

信息科学与技术基础丛书

数理逻辑

基本原理与形式演算

(第二版)

李 未 著



科学出版社

信息科学与技术基础丛书

数理逻辑

基本原理与形式演算

(第二版)

李 未 著



科学出版社

北 京

内 容 简 介

数理逻辑是以公理系统和数学证明为研究对象的数学分支,对信息科学与技术的发展具有指导作用。本书共分十章,系统介绍数理逻辑的基本原理与形式演算。前五章涵盖了经典数理逻辑的核心内容,包括一阶语言的语法与模型,形式推理系统,可计算性与可表示性,哥德尔定理。后五章的内容是作者的研究心得。这部分阐述了形式理论的版本序列及其极限,修正演算系统,过程模式及其性质,以及归纳推理理论,给出了三个语言环境的思想以及元语言环境的基本原理,并描述了信息社会中科学研究的工作流程。

本书前五章可作为大学本科生的数理逻辑教材,而后五章可向有关专业的研究生讲授。本书也可供数学、信息与技术和其他自然科学专业的本科生、研究生和有关科研人员参考。

图书在版编目(CIP)数据

数理逻辑:基本原理与形式演算/李未著. —2版. —北京:科学出版社, 2014.3

(信息科学与技术基础丛书)

ISBN 978-7-03-040155-7

I. ①数… II. ①李… III. ①数理逻辑 IV. ①O141

中国版本图书馆CIP数据核字(2014)第045299号

责任编辑:林 鹏 陈玉琢/责任校对:林青梅

责任印制:钱玉芬/封面设计:王 浩

科学出版社 出版

北京东黄城根北街16号

邮政编码:100717

<http://www.sciencep.com>

北京通州皇家印刷厂 印刷

科学出版社发行 各地新华书店经销

*

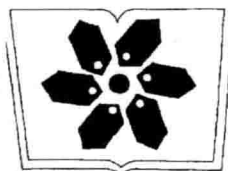
2014年3月第 二 版 开本:720×1000 1/16

2014年3月第一次印刷 印张:19 1/2

字数:364 000

定价:98.00元

(如有印装质量问题,我社负责调换)



中国科学院科学出版基金资助出版

《信息科学与技术基础丛书》编委会

主 编：李 未

副 主 编：何积丰 卢锡城

编 委：王东明 孙家广 邬贺铨 李国杰

李德毅 应明生 林惠民 周志华

郭光灿 梅 宏 傅育熙 潘云鹤

执行编委：王东明

《信息科学与技术基础丛书》序

迅猛发展的信息科学与技术将人类社会带入了信息时代. 半个多世纪以来, 众多学者为信息科学的建立、信息技术的发展和信息社会的进步作出了不懈的努力. 然而, 要为信息科学与技术奠定坚实的基础, 我们仍然面临很多问题与挑战. 无论是科学研究和技术革新, 还是基础教育和学科建设, 都需要我们继续付出艰苦的劳动.

如何加强和深化信息科学的基础研究, 完善已有的创新成果, 为信息技术的持续发展提供保障? 如何将重要的学术思想、理论和方法系统化, 将信息科学与技术各个分支中的基本原理、核心内容和最新进展结合起来? 如何让学术论著同时服务于科研与教学, 使其相辅相成、相互促进? 这些问题的解决将会使我国信息科学与技术的基础教育和学术研究的质量得到显著提高.

《信息科学与技术基础丛书》是为解决上述问题所做的一种尝试. 这套丛书旨在收录信息科学与技术领域的研究专著. 要求每部著作中都有一定篇幅的内容是著者的原创性研究成果, 并且结构合理、内容充实、论证严密、写作规范. 这套丛书可以作为研究生和高年级本科生的教材和教学参考书. 入选这套丛书的每部著作都至少需要通过两位同行专家的评审.

这套丛书的出版离不开作者、出版社与编委会的通力合作和诸多同仁的支持. 在此我们向支持丛书出版工作的所有个人和单位表示诚挚的谢意, 并热切期待来自读者的批评、指正和帮助, 使这套丛书能够不断完善, 成为信息科学与技术方面的重要文献.

