图灵机在康托集上的可计算性

张汉昭

（南京大学 数学系，南京 210093）

摘　要：对定义在集合Σ上的函数$f:\subseteq(\Sigma^*)^\omega\to\Sigma^*$引入可计算性，该方法不能应用于不可数集合$M$，如实数集$\mathbf{R}$。利用无限符号序列作为名，同时定义作用于这些无限序列函数的可计算性，将可计算性拓展到以上不可数集合，从而引入型2图灵机。部分可递归函数（即可计算数函数）集合$P^{(1)}$中的概念“有效的哥德尔数”$\varphi:N\to P^{(1)}$是一般递归理论的基础，而φ由通用图灵机定理等价唯一定义。利用Σ^ω将φ的概念推广到可计算函数$f:\subseteq\Sigma^\omega\to\Sigma^\omega$和某些连续函数$f:\subseteq\Sigma^\omega\to\Sigma^\omega$，其中$a,b\in\{\omega,*\}$，这就是型2图灵机的表示。

关键词：型2图灵机；标准表示；可计算性；名

中图分类号：TP18　　**文献标识码**：A　　**文章编号**：1673-9833(2008)05-0018-05

Computability of Turing Machine on Cantor Sets

Zhang Hanzhao

（Department of Mathematics, Nanjing University, Nanjing 210093, China）

Abstract: In view of computability for functions $f:\subseteq(\Sigma^*)^\omega\to\Sigma^*$ on the set Σ, this method cannot be applied for introducing computability on uncountable sets M like the set $\mathbf{R}$ of real numbers. The above concepts will be extended to uncountable set by using infinite sequences of symbols of names and by defining computability for functions which transform such infinite sequences with Type-2 machine. In recursion theory the concept of an "effective Gödel numbering" $\varphi:N\to P^{(1)}$ is fundamental. This number φ is defined uniquely up to equivalence by the universal turning machine theorem. We generalize the number φ to notations of the computable functions $f:\subseteq\Sigma^\omega\to\Sigma^\omega$ by means of Σ^ω and the representations of certain continuous functions $f:\subseteq\Sigma^\omega\to\Sigma^\omega$, where $a,b\in\{\omega,*\}$. This is the presentation and notation of Type-2 machine.

Key words: Type-2 machine; standard presentation; computability; name

1　背景知识

在数学上，递归函数和λ可定义函数均等价于图灵机定义的可计算函数。图灵机能表示算法、程序和符号行的变换，因而可作为电子计算机的数学模型，也可用作控制算法的数学模型，在形式语言理论中还可用来研究短语结构语言，即递归可数语言。

对图灵机的研究主要集中在两个方面：第一，研究图灵机所定义的语言类，该语言类称为递归可枚举集合；第二，研究图灵机所计算的函数类，该函数类称为部分递归函数。

作为有效过程或算法的形式模型，图灵机的每个动作过程都应该是有穷可描述的。其次，每个过程应由离散的步骤组成，每一步都能够机械地实现。图灵机有非确定型、多维型、多带多头型等模型，它们在计算能力上是等价的，是图灵机基本模型的变种。

一般情况下，研究者只是明确地为定义在集合Σ^*上的函数$f:\subseteq(\Sigma^*)^\omega\to\Sigma^*$引入可计算性，其中$\Sigma^*$是由$\Sigma$生成的，而$\Sigma$由任意的有限字母表中的有限字母构成。对于计算其他集合M（如自然数集、有理数集）上的

收稿日期：2008-08-21

作者简介：张汉昭（1986-），男，湖南株洲人，南京大学数学系学生，主要研究方向为计算机逻辑和人工智能.

函数，字母被视为 M 中基本元素的“名”。因集合$\sum^*$中属于有限字母表的字母仅仅是可数的，所以这种方法不能应用于不可数集合 M，如实数集 **R**、**R** 上开区间，或在[0,1]上的实连续函数 $C[0,1]$上引入可计算性。

利用无限符号序列作为名，同时定义作用于这些无限序列函数的可计算性，将拓展以上的概念。由有限字母表(假设其中至少有两个元素)$\sum$中元素构成的集合$\sum^\omega$与实数集(连续势)有相同的势[1]，因此，它能作为任何一个最多连续势的的名，如惯用的以十进制小数表示实数、π用 3.141…表示。

以下约定$\sum$是一个固定且有限的字母表，且包含元素 0 和 1，并假设$\sum$在一些例子中足够大。

2　型 2 图灵机和可计算字符串函数

对于$k \geq 0, Y_0, Y_1, \cdots, Y_k \in \{\sum^*, \sum^\omega\}$,用带有 k 条单向输入纸带、有限条工作纸带和一条单向输出纸带的图灵机定义可计算函数，可得如图 1 所示图灵机。

