现代数学基础丛书

递归论

● 莫绍揆 著

科学出版社

现代数学基础丛书

递　归　论

莫绍揆著

科　学　出　版　社

1997

内 容 简 介

本书是一本入门书，对递归论的各个发展方向(古典的与新兴的)都作了比较详细而有系统的介绍. 前四章是初等部分，详细讨论了递归函数类及其各重要子类，并以算子概念贯穿整个讨论，使读者有巩固的基础知识. 后四章分别介绍递归枚举性、判定问题、谱系与计算复杂性、化归与不可解度论，将读者引导到科研前沿. 本书可供大学数学系本科生或研究生作为递归论的教材或参考书.

现代数学基础丛书
递 归 论
莫绍揆 著
科学出版社出版
北京东黄城根北街 16 号
邮政编码：100717
中国科学院印刷厂印刷
新华书店北京发行所发行 各地新华书店经售
*
1987 年 11 月第 一 版 开本：850×1168 1/32
1997 年 8 月第二次印刷 印张：10
印数：2951—4950 字数：259,000
ISBN 7-03-000050-1/O · 12
定价：18.00元

序　　言

递归论在数理逻辑中是发展较快的一个分支．它不但与计算机科学有着十分密切的关系，彼此互相促进，共同发展，而且就纯理论而言，递归论无论在内容上还是在方法上，在短期间内都有了较大的发展．在目前，要想用一本书来综述其全部成果几乎是不可能的．

本书是一本入门书，在递归论的各个发展方向上，例如可偏函数，递归枚举性，判定问题，谱系理论，计算复杂性，化归与不可解度等(不包括其推广，即所谓广义递归论)，作了一些扼要的介绍．关于广义递归论以及其他一些推广，本书不予叙述．

介绍递归论中比较深一些的理论的书，对于初等部分一般只作简略叙述，这就要求读者对比较初等的部分预先有透彻的理解．我认为，在研讨比较深的理论的同时，仍需对初等部分加以巩固和提高，这对读者是有好处的．因此，本书前四章对初等部分作了详细叙述，并用算子概念贯穿整个讨论．

本书在许多地方用作者自己的独立见解加以组织和整理，以期能更加明晰而有系统．如有错误，希望读者批评指正．

莫绍揆

于1986年2月

目　录

绪　论

§01. 递归论的对象

递归论又叫做能行性论，主要讨论有关"能行计算"、"能行判定"的问题.

凡与"能行性"有关的讨论，都是处理离散对象的.因为非离散的对象，所谓连续的对象，例如处处稠密的对象，是很难能行地处理的. 离散对象以自然数为最简单又最常见，因此能行性论便以讨论自然数为主. 在本书中所谓自然数，包括正整数及 0（而不象一般书中那样，只限于正整数).

人类认识自然数已经有数千年的历史了，并对它进行了四则运算(即加、减、乘、除)，但由于对自然数一般不能够进行无限制的减与除，人们便推广而引进负(整)数及有理数，还进而引进实数与复数，这都给人们带来更大的方便，也使人们对外界的认识更为深入，但同时却使人们对自然数本身的特性(离散性与能行性)有所忽略了，亦即，对自然数本身所特有的特性没有进行深入的研究.

十六世纪 Pascal 开始明确地使用数学归纳法，这是对自然数的特性的首先注意. 以后，Dedekind 与 Peano 在发展自然数论时，不但更频繁地更有系统地使用数学归纳法，而且也使用原始递归式来定义新函数. 到 1923 年，Skolem 便明确地宣称，所有初等数论中的数论函数都可以通过原始递归式而定义，这样原始递归函数便开始展现其重要性. 1931 年当 Gödel 证明他的有名的不完全性定理时，原始递归函数起到了极重要的作用. 到此，原始递归函数论便正式出现了.