李未

2007年11月

于北京航空航天大学

再版前言

本书自 2007 年第一版发行起,至今已整 7 年了. 2009 年第二次印刷前,我曾对当时已发现的疏漏进行了修改,但仍有几个问题自认为解决得不够理想. 这几年,我一直在思考这些问题,且取得了一些新进展. 去年年底,与眭跃飞、罗杰一起重读了此书,以将取得的进展纳入书中加以介绍. 此次再版的增补和修改主要有以下四个方面:

一、关于 Gentzen 逻辑推理系统 (**G** 系统) 完全性定理的证明. 第一版在讨论使用 **G** 系统进行形式证明时,我们给出了一个宽度优先搜索算法,称为 *CP* 过程,并证明对任意给定输入序贯 $\Gamma \vdash A$, 如果 *CP* 过程停机并输出一棵证明树,那么 $\Gamma \vdash A$ 可证 (引理 3.1). 由此只要再证明: 如果 *CP* 过程输出的是有穷推理树但非证明树或者输出一棵无穷推理树,那么输入序贯有反例. 这就是第二版新增的引理 3.6. 有了这两条引理,用反证法就可证明 **G** 系统的完全性,即如果 $\Gamma \models A$ 有效,那么 $\Gamma \vdash A$ 可证. 这种证明方法的好处一是删除规则不再作为 **G** 系统的一条基本规则出现,证明过程简单直观,更符合本书强调的“将数学证明转变为符号演算”的基本思想;二是引理 3.1 和引理 3.6 的证明都是构造性的,对使读者了解 **G** 系统每一条规则在形式证明中的作用是有益的.

二、关于科学发现的逻辑. 第二版新增加了一节,即 7.6 节,用于讨论 Karl Popper 在科学哲学层面给出的科学发现的逻辑 [Popper, 1959] 与本书的关系. 通过对爱因斯坦发现狭义相对论和达尔文创立进化论的历史过程的分析,本书首先将 Popper 关于科学发现的逻辑具体化为科学家在工作层面可以使用的操作过程. 然后,在此操作过程的每一步,使用本书引入的 **G** 系统或 **R**-演算,完成对这两个典型科学发现事例的形式化验证. 进而在—阶语言可描述的范围内,将科学发现的操作过程转变为形式化的演算过程.

三、关于不含删除规则的 **R**-演算. 第一版第 7 章引入了 **R**-演算,它由 7 条规则组成. 当一个形式理论受到事实反驳时,使用 **R**-演算可以推演出与该反驳协调的现有理论的所有极大子集. 这个演算的不足是: 第一版给出的 **R**-演算包含了一条 **R**-删除规则,即如果现有理论的某一个逻辑结论受到反驳,那么该理论中,至少一个充当此逻辑结论必要前提的原理必须被删除. 这条规则的优点是直观,但它不是关于逻辑连接词和量词符号的演算规则,会增加实现 **R**-演算自动推理的难度. 它更适合作为一条引理. 为了弥补此缺憾,第二版第 7 章新增加了不含删除规则的 **R**-演算一节

(7.7 节). 此节由罗杰完成, 增加了几条关于逻辑连接词符号和量词符号的规则, 将 **R**-删除规则替换下来. 这个演算系统称为不含删除规则的 **R**-演算.

四、关于程序调试的结构操作语义. 在软件开发过程中, 如果某个程序的执行结果与程序设计者的预期不一致, 那么就需要定位程序错误并加以改正, 这就是程序调试的任务. 从发现错误并加以改正这个角度来看, 这与 **R**-演算的宗旨是相通的. 所不同的只是语言环境: 前者是程序语言, 后者是一阶语言. 那么程序调试中与 **R**-演算对应的演算系统应该是什么? 本书的答案是程序调试的形式语义. 再版第 7 章新增加了一节 (7.8 节), 对程序的几个基本结构给出了程序调试的操作语义.

以上就是此次再版增补修改的主要内容.

本书的内容大致分为两部分. 第一部分是数理逻辑基础, 包括前五章, 即一阶语言的语法、语义、推理系统、可计算性、不完备性等理论. 第二部分由后五章组成. 可称为知识进化的逻辑基础, 包括一阶语言的版本序列、修正演算、过程模式、归纳推理和语言环境等. 对于熟悉一阶语言希望了解知识进化的逻辑的读者, 作者建议: 可以跳过第一部分, 或在浏览、阅读前三章之后, 跳过第4和第5两章, 直接阅读第二部分内容.

