

应用数学丛书

可计算性理论

张 鸣 华

清华大学出版社

51.8
551
G.2

清 华 大 学
应 用 数 学 丛 书

第 1 卷

可 计 算 性 理 论

张 鸣 华

清 华 大 学 出 版 社

内 容 简 介

可计算性理论是数学的一个分支，是计算机科学的基本理论。

本书分为四章。前三章讨论三种计算模型：递归函数、Turing 机以及 Post 系统，同时讨论递归集、递归可枚举集以及递归谓词、递归可枚举谓词。第四章讨论判定问题以及不可解级。

本书是数学及计算机科学工作者参考用书，也可供数学及计算机科学有关专业高年级学生和研究生作为教材。

可 计 算 性 理 论

张 鸣 华



清华大学出版社出版

北京 清华园

清华大学印刷厂印刷

新华书店北京发行所发行 各地新华书店经售



开本：850×1168 1/32 印张：11 6/16 字数：295 千字

1984 年 1 月第一版 - 1984 年 1 月第一次印刷

印数：1~20000

统一书号：15235.98 定价：1.50 元

关于《应用数学丛书》

为了满足广大科技人员、高等院校教师、研究生进一步学习应用数学的需要，我们编辑出版本丛书。丛书内容将包括应用数学的各个方面、有关的边缘科学以及应用数学的方法等。限于我们的水平和经验，丛书中难免有不少错误和不足之处，诚恳希望广大读者批评指正。

清华大学《应用数学丛书》编辑委员会

1983.4

主编 赵访熊

编委 常 迥 栾汝书 孙念增 黄克智 肖树铁

37803

序 言

可计算性理论是数学的一个分支，是计算机科学的基本理论。

可计算性理论是三十年代中开始形成的。过去除了数理逻辑学家之外，广大的科学工作者，包括数学工作者，对它不太熟悉。但从五十年代以来，由于计算机科学的蓬勃发展，许多数学家及计算机科学家都从事这个方面的研究。可计算性理论的一些基本概念，如 Turing 机、递归集及递归可枚举集等等，是每个从事计算机科学的人员都需要了解的。随着计算机的广泛应用，更多的人将对它进行研究并使它进一步发展。

可计算性理论所要回答的一个主要问题是：究竟什么是计算，计算的精确含意是什么。所谓计算，并不仅仅是数的计算，它有更广泛的内容。大体上讲，凡是电子计算机所能进行的工作都可以叫计算。

电子计算机在开始的时候，只进行科学计算，例如求代数方程的根、求微分方程的数值解等等。这一些是传统的所谓计算，现在往往叫它数值计算。后来计算机越来越多地进行非数值计算。目前，计算机的应用领域已十分宽广，任何一个现代化的企业、实验室以及工程设计，几乎都离不开计算机。它一方面从事于一些过去认为需要高度智力的工作，如编制图书资料索引、进行文字翻译以至证明数学定理；另一方面它开始进入人们的日常生活领域。因此，计算机似乎具有无限的能力。究竟它能够干些什么？它的能力有没有限度？这就是可计算性理论所要研究的基本问题。搞清楚什么是可计算、什么是不可计算，在某种意义上就说明了计算机能力的一种限度。

本书前三章讨论三种计算模型:递归函数、Turing 机以及 Post 系统。回答了什么是计算,什么是可计算。在第一章中,还讨论了递归集、递归谓词以及递归可枚举集、递归可枚举谓词。

这里要说明一下,可计算性理论不仅是建立计算机的理论模型,它也给程序及形式语言提供了理论模型。各种自动机及形式语言的语法就是在 Turing 机及 Post 系统的基础上发展起来的。

第四章讨论判定性,着重讨论不可判定的问题。判定性的概念有重大的实际意义。在计算机科学中,如对程序及形式语言的研究中,有大量的判定问题。这些问题必须与程序及形式语言联系起来讨论,但需要有关于判定问题的一般性理论,这就是本书所要讨论的。