从 Skolem 的宣称开始，人们便发问：是否一切可计算的数论函数都是原始递归函数. 这个问题，很快便由 Ackermann 给以

解答，答案是否定的. Ackermann 具体地作出一个函数，它的增长速度远远超过一切原始递归函数，但它却的确是可计算的. 因此，找寻更为广泛的可计算函数类便提到日程上来了.

根据 Herbrand 一封信的暗示，Gödel 建议引进一般递归函数，经过 Kleene 的改进与阐明后，现在已经成为一般书中所采用的一般递归函数的定义. 这样定义的一般递归函数，它比原始递归函数更广，那是没有疑问的. 人们要问：这样定义的函数是否已经包括很广以致于人们所说的"可计算函数"都在内呢？如果还有更广的(可计算函数)，这类更广的函数又该怎样定义呢？

1936 年，Church 提出一个有名的论点：人们一般所说的可计算函数恰巧便是一般递归函数. 同年，Turing 也提出一个论点：人们一般所说的可计算函数，恰巧便是可用 Turing 机（由 Turing 所构造的一种机器)所计算的函数. 很快，便由 Kleene 证明了，一般递归函数也就是可用 Turing 机所计算的函数. 因此这两个论点实质上是同一个论点，现在通称为 Church-Turing 论点.

当这两个论点提出时，看来过于武断，很可能会找出一些可计算的但不是一般递归的函数. 但随着时间的流逝，反例不但没有找出，而有利于这个论点的事项却越来越多，因此，在数理逻辑界，现在一般都承认 Church-Turing 论点，反对的人几乎是没有了.

这个论点得到普遍的承认并不意味着递归论(能行性论)已经大功告成，不能再行发展了. 恰恰相反，根据这个论点，我们可以做很多事情，为以前所不能做的.

第一，有了这个论点以后，我们可以断定某些问题是不能能行地解决的或不能能行地判定的. 在未有这个论点之前，我们只能设法去找出能行的解决办法或能行地判定的办法，找出以后便说，某某问题可以能行解决，某某问题可以能行判定. 万一这种办法找不到，我们却不敢断定，某某问题不能能行解决或某某问题不能能行判定，因为"找不出"不等于"不存在". 只有接受 Church-Turing 论点以后，把"能行地解决"解释为"有一般递归函数"，把

"能行地判定"解释为"有一个一般递归谓词",那末当我们证明"没有相应的一般递归函数"或"没有相应的一般递归谓词"时,我们便可以作出否定的答案了. 正因为如此,长久不能解决的 Hilbert 第十问题(判定不定方程是否有解)到现在便得到否定的答案了. 这是能行论的一个重要成就.

不但如此,根据这个论点而把能行性等同于递归性以后,我们还可以不断地深入探究,无论在能行性方面以及非能行性方面都有好些深入的有意义的成果,远非以前所能达到的.

可计算的函数既等同于一般递归函数,我们还要求讨论可计算的复杂性,这就要求对一般递归函数再作分类,将一般递归函数分成各阶层,由最简单的到较复杂的,最后到最复杂的. 在这方面,以前已有原始递归函数,它是一般递归函数的特例. 现在还进而研究各种各样的特例,在原始递归函数与一般递归函数之间的有多重递归函数,在原始递归函数之内的(作为其特例的)又有初等函数,初基函数乃至五则函数等等.对这些函数的性质的深入研究,不但使我们对数论函数有更深入的了解,而且对自动机、计算机的研究也大有补益.

其次,我们对非递归函数也作了深入研究,从而发展了不可解度论、分层论等等,这些都从反面使我们对能行性有更进一步的了解.

由此看来,递归论的内容非常丰富,很值得我们深入探讨.

另外,人们又将递归论从各方面作推广,得出所谓广义递归论.

一个推广的方向是把递归论的对象从自然数推广到一般(有限)字母表上的字去,即不但讨论数论的递归函数,而且讨论字的递归函数,并且把前者作为后者的特例(数论的递归函数可以看作是单字母表上的字的递归函数). 由于计算机所能处理的,实质上都是一些表达式,从而计算机实际上是处理两个字母表上的字的函数,因此这种推广更能直接应用于计算机,从而更显出其重要性.

