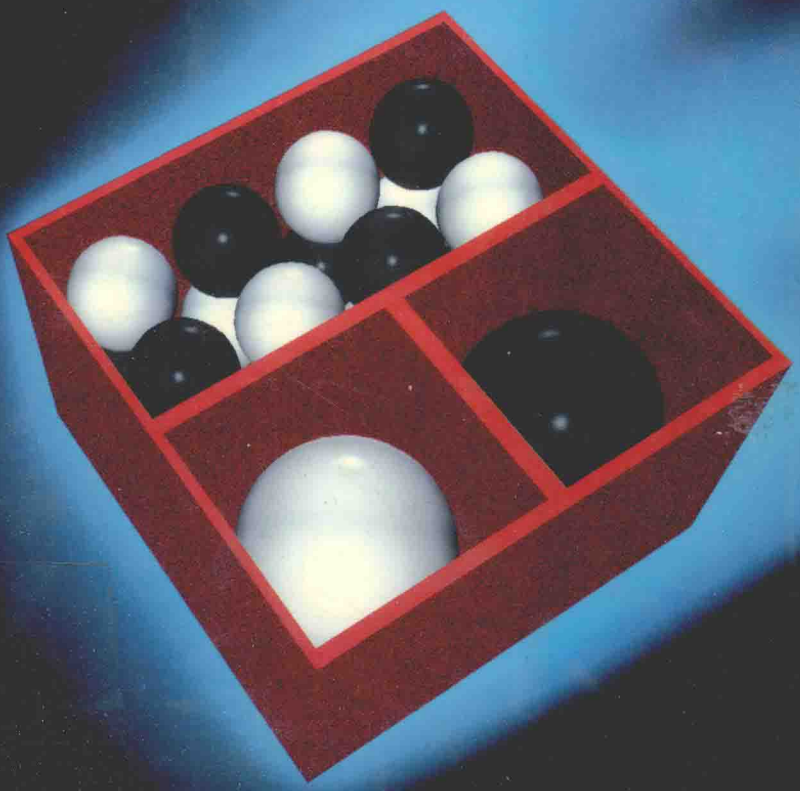
世界著名计算机教材精选

计算理论基础 第2版

ELEMENTS OF THE THEORY OF COMPUTATION SECOND EDITION

Harry R. Lewis, Christos H. Papadimitriou 著

张立昂 刘田 译



清华大学出版社
http://www.tup.tsinghua.edu.cn

PRENTICE HALL
http://www.prenhall.com

世界著名计算机教材精选

计算理论基础

（第2版）

Harry R. Lewis
Christos H. Papadimitriou 著

张立昂 刘 田 译

清华大学出版社 Prentice Hall

(京)新登字 158 号

内 容 简 介

计算理论是计算机科学的理论基础。本书介绍了计算理论最核心、最基本的内容，包括形式语言与自动机、可计算性和计算复杂性三大部分。全书共分七章，分别为：集合、关系和语言；有穷自动机；上下文无关语言；Turing 机；不可判定性；计算复杂性；NP 完全性。本书突出了算法，从而使计算机专业的学生更易接受，也更有收益。

本书适合作为计算机专业及数学专业本科生或研究生的教材，也可供从事计算机科学的教学与研究人员参考。

Elements of the Theory of Computation(Second Edition)
Harry R. Lewis, Christos H. Papadimitriou

书　名：计算理论基础(第 2 版)
作　者：Harry R. Lewis　Christos H. Papadimitriou 著
出版者：清华大学出版社(北京清华大学学研楼，邮编 100084)
http://www.tup.tsinghua.edu.cn
印刷者：北京昌平环球印刷厂
发行者：新华书店总店北京发行所
开　本：787×1092 1/16　**印张**：16　**字数**：367 千字
版　次：2000 年 7 月第 1 版　2000 年 7 月第 1 次印刷
书　号：ISBN 7-302-03948-8/TP・2310
印　数：0001～6000
定　价：29.00 元

译 者 序

半年前我先后接到清华大学出版社第四编辑室室主任薛慧同志和机械工业出版社华章图文信息有限公司总编辑傅豫波同志的电话，前者请我翻译这本书，后者请我翻译Sipser的Introduction to the Theory of Computation (Second edition)，我都欣然接受，并且邀请王捍贫、刘田和黄雄三位博士共同翻译。其实，翻译教材并不是一件美差，对我来说，更谈不上是“好事”，由于眼疾，看书时间一长，心里总犯怵。现在总算完成了任务，两本书都即将出版(尽管中间眼睛还是又犯了一次病)。我之所以愿意翻译这两本书，一是因为它们确实是两本关于计算理论的优秀教材；二是因为国内太缺乏这样的书了，尽管计算机类书刊极其繁多，可是计算理论的书却难觅踪影。清华大学出版社和华章图文信息有限公司购买这两本书的版权翻译出版，此乃善举！我怎好推却。

计算理论在计算机科学技术专业的教学计划中的地位是有争议的。国内计算机专业本科几乎没有开计算理论课的，研究生学这门课的好像也不多，倒是那些想去美国读书的学生为了应付GRE Subject测试，自己在找有关计算理论的书看。我希望这两本书的出版能有利于提高计算机教育界对计算理论的重视，特别希望立志致力于计算机科学技术事业的青年学生都来认真地学一点计算理论，那将可能是终身受益的。

