# 论可计算数及其在判定问题中的应用

A.M.Turing

1．计算机

2．定义

自动机器

计算机

可循环数与不可循环数

可计算的数列和数

3．计算机的例子

4．微型桌面

更有深度的例子

5．列举可计算的数列

6．一般计算机

7．一般计算机的详细描述

8．对角线方法的应用

9．可计算数的数组的例子

10．一个大的可计算的数组的例子

11．可计算数在其判定问题中的应用

附录

Endnotes：

可计算的数可以简要描述为那些十进制形式可通过有限步骤来计算是实数。虽然这篇论文表面上讲的是可计算数，但它也研究了可计算方程，无论整数、实数或可计算变量，等等。这篇论文简单给出了可计算数、方程等之间的关系。这会包括函数理论的发展和实变量用可计算数表示。一个数，如果它是十进制形式，还可以被机器识别，它就是可计算的。

在9.10中给出了一些关于所有数都是可计算的 的讨论。其实，作者给出了一定大的数组都是可计算的。它们包括，算术数学中的实数部分，Bessel函数中是实数部分的零，如x,e等。然而可计算数不全包括可定义是数，其中一个例子就是能找到一个不可计算却可定义的数。

即使可计算数组如此大，在许多方面它与实数集很相似。从正确讨论的应用中，结论与Godel<1>的很相似，着都是有价值的证明。

在Alonzo Church的最近的一篇论文中介绍了“有效计算”的思想，它与作者“计数”的思想很接近，不过又定义的十分不同。Church还探求了有关判定问题的结果。有关“计数”和“有效计算”本质相同的证据，列在了附录中。

1. 计算机

我们知道可计算数是那些十进制可以通过有限步骤计算的，不过我们需要更明确的定义 。而计算机的需求是因为人类的记忆是有限的。

我们可以设想一个人将一个个实数输入一个只能存有限数q1，q2……qr的机器，将被叫做“m—配置”，这个机器被纸带供应，被分为“方格”（也叫做square），每次能生成一个符号。某时只有一个方格叫做“r—th”，生成S(r)的符号。我们就将它称为“扫描方格”，符号被称为“扫描符号”。这样可以用机器来做计算，当符号被写下。会形成数列或是数字，数列与数字都是十进制的可以计算的。其余的记录全都用来记忆。而正是这些记录我们才认为计算是可信任的。

这就是作者的关于书的计算的论点，下节会有关“machine机器”“纸带”和“扫描的”含义与理解。

1. 定义，解说
2. 自动机器

如果这台机器的每个动作都完全由配置决定，我们就将这个机器叫做“自动机器”（或a—机器）。因为一些原因我们会使用部分动作由配置决定的机器（选择机器或c—机器）。当这样一台机器，任意的外部操作都会影响它的运行，没有外部操作它也根本无法运行。这篇论文只研究自动机器。

1. 计算机

如果一台自动的机器包括两种符号，第一种（也叫数字）只包括0和1，其余的叫符号或第二种符号，那么这台机器就可称为计算机。符号打印的子数列称为“机器计算的数列”。二——十进制的数称为“机器计算的数”。

纸带的符号和m—配置被称为完全配置。机器上合纸带上的改变叫做机器的一个步骤。

1. 循环和不循环的机器

如果一台机器从来不写下多于有限个第一类数（0和1），那么它就叫做可循环的机器，否则就叫做不可循环的机器。

1. 计算数列与数

如果一个数列可以被不可循环的机器运算，那么它就是可计算的。

1. 计算机的例子

如果一个数列010101……一个机器有4个m—配置“b”“c”“z”“e”。而且可以打印“0”和“1”。机器的表现为描述以下桌面，“R”表示“机器正在扫描从上次扫描处右侧”，“L”表示从左侧，“E”表示擦除，“P”表示打印。

就像这样

|  |  |  |  |
| --- | --- | --- | --- |
| 配置 | | 表现 | |
| b | 无 | P0,R | c |
| c | 无 | R | e |
| e | 无 | P1,R | z |
| z | 无 | R | b |

一个难一点的例子，我们要计算数列001011011101111011111……

会引入“o”“p”“q”“f”“b”打印“e”“x”“0”“1”。

可替换方块中的一个被称为F-方块，其余的称为E-方块，E-方块的特点是易于擦除。F-方块中的数应该是连续的，而在它运行到结尾之前是没有空格的。两个F-方格之间有一个E-方格。E-方块的明显的可以满足符号打印在E-方块上的需要。如果一个符号J-是在E-方块的s的右边。然后S和J会被称为标记为I。这个进程可以打印为I，I会被记为J（或S）和I记在一起。

1. 微型桌面

几乎所有的机器都在用对应的进程，而且，一些机器用了许多链接。这些工程包括复制符号数列，比较两个数列，擦除已给形式的所有符号，等等。这样我们可以考虑用m-配置用于基于缩写有关的桌面。这些都是“变量”的自然意义。通过吧希腊文转化为德文，我们就获得了m-配置。我们可以用基干桌面造出微型桌面。

