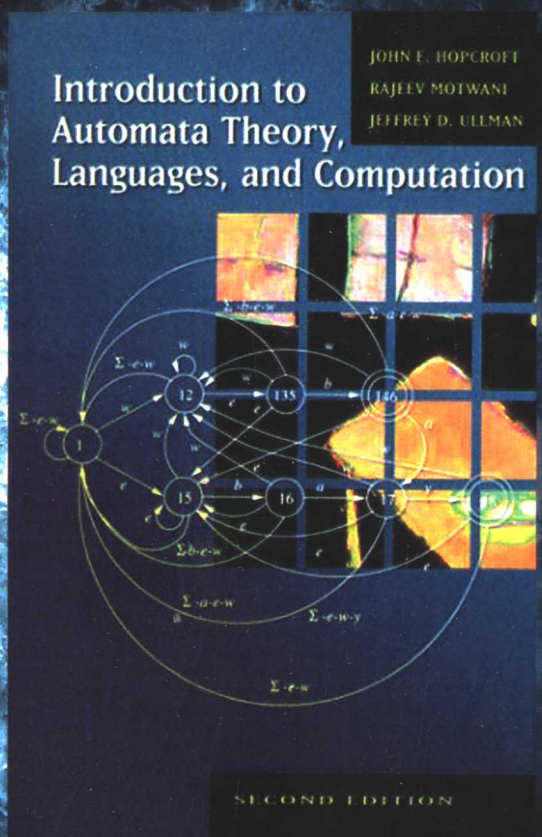
计 算 机 科 学 丛 书

原书第2版

自动机理论、语言和计算导论

(美) John E. Hopcroft Rajeev Motwani Jeffrey D. Ullman 著 刘田 姜晖 王捍贫 译



Introduction to Automata Theory, Languages, and Computation

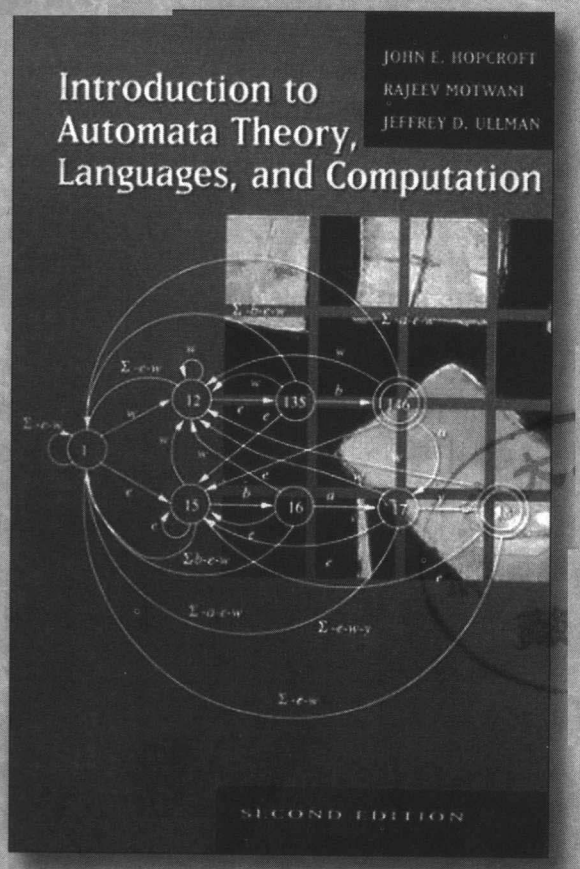
Second Edition

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

原书第2版

自动机理论、语言和计算导论

(美) John E. Hopcroft　Rajeev Motwani　Jeffrey D. Ullman 著　刘田　姜晖　王捍贫　译



Introduction to Automata Theory, Languages, and Computation

Second Edition

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

本书是关于形式语言、自动机理论和计算复杂性方面的经典之作。书中涵盖了有穷自动机、正则表达式与语言、正则语言的性质、上下文无关文法及上下文无关语言、下推自动机、上下文无关语言的性质、图灵机、不可判定性以及难解问题等内容。本书在定义和证明中使用了很多细节和直观说明，使用图来帮助阐明思想，并包含了大量的难度各异的示例和习题，以便读者确认和加深对内容的理解。