本书包括形式语言与自动机、可计算性和计算复杂性三大部分，这是计算理论最核心、最基本的内容。本书的一大特色是突出了算法，从语法分析到NP难问题的实用算法，算法贯穿全书。这些内容不仅透彻生动地提供了算法设计与分析的基本理论，而且使读者清楚地看到，计算理论不是没有用，也不是离实际很远的。恰恰相反，不仅当今的许多计算机技术可以在计算理论中找到它的思想渊源，而且在许多实际问题中充满着计算理论。

我们按照作者在第二版序言中提供的电子邮件地址得到本书的勘误。在翻译过程中我们又发现了一些错误，也都予以更正。这些错误基本上都是非实质性的，因此没有在书中一一指明。我们已用电子邮件把发现的错误发给作者。

导言和第1章到第3章由张立昂翻译，第4章到第7章由刘田翻译。译文中定有错误和不当之处，敬请读者赐教。

张立昂

1999年9月写于

北京大学燕北园

第一版序言

本书在大学本科水平上介绍经典的和当代的计算理论。粗略地说，内容包括自动机理论与形式语言、Turing(图灵)机的可计算性与递归函数、不可计算性、计算复杂性以及数理逻辑，采用数学的处理方法和计算机科学的观点。于是，在关于上下文无关语言的一章中包含语法分析的讨论，在关于逻辑的几章中给出定理归结证明的有效性和完备性。

在本科教学计划中，这门课通常安排在后面和算法设计与分析课同时上。我们的看法是学计算机科学的学生应该早一点接触这些材料，比如，在二年级或三年级。这是因为，其一，对这些材料的深入研究形成计算机科学中的一些专题；其二，它有助于提供必要的数学示范。但是，我们发现由于一些更高级的教科书要求学生具有较好的数学基础，教这门严格的大学生课是一项困难的任务。我们写这本书的目的是用数学的语言、但不要求有专门的数学经验使得这门课的实质易于被大学生们理解。

全书提供一学年的课程。我们都讲过一个学期的课程，包括第 1 章到第 6 章的大部分内容，根据情况删去了关于语法分析、递归函数及具体的不可解判定问题的部分内容，其他的选择是可能的。例如，强调可计算性和机械逻辑基础的课程可以很快地跳过第 1 章到第 3 章，集中力量讲第 4、6、8 和 9 章。不管怎样使用它，我们都强烈地希望本书对下一代计算机科学家的智力发展有所贡献，在他们受教育的早期阶段向他们介绍关于计算问题的新鲜的和有条理的思想。

我们借这个机会感谢所有让我们从他们那儿学到东西的人，包括教师和学生。特别感谢 Larry Denenberg 和 Aaron Temin 校对早期的草稿，Michael Kahl 和 Oded Shmueli 作为助教对我们的帮助和建议。1980 年春季，Albert Meyer 在 MIT(麻省理工学院)用这本书的草稿讲课，提出很好的批评和修改意见，对此我们表示衷心的感谢。当然，遗留下任何错误都应该由我们负责。Renate D'Arcangelo 以她特有的、非凡的完美和迅速打手稿和画图。

第二版序言

自《计算理论基础》问世以来，在这15年中许多东西变了，也有许多东西没有变。计算机科学现在是一门成熟得多的公认的学科，在这个计算无处不在、信息全球化和复杂性迅速增长的世界上扮演越来越重要的角色，因此有更多的理由关心它的基础。《计算理论基础》的作者们自己现在更加成熟和忙碌——这是何以这么久才出第二版的原因。我们写第二版，是因为我们感觉到有几处可以说得更好些，有几处可以简单些，还有一些可以删掉。更重要的是，我们希望本书反映计算理论和学它的学生在这些年的进步。虽然就绝对而言现在教计算理论的比过去多了，但是它在计算机科学教学计划中的相对地位，例如与算法课程相比，没有得到加强。实际上，算法设计与分析领域现在已经很成熟，有理由认为它的基本原理是关于计算理论的基础课程的一部分。此外，今天的大学生有广泛的和早期的计算经验，清楚地知道自动机在编译程序中的应用。但是，例如在讲像Turing(图灵)机这样的简单模型，用它们作为一般的计算机模型时，他们的疑问更多。总之，对这些问题的处理需要作某些修改。

具体地说，与第一版的主要区别有：

· 早在第1章就形式地介绍算法设计与分析的入门(与闭包有关)，并且算法问题的讨论贯穿全书。在第2、3章有几节讨论与有穷自动机和上下文无关文法有关的算法问题(包括状态最小化和上下文无关识别)。有关于NP完全问题的易解变形的算法，还有一节考察“对付NP完全性”的算法技术(特殊情况的算法、近似算法、回溯与分支定界、局部改进以及模拟退火算法)。

· 第4章对Turing机的处理更形式化，模拟论证更加简单和更加定量。引入随机存取Turing机，以帮助跨越Turing机的笨拙与计算机和程序设计语言能力之间的鸿沟。