在本书的出版过程中,贾磊同志曾协助进行文字的校对工作,在此表示感谢。

张鸣华

83 年 4 月

目 录

引论	1
(一) 计算及可计算性	1
(二) 可计算性理论的发展	3
(三) Gödel 编号	5
(四) 本书内容简介	10
第一章 递归函数	13
第一节 原始递归函数	15
§1.1 原始递归函数	16
(一) 原始递归函数的定义	16
(二) 原始递归函数的运算	20
§1.2 原始递归谓词	23
(一) 原始递归谓词的定義	23
(二) 原始递归谓词的运算	27
§1.3 原始递归函数的其它定义	29
(一) 递归与迭代	29
(二) 一元原始递归函数	34
§1.4 关于递归运算	37
(一) 多步递归	37
(二) 多变元递归	39
(三) 联立递归	41
(四) 嵌套递归	42
第二节 递归函数	42
§1.5 μ 递归函数及递归谓词	43

(一) μ 递归函数的定义	43
(二) 递归谓词	46
(三) Gödel β 函数	48
(四) 利用 μ 运算消去递归运算	50
§1.6 Ackermann 函数	52
§1.7 一般递归函数	61
§1.8 μ 递归函数与一般递归函数	67
第三节 递归可枚举集及递归集	79
§1.9 递归可枚举集	80
(一) 递归可枚举集	80
(二) 递归可枚举集的运算	81
(三) 递归函数的图形定理	83
§1.10 递归集	89
(一) 原始递归集及递归集	89
(二) 原始递归集及递归集的运算	92
(三) 递归集与递归可枚举集的关系	93
§1.11 递归可枚举谓词	98
第四节 通用函数、递归定理	103
§1.12 Kleene 法式定理、通用函数	104
(一) Kleene 法式定理	105
(二) 原始递归函数的通用函数	107
(三) 递归函数的通用函数、枚举定理	111
§1.13 s-m-n 定理、递归定理	113
(一) 递归函数的 Gödel 编号	113
(二) s-m-n 定理	118
(三) 递归定理	120
§1.14 关于集合及谓词的若干性质	122
(一) 递归可枚举集的编号	122

(二) 谓词的 Kleene-Mostowski 分层.....	126
第五节 字函数.....	133
§1.15 递归字函数的算术化定义.....	133
(一) 字的算术化.....	133
(二) 递归的字集及字函数.....	136
§1.16 递归字函数的直接定义.....	138
(一) 直接定义.....	139
(二) 直接定义等价于算术化定义.....	143
第二章 Turing 机.....	149
第一节 Turing 机的基本概念.....	151
§2.1 Turing 机的概念.....	151
(一) Turing 机及其计算.....	151
(二) Turing 机计算及符号变换.....	157
(三) Turing 机的四元组指令.....	159
§2.2 Turing 机的标准化及组合.....	162
(一) Turing 机的标准化.....	162
(二) Turing 机的组合.....	167
第二节 Turing 可计算函数.....	168
§2.3 递归函数是 Turing 可计算函数.....	168
§2.4 Turing 可计算函数是递归函数.....	180
第三节 通用 Turing 机.....	186
§2.5 通用 Turing 机.....	186
(一) 通用 Turing 机.....	186
(二) Turing 机的枚举.....	192
§2.6 关于小的通用 Turing 机.....	192
第四节 Turing 机的变形.....	197
§2.7 多头多带 Turing 机.....	198
(一) 半无限带的 Turing 机.....	199

(二) 多读写头的 Turing 机.....	201
(三) 多带 Turing 机.....	204
§2.8 W 机、URM 及算子算法.....	205
(一) W 机	205
(二) URM	211
(三) 算子算法.....	216
§2.9 Minsky 机.....	218
(一) Minsky 机与算子算法.....	218
(二) 三带 Minsky 机.....	220
(三) 二带 Minsky 机.....	223
第五节 递归函数的子类.....	227
§2.10 原始递归函数的层次.....	228
(一) Grzegorzcyk 层次.....	228
(二) 初等函数.....	231
§2.11 初等函数的层次.....	233
(一) 基本概念.....	233
(二) 函数类 $\mathcal{F}^0$	237
(三) 函数类 $\mathcal{F}^i$	242
§2.12 可计算函数的复杂性类.....	243
(一) 计算复杂性的尺度.....	244
(二) 复杂性类.....	248
(三) 分层问题.....	250
第三章 Post 系统.....	255
第一节 半 Thue 系统及 Thue 系统.....	256
§3.1 半 Thue 系统及 Thue 系统.....	256
(一) 半 Thue 系统.....	256
(二) Thue 系统.....	259
§3.2 半 Thue 系统与可计算性.....	260