本书适合作为计算机专业高年级本科生及研究生计算理论课程的教材和教学参考书。

本书版权登记号：图字：01-2001-4878

图书在版编目（CIP）数据

自动机理论、语言和计算导论（原书第2版）/（美）霍普克罗夫特（Hopcroft, J. E.）等著；刘田等译. -北京：机械工业出版社，2004.6

（计算机科学丛书）

书名原文：Introduction to Automata Theory, Languages, and Computation, Second Edition

ISBN 7-111-14452-X

I. 自… II. ① 霍… ② 刘… III. ① 自动机理论-高等学校-教材② 形式语言-高等学校-教材 IV. TP301

中国版本图书馆CIP数据核字（2004）第043711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

北京牛山世兴印刷厂印刷 · 新华书店北京发行所发行

2004年6月第1版第1次印刷

787mm × 1092mm 1/16 · 24印张

印数：0 001-5 000册

定价：39.00元

凡购本书，如有倒页、脱页、缺页，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

前　言

在本书1979年的前一版的前言中，Hopcroft和Ullman对下面的事实感到惊异：与1969年写第一本书时的状况相比，自动机的专题已经有了飞速的发展。确实，1979年的书包含了以前的著作中找不到的许多主题，篇幅大约增加了一倍。如果读者把本书与1979年那本书相比就会发现，像20世纪70年代的汽车那样，本书是“外看大，内看小”。这听起来像是退步，但是有理由对此变化感到高兴。

首先，在1979年，自动机和语言理论还是一个活跃的研究领域。那本书的目标是鼓励有数学倾向的学生为这个领域做出新的贡献。今天，几乎不再直接研究自动机理论（而是研究其应用），因此就几乎没有理由保持1979年那本书的简洁和高度的数学风格。

其次，自动机和语言理论的角色在过去20年里已经发生了改变。在1979年，自动机主要还是研究生水平的专题，那时假定读者是高年级研究生，特别是使用这本书后几章的那些读者。今天，这个专题是本科生课程的主要内容。由于这个原因，本书的内容必须假设对于学生只有很少的预备知识的要求，因此就必须比以前的书提供更多的背景和论证的细节。

环境的第三个变化是，计算机科学在过去20年里已经增长到了几乎无法想像的程度。在1979年，用我们觉得将会经得起下一轮技术考验的材料来填充课程计划，常常是一项挑战；而在今天，却有非常多的子科目在竞争本科生课程的有限空间。

第四，计算机科学已经成为更加职业化的专题，在许多学生中存在着严重的实用主义。我们仍然相信，自动机理论的方方面面是各种新兴学科的重要工具，并且我们相信，无论有多少学生宁愿只学那些最能直接赚钱的技术，在典型的自动机课程中那些理论的和扩展心智的习题还仍然保持其价值。然而，为了保证这个专题在供计算机科学专业学生选择的课程表中持续占有一席之地，我们相信有必要在强调数学的同时也强调应用。因此，我们已经把上一版中许多深奥的主题换成了今天如何使用这些思想的例子。虽然自动机和语言理论对于编译器的应用现在广为人知，以致于这些内容通常包含在编译课程中，但是还有各种较新的用途，包括验证协议的模型验证算法和以上下文无关文法为模式的文本描述语言等。

对于这本书内容既增长又收缩的最后一项解释是，现今能够利用Don Knuth和Les Lamport开发的TEX和LATEX排版系统。特别是后一种系统鼓励“开放”式的排版，让书籍篇幅更大，但更容易阅读。感谢这两个人的努力。