基干桌面应该与微型桌面一样：它们不是必须的。

我们可以说这个机器会有m-配置q和所有可得到的m-配置，会成为一个可以把c转化成p(c)的m-配置代替。这样就会得到qYp(q),p(p(q)),p(p(p(q)))。

如果我们用q来代替C，用r代替B，用x代替I，我们会得到一个完整的m-配置，f(q(r,x))的桌面，f就被称为m-配置方程，或m-方程。

可采纳的用来代替的都是m-方程的m-配置机器的符号。那些用明显的列举出来：他们可能包括表示像p(e,x)一类的符号，他们也一定需要m-方程。如果我们不支持明确的列举，而是简单的陈述，机器用当然的m-配置，而且所有m-配置获得的通过用m-方程代替m-配置。然后它会在qYp(q),p(p(q)),p(p(p(q)))，就像m-配置。然后我们的就是这样，我们给出了m-配置关于机器的名称，大部分m-配置会用m-方程来表示。我们也给出了它相对应的基干桌面，所有我们想要的是为m-配置完成机器桌面。这样会得到基干桌面通过反复操作来替代。

更深远的的例子

用符号“/”表示机器进入m-配置。

1. 列举可计算的数列

一个可计算的数列0是决定于一个机器，可以计算0的机器对它的描述。这样，数列001011011101111……就为一个桌面所决定，而且实际上，任何一个可计算的数列都能被这样一个桌面描述。

把这些桌面放到一个标准形式中会很有用。开始让我们假设这个桌面与给出的第一个桌面一样，就是说，那个在运算中的项一般是下列中的一个：E,R:E,L:Pa:Pa,R:Pa,L:R:L:或没有项。这个桌面一般会可以通过介绍更多m-配置成这个形式。现在，让我们给一些数m-配置，称它们为q1,q2……qn。最初的m-配置常被称为q1，我们也给符号S1,S2……Sm一些数，其中特殊的是，空格=S0，0=S1，1=S2，这个桌面的线现在形式……

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| qi | sj | Psk,L | qm | (N1) |
| qi | sj | Psk,R | qm | (N2) |
| qi | js | psk | qm | (N3) |

像下面的线：

Qi sj E,R qm

被写成：

Qi sj PSO,R qm

并且像这样的线：

Qi sj R qm

可以被写成

qi sj psj,R qm

用这种方法我们将桌面上的每条线都缩减为（n1）(n2)(n3)三种形式中的一种。

从（n1）中的每条线来看，我们建立了q1,si,sk,l,qm的表达方式；从(n2)中的每条线来看，我们建立了qi,si,sk,r,qm的表达方式；从(n3)中的每条线来看，我们建立了qi,sj,sk,nqm的表达方式，来为机器从桌面上建立他们，并且用分号分开它们。用这种方法我们得到了一个完整的关于机器的描述。在这样的描述中，我们可以将qi替换为“D”后加一个”A”字母重复i次，并且sj替换为”D”后加个”C“重复j次。这种关于机器新的描述可以被称为”标准描述“（SD）。它完全由字母”A””C””D””L””R””N”和符号”；“组成。

最后，如果我们将”A”替换为L””C”替换为“Z”“Ｄ＂替换为＂３＂＂Ｌ＂替换为”４“，”Ｒ＂替换为“５”，“Ｎ＂替换为”６“，并且”Ｊ替换为“７”，我们就有了一种关于机器的描述，这种描述的形式是阿拉伯数字。这种用数字的整数来代替的形式可被称为机器的数字性描述（Ｄ。Ｎ）。“Ｄ．Ｎ＂决定了”Ｓ。Ｄ“并且机器的构造是唯一的。Ｄ．Ｎ是Ｎ的机器可被描述为Ｍ（Ｎ）。

对每个可计算的数列，对应至少一种描述数字，同时对于没有描述数字的，对应着多于一种可计算的数列。因此，可计算的数列和数字是可列举出来的。

下面，找出＆３中机器Ｉ的描述数字。当我们重新命名。Ｍ－配置时，它的桌面就变成了：

ｑ１　　　　　　　　　　ｓｏ　　　　　　　ｐｓ１，Ｒ　　　　　　　　ｑ２

ｑ２　　　　　　　　　ｓｏ　　　　　　　　ｐｓｏ，Ｒ　　　　　　　　ｑ３

ｑ３　　　　　　　　　ｓｏ　　　　　　　　ｐｓ２，Ｒ　　　　　　　　ｑ４

ｑ４　　　　　　　　　ｓｏ　　　　　　　　ｐｓｏ，Ｒ　　　　　　　　ｑ１

其他的桌面可以由上面两个不同的线相加得到，例如；

ｑ１　　　　　　　　ｓ１　　　　　　ｐｓ１，Ｒ　　　　　　　　　ｑ２

我们最初的标准形式是：

ｑ１ｓｏｓｓ１ｒｑ２；ｑ２ｓｏｓｏｒｑ３；ｑ３ｓｏｓｏｒｑ４；ｑ４ｓｏｓ２ｒｑ１；

标准的描述是：

ＤＡＤＤＣＲＤＡＡ；ＤＡＡＤＤＲＤＡＡＡ；

　　　　　　DAAADDCCRDAAAA;DAAAADDRDA;