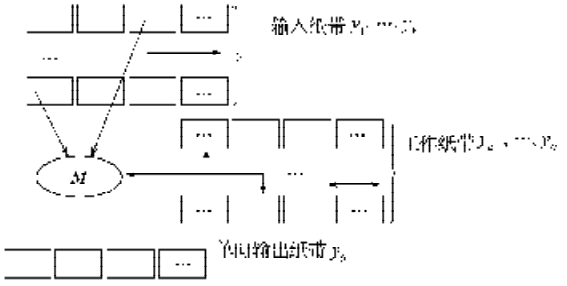

图 1　利用图灵机计算 $f_M(y_1, \cdots, y_k)$

Fig. 1　Computing $f_M(y_1, \cdots, y_k)$ by turing machine

定义 1　与一般的图灵机不同，型 2 图灵机允许无限输入和输出，它有 k 条输入纸带和一个类型规格$(Y_1, \cdots, Y_k, Y_0)$，其中$Y_i \in \{\sum^*, \sum^\omega\}$，它们给出输入 / 输出纸带的类型。定义字符串函数$f_M: \subseteq Y_1 \times \cdots \times Y_k$由型 2 图灵机计算。

定义 2　对于输入$(y_1, \cdots, y_k) \in Y_1 \times \cdots \times Y_k$来说，初始的纸带配置如下：对于每条输入纸带 i，序列 $y_i \in Y_i$(有限或无限)立刻被置于纸带右边开始处，这些纸带的其它格子写入符号 B（其中 B 不包含在输入输出符号集$\sum$中）；其它无输入纸带的所有格子都包含符号 B。对于所有的 $y_0 \in Y_0$，$y_1 \in Y_1$，…，$y_k \in Y_k$，定义：

1）若$Y_0 = \sum^*$，$f_M(y_1, \cdots, y_k) = y_0 \in \sum^*$，当且仅当 M 在输入$(y_1, \cdots, y_k)$后在输出纸带上输出 y_0；

2）若$Y_0 = \sum^\omega$，$f_M(y_1, \cdots, y_k) = y_0 \in \sum^\omega$，当且仅当 M 在输入$(y_1, \cdots, y_k)$永远地计算，最后在输出纸带上输出 y_0。

一个字符串函数$f_M: \subseteq Y_1 \times \cdots \times Y_k$被认为是可数的，当且仅当它是由型 2 图灵机 M 计算的。

如果机器永远计算下去，但只在输出纸带上写入有限的符号，这样的函数$y_0 = f_M(y_1, \cdots, y_k)$是没有定义的，定义的语言中不利用这部分结果。因为型 2 图灵机与图灵机一样现实和强有力。显然实际中不存在无限的输入和输出，无限的计算也无法完成。但在物理设备上，只要时间足够长且内存允许，能实现输入的有限初始部分进行有限的计算，从而产生有限的初始部分输出。型2图灵机现实性的疑问，将在图 2 中解释。

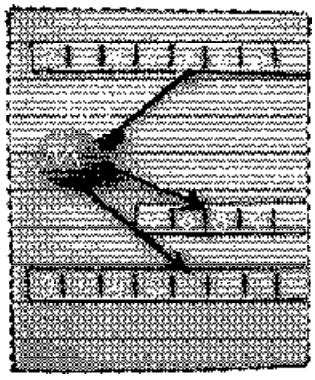

图 2　型 2 图灵机的实现

Fig. 2　A realization of Type-2 machine

电子计算机能模仿型 2 图灵机，因此，能用有限的物理计算来近似型 2 图灵机的无限计算，并达到任意精度。单向输出纸带的限制是有限初始的任何部分输出结果将来都不能修改，是最后结果。因此，双向输出纸带的计算方式不会有太大用处。

因具体设计一图灵机很麻烦，所以仅提出算法。如只考虑可计算性，则更倾向于算法语言而不是图灵机语言，只要这些语言至少在理论上能被翻译成图灵机语言。可计算字符串函数的定义包含标准的可计算字符函数的定义（选择$Y_0 = Y_1 = \cdots = Y_n = \sum^*$）。

定义 3　1）任何元素$\omega \in \sum^*$是可计算的；

2）序列$\rho \in \sum^\omega$是可计算的，当且仅当常值函数$f: \{()\} \to \sum^\omega$是可计算的。

3）元组$(y_1, \cdots, y_k)$，其中$y_1 = \sum^*$或$y_1 = \sum^\omega$是可计算的，当且仅当每个组成元素 y_1 是可计算的。