另一个推广是把自然数推广为一般的序数(包括无穷序数),

从而不但研究自然数上的递归函数，而且研究无穷序数上的递归函数；这样，研究的对象更广了．

但在本书中，我们不想讨论这两种推广．对前者的推广而言，我们觉得，原来的递归函数论专就自然数本身立论，不牵涉到它的表达式，这是更简单而且是更根本的．如果讨论 n 个字母的字母表上的字的函数，表面看来是推广了(从一个字母推广到 n 个字母)，实质上却是把注意力从自然数本身而转移到自然数的 n 进制表示上去．除非我们专门集中注意力于 n 进制，否则这种转移是没有好处的．因此作者认为，如想作这种推广，最好放在"算法论"内去讨论，在递归论内可不考虑自然数的表达式．

至于把递归函数推广到无穷序数上去，这种推广与原来的着眼点大不相同，这时讨论的主要精力集中在"无穷序数"上去，而"递归性"(能行性)反而隐而不现，消失于无形了．作者认为，有关这部分的推广，最好另行处理，不要与递归论放在一起．

因此，本书仍然集中力量于自然数上的递归论．

§02. 基本概念，组成规则

如上所述，在递归论中我们以自然数集为讨论的全域．在数理逻辑中我们主要讨论公式及谓词，在递归论中，我们主要讨论项及函数，但也不完全排除公式及谓词．

既然我们讨论的全域为自然数集，从而个体便是自然数(以后省称为数)．个体变元即数变元，个体常元即常数．

将数变成数的叫做(数论)函数，将数变成真假的叫做(数论)谓词，将真假变成真假的叫做(命题)联结词，而将真假变成数的叫做特种函数．我们只用一个特种函数 ct，它把真、假分别变为 0，1．即 ct (真) = 0，ct (假) = 1．

此外，把函数变成函数的叫做算子，把谓词变成函数的叫做摹状词，把谓词变成谓词的叫做量词．至于把函数变成谓词的迄今尚无人使用，可不必讨论．这些(算子、摹状词、量词)都叫做约束

词.

各种词的性质须分别具体规定，但其公共的性质却可以用组成规则确定如下.

递归论系统的组成规则

数变元 $x_1, x_2, \cdots$

数常元 $0, 1, 2, \cdots$

函数符号 $f_1, f_2, f_3, \cdots$(并指定其元数)

常函数 具体指定(见后)

谓词符号 $P_1, P_2, P_3, \cdots$(并指定其元数)

常谓词 具体指定(见后,有一个为二元相等词 $=$)

特种函数 ct(一元)

联结词 $\neg$(非), $\wedge$(且), $\vee$(或), $\longrightarrow$(如果则), $\longleftrightarrow$(恰当)

算子符号 $\tau_1, \tau_2, \cdots$(并指定其型)

常算子 具体指定(见后)

量词符号 $Q_1, Q_2, \cdots\cdots$(并指定其型)

常量词 $\forall, \exists$(其余见后)

摹状词符号 $\iota_1, \iota_2, \cdots$(并指定其型)

常摹状词 ι, μ(其余见后)

组成规则

项及**公式**递归地定义如下:

(1) 数变元与数常元为**项**;

(2) 如果 f 为 n 元函数而 $\xi_1, \cdots, \xi_n$ 为 n 个项,则 $f(\xi_1, \cdots, \xi_n)$为**项**;

(3) 如果 P 为 n 元谓词而 $\xi_1, \cdots, \xi_n$ 为项,则 $P(\xi_1, \cdots, \xi_n)$ **为公式**;

(4) 如果 α 为公式则 $\mathrm{ct}\alpha$ 为**项**;