数字性描述是：

31332531173113353111731113322531111731111335317

并且可以写成如下:

31332531173113353111731113322531L1173111133531731323253117

一个可循环计数机器的数字性描述的数字可被叫做一个满足的数字。在&8中表明还没有通常的办法来判定一个给定的数字是不是满足的。

6普遍性的计算机

发明一种独立的机器，用来计算任何可计算的数列是可以实现的。

如果我将这部机器在开始时配置一纸带，用来记录某些可计算机器M的S.D，那么我能计算出和M相同的数列。

首先，我们假设我们有一部机器M‘，它将记录F-方格，继续完成机器M的构造。这可能于P235表达形式是一样的，用第二种描述©，将所有符号都写到一条线上。或者，我们可以将这种描述(&5)变换为将每个M-配置替换为“D“后加字母A重复合适的次数。并且A和C的次数与&5中被选的数字一致，这样的话，特别的0被DC替代，1被DCC替代，空格被D替代。这些替代都将实现，在配置被完全结合在一起之后，正如（C）。如果我们首先做替代的话难度会增大。在每个完成的配置中，所有的空格都必须用D来替代，因此完整的配置不能被一个有限的符号来表示。

如果在&3中，机器2的描述中，我们用DAA替代0，DCCC替代E，DAAA替代Q，那么（C）列就变成了

DA：DCCCDCCCDAADCDDC:DCCCDCCCDAAADCDDC:………..(C1)

（这就是F-方格上符号的列）

不难知道，如果M能被制造，那么M‘也能。计算M‘的方式可以被制定，依赖于它本身内部的地方有计算规则；每部都能进行下去，根据这些规则。

我们只需要将这些比率认为能可取的并且可交换的或者，我们有与普通机器很相似的东西。

第一件就是：目前M‘机器还没有出版的图形。我们可以修正它，通过出版完整配置在每个连续对之间的图形，他将出现新的配置而不是旧的。那么（C1）变为DDA:0:0:DCCCDCCCDAADCDDC:DCCC……….(c2)

这并不能很明显的看出Ｅ－方格为必须的“粗糙工作“留足了空间，但事实就是这种情况。

在冒号之间的字母数列被表示为（Ｃ１），可能用完整的配置的标准描述。当字母被数字所代替，如＆５中，我们将得到完整配置的数字性描述，我们将它称为描述性数字。

７普通机器的细节描述

一张关于普通机器作用的表格如下图。机器的Ｍ－配置是所有在表格中第一个和最后一个出现的冒号，与我们写不可缩短的表格时那些以Ｍ－配置的形式出现在表格中的一样，出现在表格中，且是一个Ｍ－配置。它不可缩短的表格是：（Ｐ２３９）

ｅ（ａｎｆ）～ｅ　　　　　　Ｒ　　　ｅ１（ａｎｆ）

ｎｏｔｅ　　　Ｌ　　　ｅ（ａｎｆ）

　　ｅ（ａｎｆ）～Ａｎｙ　　　　Ｒ，Ｅ，Ｒ　ｅ１（ａｎｆ）

　　　　　　　　　　ｎｏｎｅ　　　　　　　　ｅ（ａｎｆ）

最后，ｅ１（ａｎｆ）是Ｉ的一个ｍ－配置

　　　当Ｉ准备开始工作后，纸带旋转，当它承受了Ｆ－方格中的符号Ｅ并且在接下来的Ｅ－方格中；在这之后，只在Ｆ－方格中，机器的Ｓ．Ｄ产生，紧跟着双冒号“：：“。Ｓ．Ｄ包含一系列的指导，用分号分开。

　　　每个说明包含五部分。

１）“Ｄ“后跟着字母Ａ列，这描叙了适宜的Ｍ－配置。

２）“Ｄ“后跟着字母ｃ列，这描叙了扫描信号。

３）“Ｄ‘后跟字母Ｃ的另外一列，这描叙了扫描信号的符号要改变。

４）“L”“R”“N”，描叙了机器是否向左移动，向右移动，或者根本不动。

5）“D”后跟字母A列，这描叙了最终的m-配置。

这部机器能够发行”A””C””D””L””u””V””W””x””y”’Z”S.D是由”j””A””C””D“”L“”R””N”来组成。

8 对角线程序的应用

证明真数不可数的理由也是证明可计算数字和列不可数。可以这样认为：数列的极限一定可计算。

或者我们可以应用对角线程序:”如果可计算数列是可数的。令Ln为nth可计算的数列。令Yn(m)是mth的数字。令J为I-Yn(u)的列。因为Jis是可计算的，存在一个K，使得

1-Yn(m)=YK(n) all n.令n=k，我们得到I=2YK(k)，那就是说，I是复数。这是不可能的。

可计算数列因此是不可能的。

最简单也最直接的证明是证明：如果通用的程序存在，那么就有一个计算J的机器。这个证明，有一个缺点是给读者一种感觉是：“一定有些东西是错误的。“它不解J，但是解J‘，它的n阶数字是Yn(n)。

假设有下面的程序：我们可以发明已机器D，当S.D应用到任何一机器m上时，M将会测试S.D并且M是循环的，将用符号“U“来标记S.D，如果它自由循环将被标记为”S“。