本书用法

本书适合于三年级以上学生的一季度或一学期的课程。在斯坦福大学，我们把这份讲义用在CS154课程中，这是自动机和语言理论的课程。这是一季度的课程，Rajeev和Jeffrey曾经教过。由于授课时间有限，第11章不包括在内，后面一些材料（比如较难的10.4节中的*多项式时间归约*）也都省略了。本书的网站（参见下面）包括CS154课程的笔记和教学大纲等材料。

若干年前作者发现，来斯坦福大学的许多研究生都学过一门不包括难解性理论的自动机理论课程。由于斯坦福大学的全体教员相信，对于每一个计算机科学家来说，充分了解这些思想都是重要的，所以就有了另一门课程CS154N，选这个课的学生可以只学习第8、9和10章。这些学生实际上参加CS154课程大约后三分之一的学习，来满足CS154N的要求。即使在今天，作者仍然发现，每季度都有一些学生采取这个选项。由于这样做几乎不要求额外的努力，所以作者推荐这个方法。

预备知识

为了最佳地使用本书，读者应当在以前学过一门关于离散数学（如图、树、逻辑、证明技巧等）的课程。还假设读者学过一些程序设计课程，熟悉普通的数据结构、递归和主要系统组件（如编译器）的作用等。这些预备知识一般应当包含在一、二年级计算机科学课程计划中。

习题

本书包含大量习题，几乎每节都有。把较难的习题或习题部分用叹号标出。最难的习题有两个叹号。

某些习题或习题部分标有星号。对于这些习题，其解答方法可以通过本书的网页获得。这些解答可以公开获得，应当用于自我测试。注意，在少数情况下，习题*B*要求修改或改编对另一个习题*A*的解答。如果*A*的某些部分有解答，那么就应当预期*B*的对应部分也有解答。

万维网上的支持

本书的主页是

`http://www-db.stanford.edu/~ullman/ialc.html`

其中包含带星号的习题的解答、已知的勘误表、后备材料等。作者希望提供所教CS154课程的每种材料的笔记，包括课后作业、解答和考试等。

致谢

Craig Silverstein关于“如何进行证明”的讲义影响了第1章中的一些材料。对本书手稿的评论和勘误来自Zoe Abrams、George Candea、Haowen Chen、Byong-Gun Chun、Jeffrey Shallit、Bret Taylor、Jason Townsend和Erik Uzureau等。非常感谢他们。当然，剩下的错误都由我们负责。

J. E. H.

R. M.

J. D. U.

纽约州，艾萨克；加州，斯坦福

2000年9月

目　录

第1章 自动机：方法与体验

自动机理论研究抽象计算装置或“机器”。在20世纪30年代计算机出现之前，图灵研究过一种抽象机器，这种机器具备了今天计算机的所有能力，至少在计算能力上是这样的。图灵的目标是精确地描述一条界线，这条界线区分计算机能做什么和不能做什么；图灵的结论不仅适用于抽象的图灵机，也适用于今天的真实计算机。

在20世纪40和50年代，许多研究者研究过更简单类型的机器，今天称为“有穷自动机”。起初建议用这些自动机来为人脑功能建立模型，后来发现这些自动机对于1.1节提到的各种其他目的也极为有用。在20世纪50年代后期，语言学家乔姆斯基（N. Chomsky）开始研究形式“文法”。尽管这些文法不是严格意义上的机器，但与抽象自动机有密切关系，而且目前是一些重要软件部件（包括部分编译器在内）的基础。

在1969年，库克（S. Cook）扩展了图灵对什么能被计算和什么不能被计算的研究。库克设法分离出了两类问题，一类问题是计算机能有效解决的，另一类问题是计算机原则上能解决，但实际上要花费太长时间，以致于除了非常小的问题实例以外，计算机是毫无用处的。后一类问题称为“难解的”或“NP-难的”。计算机硬件一直遵循着计算速度呈指数增长的规律（摩尔定律），但这也不太可能显著地影响解决难解问题大实例的能力。

