以推出 $R \in R$,可见 $R \in R$ 或者 $R \notin R$ 都会推出矛盾,这就是罗素悖论。

人们需要用其他方法来表示集合以避免上面的困境。一个解决办法是规定集合的构造方法,从给定的初始集合开始,规定新的集合只能通过特定的、安全的方法从已有的集合中构造出来。我们假定总存在这样的初始集合和构造方法。避开罗素悖论后,集合论强大的表述能力可以为我们提供数学上的理论支撑。

1.2.2 一些重要集合

本书用特定的符号字母表示常用的基本集合。约定:

- $\emptyset$ 表示空集。
- ω 表示由自然数 0,1,2,…组成的集合。

3

我们一般用{"a","b","c","d","e",…,"z"}这样的集合表示符号,尽管它也可以表示为特定的数字。符号集合上的相等关系是指语法恒等式给出的相等性; 两个符号相等当且仅当它们是相同的符号。

1.2.3 集合的构造

我们先给出集合上的几个运算,以便从给定的集合构造新的集合。

概括 如果 X 是一个集合,$P(x)$ 是一个性质,则我们能构造一个集合

$$\{x \in X \mid P(x)\}$$

它也可以记作

$$\{x \mid x \in X \,\&\, P(x)\}$$

它是集合 X 的子集,由 X 中所有满足 $P(x)$ 的元素 x 组成。

我们还可以使用进一步简化的记号,设 $e(x_1,\cdots,x_n)$ 是某一表达式,它对特定的元素 $x_1 \in X_1,\cdots,x_n \in X_n$ 产生一个特定的元素,并且 $P(x_1,\cdots,x_n)$ 是这些 $x_1,\cdots,x_n$ 的一个性质,则我们把

$$\{y \mid \exists x_1 \in X_1,\cdots,x_n \in X_n.\, y = e(x_1,\cdots,x_n) \,\&\, P(x_1,\cdots,x_n)\}$$

简记为

$$\{e(x_1,\cdots,x_n) \mid x_1 \in X_1 \,\&\, \cdots \,\&\, x_n \in X_n \,\&\, P(x_1,\cdots,x_n)\}$$

例如,

$$\{2m+1 \mid m \in \omega \,\&\, m > 1\}$$

表示大于 3 的奇数集合。

幂集 集合的幂集是由该集合所有子集组成的集合,记作

$$\mathcal{P}ow(X) = \{Y \mid Y \subseteq X\}$$

加标集合 假设 I 是集合,对任意的 $i \in I$,存在一个惟一的对象 x_i,(x_i 本身可能是集合),于是

$$\{x_i \mid i \in I\}$$

[4] 是一个集合。称元素 x_i 由元素 $i \in I$ 加标。

并集 两个集合 X 和 Y 的并集定义为

$$X \cup Y = \{a \mid a \in X \text{或} a \in Y\}$$

广义并集 设 X 是集合的集合，则集合

$$\bigcup X = \{a \mid \exists x \in X.\, a \in x\}$$

称为广义并集。如果 $X = \{x_i \mid i \in I\}$，I 为某个加标集合，则 $\cup X$ 通常记作 $\cup_{i \in I} x_i$。

交集 两个集合 X 和 Y 的交集 $X \cap Y$ 的元素是同时在这两个集合中的元素，即

$$X \cap Y = \{a \mid a \in X \,\&\, a \in Y\}$$

广义交集 设 X 是集合的非空集合，则集合

$$\bigcap X = \{a \mid \forall x \in X.\, a \in x\}$$

称为广义交集。如果 $X = \{x_i \mid i \in I\}$，I 为非空加标集合，则 $\cap X$ 通常记作 $\cap_{i \in I} x_i$。

积 两个元素 a, b 的序偶记作 (a,b)。如果用集合来表示序偶，则序偶 (a,b) 可以表示成 $\{\{a\},\{a,b\}\}$——这是将序偶编码为集合的一种特殊方式。显然，用这种方式表示的两个序偶相等当且仅当它们的第一个分量和第二个分量都分别相等，即

$$(a,b) = (a',b') \Longleftrightarrow a = a' \,\&\, b = b'$$

不管用什么方法将序偶表示为集合，在证明序偶的性质时，这个性质应该总是成立的。

练习 1.1 对于推荐的序偶的表示方法，试证明上面的性质成立。（不要以为这很容易！在遇到困难的情况下，读者可以参考文献[39]的第 36 页或文献[47]的第 23 页。） □

集合 X 和 Y 的积定义为

$$X \times Y = \{(a,b) \mid a \in X \,\&\, b \in Y\}$$

它是由 $a \in X$ 和 $b \in Y$ 构成的序偶 (a,b) 组成的集合。

推广这个概念，三元组 (a,b,c) 是集合 $(a,(b,c))$，并且积 $X \times Y \times Z$ 是三元组的集合 $\{(x,y,z) \mid x \in X \,\&\, y \in Y \,\&\, z \in Z\}$。更一般地，积 $X_1 \times X_2 \times \cdots \times X_n$ 是 n 元组 $(x_1,x_2,\cdots,x_n) =$
[5] $(x_1,(x_2,(x_3,\cdots)))$ 组成的集合。

不相交并集 在由并运算得到的新集合中，如果每一个元素都能够指明自己来源于哪个集合，就称这样的并运算为不相交的并。可以将集合中的元素拷贝过来加以标志，以便在它们来自不同的集合时使其可以被强制区分。不相交并集定义为

$$X_0 \uplus X_1 \uplus \cdots \uplus X_n = (\{0\} \times X_0) \cup (\{1\} \times X_1) \cup \cdots \cup (\{n\} \times X_n)$$

特别地，对于 $X \uplus Y$，$(\{0\} \times X)$ 和 $(\{1\} \times Y)$ 是不相交的，即