(5) 如果 α, β 为公式,则 $\neg\alpha, \alpha\wedge\beta, \alpha\vee\beta, \alpha\rightarrow\beta, \alpha\longleftrightarrow\beta$ 为**公式**;

(6) 如果 τ 为(m,n,s)型算子, $e_1, \cdots, e_m$ 为数空位, $\xi_1, \cdots, \xi_n$ 为项, $\eta_1, \cdots, \eta_s$ 为函数,则 $\tau(e_1\cdots e_m)\rightarrow(\xi_1\cdots\xi_n)\{\eta_1, \cdots,$

$\eta_s\}$ 为**项**；

(7) 如果 Q 为(m,n,s)型量词，$e_1,\cdots,e_m$ 为数空位，$\xi_1,\cdots,\xi_n$ 为项，$\alpha_1,\cdots,\alpha_s$ 为谓词，则 $Q(e_1,\cdots,e_m)\rightarrow(\xi_1,\cdots,\xi_n)\{\alpha_1,\cdots,\alpha_s\}$ 为**公式**.

(8) 如果 L 为(m,n,s)型摹状词，$e_1,\cdots,e_m$为数空位，$\xi_1,\cdots,\xi_n$为项，$\alpha_1,\cdots,\alpha_s$ 为谓词，则 $\iota(e_1,\cdots,e_m)\rightarrow(\xi_1,\cdots,\xi_n)\{\alpha_1,\cdots,\alpha_s\}$ 为**项**.

所谓**项**及**公式**仅限于根据以上而作成的.

这里引进几个定义.

定义算子、量词、摹状词合称**约束词**.

(2) 中的 $f(\xi_1,\cdots,\xi_n)$ 叫做 f 在 $(\xi_1,\cdots,\xi_n)$ 处的**值**，也叫做 f **填式**，它是 f 作用于 $(\xi_1,\cdots,\xi_n)$ 的结果.

(3) 中的 $P(\xi_1,\cdots,\xi_n)$ 叫做 P 在 $(\xi_1,\cdots,\xi_n)$ 处的**值**，也叫做 P **填式**，它是 P 作用于 $(\xi_1,\cdots,\xi_n)$ 的结果.

对(4),(5)中的 $\mathrm{ct}\alpha$ 等仿此定义.

(6),(7),(8) 中的 $(e_1,\cdots,e_m)$ 叫做**约束空位组**，明确地说是相应约束词 τ，Q，ι 的指导空位组. 而 $(\xi_1,\cdots,\xi_n)$ 叫做**新添项**. $\{\eta_1,\cdots,\eta_s\}$ 叫做**作用域**（第一作用域至第 s 作用域）. $\{\eta_1,\cdots,\eta_s\}$ 中的变元 $(e_1,\cdots,e_m)$ 叫做受相应的算子、量词、摹状词所约束的**约束空位**. 作用域中与约束空位不同的空位叫做**参数空位**. 项 $\xi_1,\cdots,\xi_n$ 中的空位，不管是否 $e_1,\cdots,e_m$ 都叫做**自由空位**.

参数空位或参变元亦可作代入，但我们规定：对被作用函数(即作用域)中的空位或参数作代入时，不允许把所代入的项填入作用域中的参数空位或参变元处去，而必须写在约束词的下面，与约束空位、新添项并列而用分号隔开. 例如，对 $\alpha_{e_1\rightarrow u}f(e_1,e_2,v)$ 作代入 $e_2\rightarrow A$，$v\rightarrow B$ 时，不能写为 $\alpha_{e_1\rightarrow u}f(e_1,A,B)$ 而必须写为 $\alpha_{e_1;e_2,v\rightarrow u;A,B}f(e_1,e_2,v)$. 至于在什么情况下它与 $\alpha_{e_1\rightarrow u}f(e_1,A,B)$ 相等，须作详细的讨论.