在第二版即将付梓之时, 我要感谢所有关心此书, 指出书中错误和疏漏的读者. 正是他们的批评和建议才促使我认真思考这些问题, 提高本书的质量. 我特别要感谢陆跃飞和罗杰. 他们和我一起完成了本书第二版的修订补充工作. 这两位年轻朋友概念清楚、思维缜密、工作认真, 若无他们的参与, 第二版还不知要拖多久才能完成. 我还要感谢夫人孟华、女儿李晓耕对我始终如一的支持和鼓励, 以及在各个方面的照顾和帮助. 最后, 我要衷心感谢科学出版社长期的帮助和支持.

作 者

2013 年冬至写于北京

第二次印刷说明

本书出版后,为了检验它是否适合读者自学之用,我组织了一个讨论班,邀请了计算机和数学专业的十二位同学和两位老师参加.他们中间有博士生和硕士生,也有大学三年级的学生.讨论班的做法是参加者以自学为主,每人在讨论班上讲一章或其相关内容,之后进行讨论.半年的实践证明这样做的效果是好的.对于像一阶语言形式推理系统的完全性、可表示性定理、哥德尔不完全性定理和修正演算等,这些我过去花了数年时间才搞明白的内容,他们一次就基本抓住了要点.

这个讨论班的副产品是,对本书中的错误和疏漏又进行了一次像软件测试一样的排查.根据排查的结果,我对本书作了修订.其中主要的修改包括:第3章推理树的定义,第4章例4.3关于循环不变式的表示和第9章引理9.3的证明.这些修改将使本书在第二次印刷后错误更少.在这里我要向参加讨论班的同学和老师表示衷心的感谢.

最后需要特别指出,本书的写作和出版都是在国家重点基础研究计划关于海量信息的组织与管理的两个项目的支持下完成的.项目编号是G1990032700和2005CB321900.

作 者

2008年9月5日

前 言

经典数理逻辑是数学基础的重要组成部分，是以数学的方法，特别是公理系统和数学证明为研究对象的数学分支。它的核心内容包括一阶语言的语法理论、一阶语言的模型理论、形式推理系统及其可靠性和完全性、可计算性理论和 Gödel 的两个定理，即形式理论的不完全性定理和关于形式理论的协调性定理。从数理逻辑的发展历史来看，这些理论都是百余年来从数学的研究方法中总结和抽象出来的，主要结论大多是在 20 世纪 40 年代之前证明的。

1990 年，作者在德国萨尔布吕肯大学 (Universität des Saarlandes) 担任了一年祖思 (Konrad Zuse) 讲座教授，给计算机系学生开设了一门数理逻辑课程；之后又在丹麦和国内讲了几次，并对其中的某些内容做过多次专题讲座。随着作者对数理逻辑理解的加深，对它的研究兴趣也越来越浓，并由此萌生了为生活在信息社会中的大学生和从事科学研究的人们，写一本数理逻辑书的想法。

产生这一愿望的原因来自下述认识。20 世纪 40 年代，人们发明了计算机，不久又定义了高级程序设计语言并在计算机上加以实现。自此之后，计算机科学、人工智能和计算机软硬件技术得到了迅猛发展。这对数理逻辑产生了深刻影响。数理逻辑的概念、理论和方法在这种发展中得到了广泛的应用，而这种发展反过来又对数理逻辑的研究提出了新的要求，这可以从两方面加以说明。

一方面，数理逻辑原本是关于数学的公理系统和数学证明的一般理论，但它所建立的概念、理论和方法，不仅在程序设计语言的研究中获得了完全的认同，而且在计算机软硬件设计和实现技术方面起到了原则性的指导作用。例如，结构归纳方法是定义一阶语言的语法而发明的，可是现在每一个程序设计语言都是用这种方法定义的，所以有关程序设计语言的性质，原则上都可以用一阶语言的方法加以研究。又如，根据一阶语言及其模型理论，Peano 算术是一阶语言的一个形式理论，而小学课本中包括加、减、乘、除四则运算的自然数系是该理论的一个模型。这种区分在理论研究方面是重要的，只有如此，Gödel 定理才得以证明，可是这种做法，对人们日常的数学和计算实践，既不自然，也无必要。但是就程序设计语言来说，如果将用此语言写的程序，与经过编译之后产生的可执行代码加以区分，并将后者与编译程序合在一起，作为该语言的模型，那么这种区分就不仅仅是理论研究上的需要，而且在实践上也是不可或缺的。若非如此，程序设计语言就无法在计算机上使用。再例如，经典数理逻辑中最难懂的部分是 Gödel 不完全性定理的证明，特别是证明中项