· 第5章介绍不可判定性，包括某些递归函数论内容(到Rice定理为止)。介绍文法和递归数值函数，证明它们等价于Turing机，证明更加简单。与上下文无关文法有关问题不可判定性的证明采用简单的直接论证，而不求助Post对应问题。保留铺砖问题，这个问题在NP完全性一章还要见到。

· 处理复杂性的方式相当的新颖：在第6章，除多项式时间界限之外没有定义其他时间界限——于是，P是接触到的第一个复杂性类和概念。然后，用对角化方法证明存在不在P中的指数问题。同时介绍现实生活中的问题与它们的语言表示(两者的区别被故意地弄模糊了)，广泛地考察它们的算法问题。

· NP完全性是单独的一章。在这一章中，有一组新的、广泛的、我们认为有助于教学的NP完全性归约，最后一个是正则表达式的等价问题——回到本书的第一个问题。正如上面所说，本书的最后一节是关于“对付NP完全性”的算法技术。

· 在这个新版中删去关于逻辑的几章。这是一个困难的决定。作出这样的决定有两

个原因：种种迹象表明，这几章是本书过去读得最少和教得最少的几章；现在有几本更好的论述这个题目的书。但是，在第 6 章将详细地论述布尔逻辑和它的可满足性问题。

·总的说来，证明和讲解被简化了，并且在一些关键的地方更加形式化。有几处，如在上下文无关语言与下推自动机的等价性的证明中，把长篇技术性归纳陈述的证明改作习题。每一节的后面有几个习题。

经过这些改动之后，现在有不只一种的方式讲授本书的材料（还有在第一版中提出的方式以及使用中形成的方式）。以讲授计算理论和算法的基础为目标的一学期长的课程可以以从第 2 章到第 7 章中选取的材料为基础。

我们衷心地感谢在这 15 年中所有向我们提供反馈、想法、错误和更正的我们的学生和同事——不可能在这里把他们全部列出来。特别感谢 Martha Sideri 帮助修订第 3 章。还要十分感谢本书的编辑 Alan Apt 和 Prentice Hall 的朋友们——Barbara Kraemer，Sondra Chavez 和 Bayani de Leon——他们都非常有耐心而且乐于助人。

最后，我们乐于收到指出错误的报告和其他评论，最好是通过电子邮件，地址是：elements @ cs. berkeley. edu. 有关本书的错误、更正和其他信息也可以通过这个地址得到。

导　言

看看你的周围，计算无处不在，无时不在，每一个人都计算，计算影响着所有的人。所以能够如此，是因为在过去的几十年中计算机科学家发现了很复杂的方法用来管理计算机资源，实现通讯，翻译程序，设计芯片和数据库，创造更快、更廉价、更容易使用、更安全的计算机和程序。

和所有的主要学科一样，计算机科学的成功实践建立在优美和坚实的基础之上。自然科学有一些基本问题，如物质的本质是什么？有机体生命的基础和起源是什么？计算机科学同样有它自己的基本问题：*什么是算法？什么是能计算的？什么是不能计算的？什么时候可以认为一个算法是实际可行的？*60 多年来（甚至从电子计算机出现之前就开始了）计算机科学家一直在考虑这些问题并且给出充满智慧的回答，这一切对计算机科学产生了深刻的影响。

本书的目的是介绍渗透在计算机科学中的这些基本思想、模型和结果，它们都是该领域的基本范例，它们有很多理由是值得学习的。首先，现代计算机科学中的很多东西直接或间接地以它们为基础——而其余的应该……。其次，这些思想和模型是漂亮、有力的，是数学建模的杰出例子，是有长久价值的。此外，它们更是历史的一部分，是我们这个领域的“共同潜意识”。不首先了解它们，是很难理解计算机科学的。

这些思想和模型在本质上是数学的，这大概不会让人感到奇怪。虽然计算机肯定是一个物理实体，但是同样肯定的是关于它的物理方面，如它的分子和它的形状，能够值得说的很少；对计算机最有用的抽象显然是数学的，论证这些抽象所必需的手段也一定同样是数学的。此外，实际的计算工作需要只有数学才能提供的铁的保证（希望我们的编译程序正确地翻译，应用程序最终停机，等等）。但是，在计算理论中使用的数学与其他应用学科中使用的数学有很大的区别，它一般是*离散的*，在这里不强调实数和连续变量，而强调有穷集合和序列。它以很少几个基本的概念为基础，通过小心、耐心、仔细、一层一层地处理这些概念，勾画出它的能力和深奥之处——这很像计算机，在第 1 章将使你回想这些基本的概念和方法（其中有集合、关系和归纳法），介绍在计算理论中使用它们的风格。

第 2 章和第 3 章描述某些受限制的计算模型。它们能够完成一些非常特殊的字符串处理工作，例如回答一个给定的字符串是否出现在给定的正文中，如字 punk 是否出现在莎士比亚文集中，或者检查一个给定的括号字符串是否正好平衡——像()和(())()，而不是)()的样子。这些受限制的计算模型（分别叫做*有穷自动机*和*下推自动机*）竟然作为像电路和编译程序之类很普通的系统的非常有用和高度完善的部分出现在现实生活中。在这里它们为我们探索算法的一般的形式定义提供极好的热身练习。此外，看到由于增加或删除各种特性这些装置的能力如何增强和减弱（或者，更经常地是，保持不变）是有教益的。其中最值得注意的特性是非确定性，计算的这个迷惑人的特性既是重要的，又是（相当荒谬地）非现实的。