H机器能被分为好几部分。在N-1部分里，包含其他东西。在N-th部分里，D测试数字n 。如果N是满足的，是一个D.N自由循环机器。R（N）中数列的数字，N在D.N中是可计算的。

从H的建造中，我们可以看出H是自由循环的，H中每部分在有限的计算之后都会结束。如果N不恰当的话，那么N-th部分被完成了。如果N恰当的话，这就意味着机器M（N），它的D.N是自由循环的N，因此它的R-th数据可以在有限部计算出来。

使k成为D.N中H的话，假设有两个“s””u”,一方面verdict不能为”s“，另一方面，又不能为’U”，因此我们得出不存在这样的机器D。

我们也可以证明，当应用到机器M中S.D时，也不存在机器R，将决定M是否曾经产生了一个给定的符号。

我们假设，如果存在机器R的话，也就存在一个程序，由一个给定的机器不断的产生0来决定。令M1成为和M产生相同数列的机器，除了第一个由M产生的D，M1产生%，M2将有前两个0符号代替%。因此，如果M被创造，那么，M将产生，

ABA01AAB0010AB……….,

M1产生AB A%! A AB

M2产生AB A% 1AAB% 010…..

通过这些程序的组成，我们有了一个确定M产生有限数据的程序。这种应用可以判定是否我们能判断我们的定义：“可计算“。对每个”普通程序“问题可被表达为一个关于确定一个给定的整数n是否有合适G（n）的问题，并且这就等价于计算一组数，它的n-th数据是1，如果G(n)是正确的，或者的0，如果G(n)是错误的。

9。可计算数的拓展

真正的问题是：可以计算的数据的可能程序是什么？

下面有三种类型：

a直接应用 b两个定义等价的证明 c给出大量可计算数据的例子。

一旦找出所有“可计算“的其他合适位置，尤其是，如果有一个通用程序来确定是否一个Hilber程序计算是可行的，那么就可以确定它可能被一个机器运行。

通常计算是通过在纸上写某些符号来完成的，我们假设这张纸被分为几个部分。在初级计算中，这张纸有事被利用了，但这种使用总被禁止。假设符号的数目可以被有限的产生，那么，如果允许一个无限符号的话，那就存在一个被扩展的符号。

计算机的表现在任何情况下都是由它观察到的符号所决定。假设存在一个B,对于符号或数列的数目来讲可被瞬间观测到。如果它要观察更多的话，必须利用更大的系统。如果我们承认一个无限的思维状态，那么它们中的一些将被关闭或被拒绝。并且，这种拒绝不会严重影响到运算，我们假设符号改变的方块总是可观察到的方块。

除了这些符号的改变，简单的运算必须包含观察方格的改变。最新观察的方格必须立刻被机器识别。每个最新被观察的方格都在L方格中。与“立刻识别“联系，可以认为有其他种类的方格被立刻识别，尤其是那些被标记的方格。那些只被符号标记的方格如果只是一种有限的数字的话，我们就可以证明理论。通过将这些标记的

如果，另一方面，他们通过一个符号序列来标记，我们不能把识别的过程当做一个简单过程。这是一个基本点，应该被说明。在大多数数学论文中方程和定理被编号。通常这些数不超过1000。因此，通过它的编号一眼能认出这个定理。但是如果论文非常长，我们可能接触Theorem157767733443477；然后进一步在文件中，我们也许能找到更具体的定义。为了确定哪一个是相关定理，我们必须一个数字一个数字地比较，很可能把这些数字标记在铅笔上来确保没被数两次。如果尽管如此，仍有其他“立即识别”的方格，只要这些方格能通过我的机器的一些程序被找到，那我就不会失望。

因此简单的运行必须包括：

1. 符号的改变在观察过的方格上
2. 在L方格中，方格中一个的改变遵循先前观察过的方格中的另一个方格

也许这些改变中的一些必须卷入头脑状态的改变，因此最通用的单个操作必须被带到下面中的一个:

1. 一个可能的符号改变伴随一个可能的头脑状态的改变
2. 被观察过的一个方格的改变，伴随一个可能的头脑状态的改变

这个操作被执行是有限制的，由电脑的思想状态和观察的符号决定。特别的，当程序被操作后，他们决定计算的想法状态。

我们现在也许能造一个机器来做这个计算的工作。使计算机的每一个想法状态符合一个“m-配置”的机器。这个机器的扫描B方格与被计算机观察过的方格相符合。机器的每一个动作可以改变在扫描方格上的符号，或可以把扫描方格中的任意一个变成另一个远的方格，不仅仅是L方格中扫描方格的另一个。这个动作做完了，继续的配置，由扫描符号和m-配置决定。这个机器与能计算相同序列的计算机器差不多，这就是说数列通过计算机来计算。

我们把k的动作分开：成为节。第n节是为了找数列I中的第n个数，（n-1）后面以双冒号结束，而且后继的工作在双冒号右边的方格上完成。第一步是写下字母“A”,后面是公式（An）,然后是“B”,后面是（Bn）.机器k:然后开始工作，但是每当一个可证明的公式被找到，这个公式与（An）和（Bn）比较。如果与（An）相同，然后数字“1”被打印，第n节完成。如果是（Bn），“0”被打印，这一节也完成。如果它与两个都不同，然后k的工作继续从刚才断的点开始，迟早会到达公式（An）或（Bn）。因此第n节最终将被完成。