在上面的组成规则中，我们只有新项及新公式的组成，而没

有造成新函数、新谓词、新约束词的方法. 对于函数、谓词以及约束词可以说我们只能限于开始时引入的基本的函数、基本的谓词以及基本的约束词. 即组成规则中所列举的. 在此以后只有含变元的新项及新公式,并没有新函数、新谓词. 这是沿用数学界自古以来的习惯: 大量使用(含变元的)项及公式, 而(除基本的以外)少使用函数及谓词.

这种做法有很多的优点. 最大的优点是: 根本上取消了高级函词与高级谓词,亦即以函数或谓词为变目的那种高级函词谓词. 凡通常所说的以函词或谓词为变目的, 这里都改为以含变元的项或公式作变目了. 这里不但使用项对函数、谓词的填入,即由 f 及 $\xi_1,\cdots,\xi_n$ 而造 $f(\xi_1,\cdots,\xi_n)$, 并且还使用了算子量词等约束词. 这些约束词对应于高级函词高级谓词, 但它们都以(含变元的)项或公式作为变目, 从而根本堵绝了高级函词谓词产生的可能. 当然,约束词本身也带来一些新奇性, 下面再细论. 但容易想象到, 函数及谓词本身是必要的,不能恒用项或公式来代替,而且除基本的函数及谓词外,由任意新造成的项也可抽象而得一个新函数,由任意新造成的公式也可抽象而得一个新谓词. 这个抽象运算 (它本身也是一个约束词)是必要的. A. Church 引进符号 λ,从而当 η 为项时, $\lambda_{x_1\cdots x_m}\eta$ 便是依 $x_1,\cdots,x_m$ 对 η 抽象而得的 m 元函数, 即 $(\lambda_{x_1\cdots x_m}\eta)(x_1,\cdots,x_m)=\eta$. 又如果 α 为公式时, $\lambda_{x_1\cdots x_m}\alpha$ 便是依 $x_1,\cdots,x_m$ 对 α 抽象而得的 m 元谓词,即 $(\lambda_{x_1\cdots x_m}\alpha)(x_1,\cdots,x_m)\longleftrightarrow\alpha$. 换句话说,如把 $\lambda_{x_1\cdots x_m}\eta$ 与 $\lambda_{x_1\cdots x_m}\alpha$ 分别记为 f 及 P, 则有 $f(x_1,\cdots,x_m)=\eta$ 与 $P(x_1,\cdots,x_m)\longleftrightarrow\alpha$.

这个约束词与上面的约束词不同, 上面的约束词的值是项或公式,而这个约束词的值却是函数(f)或谓词(P).

通常很少使用约束词 λ, 很多书甚至于根本不提它. 当使用函数 $\lambda_x\eta$ 时,就使用项 η 本身(含变元的项), 当使用函数 $\lambda_x\eta$ 在 y 处之值,即 $(\lambda_x\eta)(y)$ 时就用 $\underset{x\to y}{代}\eta$ 或 $\eta\binom{x}{y}$. 在通常的数理逻辑或数学中,"代入"运算 (处处将 x 代入以 y) 是一个非常重要的

运算，实质上它是“将 η 代 x 抽象而得一函数再求该函数在 y 处之值”. 从此可见，有“代入”运算后可以代替抽象运算 λ，反之，有抽象运算后，代入运算可以表示为 $\underset{x\to y}{代}\eta = (\lambda_x\eta)(y)$. 通常书中不用运算 λ，故处处使用“代入”了.

由于约束词(以及所谓“约束变元”)的出现，使得项的代入、函数的代入、谓词的代入遇到非常的困难. 最初数理逻辑家所表达的代入原则每每有毛病，经过十多年的探讨修正，才初步获得一些正确的表述，但正确的表述每每麻烦而琐碎，很难认为是合用的表达形式.