在第 4 章，我们研究算法的一般模型，其中最基本的模型是 Turing 机[①]。Turing 机是第 2 章和第 3 章中字符串处理装置的相当简单的推广，结果令人吃惊地成为描述任意算法的一般框架。为了证明这一点，即通常所说的 Church-Turing 论题，我们引入越来越复杂的计算模型(Turing 机更强的变种，还有随机存取 Turing 机和递归定义的数值函数)，并且证明它们在能力上全都完全等价于基本的 Turing 机模型。

再下一章处理不可判定性。某些自然的明确的计算任务具有这种出人意料的性质，可以证明它们超出了算法能够解决的范围。例如，问你是否能用给定的若干种形状的砖铺整个平面。如果这些形状中包括一个正方形，或者甚至一个三角形，则答案显然是“能够”。但是，如果是几种稀奇古怪的形状，或者规定几种形状必须至少使用一次，那会怎样？这肯定是很复杂的问题，你想用机器给出回答。在第 5 章我们使用 Turing 机的形式表示证明这个问题和许多其他问题是根本不可能用计算机解决的。

即使当一项计算任务能够用某个算法解决的时候，也可能不存在解决它的适当快的、实际可行的算法。在本书的最后两章，我们说明怎么用它们的复杂性对现实生活中的计算问题进行分类：某些问题能够在合理的，即多项式的时间界限内解决，而另一些问题似乎需要以天文速度，即指数地增长的时间。在第 7 章我们定义一个叫做 NP 完全的问题类，它们是一些普通的、实际的、但难得出名的问题(旅行商问题仅是它们中的一个)。我们证明所有这些问题在下述意义下是等价的：如果它们中的一个有有效的算法，则它们每一个都有有效的算法。普遍地相信所有 NP 完全问题具有固有的指数复杂度；这个猜想是否实际上为真是著名的 P≠NP 问题，它是当代数学家和计算机科学家面临的最重要和最深刻的问题之一。

本书有很多篇幅是有关算法及其形式基础的。然而，大概你也意识到了，在今天的计算机科学教学计划中，算法课程，包括算法的分析和设计，相当地脱离计算理论的课程。在本书的现行版本中，我们试图恢复这门课程的某种统一性。结果是，本书对算法也提供了相当不错的介绍，只是有些专门和非传统。在第 1 章形式地介绍算法及其分析，并且在第 2 章和第 3 章研究受限制的计算模型和由它们产生的自然的计算问题的时候一再重新提起。这样一来，在后面探索算法的一般模型的时候，读者处在较好的位置，能正确评价这种探索的目标并且断定是能成功的。在讲解复杂性时算法同样担任主角。因为鉴别复杂问题的最好方法是把它与另一个经得起有效算法考验的问题进行比较。最后一章用一节叙述对付 NP 完全性作为结束，给出在遇到 NP 完全问题时已经成功地使用过的算法技术(近似算法、穷举算法、启发式局部搜索等)。

计算是必不可少的、有力的、优美的、具有挑战性和不断发展的——它的理论也是如此。本书仅讲了一个激动人心的故事的开头，它是对从计算理论宝库中精选出的少量基本论题的朴素介绍。我们希望它将激发读者去作进一步的探索，每一章最后的参考文献指出了好的出发点。

---

① 以卓越的英国数学家和哲学家 Alan M. Turing (1912—1954)的名字命名。他在 1936 年发表的开创性的论文标志计算理论的诞生。Turing 也是人工智能和计算机下棋以及生物学中形态形成领域中的先驱者。在第二次世界大战中，他帮助破译德国海军密码 Enigma。想更多地了解他迷人的一生(以及他最后悲惨地死在官方残忍和偏执的手中)请阅读《Alan Turing: The Enigma》，Andrew Hodges 著，New York: Simon Schuster，1983 年。

目　录

第 1 章　集合、关系和语言

1.1 集　　合

人们说数学是科学的语言，它当然也是我们在本书将要学习的科学学科——计算理论——的语言。数学语言涉及集合以及它们重叠、相交和用集合构成新的集合的各种复杂方式。

集合是对象的汇集。例如，4 个字母 a,b,c 和 d 的汇集是一个集合，把它叫做 L，记作 $L=\{a,b,c,d\}$。构成集合的所有对象叫做它的**元素**或**成员**。例如，b 是集合 L 的一个元素，表成 $b\in L$，有时直接说 b 在 L 中，或 L 包含 b。另一方面，z 不是 L 的元素，记作 $z\notin L$。

在集合中，不区分元素的重复。于是，集合{red,blue,red}与{red,blue}是相同的集合。类似地，所有元素的顺序是无关紧要的。例如，{3,1,9}，{9,3,1,}和{1,3,9}是相同的集合。总之，两个集合相等(即，相同)当且仅当它们有相同的元素。