这是通过微积分函数所描述的计算机器。

我们假设计算在一个带上执行，但是我们避免介绍“想法状态”，而是通过想一个更自然地和它明确对应的部分。计算机先中断它的工作离开，再回来继续是可能的。如果这样做，它就必须留下指令（用一些标准形式写）来解释这项工作怎样继续进行。这个指令是“想法状态”的对应部分。我们认为计算机根据这样一个不连贯的习惯工作，它绝不是一次做超过一步的工作。指令的标注一定能使它执行完一步并且写下另一个注解。然而计算过程的状态在哪一阶段完成是由指令的标注和带上的符号决定的。即，系统的状态通过一个单一的表示（符号的序列）来描述，由带上的符号和指令的的注解组成，这种表示被称作 “状态公式”。我们知道在任意一个给定阶段上的状态公式由上一步的状态公式决定，我们假定这两个公式之间的关系在基础微积分中是可表示的，换句话说，我们假定有一个公理U,它表达了指导计算行为的准则，根据在任一阶段状态公式与在进行阶段的状态公式的联系，我们能构造一个机器来写下连续的状态公式，并且来计算需要的数字。

10一个大的可计算数组的例子

开始一个可计算函数的定义在整数变量和可计算变量上是有用的。有许多等价的方法定义一个可计算的整数变量的函数。最简单的如下，如果O是一个可计算数列，在其中0出现无限次，n是一个整数，然后我们定义W（0,n）为第n个和第（n+1）个0之间数字1的个数，然后Y（n）是可计算的，如果对所有n和一些0.，Y（n）= W（0,n）。

Y被称作一个可计算函数

我们不能定义普遍的实数变量的可计算函数，因为没有通用的描述一个实数的方法，但是我们可以定义一个可计算变量的可计算函数。

相似的定义关于几个变量的可计算函数能被给出及可计算数值的整数变量的函数。关于计算的法则。

1. 一个整数的可计算函数或可计算变量的可计算函数是可计算的。
2. 任意一个整数变量的根据可计算函数定义的循环函数是可计算的。
3. 如果Y（m,n）是两个整数变量的可计算函数，Y（n,n）是关于n的一个可计算函数
4. 如果Y（n）是一个可计算函数，它的值总是0或1，那么第n行为Y（n）的数列是可计算的。Dedekind的定理不包括前面的形式，如果我们用可计算的代替实数的，但它包括下面的形式:
5. 定理包括可计算的任一部分，有一个通用的过程来决定一个给定数属于哪个组。

因为Dedekind的定理的限制，我们不能说一个可计算的有界递增数列的可计算数有一个可计算的极限

1. 如果I,J都是可计算的，并且I<J,Y（I）<0<Y（J）,Y（i）是一个可计算的增长的连续函数，有一个单独的可计算数0，满足I<0<J,则Y（0）=0

可计算收敛

1. 一个幂指数系列，它的系数是可计算数在收敛区间内部的所有可计算点形成的一个可计算数列
2. 可计算的收敛数列的极限是可计算的。

并且有“一致可计算收敛”的明显定义

1. 有一致极限的可计算收敛与可算函数的可计算数列是可计算函数，因此
2. 一个幂系列，它的系数形成的可计算数列的和是在它收敛区间内部的一个可计算函数

11.Entscheidungsproblem的应用

一些定理的结果有重要应用，特别的，被用来说明Hilbert Entscheidungsproblem问题可能没有解。现在我将要限制自己来证明这个特别的定理。对这个问题的系统而确切的说明我向读者推荐 Hilbert 和 Ackermann合著的Grundzuge der theoretischen logik第三章。

因此，我建议，为了说明可能没有普通方法来决定一个给定的关于函数微积分Z的公式U是否是可证明的，也就是说，没有这样的应用这些公式中的任意一个U的机器，能够最后说明U是否可以被证明。

它也许被说我将证明的与大家都知道的Godel的结果相当不同，Godel已经说明有陈述U满足U或-U都不能证明。作为这个结果，它说明没有PRINCIPIA MATHEMATICA的一致性证明能被给出在那种形式之内。另一方面，我得说明没有通用的方法，它告诉一个给定的公式U在Z上是否能证明，与之相同的，由Z和-U联系起来作为额外的公式组成的系统是否是一致的。

如果对Godel的说明的否定被证明，对每一个U，U或-U是可证明的，我们将有Entscheidungsproblem的一个立即的结果。我们能发明一个机器K，它能连贯地证明所有可证明的公式，迟早K能到U或-U，如果到U，我们知道U是可证明的，如果到-U，既然Z是一致，我们知道U是不可证明的。

因为Z中整数的空缺，所以证明看起来有点长。下面列出的想法是相当简明的。

于每一个计算的机器M对应，我们构造一个公式Un（M）是否是可证明的，那么有一个通用的方法来确定M是否印出过0。

与I相应证明

1. 如果S1在带上出现，并且在一些M的完整配置中，那么Un（M）是可证明的。
2. 如果Un（M）是可证明的，那么S1出现在M的一些完整配置的带上。

当这个被做完，这个定理的暗示是平凡的。

LEMMA1：如果S1出现在M的一些完整配置的带上，那么Un（M）是可证明的。

我们必须展示怎样证明Un（M），让我们假设在第n个完整配置上，在带上的符号序列是Sr!n,0″,Sr!n,1″,……，Sr!n,n″,后面是空白，并且扫描符号是i（n）-th,m-配置是去qk!n.然后我们构造陈述缩写为CCn.前面的，F(u,u′)&F(u,u″)&……&F(U〖6-1〗,U【6】),简写为F【6】.