所有这些理论进展都直接影响了计算机科学家今天的工作。有些概念，比如有穷自动机和某些类型的形式文法，用于设计和构造重要类型的软件。另一些概念，比如图灵机，则帮助理解能期待软件做什么。特别是，难解问题的理论允许推断是否有可能“正面”处理一个问题，编写解决这个问题的程序（因为这个问题不属于难解的一类），或者是否需要找到某种方法来迂回处理这个难解的问题：找近似算法、用启发式算法或者用某种其他方法来限制程序解决这个问题所花费的时间。

入门性的本章从概述自动机理论的内容和用途开始。本章许多篇幅用来总结证明技术和发现证明的技巧。这些内容包括：演绎证明、命题变形、反证法、归纳证明以及其他重要概念。最后一节介绍一些贯穿自动机理论的概念：字母表、串以及语言。

1.1 为什么研究自动机理论

对于自动机和复杂性的研究是计算机科学核心的重要部分有几个理由。本节要向读者介绍主要的动机并概括本书讨论的主要主题。

1.1.1 有穷自动机简介

有穷自动机是许多重要类型的硬件和软件的有用模型。从第2章开始会看到如何使用这些概念的例子。目前只列出一些最重要的类型：

1. 数字电路的设计和性能检查软件。

2. 典型编译器的“词法分析器”，也就是说，把输入文本分解为诸如标识符、关键字和标点符号等逻辑单元的编译器部件。

3. 扫描大量文本（比如收集到的网页）来发现单词、短语或其他模式的出现的软件。

4. 所有类型的只有有穷多个不同状态的系统（比如通信协议或安全交换信息的协议）的验证软件。

很快就会遇到各种类型自动机的精确定义，非形式化的介绍从概述有穷自动机是什么和做什么开始。可以认为许多系统或部件（如上面列举的那些）始终处在有穷多个“状态”之一。状态的目的是记住系统历史的有关部分。由于只有有穷多个状态，在一般情况下不可能记住整
2 个历史，所以必须仔细地设计系统，以记住重要的历史而忘记不重要的。只有有穷多个状态的好处是用固定的资源就能实现系统。例如，可用硬件将其实现为电路，或者实现为简单形式的程序，这种程序只查看有限的数据就能做出决定，或者用代码本身中的位置来做出决定。

例1.1 也许最简单的非平凡有穷自动机是两相开关。这个装置记住处在“开”状态或“关”状态，允许用户按一个按钮，这个按钮根据开关状态起不同作用。也就是说，如果开关处在关状态，按这个按钮就变成开状态；如果开关处在开状态，按这个按钮就变成关状态。

这个开关的有穷自动机模型如图1-1所示。对所有的有穷自动机都一样，状态用圆圈表示。在这个例子中，命名了状态on（开）和off（关）。状态之间的箭头用“输入”来标记，表示作用在系统上的外部影响。这里两个箭头都用Push（按）来标记，表示用户按这个按钮。这两个箭头的含义是：无论系统处在什么状态，当收到输入Push时，就进入另一个状态。

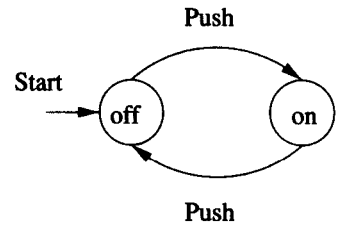


图1-1 为两相开关建立模型的有穷自动机

把一个状态指定为“初始状态”，即系统开始时所处的状态。在这个例子中，初始状态是off，习惯上用单词Start（开始）和一个指向这个状态的箭头来说明初始状态。

通常需要说明一个或多个状态是“终止”或“接受”状态。在经历一个输入序列后进入这些状态之一，说明这个输入序列在某种程度上是好的。例如，可以认为图1-1中的on状态是接受状态，因为在这个状态下，这个由开关控制的装置将会运行。习惯上用双圆圈来说明接受状态，但在图1-1中没有做这样的说明。 □