一个集合的元素之间不需要有什么关系(除去它们都是同一个集合的元素外)。例如，{3,red,{d,blue}}是有 3 个元素的集合，其中一个元素的本身是一个集合，集合可能只有一个元素，这样的集合叫做**单元集**。例如，{1}是以 1 作为它的唯一一个元素的集合，{1}与 1 是完全不同的。还有一个根本没有元素的集合。当然，只可能有一个这样的集合，它叫做**空集**，用 $\varnothing$ 表示。称除空集以外的任何集合是非空的。

到现在为止，我们用直接列出所有的元素来规定集合，元素用逗号隔开，放在一对花括号内。有些集合是无穷的，不可能用这种方式写出来。例如，自然数集合 $\mathbf{N}$ 是无穷的。写成 $\mathbf{N}=\{0,1,2,\cdots\}$，在无穷长的表处用 3 个点可以直觉地示意出它的元素。不是无穷的集合是有穷的。

规定集合的另一个方法是求助别的集合与元素可能具有或不具有的性质，设 $I=\{1,3,9\}$和 $G=\{3,9\}$，可以把 G 描述成由 I 的大于 2 的元素组成的集合，写成

$$G=\{x:x\in I \quad \text{且 } x \text{ 大于 } 2\}$$

一般地，设集合 A 已经有定义，P 是 A 的元素可能具有、也可能不具有的性质，则可以定义一个新的集合

$$B=\{x:x\in A \text{ 且 } x \text{ 具有性质 } P\}$$

下面是另一个例子，奇自然数集合为

$$O=\{x:x\in \mathbf{N} \text{ 且 } x \text{ 不被 2 整除}\}$$

如果集合 A 的每一个元素都是集合 B 的元素，则称 A 是 B 的**子集**，记成 $A\subseteq B$。于是，$O\subseteq\mathbf{N}$，因为每一个奇自然数都是自然数。注意任何集合是它自身的子集。如果 A 是 B 的子集且 A 不等于 B，则称 A 是 B 的**真子集**，记作 $A\subset B$。还要注意空集是每一个集合的

子集。设 B 是任一集合，因为$\varnothing$的每一个元素(没有这样的元素)也是 B 的元素，故$\varnothing \subseteq B$。

要想证明两个集合 A 和 B 相等，可以证明 $A \subseteq B$ 和 $B \subseteq A$。于是，A 的每一个元素一定是 B 的元素，B 的每一个元素也一定是 A 的元素，从而 A 和 B 有相同的元素，故 $A=B$。

和用算术运算，比如用加法，把数结合在一起一样，可以用各种集合运算把两个集合结合成第三个集合。**并**是一种集合运算：两个集合的并是一个集合，它的元素至少在这两个给定集合中的一个中、也可能在这两个中。用符号$\cup$表示并，因而

$$A \cup B = \{x : x \in A \text{ 或 } x \in B\}$$

例如，

$$\{1,3,9\} \cup \{3,5,7\} = \{1,3,5,7,9\}$$

两个集合的**交**是这两个集合共有的全部元素组成的集合，即

$$A \cap B = \{x : x \in A \text{ 且 } x \in B\}$$

例如，

$$\{1,3,9\} \cap \{3,5,7\} = \{3\}$$

和

$$\{1,3,9\} \cap \{a,b,c,d\} = \varnothing$$

最后，两个集合 A 和 B 的**差**，记作 $A-B$，是 A 的不在 B 中的所有元素组成的集合。即

$$A - B = \{x : x \in A \text{ 且 } x \notin B\}$$

例如，

$$\{1,3,9\} - \{3,5,7\} = \{1,9\}$$

这几个集合运算的某些性质容易从它们的定义得到。例如，设 A、B 和 C 是集合，下述算律成立。

幂等律	$A \cup A = A$
	$A \cap A = A$
交换律	$A \cup B = B \cup A$
	$A \cap B = B \cap A$
结合律	$(A \cup B) \cup C = A \cup (B \cup C)$
	$(A \cap B) \cap C = A \cap (B \cap C)$
分配律	$(A \cup B) \cap C = (A \cap C) \cup (B \cap C)$
	$(A \cap B) \cup C = (A \cup C) \cap (B \cup C)$
吸收律	$(A \cup B) \cap A = A$
	$(A \cap B) \cup A = A$
De Morgan 律	$A - (B \cup C) = (A - B) \cap (A - C)$
	$A - (B \cap C) = (A - B) \cup (A - C)$

例 1.1.1 证明第一条 De Morgan 律，令

$$L = A - (B \cup C)$$

$$R = (A - B) \cap (A - C)$$

要证 $L=R$。为此要证(a) $L\subseteq R$ 和(b)$R\subseteq L$。

(a) 设 x 是 L 的任一元素,则 $x\in A$ 但 $x\notin B$ 且 $x\notin C$。因而 x 是 $A-B$ 和 $A-C$ 两者的元素,故是 R 的元素。所以 $L\subseteq R$。

(b) 设 $x\in R$,则 x 是 $A-B$ 和 $A-C$ 两者的元素,因而在 A 中但不在 B 和 C 中。因此 $x\in A$ 但 $x\notin B\cup C$,故 $x\in L$。因此 $R\subseteq L$。