我们应该证明形如AM&F【5】\CCn的所有公式是可证明的。CFn的意思是“M的第n个完整配置是so and so”,这里so and so”代表M的第n个实际完整配置。因此，我们期望应CFn该是可证明的。

CF0当然是可证明的，因为在完整配置中，符号全是空白，m-配置是q1,并且扫描方格是u,CC0是（y）RS6(u,y)&I（u,u）&Kq7（u）.

A（M）\CC0是平凡的。

我们接着说明CFn\CFn+1对每一个n是可证明的，有三种情况要考虑：依照从n到n+1的数位配置，机器向左或向右移动或保持静止。我们假设第一种情况是成立的，也就是说，机器向左移动。相似的论点在其他情况也适用。如果r(n,i(n))=a, r(n+1,i(n+1))=c, k(i(n))=b, 同时 k(i(n+1))=d,那么Inst{*q*a *S*b *S*d *Lq*c}一定是Des（M）中的一项，也就是说

# Des(M) \ Inst{*q*a *S*b *S*d *Lq*c}.

# 因此*A*(M) & *F*[5+1] \ Inst{*q*a *S*b *S*d *Lq*c}& *F*[5+1] .

但是Inst{*q*a *S*b *S*d *Lq*c} & *F*[5+1] \ (*CC*n\*CC*n+1)是可证明的。因此有

*A*(M) & *F*[5+1] \ (*CC*n\*CC*n+1)

{262} 以及 *A*(M) & *F*[5] \ *CC*n) \ (*A*(M) & *F*[5+1] \ *CC*n+1)

也就是说*CF*n \ *CF*n+1.

*CF*n对于每个n都是可证明的。现在对这个论点的假设是S1以一种完整的数位配置出现在M标记的字符数列？？？？？中，也就是说，对于整数N和K，*CC*N有*R*S7(*u*[9], u[7])这一项。因此*CC*N\RS7(u[9],u[7])是可证明的。这样我们就有

CCN\ RS7(u[9], u[7])

以及A(M) & F[5] \ CC9

同时我们有(`u)A(M) \(`u)(`u') ... (`u[9Z]) A(M) & F[9],

其中N' = max (N, K). 因此

(`u)A(M) \(`u)(`u') … (`u[9Z]) *R*S7(u[9], u[7]),

(`u)A(M) \(`u)(`u[9])(`u[7]) (`u[9], u[7]),

(`u)A(M)\(`s)(`t)RS7(s,t),

因此，Un(M)是可证明的

这样我们完成了对定理1的证明。

定理2 如果Un(M)是可证明的，那么S1就会出现在M表示的完整数位配置的带中。

如果我们用任意建议的函数代替一个可证明的公式中的函数变量，就会得到一个真命题。特别的，如果我们用259-260页上列出的意义替换Un(M)，我们将会得到一个真命题“S1会出现在M表示的玩整数位配置的带中”。

现在我们将证明Entseheidungsproblem是不能解决的。我们将从反面证明。即存在普遍的（机械的）方法来决定Un(M)是否是可证明的。根据定理1 和定理2，这就意味着存在一个可以决定M是否能标记为0程序，但根据§8，这是不可能的。因此Entseheidungsproblem是不能解决的。

考虑到Entseheidungsproblem的大量特殊情况下的解 （263）以把所有可计算数放在前端的方式来表示Un(M)是很有趣的，即

(u)(`x)(w)(`u1) ... (`un)B, （Ⅰ）

其中B不包含可计算数，n=6。通过微小的变化我们可以得到一个具有Un(M)全部基本特性的公式，这个公式具有（Ⅰ）的形式，n=5。

附录

1936年8月28日增补

可计算性和有效可计算性

所有的有效可计算数列（V-definable）都是可计算的，这个定理将在以下概要中证明。假设Church和Kleene所使用的词“完整结构公式”（well-formed formula W.F.F.）和“转换”(conversion)已经被理解。在这些证明中的第二个中，一些公式的存在是假设的，没有被证明。这些公式可能是直接在诸如Kleene的“A theory of positive integers in formal logic”的帮助下建立的。

W.F.F.代表的整数能够记做Nn。我们应该说第n项是Y9（n）的数列O是V-definable或有效可计算的，如果1+Y9（u）是n的有效可计算函数，也就是说，如果存在W.F.F.M9，对于所有整数n，