或逻辑公式的 Gödel 数和 Gödel 项的使用. 它之所以难懂, 是因为从纯数学的角度来看, 很难找到对这种证明方法的直观解释. 但是从程序设计的角度来看, 其直观解释就是指针的思想. 具体地说, 用指针将每一个项或逻辑公式指向其相应的 Gödel 项, 而其 Gödel 数就是指针指向的地址, 在此地址中存储着 Gödel 项. 只有这样做, 才能将项和逻辑公式这两种不同类型的语法对象都转换为一阶语言的项, 才能在一阶语言中解决“自指语句”的描述问题.

由于上述原因, 作者希望写这样一本书, 它不仅能从数学基础的角度研究数理逻辑, 而且还能在方法论层面上, 对生活在信息社会、从事科学技术研究的人们有所启迪. 因此, 这本书不仅要用抽象代数、数论等数学理论解释数理逻辑的概念、理论和方法, 还要用计算机、程序设计语言和软件的实践来解释它们, 要将数理逻辑的概念、理论和方法与计算机、程序设计语言和软件的研究与发展联系起来.

另一方面, 计算机科学、人工智能和计算机软硬件技术经过六十多年发展所积淀下来的知识, 蕴涵着经典数理逻辑所不能涵盖的概念、理论和方法. 为此, 人们进行了不懈的努力, 力求将它们抽象和总结出来, 进而丰富和充实经典数理逻辑的内容, 以便对信息科学和技术的发展起到理论上的指导作用. 因此我又希望, 这本书能在经典数理逻辑现有成果的基础上, 将自己对这些概念、理论和方法的认识和理解加以总结, 作为对经典数理逻辑进行拓广和发展的一次尝试. 概括起来, 我的主要研究工作是以下述四个基本问题为出发点展开的.

一、关于软件版本问题 首先, 每一个软件系统都是用程序设计语言编写的, 是一个形式系统, 而这个系统的需求说明, 也称规约, 可以用一阶语言的形式理论来描述. 其次, 每一个软件系统的实现都不是一蹴而就的, 它至少要由需求说明、软件编制和软件测试三部分人员, 经过频繁交流与紧密合作, 对系统进行多次测试与修改, 方可完成. 因此, 在软件开发和应用的实践中, 每一个软件系统都是以版本的形式出现的. 只有对软件系统的版本加以区分, 才能支持设计、编程和测试人员的交流与合作. 由此可见, 软件系统的版本是一个基本概念, 版本反映软件系统的进化. 随之而来的版本序列同样是基本概念, 因为它记录了一个软件系统的进化过程和变化趋势. 所以, 数理逻辑要发展, 就应该将形式理论的版本和版本序列作为新的研究对象; 只有这样做, 才能描述和研究形式理论的进化过程.

二、关于软件测试问题 测试是软件开发中不可缺少的重要环节, 只有经过严格测试的软件才能被使用. 为此, 人们研制了许多软件工具, 使测试工作日趋专业化. 尽管如此, 对软件的测试仍需要投入大量人力, 而且测试人员的素质和经验仍然与测试质量和效率密切相关, 所以软件测试尚处于“工艺性技术”阶段. 总体上说, 软件测试是由测试样例的制定和对软件错误的修正两部分组成的. 这两部分都是逻辑推理的结果. 首先, 测试样例是人们通过对软件系统的需求说明 (specification) 进行

逻辑分析之后设计出来的, 因此它是逻辑推理的结果. 其次, 在使用测试样例对软件进行测试的过程中, 如果测试结果与软件需求说明的预期不一致, 那么人们必须以需求说明的预期结果为依据, 通过逻辑推理进行错误定位, 找出出错原因, 再加以改正. 所以对软件错误的修正也是逻辑推理的结果. 如果我们将数学证明与软件错误的修正加以比较, 那么就不难看出, 数学证明是逻辑推理, 是经典数理逻辑的研究对象. 我们可以建立形式推理系统, 并证明这个推理系统的可靠性和完全性, 从而可以使用计算机作为辅助工具进行数学定理的证明. 与之相比, 软件测试中对软件错误的修正也是逻辑推理, 所以它也应该像数学证明一样, 成为数理逻辑的研究对象. 我们也可以建立形式化的错误修正系统, 进而可以使用计算机作为辅助工具, 实现软件纠错的“机械化”. 如果这个目标能实现, 那么这种在数理逻辑抽象层面上的研究成果, 将会对提高软件测试效率起到理论上的指导作用.