这就证明了 $L=R$。 ◇

如果两个集合没有共同的元素,即它们的交是空集,则称它们**不相交**。

可以定义两个以上集合的并和交。设 S 是任一集合的汇集,$\cup S$ 表示 S 中所有集合的全部元素组成的集合。例如,如果 $S=\{\{a,b\},\{b,c\},\{c,d\}\}$,则 $\cup S=\{a,b,c,d\}$;如果 S 是所有以自然数作元素的单元集$\{\{n\}:n\in\mathbf{N}\}$,则 $\cup S=\mathbf{N}$. 一般地,

$$\cup S=\{x:\text{对于某个集合 } P\in S, x\in P\}$$

类似地,

$$\cap S=\{x:\text{对于每一个集合 } P, x\in P\}$$

集合 A 的所有子集的汇集本身是一个集合,叫做 A 的**幂集**,记作 2^A。例如,$\{c,d\}$的全部子集是$\{c,d\}$本身、单元集$\{c\}$和$\{d\}$,以及空集$\varnothing$。因此

$$2^{\{c,d\}}=\{\{c,d\},\{c\},\{d\},\varnothing\}$$

非空集合 A 的划分是 2^A 的一个子集 Π,使得$\varnothing$不是 Π 的元素并且 A 的每一个元素在且只在 Π 中的一个集合中。即,如果 Π 是 A 的子集的集合使得

(1) Π 的每一个元素非空,

(2) Π 的不同元素是不相交的,

(3) $\cup\Pi=A$,

则 Π 是 A 的一个划分。

例如,$\{\{a,b\},\{c\},\{d\}\}$是$\{a,b,c,d\}$的一个划分,而$\{\{b,c\},\{c,d\}\}$不是它的划分。奇自然数集合和偶自然数集合构成 $\mathbf{N}$ 的一个划分。

习　题

1.1.1 判断下述每一条是真是假:

(a) $\varnothing\subseteq\varnothing$

(b) $\varnothing\in\varnothing$

(c) $\varnothing\in\{\varnothing\}$

(d) $\varnothing\subseteq\{\varnothing\}$

(e) $\{a,b\}\in\{a,b,c,\{a,b\}\}$

(f) $\{a,b\}\subseteq\{a,b,\{a,b\}\}$

(g) $\{a,b\}\subseteq 2^{\{a,b,\{a,b\}\}}$

(h) $\{\{a,b\}\}\in 2^{\{a,b,\{a,b\}\}}$

(i) $\{a,b,\{a,b\}\}-\{a,b\}=\{a,b\}$

1.1.2 用花括号、逗号和数字写出下述集合:

(a) $(\{1,3,5\}\cup\{3,1\})\cap\{3,5,7\}$

(b) $\bigcup\{\{3\},\{3,5\},\bigcap\{\{5,7\},\{7,9\}\}\}$

(c) $(\{1,2,5\}-\{5,7,9\})\cup(\{5,7,9\}-\{1,2,5\})$

(d) $2^{\{7,8,9\}}-2^{\{7,9\}}$

(e) $2^{\varnothing}$

1.1.3 证明下述等式：

(a) $A\cup(B\cap C)=(A\cup B)\cap(A\cup C)$

(b) $A\cap(B\cup C)=(A\cap B)\cup(A\cap C)$

(c) $A\cap(A\cup B)=A$

(d) $A\cup(A\cap B)=A$

(e) $A-(B\cap C)=(A-B)\cup(A-C)$

1.1.4 设 $S=\{a,b,c,d\}$。

(a) S 的成员最少的划分是什么？成员最多的划分是什么？

(b) 列出 S 的恰好有两个成员的所有划分。

1.2 关系与函数

数学研究关于对象以及它们之间的关系的陈述。例如，说“小于”是某种对象(即，数)之间的关系是很自然，这种关系在 4 与 7 之间成立，但在 4 与 2 和 4 与 4 之间不成立。但是现在可供我们使用的数学语言只有集合的语言，怎么用这种语言表达对象之间的关系？把关系本身看作集合。属于关系的对象在本质上是直观上使得关系成立的个体的组合。因而小于关系是第一个数小于第二个数的所有数对组成的集合。

但是我们走得快了一点。在属于关系的一对数中，必须能够区分这对数的两部分，而我们还没有说明如何区分。不能把这些数对写成集合，因为$\{4,7\}$和$\{7,4\}$是一样的。最简单的办法是引进一个新的组合对象的方法，把它叫做**有序对**①。

a 和 b 的有序对记作(a,b)，a 和 b 叫做它的**分量**。有序对(a,b)不同于集合$\{a,b\}$。首先，顺序是有关系的：(a,b)与(b,a)不一样，而$\{a,b\}=\{b,a\}$。其次，有序对的两个分量不必是不同的，$(7,7)$是有效的有序对。注意：仅当 $a=c$ 且 $b=d$ 时两个有序对(a,b)和(c,d)相等。