{M9} (Nn) conv N&9!n"+1,

也就是说，{M9} (Nn)是可以转变成Vxy.x(x(y))或 Vxy.x(y)，如果V的第n项是1或0。

为了证明每个V-definable数列O都是可计算的，我们必须说明如何建造一个用来计算O机器。为了使机器使用方便在转化运算中需要微小的变化。这种变化存在于使用x, x', x", ...作为变量而不是使用a,b,c,...我们现在建造一个机器L，当提供给机器公式M9时，机器写下数列O。机器L的建造有些类似用来证明所有关于函数计算的可证明公式的机器K的建造。首先我们建造一个选择机器L1，提供给它一个W.F.F.，即M，适当地操作它，将会得到可由M转化而成的任何公式。然后L1可以被转化来产生一个自动机器L2，L2成功包含了所有的可有M转化而成的公式{264}(cf- foot-note p.252)。机器L 包括L2。当提供给公式M9时，机器L的运动分为多步，其中第n步是为了找到数列O的第n项。第n步的第一阶段是{M9} (Nn).的形成。然后这个公式被提供给机器L2，机器L2把它转化为各种各样的其他公式。每个转化出的公式最终都会以如下形式出现

Vx[V'x[{x}({*x*}(x'))]] 即N2

以及Vx[Vx'[{x}(x')]] 即N1

如果结果与第一个相同，那么机器就印出1，第n步就完成了。如果结果与第二个相同，那么机器就印出0，这一步就完成了。如果结果和两个都不同，那么L2的结果就会被送回。通过假设，{M9} (Nn)可以转变为公式N2或N1中的一个；从而第n步最终被完成了，也就是说，数列O的第n项最终被写下。

为了证明每个可计算数列都是V-definable，我们必须说明如何用公式M9，对于所有整数n,

{*M*9} (*N*n) conv *N*1+&9!n".

假设M是一个可以计算数列O的机器，让我们用数字方法对M的完整数位配置做一下形容。例如，我们可以取§6中形容的完整数位配置中的D.N，将第n个完整数位配置中的D.N记做W(n)。机器M的桌面将会给出W(n+1)和W(n)的关系如下

W(n + 1) = p9(W(n))

其中p9是一个严格限定的函数，尽管不是非常简单的形式。M的桌面决定了p9是V-definable，也就是说，存在W.F.F. A9 使得 对于所有整数n，

{A9} (Nw!n") conv Nw!n+1".

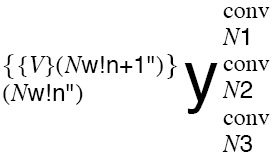
使U9代表

Vu[~{u}(A9)x(Nr)]

其中 r = W(0); 那么，对于所有整数 *n*,

{U9} (*N*n) conv *N*w!n".

{265} 可以证明存在公式V使得

如果由第n到第n+1个完整配置转变，数字0被印出。

否则，如果数字1被印出

使 W% 表示

Vu[~{V}({A9}({*U*9}(*u*)))x({*U*9}(u))]

这样，对于每个整数

~{{V}(Nw!n+1")x (w!n") conv {W9} (Nn),

使Q代表一个公式使得

~{Q}(W9)x(Ns) conv Nr!z"

其中*r*(*s*)是第s个整数q，对于q，(W9) (Nn) 可以转化为N1或N2。那么，如果M9代表

Vw {W9} {Q}(W9) (w)

它就将拥有所要求的特性。[16]

毕业学校

普林斯顿大学

新泽西州

U.S.A.

{544}

对论文的修正

在一篇名为*On computable numbers, with an application to the*

*Entseheidungsproblem* [17]的论文中，作者给出了对the

Entseheidungsproblem of the “engere Funktionenkalkül”.[18]不可解决的证明。这个证明有一些错误，接下来将修正这些错误，这篇论文中还有一些其他应该改正的陈述，尽管它们并不是错误的。

260页的表达Inst{qi Sj Sk Lqo}应该读作

(x,y,x',y') ~(RSg(x,y) & I(x,y) & Kqf(x) & F(x,x') & F(y',y))

\ (I(x',y') & RSh(x',y) & Kqo(x') & F(y',z) v

[(RS6(x,z)\(RS6(x',z))

& (RS7(x,z)\(RS7(x',z)) & ... &

(RSy(x',z))])},

S0, S1, …, SM是M可以打印出的符号。

261页33行的叙述“Inst{qa Sb Sd Lqc} & F[5+1] \ (CCn \ CCn+1)是可证明的”是错误的（即使使用对Inst {qa Sb Sd Lqc}新的表述）我们不可能推论出诸如F[5+1] \(–F(u,u"))的命题，因此我们永远不能在in Inst {qa Sb Sd Lqc}中使用词组

F (y',z) v [(RS6(x,z) \RS6(x',z)) & … & (RSy(x,z) \RSy(x',z))]{545}. 为了修正这个错误我们将介绍一个新的函数变量G [G(x,y)来描述“x precedes y”.]。那么，如果Q是

(x)(`w)(y,z) F(x,w) & F(x,y)\G(x,y) & F(x,z) & G(z,y)\G(x,y)

& [G(z,x) v (G(x,y) & F(y,z)) & (F(x,y) v F(z,y)) \ (–F(x,z))]}

的缩写

修正后的公式Un(m)将成为

(`u)A(M)\(`*s*)(`t)RS7(*s*,*t*),

其中A(M)是*Q* & (y)RS6(u,y) & I(u,u) & Kq7(u) & Des (M)的缩写

那么261页33行的叙述将被读作

Inst{qa Sb Sd Lqc} & Q & F[5+1] \ (CCn\ CCn+1)

29行将被读作

r(n,i(n))=b, r(n+1,i(n))=d, k(n)=a, k(n+1)=c.