三、关于软件开发方法问题 每一个软件产品都是开发出来的, 软件的开发方法决定了软件的质量和成本. 总体上说, 软件开发方法主要是由构造软件系统的若干规则和流程组成的, 它的每一步, 虽然不像程序那样可以在计算机上执行, 但都必须具有可操作性, 或可构造性. 由于可计算性是经典数理逻辑的研究对象, 我们可以定义指令系统和程序设计语言, 通过编制程序解决可计算的问题. 由于计算过程只是一种特殊的构造过程, 所以构造过程也应作为数理逻辑的研究对象. 我们可以通过定义可构造性, 设计描述软件开发方法的构造性语言, 进而从数理逻辑的抽象层面, 研究软件开发方法所应具有的性质.

四、关于元语言环境问题 一阶语言和它们的模型是在元语言环境中定义并加以说明的, 而许多重要的定理也都是在元语言环境中加以证明的. 这必然对元语言环境有所要求, 有所限制. 数理逻辑发展到这一阶段, 对一阶语言的元语言环境所必须遵守的原理问题, 应该给出明确的回答.

从更广泛的意义上来说, 每一个数学和自然科学理论的形成都是一个进化过程, 而在这一过程中, 这些理论在不同阶段都具有版本的特征, 只不过它们, 特别是数学, 与软件开发相比, 通常是由为数不多的专家群体, 在很长时间内完成的, 其原理和公理的规模远比软件系统来得小, 而产生的时间又远比软件版本来得长, 所以版本的问题不像在软件系统的设计与生产中显得那么突出. 实际上, 从版本和版本序列的观点来看, 与其说经典数理逻辑是以公理系统为研究对象, 不如说它是以公理系统的一个阶段版本为研究对象更为准确. 总之, 在数学和自然科学理论不断完善的过程中, 也同样存在着版本和版本序列、理论的修正、科学研究的方法以及元语言环境等问题. 这些都是数理逻辑发展到现阶段应该解决的问题.

十几年来, 作者对这些问题的研究和探索始终没有停止过, 它已成为我生活的一部分. 研究的心得和成果, 择其要者, 都写进了这本书中. 本书共有十章, 内容分

为两部分. 第一部分由前五章组成, 主要介绍经典数理逻辑的核心内容. 第二部分由后五章组成, 主要研究形式理论的版本、版本序列及其极限、形式理论的修正、过程模式、形式理论的进化、归纳推理问题以及元语言环境的基本原理. 相对于经典数理逻辑而言, 第二部分内容都是新的, 是作者的研究心得.

由于是一本数理逻辑的专著, 本书在写作深度和严谨性方面, 以数理逻辑经典著作作为标准: 凡概念都给出了严格的定义, 并通过例子加以说明; 凡定理都给出了严格的证明, 并力求给出证明的细节; 凡引用的结论和方法均给出了原创者和出处. 本书可以作为信息科学类研究生的专业基础课教材, 前五章还可以作为大学本科生的数理逻辑教科书.

尽管在过去的几年中, 作者对书稿进行了多次重大修改, 但摆在读者面前的这一版, 仍难免有不足, 甚至错误. 作者真诚地欢迎读者的每一个批评和建议.