两个集合 A 和 B 的**笛卡儿积**是 $a\in A$ 且 $b\in B$ 的所有有序对(a,b)构成的集合，记作 $A\times B$。例如

$$\{1,3,9\}\times\{b,c,d\}$$
$$=\{(1,b),(1,c),(1,d),(3,b),(3,c),(3,d),(9,b),(9,c),(9,d)\}$$

两个集合 A 和 B 上的二元关系是 $A\times B$ 的子集。例如：$\{(1,b),(1,c),(3,d),(9,d)\}$是$\{1,3,9\}$和$\{b,c,d\}$上的二元关系。$\{(i,j):i,j\in\mathbf{N}$ 且 $i<j\}$是小于关系，它是 $\mathbf{N}\times\mathbf{N}$ 的子集(用二元关系关联的两个集合常常是相同的)。

更一般地，设 n 是任一自然数。如果 $a_1,\cdots,a_n$ 是任意 n 个对象，不必是不相同的，则$(a_1,\cdots,a_n)$是一个**有序组**。对每一个 $i=1,\cdots,n$，a_i 是$(a_1,\cdots,a_n)$的第 i 个分量。有序 m 元

① 在集合论中不把有序对(a,b)看作新的对象类型，它等同于$\{\{a\},\{a,b\}\}$。

组 $(b_1,\cdots,b_m)$ 和 $(a_1,\cdots,a_n)$ 相同当且仅当 $m=n$ 且 $a_i=b_i, i=1,\cdots,n$。于是,(4,4),(4,4,4),((4,4),4)和(4,(4,4))都是不相同的。有序二元组就是前面讨论的有序对。有序 3、4、5 和 6 元组分别叫做**有序三元组**、**四元组**、**五元组**和**六元组**。此外,**序列**是一个有序 n 元组,其中 n 是某个未指定的数,叫做该序列的**长度**。设 $A_1,\cdots,A_n$ 是任意 n 个集合,**n 重笛卡儿积** $A_1\times\cdots\times A_n$ 是所有有序 n 元组 $(a_1,\cdots,a_n)$ 的集合,其中对于每一个 $i=1,\cdots,n, a_i\in A_i$。当所有的 A_i 都相同(设为 A)时,A 自身的 n 重笛卡儿积 $A\times\cdots\times A$ 也写成 A^n。例如,$\mathbf{N}^2$ 是所有自然数有序对的集合。集合 $A_1,\cdots,A_n$ 上的 **n 元关系**是 $A_1\times\cdots\times A_n$ 的一个子集。1、2 和 3 元关系分别叫做**一元**、**二元**和**三元关系**。

另一基本的数学概念是*函数*。直观上,函数是把一类中的每一个对象关联到另一类中的一个唯一的对象。例如,人与他们的年龄,狗与它们的主人,数与它们的后继等等。利用把二元关系看作有序对集合的思想,可以给出这个直观概念的具体定义。从集合 A 到集合 B 的**函数**是 A 和 B 上的具有下述特殊性质的二元关系 R:对于每一个元素 $a\in A$,在 R 中恰好有一个有序对以 a 为第一个分量,用下面的例子说明这个定义。令 C 是美国的城市集合,S 是州集合,

$$R_1=\{(x,y):x\in C,y\in S \text{ 且 } x \text{ 是州 } y \text{ 的一个城市}\},$$

$$R_2=\{(x,y):x\in S,y\in C \text{ 且 } y \text{ 是州 } x \text{ 的一个城市}\}。$$

那么,R_1 是一个函数,因为每一个城市在且只在一个州内;R_2 不是函数,因为一些州有一个以上城市①。

一般地,用字母 f,g,h 等表示函数,用 $f:A\mapsto B$ 表示 f 是从 A 到 B 的函数。把 A 叫做 f 的**定义域**。设 a 是 A 的任一元素,用 $f(a)$ 表示 B 中使得 $(a,b)\in f$ 的元素 b。因为 f 是一个函数,恰好存在一个 $b\in B$ 具有这个性质,所以 $f(a)$ 表示一个唯一的对象。$f(a)$ 叫做 a 在 f 下的**象**。为了详细说明函数 $f:A\mapsto B$,只要对每一个 $a\in A$,指定 $f(a)$。例如,要详细说明上述函数 R_1,只要对每一个城市指出它所在的州。设 $f:A\mapsto B$,A' 是 A 的子集,定义 $f[A']=\{f(a):a\in A'\}$(即,$\{b:$对某个 $a\in A', b=f(a)\}$)。称 $f[A']$ 是 A' 在 f 下的**象**。f 的**值域**是它的定义域的象。

当函数的定义域是笛卡儿积时,通常省略一对圆括号。例如,如果定义 $f:N\times N\mapsto N$ 使得有序对 (m,n) 在 f 下的象是 m 与 n 的和,则按照习惯记法写成 $f(m,n)=m+n$,而不写成 $f((m,n))=m+n$。

设 $f:A_1\times\cdots\times A_n\mapsto B$ 是一个函数,$f(a_1,\cdots,a_n)=b$,其中 $a_i\in A_i, i=1,\cdots,n$ 且 $b\in B$,有时称 $a_1,\cdots,a_n$ 是 f 的**自变量**,而 b 是 f 对应的**值**。于是,对自变量的每一个 n 元组给出 f 的值可以指定函数 f。