对于260页15行的“logical sum”。在这种转变后此证明是正确的。Un(m)就能够写成（Ⅰ）的形式，其中n=4。

当用特殊的方法定义可计算数时一些困难随之出现（233页）。如果可计算数要满足直观的要求我们需要：

我们是否能够给出一个将每个整数和两个有理数an,bn联系起来的法则，且满足an≦an-1<bn+1≦bn,bn-an<2-n，从而存在一个可计算数使对于每个n，an≦bn.

（A）

可以根据一般的数学标准给出对此的有效证明，但是除了对定理的实际应用。也就是说下面的说法是错误的：

给出（A）中数列an,bn的构成，存在一个法则，凭借它可以找到一个D.N.使机器计算出1。

（B）

（B）是错误的，如果我们为了方便，m/2形式的十进制数字都将会以0终结。假设N是某台机器，给cn如下定义：cn=！-2-4-3 如果M在第n步完整数位配置时没有印出数字0，那么cn =!-2-4-3。如过0在第m步完整数位配置时被先印出（m小于等于n）使an=cn—2-5-2，bn=cn+2-5-2.那么不等式（A）就满足了，并且如果N印出0那么I的第一个数字是0，否则印出是1。如果（B）是正确的我们就有一种方法来在给出D.N.中的N的情况下找出数字1；也就是说，我们能够判定N是否能印出0，与论文§8引用的结果相反。这样尽管(A)说明一定存在能够计算Euler常数的机器（举例），我们现在也不能描述出任何这样的机器，因为我们还不知道Euler常数是不是m/2\*的形式。

通过改变可计算数与可计算数列联系的方法，而可计算数的和扔保持左不可改变，上面这种不符合的情况就可以被避免。有多种方法可以解决，[19]就是一个例子。假设一个可计算数列O的第一个数字是i，然后是n个重复的1，然后是0，最后数列的第r个数是cr。那么数列O就对应于下面的实数：

(2i-1)n+∑(2cr-1)(2/3)r

如果计算O的机器被认为就是在计算这个常数，那么(B)就成立。这种用数字构成的数列重新表述实数的独特性现在已经丢失了，但是这种方法在理论上并不十分重要，因为D.N.’s在任何情况下都不是独特的。

毕业学校

普林斯顿大学

美国

尾注

1-5相关参考论文

6.如果我们认为一个符号是被逐字印出在一个正方形上，我们可以认为这个正方形是0≤x≤1,0≤y≤1.这个符号被定义为这个正方形内的一个点集。这个点集被打印机的墨水印出。如果这些集合被限制是可计算的，我们可以定义两个符号间的“距离”为把一个符号变成另一个的代价，如果移动单位面积的打印机的墨水单位距离的代价是一体的，并且当x=2,y=0时有无限的墨水供应。根据这种方法，这些符号组成了一个有条件的紧密结合的空间。

7.“函数微积分”的表达是指严格的希尔伯特函数微积分。

8.建造一个选择机器来做这件事情的最自然的。但是之后建造所需的自动机器就很简单。我们可以认为选择永远是在两种可能性0和1之间的选择。那么每个证明都将被以一系列的i1, i2, …, in (i1 = 0 or 1, i2 = 0 or 1, …, in = 0 or 1)决定，因此2n+i1,2n+1+i2,2n+2+in,完全决定了证明。这个自动机器成功实施了证明1，证明2，证明3……

9.作者已经发现了这样的机器的描述。

10.否定标志是写在表达式前面的，而且不超过它。

11.质数数列记成(r)

12.如果计算M，那么γ是否能无限地印出0的问题经常是和M是否是不可循环的是等价的。

13.一个函数αn可以被多种方法定义，以处理可计算数。

14.尽管不可能找到一个决定某个给定数字是否是满足的普遍方法，说明某个确定等级的数字是满足的却是可能的。

15.Loc,cit

16.在完整的V-definability的可计算数列的证明中，最好是改变这种方法，即用一种能够被我们的机器更容易处理的表达来代替原本的对完整数位配置的数字表达。让我们选择特定的整数来代表机器的符号和m-数位配置。假设在某个特定的完整数位配置中，代表带上的连续的符号的数字是s1,s2,…,sn,并且m-数位配置有数字t，纳闷我们就能用如下公式表达这个完整数位配置：

[[NS1,NS2,…,NSM-1],[Nt,Nm],[Nsm+1,…,Nsm]]

其中[a,b] 代表λu[{{u}(a)}(b)],

[a,b,c] 代表λ u[{{{u}(a)}(b)}(c)].

17.

18.

19.重叠区间在实数定义中的应用最初是由Brouwer指出的。