(一) Turing 机可计算是半 Thue 系统可计算.....	260
(二) 半 Thue 系统的字集合与 递归可枚举集.....	262
第二节 Post 系统	264
§3.3 正规系统.....	265
§3.4 tag 系统.....	268
§3.5 标准系统.....	276
(一) 标准系统.....	276
(二) 标准系统与正规系统.....	278
第三节 马尔科夫算法.....	288
§3.6 马尔科夫算法.....	289
(一) 马尔科夫算法的基本概念.....	289
(二) 马尔科夫算法的组合.....	292
§3.7 马尔科夫算法与递归函数.....	299
第四章 判定问题.....	305
第一节 基本概念.....	306
§4.1 判定问题的基本概念.....	306
(一) 历史背景.....	306
(二) 基本定义.....	308
(三) 基本方法.....	310
§4.2 非递归函数的例子.....	310
第二节 若干递归不可解的判定问题.....	314
§4.3 Turing 机的停机问题.....	314
(一) 停机问题.....	314
(二) Turing 机的完全性及等价性.....	315
§4.4 字问题及 Post 对应问题.....	319
(一) 半 Thue 系统及正规系统的字问题.....	319

(二) Thue 系统的字问题.....	321
(三) Post 对应问题.....	322
第三节 不可解级.....	326
§4.5 相对递归及 Turing 级.....	327
(一) 相对递归.....	327
(二) 带有解算装置的 Turing 机.....	329
(三) Turing 归约及 Turing 级.....	330
§4.6 $(m, 1)$ 级及 $(1, 1)$ 级.....	333
(一) $(m, 1)$ 归约及 $(m, 1)$ 级.....	333
(二) $(1, 1)$ 归约及 $(1, 1)$ 级.....	337
§4.7 递归可枚举级.....	338
(一) 创造集及生产集.....	338
(二) 完全集.....	342
(三) 简单集及禁集.....	344
参考文献.....	348

引 论

(一) 计算及可计算性

无论在日常生活中，还是在工程设计和科学研究中，都需要计算。因此，人们一直在广泛地进行计算。但是，只是到最近几十年，才对计算本身进行深入的研究。

所谓计算当然包括数的加减乘除，也包括函数的微分、积分，微分方程的求解等等。还可以包括定理的证明推导。抽象地说，所谓计算就是从一个符号行 ξ 得出另一个符号行 η 。譬如说从符号行 $12+3$ （这是由 1、2、+ 及 3 等四个符号组成的符号行 ξ ）得出 15

（这是由 1 和 5 两个符号组成的符号行 η ），这个计算就是加法。如果符号行 ξ 是 $x \cdot x$ ，而符号行 η 是 $2x$ ，从 ξ 到 η 的计算就是微分。从这个角度来看，文字翻译也是计算。譬如 ξ 代表一个英文句子，就是由英文字母及标点符号组成的符号行，而 η 是含意相同的中文句子，那末从 ξ 得出 η 就是把英文翻译成中文。所以，文字翻译是计算。定理证明也是。令 ξ 表示一组公理及推理规则，令 η 是一个定理，那末从 ξ 得出 η 就是定理 η 的证明。

这里要强调一下，所谓从 ξ 得到 η 是指能够从 ξ 出发，在有限步内真正具体求出 η 。计算必须是能够具体进行的。譬如看下列函数：

$$f(x) = \begin{cases} 1 & \text{若在 } \pi \text{ 的十进位数表示中有连续 } x \text{ 个 } 5; \\ 0 & \text{其它情况} \end{cases}$$

这是一个完全定义的函数。给出 x ，就有对应的函数值 y 。因此，

也可以说从 x 能得到 y 。但是实际上 y 值求不出来,当 $x=100$ 时, y 等于多少? 所以, 计算(或者说算法)应该是具体的, 是真正可以进行的一个过程。

但是, 什么叫真正可以进行? 其含意模糊不清。从英文翻译成中文, 从 x^2 计算 $2x$, 从这个定理推出那个定理, 或是从一个数求出它的平方根, 这一切当然都是可以真正进行的。它们之间有什么共同点? 为什么把它们都叫做计算?