试就算子 $\alpha_{e\to v}$ 而论，假设它把 $f(e,e_1,u)$ 改造为 $g(v,e_1,u)$，即有(e_1 为参数空位，而 u 为参数)：

$$g(v,e_1,u) = \alpha_{e\to v}f(e,e_1,u)$$

如果我们把项 A 代入空位 e_1 处，又把参变元 u 代入以项 B，依现时流行的说法，我们不能无条件地得到

$$g(v,A,B) = \alpha_{e\to v}f(e,A,B).$$

要使它成立，必须 A,B 中没有自由空位 e，而且在 $f(e,A,B)$ 中，A,B 的自由空位不受约束. 数理逻辑家为此需作相当长的讨论.

依我们看来，左边的 $g(v,A,B)$ 显然是“先实施算子 $\alpha_{e\to v}$ 再作代入 A,B”，而右边则是“先作代入 A,B，再实施算子 α”，两者是不同的运算过程，因此它们有时相等，有时则不相等，那是很清楚，很容易理解的事，用不着大惊小怪. 问题是：以前我们没有表示“先作算子再代入”的符号，只能先引入新符号 $g(e_1,u)$ 表示实施算子 α 的结果再作代入，至于记号

$$\alpha_{e\to v}f(e,e_1,u)\Big|_{\substack{e_1=A\\u=B}}$$

则不是正式的记号，以致当两边不相等时，我们竟然无法表示“先实施算子(约束词)再代入”这个过程. 现在，我们建议下列记号：

$$g(v,A,B) = \underset{(e;e_1,e_2)\to(v;A,B)}{\mathrm{rti}} f(e,e_1,e_2)$$

分号“;”之左是约束空位与新添项，分号“;”之右是参数空位及其代入项. 凡代入必不能在约束词的作用域中对其参数或空位作代

入，如要代入，只能写在约束词的下方，如上式所示．这时代入永远是可行的没有任何限制．如要在作用域中作代入，则有：如果代入项（上文的 A,B）中的自由空位没有与某约束词的约束空位同名，则这些代入项可以代入到该约束词的作用域中去．因为这时约束关系不变，故代入合法．

例如，当 f 为基本函词而 A,B 中没有空位为 e 时，我们有：

$$\underset{(e_1e_1,e_2)\to(\iota A,B)}{\mathrm{rti}} f(e,e_1,e_2) = \underset{e}{\mathrm{rti}} f(e,A,B).$$

这也正是现行说法中的合法代入．

当 A，B 中含有空位 e 时，现行说法既没有 $\underset{(e_1e_1,e_2)\to(\iota A,B)}{\mathrm{rti}}$ 的记号，只能将约束空位改名，例如改为 e_3（不出现于 A，B 中的），便得

$$\underset{e_3}{\mathrm{rti}} f(e_3,A,B).$$

当然，这式与 $\underset{(e_1e_1,e_2)\to(\iota A,B)}{\mathrm{rti}} f(e,e_1,e_2)$ 是相等的，但由于这种相等便承认永远可以改名，从而永远可在作用域中作代入，而不引用表示“先实施算子再代入”的符号，永远只使用“先代入再实施算子”的过程，这在根本上便是不够妥当的．我们应该有完备的记号系统才对．

和代入相类似的还有一种很重要的运算：替换．很多书把“代入”与“替换”当作相同的运算，（在国外甚至于有使用相同的名称的），但两者是不相同的．代入是对变元的处处代入，替换则是对某复杂项的一处或多处（当然可以处处）的替换，两者本质是不相同的．

例如，设 α 为 $3x = y \to 3 \cdot 3x = 3x+2y$，则 $\underset{x\to u+2}{代}\alpha$ 将是

$$3(u+2) = y \to 3 \cdot 3(u+2) = 3(u+2) + 2y.$$

这里各处的 x 都须代以 $u+2$．反之，在 α 中将 $3x$（看作一复杂项）替换以 B，则可以有下列各结果：

$$B = y \to 3 \cdot 3x = 3x + 2y,$$
$$B = y \to 3 \cdot B = 3x + 2y,$$
$$3x = y \to 3 \cdot B = B + 2y.$$