例1.2 有时候，一个状态记忆的内容比一个开关选择要复杂得多。图1-2显示另一个有穷自
3 动机，它可能是词法分析器的一部分。这个自动机的任务是识别关键字then。因此这个自动机需要5个状态，每个状态表示在单词then中目前已经到达的不同位置。这些位置对应于这个单词的前缀，范围从空串（即一点也没有看见这个单词）到整个单词。

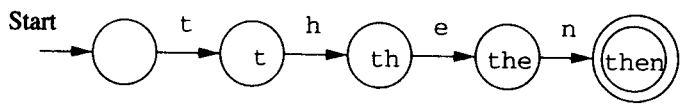


图1-2 为识别then建立模型的有穷自动机

在图1-2中，用目前已经看到的then的前缀来命名5个状态。输入对应字母。可以想像，词法分析器每次检查正在编译的程序的一个字母，要检查的下一个字母就是自动机的输入。初始状态对应空串。每个状态在then的下一个字母上转移到对应于下一个更长前缀的状态上。当输入拼写了单词then时，就进入了名为then的状态。由于这个自动机的任务是识别何时已经看到了then，所以可认为这个状态是惟一的接受状态。 □

1.1.2 结构表示法

有两种不像自动机但在自动机的研究和应用中起重要作用的重要记号。

1. 文法在设计某种软件时是有用的模型，这种软件处理带递归结构的数据。最著名的例子是“语法分析器”，即处理典型程序设计语言中递归嵌套特征的编译器部件，这些特征比如是表达式（如算术表达式、条件表达式等）。例如，像$E \Rightarrow E + E$这样的文法规则说，把两个表达式用加号连起来就形成一个表达式；对于真正的程序设计语言如何构造表达式来说，这条规则是典型的。第5章介绍通常所说的上下文无关文法。
2. 正则表达式也表示数据的结构，特别是文本串。在第3章将会看到，正则表达式描述的串模式恰好与有穷自动机描述的一样。这些表达式的风格与文法的风格有显著不同。这里只举一个简单的例子。UNIX风格的正则表达式'[A-Z][a-z]*[ ][A-Z][A-Z]'表示首字母大写的单词后面跟着一个空格和两个大写字母。这个表达式表示文本中城市和州的模式，如Ithaca NY。这个表达式不能表示有多个单词的城市名，如Palo Alto CA，这可以用更复杂的表达式

```
'[A-Z][a-z]*([ ][A-Z][a-z]*)*[ ][A-Z][A-Z]'
```

来表示。在解释这种表达式时，只需要知道[A-Z]表示从大写字母A到大写字母Z的字符范围（即任意的大写字母），用[ ]表示单个的空格字符。而且，符号*表示“任意多个”前面的表达式。括号用来对表达式各部分进行分组，而不表示所描述的文本中的字符。

4

1.1.3 自动机与复杂性

对计算局限性的研究来说，自动机是必不可少的。正如本章前言中所说，有两个重要问题：

1. 计算机到底能做什么？这种研究称为“可判定性”，计算机能解决的问题称为“可判定的”。第9章讨论这个主题。
2. 计算机能有效地做什么？这种研究称为“难解性”，计算机用不超过输入规模的某个缓慢增长函数的时间所能解决的问题，称为“易解的”。通常，把所有的多项式函数当作“缓慢增长的”，而把比任何多项式都增长得快的函数看作增长得太快了。第10章讨论这个主题。

1.2 形式化证明简介

如果在20世纪90年代以前的任何时候，在高中学习平面几何，就很可能不得不做一些详细的“演绎证明”，即用详细的步骤和理由序列来证明命题的正确性。对于为什么在高中学习这个数学分支来说，几何有实用的一面（例如，要为房间买到大小合适的地毯，就要知道计算矩形面积的公式），但学习形式化证明方法也是同样重要的理由。