对某些类型的函数特别感兴趣。如果对任意两个不同的值 $a,a'\in A, f(a)\neq f(a')$,则称函数 $f:A\mapsto B$ 是**一对一**的 。例如,设 C 是美国的城市集合,S 是州集合,而 $g:S\mapsto C$ 规定为

$$g(s)=\text{州 } s \text{ 的首府},\qquad \text{对每一个 } s\in S$$

---

① 我们认为马萨诸塞州的坎布里奇(Cambridge)和马里兰州的坎布里奇(Cambridge)不是同一个城市,而是恰巧重名的不同的城市。

那么 g 是一对一的，因为没有两个州有相同的首府。如果 B 的每一个元素是 A 的某个元素在 f 下的象，则称函数 $f:A\mapsto B$ **满射到** B。刚才规定的函数 g 不是满射到 C 的，但上面定义的函数 R_1 满射到 S，因为每一个州至少有一个城市。最后，如果函数 $f:A\mapsto B$ 既是一对一的、又是满射到 B 的，则称 f 是 A 与 B 之间的**双射**。例如，如果 C_0 是州首府集合，和前面一样，函数 $g:S\mapsto C_0$ 规定为

$$g(s) = \text{州 } s \text{ 的首府}$$

则 g 是 S 与 C_0 之间的双射。

二元关系 R 的**逆**是关系 $\{(b,a):(a,b)\in R\}$，记作 R^{-1}。显然，若 $R\subseteq A\times B$，则 $R^{-1}\subseteq B\times A$。例如，上面定义的关系 R_2 是 R_1 的逆。可见函数的逆不一定是函数。对 R_1 来说，它的逆不再是一个函数，因为某些州有一个以上城市，即存在不同的城市 C_1 和 C_2 使 $R_1(C_1)=R_1(C_2)$。设 $f:A\mapsto B$ 是一个函数，如果存在 $b\in B$ 使得对于所有的 $a\in A$，$f(a)\neq b$，则函数 f 的逆也不是函数。然而，如果 $f:A\mapsto B$ 是一个双射，则这些情况都不可能发生，f^{-1} 也是一个函数，它是 B 与 A 之间的双射 。而且，对每一个 $a\in A$，$f^{-1}(f(a))=a$；对每一个 $b\in B$，$f(f^{-1}(b))=b$。

当在两个集合之间规定一个特别简单的双射之后，有时能够把定义域中的对象与它在值域中的象看作在实质上是不可区分的：一个是另一个的换名或一种重写方式。例如，严格地说，单元集和有序一元组是不同的。但是，由于对每一个单元集 $\{a\}$，$f(\{a\})=a$ 是一个明显的双射，故偶尔地混淆两者的区别也没有多大害处。这样的双射叫做一个**自然同构**。当然，这不是一个形式定义，因为什么是“自然的”，什么差别是可以混淆的，要依赖上下文。再举几个稍微复杂一点的例子可能会更清楚一些。

例 1.2.1 对于任意 3 个集合 A，B 和 C，有 $A\times B\times C$ 到 $(A\times B)\times C$ 的自然同构，即对任意的 $a\in A$，$b\in B$ 和 $c\in C$，

$$f(a,b,c) = ((a,b),c) \qquad \diamond$$

例 1.2.2 对于任意的 A 和 B，从 A 和 B 上的所有二元关系的集合 $2^{A\times B}$ 到集合

$$\{f:f \text{ 是从 } A \text{ 到 } 2^B \text{ 的函数}\}$$

的自然同构 ϕ。即，对任意的二元关系 $R\subseteq A\times B$，令 $\phi(R)$ 为函数 $f:A\mapsto 2^B$ 使得

$$f(a) = \{b:b\in B \text{ 且 } (a,b)\in R\}$$

例如，设 S 是州集合，$R\subseteq S\times S$ 包含所有有共同边界的两个州的有序对，则自然的相关函数 $f:S\mapsto 2^S$ 规定为

$$f(s) = \{s':s'\in S \text{ 且 } s' \text{ 与 } s \text{ 有共同边界}\} \qquad \diamond$$

例 1.2.3 有时即使当函数 $f:A\mapsto B$ 不是一个双射时也把它的逆看作一个函数。这个想法是用例 1.2.2 中描述的自然同构把 $f^{-1}\subseteq B\times A$ 看作从 B 到 2^A 的函数。于是，对于任意的 $b\in B$，$f^{-1}(b)$ 等于使得 $f(a)=b$ 的所有 a 组成的集合。例如，设 R_1 的定义如前面所述（对于每一个城市，这个函数的值等于它所在的州），则对于每一个州 s，$R_1^{-1}(s)$ 等于州 s 中所有城市的集合。 $\diamond$

设 Q 和 R 是两个二元关系，它们的**合成** $Q\circ R$（可简写成 QR）是关系

$$\{(a,b):\text{对于某个 } c,(a,c)\in Q \text{ 且 } (c,b)\in R\}$$

注意到两个函数 $f:A\mapsto B$ 和 $g:B\mapsto C$ 的合成是一个从 A 到 C 的函数 h，使得对于每一个