老伴孟华是最早建议我将自己对数理逻辑的理解和研究工作总结成书的人. 这些年, 她和女儿李晓耕将出书当成是我们家最重要的事. 没有她们的爱心和坚持不懈的支持和鼓励, 这本书还不知要拖多久才能出版. 张玉平是本书每一章和每次重要修改的第一位读者, 他热情地提出了许多有益的建议. 马声明、罗杰和作者一道, 对本书进行了仔细的阅读和讨论, 并尽其所能对所发现的错误逐一修改. 罗杰完成了电子版的编辑、排版和改错工作. 周巢尘院士阅读了本书第4章的初稿; 作者根据他的建议做了修改, 给出了可表示性定理的证明路线图, 并和罗杰、马声明一道在附录3中给出了该定理的详细证明. 对他们的鼓励、支持和帮助, 作者表示最诚挚的感谢. 在本书出版之际, 作者还要衷心感谢科学出版社, 感谢他们对本书始终保持着热情和耐心!

作 者

2007年4月28日于北航

符号对照表

$\emptyset$	空集
$\supset$	真包含
$\supseteq$	包含
$\subset$	真包含于
$\subseteq$	包含于
$\cup$	并集
$\cap$	交集
$\in$	属于
$\neg$	否定
$\vee$	或者
$\wedge$	并且
$\rightarrow$	如果……那么……
$\leftrightarrow$	当且仅当
$\forall$	对所有……
$\exists$	存在……
$\doteq$	等词
$\mathcal{L}_c$	常元符号集合
$\mathcal{L}_f$	函数符号集合
$\mathcal{L}_P$	谓词符号集合
$\mathcal{A}$	初等算术语言
$+$	加法符号
$\cdot$	乘法符号
S	后继函数符号
$\&$	Gödel 编码函数
$\max$	最大值
rk	秩
$\mathbb{N}$	自然数系统
$\vdash$	形式推理关系
$\models$	逻辑推理关系

Th	形式理论闭包
Π	初等算术理论
$\longrightarrow$	程序状态转换关系
lim	极限运算符号
$\rightsquigarrow$	归结关系
$\Vdash$	力迫关系
$\vdash$	必要前提
$\Longrightarrow$	R -转换关系
$\rightarrow$	归纳推理关系
$\bowtie$	先入协调关系

目 录

《信息科学与技术基础丛书》序

再版前言

第二次印刷说明

前言

符号对照表

第 1 章 一阶语言的语法	1
1.1 一阶语言的符号	4
1.2 项	6
1.3 逻辑公式	8
1.4 自由变元与替换	9
1.5 公式的 Gödel 项	13
1.6 结构归纳证明	16
第 2 章 一阶语言的模型	20
2.1 论域与解释	22
2.2 赋值与模型	24
2.3 项的语义	25
2.4 逻辑连接词符号的语义	26
2.5 公式的语义	28
2.6 可满足性和永真性	31
2.7 关于 $\leftrightarrow$ 的永真公式	33
2.8 Herbrand 域、基和公式集	34
2.9 Herbrand 模型	36
2.10 含有变元的 Herbrand 模型 [*]	40
2.11 替换引理	43
2.12 模型的同构	47
第 3 章 形式推理系统	52
3.1 G 推理系统	56
3.2 推理树、证明树和可证序贯	58
3.3 G 系统的可靠性	64
3.4 G 系统的完全性	68
3.5 紧致性和协调性	71

3.6	若干常用推理规则	73
3.7	证明论与模型论	75
第 4 章	可计算性与可表示性	78
4.1	形式理论	79
4.2	初等算术理论	81
4.3	$\mathbb{N}$ 上的 P 过程	83
4.4	Church-Turing 论题	87
4.5	可表示性问题	89
4.6	P 过程的存储状态	90
4.7	P 过程指令的操作演算系统	92
4.8	P 过程指令的表示*	94
4.9	可表示性定理	110
第 5 章	Gödel 定理	112
5.1	自指语句	113
5.2	可判定集合	115
5.3	Π 中的不动点方程	119
5.4	Gödel 不完全性定理	122
5.5	Gödel 协调性定理	124
5.6	停机问题	127
第 6 章	形式理论序列	131
6.1	两个例子	132
6.2	形式理论序列	136
6.3	过程模式	139
6.4	归结序列	142
6.5	缺省扩充序列	145
6.6	力迫序列	148
6.7	关于过程模式的讨论	151
第 7 章	事实反驳与修正演算	153
7.1	形式结论的必要前提	154
7.2	新猜想和新公理	157
7.3	事实反驳和极大缩减	158
7.4	R-演算	161
7.5	R-演算的可达性、可靠性和完全性	170
7.6	关于科学发现的逻辑	177