在20世纪90年代的美国，流行把证明教成对命题的个人感觉。感觉要使用的命题是正确的，

这是好事，但在高中不再掌握重要的证明技术了。但证明还是每个计算机科学家需要理解的东西。某些计算机科学家采取极端的观点：程序正确性的形式化证明应当与编写程序本身同步进行。作者怀疑这样做是否有成效。另一方面，有人说在程序设计的原理中不包括证明。这个阵
5 营经常提出这样的口号："如果不能肯定程序是正确的，就运行这个程序来看吧"。

作者的立场处在这两种极端之间。程序测试肯定是必要的。但是，测试只能走这么远，因为不可能在每个输入上测试程序。更重要的是，如果程序是复杂的，比如是棘手的递归或迭代，那么在进行循环或递归调用函数时，若不能理解正在干什么，就不太可能写出正确的程序代码。当测试说明代码不正确时，还是需要进行改正的。

为了形成正确的迭代或递归，就要建立归纳假设；形式化或非形式化地推导出这个假设与迭代或递归是相容的，这样做是有帮助的。理解正确程序的工作过程，这个过程与用归纳法证明定理的过程实质上是一样的。因此，在提供某种类型软件的有用模型之外，学习形式化证明的方法学，这成了自动机理论课程的传统。也许比起计算机科学的其他核心课题来说，自动机理论本身有着更多自然的和有趣的证明，既有*演绎*类型的（有根据的步骤的序列），也有*归纳*类型的（带参数命题的递归证明，证明用到带"较小"参数值的命题本身）。

1.2.1 演绎证明

前面说过，演绎证明包含命题序列，其正确性导致从某个初始命题（称为*前提*或*已知命题*）得出"*结论*"命题。证明中每一步都必须根据某条公认的逻辑原理，利用已知的事实或演绎证明中前面的一些命题，或利用这两者的组合。

前提可能为真也可能为假，这通常依赖于参数值。前提常常包含几个独立命题，用逻辑AND（与）联结。在这些情况下，就说每个命题是一个前提或已知命题。

当从前提H到结论C时，所证明的定理是命题"如果H，则C"。说C是从H*演绎*出来的。形如"如果H，则C"的一个示范定理会解释这些要点。

定理1.3 如果$x \geqslant 4$，则$2^x \geqslant x^2$。 □

非形式化地说明定理1.3为真，这是不难的，但形式化证明要求归纳法，将其留给例1.17。首先注意，前提H是"$x \geqslant 4$"。这个前提有参数x，因此前提既不真也不假。实际上，前提的正确性依赖于参数x的值；例如，H对于$x = 6$为真，对于$x = 2$为假。

同样，结论C是"$2^x \geqslant x^2$"。这个命题也用参数x，对于特定的x值为真，对于其他值为假。例如，C对于$x = 3$为假，因为$2^3 = 8$，不大于$3^2 = 9$。另一方面，C对于$x = 4$为真，因为$2^4 = 4^2 = 16$。
6 对于$x = 5$，命题也为真，因为$2^5 = 32$，不小于$5^2 = 25$。

读者也许看出这个直观论证，这个论证说：只要$x \geqslant 4$，结论$2^x \geqslant x^2$就为真。已经看到，对$x = 4$，结论为真。当x增长到大于4时，每次x增加1，左边的2^x就翻一倍。但右边的x^2按照比值$\left(\frac{x+1}{x}\right)^2$来增长。如果$x \geqslant 4$，则$(x+1)/x$不可能大于1.25，因此$\left(\frac{x+1}{x}\right)^2$不可能大于1.562 5。由于1.562 5 < 2，每次x增长到大于4，左边的2^x就比右边的x^2增长得多。因此，只要从类似$x = 4$这样的值开始（这个值已经满足不等式$2^x \geqslant x^2$了），就可以让x随便增长，仍然满足这个不等式。