为了回答究竟什么是计算, 什么是可计算, 我们采用建立计算模型的办法。譬如说, 采用计算机作为计算模型, 那末, 凡是计算机所做的工作都叫计算; 凡是计算机所能完成的就叫可计算。当然, 这种说法有严重的缺陷。计算机有各种型号, 功能各不相同。这种计算机可以计算的, 那种计算机不一定能计算; 今天的计算机不能计算的, 明天的计算机也许可以计算。因此, 利用计算机得不到关于计算及可计算性的恰当的概念, 也就是说, 计算机不是一个合适的计算模型。

我们将看到, Turing 机是一个合适的计算模型。因此, 什么是计算? Turing 机所进行的工作就是计算。什么是可计算? Turing 机能够进行的就叫可计算。当然这些话是无法证明的, 我们无法证明 Turing 机的确是合适的模型。但是下列命题是可以接受的: 可计算就是 Turing 机可计算。这个命题叫 Church 论题或 Church-Turing 论题。关于 Church 论题的合理性下面还要讨论。

Turing 机是有确切定义的, 所以计算及可计算性也就有了确切的含意。

Turing 机与电子计算机是相似的, 它们有许多共同之处, 但也有重大的区别。Turing 机与电子计算机的主要区别是它有无穷大的存储量。因此, 根据 Turing 机得到的可计算性概念与通常直观的可计算性是不同的。由于 Turing 机的存储量无穷大, 而对 Turing 机的计算时间又没有限制, 一个计算过程, 它所用的存储

量毫无限制，而且不论是进行几小时，还是几年，甚至几千万年，只要能得到结果，就叫可计算，这种可计算实际上可能根本无法进行的。所以，可计算或 Turing 机可计算，并不是现实可计算。为了研究现实可计算，就要研究计算复杂性。如果不太复杂，可计算性是现实的；如果太复杂了，就是不现实的。关于这些问题，不是本书的内容。

虽然可计算不一定现实，但是如果一个问题不可计算，也就是 Turing 机无法计算，那么这个问题实际上更是无法计算了。因此，从这个意义上看，可计算性理论也有重大的实际意义。它在某种意义上说明了计算机的限度，也就是说明哪些问题是计算机所无法解决的。

本书的目的就是建立计算的模型，从而说明究竟什么是计算，什么是可计算，什么是不可计算。这就是可计算性理论。

最后，再解释一下算法的概念。直观上讲，算法就是一系列计算规则。给定初始数据以后，根据算法就能得出一个计算或计算过程。例如求两个数 a 和 b 的最大公因子的 Euclid 算法，这是一个算法。在具体给定 a 、 b 这两个数以后，就有一个计算过程。因此，建立了精确的计算概念，也就建立了精确的算法概念。所以本书的主要目的也就是要说明究竟什么是算法。

（二）可计算性理论的发展

计算机及计算机科学的发展推动了对计算的研究。但是可计算性理论的形成与电子计算机无关，它是在数理逻辑的研究中产生出来的。前面说过，定理的证明及推导是一种计算，而数理逻辑的研究对象正是这种计算。数理逻辑学家研究这种特殊类型的计算，从而建立了可计算性理论。

本世纪初，数学基础的研究蓬勃发展。Hilbert 认为，各数学分支都可以通过一阶逻辑来形式化。逻辑系统是一些公理及一些推理规则。从公理出发，利用推理规则所得到的命题就是定理。这个推导过程就是定理的一个证明。一个命题是否是定理，就看能不

能找到它的一个证明。在二十年代中，Hilbert 学派大力研究下列问题，这个问题他们称之为数理逻辑的基本问题，或判定问题。这个问题的大致意思是：对于一阶逻辑系统，能否找到一种有效的办法可以判断任意一个命题是否是定理？这相当于对于一个逻辑系统，寻找一个有效的证明定理的通用方法。

但是什么叫有效的方法？在三十年代中，数理逻辑学家提出了种种机构，他们认为可以由这些机构进行的就是有效的办法。既然这种机构可以有效地进行定理的证明，实际上它们也就是计算的模型。这样的模型主要有四种。一种是递归函数，这是 Gödel、Herbrand 及 Kleene 建立的（见 Gödel(1931)及 Kleene(1936)），一种是 λ 演算，这是 Church 及 Kleene 建立起来的，一种是 Turing 机（见 Turing (1936) 及 Post (1936)），还有一种是 Post 系统（见 Post(1943)）。这种种模型各不相同，表面上看区别很大，因为它们从不同的角度来看定理的证明过程，或者说计算过程。

但实际上这种种模型却是等价的，无论采用哪个模型都一样。前面说到 Church 论题是：可计算就是 Turing 机可计算。既然上述种种模型是等价的，那么 Church 论题也可以叙述为：可计算就是某种 Post 系统可计算，或者说：可计算函数就是递归函数等等。

既然各种不同的观点得到的结果都一致，那就说明这些结果一定反映了某种本质的东西。所以，Church 论题是合理的，是可以接受的。在这个意义上，Church 论题得到了证明。因此，无论是 Turing 机，还是递归函数，还是 Post 系统，都是合理的计算模型。这样，计算及可计算性的概念就有了稳固的基础。

对于计算及可计算性的概念来讲，还要提一下 Hilbert 第 10 问题。这个问题的研究与可计算性理论有密切的联系。1900 年，Hilbert 在国际数学家大会的演讲中提出了 23 个问题，其中第 10 个问题是问：有没有一个有效的办法可以判断任意一个 diophantine 方程是否有解？这样的方法没有找到。因此，人们怀疑这样的方法

根本不存在。为了说明有效的方法不存在，首先必须明确什么叫有效的办法。有效的办法就是一个算法。因此，对计算及可计算性建立精确的概念，就给 Hilbert 第 10 问题的解决提供了可能性。

关于计算及可计算性的概念是在 1936 年前后建立起来的。这是数理逻辑学家在研究数理逻辑的基本问题时得到的结果，它与电子计算机或一般的计算问题没有关系。当四十年代中开始研制电子计算机时，计算模型已经成熟，可计算性理论的基础已经奠定。不是计算机出现之后才有可计算性理论，相反，正是可计算性理论，特别是 Turing 机的理论推动了计算机的研制。von Neumann 在二十年代中大力从事数理逻辑的研究，后来他把三十年代中数理逻辑的成果带到电子计算机的设计中来。所以，正是 Turing 机提供了电子计算机的理论模型，使得电子计算机能够顺利地产生。

尽管可计算性理论给设计电子计算机提供了理论基础。但是在计算机成功之前，可计算性理论只是一部分数理逻辑学家关心的课题，并未引起数学家的普遍注意，更不用说广大的科学技术人员了。只是在计算机事业兴旺起来之后，可计算性理论才走出了数理逻辑学家的狭窄的圈子，得到了巨大的发展。现在它不仅是一个数学分支，而且是整个计算机科学的基础理论，是计算机科学家必须具备的基本理论。无论是研究算法，研究程序语言，还是研究程序，它都是必需的基本理论。

(三) Gödel 编号

计算的对象虽然是五花八门，但大致可以区别为两类：一类是数；一类是符号行。当然，数是由数字组成的，所以也可以看做是一类特殊的符号行。现在要说明的是反过来，任何符号行的计算可以看做是数的计算。这就是 Gödel 编号的作用。利用 Gödel 编号，可以把符号行的计算化为自然数的计算，这就是所谓的算术化。算术化是数学的一个重大发展。