显然，一个序列$\rho \in \sum^\omega$是可计算的，当且仅当$\rho = f(\langle i \rangle), f: \sum^* \to \sum^\omega$是可计算函数。

例 1　常值函数$f: Y_1 \times \cdots \times Y_k \to Y_0$是可计算的，当且仅当其函数值$c \in Y_0$是可计算的。据可计算函数的定义，它是由型 2 图灵机计算的，而型 2 图灵机输出值的类型是$\{\sum^*, \sum^\omega\}$，所以结论成立。

例 2　定义$f: \subseteq \sum^\omega \to \sum^*, f(\rho) = \begin{cases} 1, & \text{如果 } \rho \neq 0; \\ div, & \text{其它.} \end{cases}$

有一个型 2 图灵机 M 读取输入$a_0, a_1, \cdots \in \sum^\omega$，当存在某一 i，使得 $a_i \neq 0$ 时输出 1 停止，那么 M 计算 f。

例 3　型 2 图灵机不能计算 3 乘无限小数。假设某一型 2 图灵机 M 计算实函数$x \to 3 \times x$，那么它必须正确地计算 0.333…(即 1/3)，输出结果必须是 0.99…或者 1.000…，假设$\{0, 1, \cdots, 9, .\} \subseteq \sum$。证明这样的型 2 图灵机

不存在。

先考虑第一种情况，假定经过 k 步后，M 写出前缀输出结果的“0.”。此时机器从输入纸带上读取了有限的数字 $0.\omega$。现在考虑输入序列 $0.\omega 999\cdots$。它是一个大于 1/3 的实数。在此情况下，机器 M 将同样输出以“0.”开头的结果。但因 $3\times x>1$，M 并没有得出正确的结果(矛盾)。第二种情况可类似分析。因此，这样的机器不存在。在实数可计算性概念下，加法和乘法都不是可计算函数，因此实数可计算的概念并不十分有用，也不能被数值计算研究者和计算机科学研究者接受。以上讨论用的是单向输出纸带。事实上，有双向输出纸带的机器能计算无限小数的乘法，但这样的机器在实际中没有用处，因无限计算的有限初始部分一般不能给出一个可靠结果。

3 连续字符串函数集合的标准表示

部分可递归函数（即可计算数函数）集合 $P^{(1)}$ 中的概念“有效的哥德尔数”或者“允许的哥德尔数”$\varphi:N\to P^{(1)}$ 是一般递归理论的基础[1]。而 φ 由通用图灵机定理等价地唯一定义。本文将 φ 的概念推广到可计算函数 $f:\subseteq\Sigma^{\omega}\to\Sigma^{\omega}$ 和某些连续函数，其中 $a,b\in\{*,\omega\}$，将 Σ 的方法应用于前者，而将 Σ^{ω} 的方法应用于后者。

定义 4 1）集合 M 的表示（notation）是一个满射函数 $\gamma:\subseteq\Sigma^{\omega}\to M$；

2）集合 M 的表示（representation）是一个满射函数 $\delta:\subseteq\Sigma^{\omega}\to M$。一个命名系统就是一个表示（notation 或者 representation），有时将 $\gamma(\omega)$ 简写为 γ_{ω}，将 $\delta(p)$ 简写为 δ_p。有时说 $p\in Y$ 是 $x\in M$ 的 γ 名，当 $\gamma:Y\to M$ 是一个命名系统且 $\gamma(p)=x$。没用有限或无限的序列作为名来考虑函数 $\gamma:\Sigma^{\omega}\times\Sigma^{\omega}\to M$。

定义 5 （可计算函数的标准表示）考虑只有一条纸带且输入为 $\omega\in\Sigma$ 的型 2 图灵机，使得码字集合 TC 是递归的。对于所有的 $a,b\in\{*,\omega\}$，定义：

1）$P^{ab}:\{f:\subseteq\Sigma^{a}\to\Sigma^{b},f\text{是可计算的}\}$。

2）集合 P^{ab} 中表示（notation）$\xi^{ab}:\Sigma^{*}\to\mathbf{P}^{ab}$ 定义如下：$\xi^{ab}(\omega)\begin{cases}\text{函数}f\text{无定义，若}\omega\notin TC;\\ \text{由型 2 图灵机计算并且}\\ \quad\text{编码是}\omega\text{的函数},f\in P^{ab},\text{其它}.\end{cases}$

定义 6 （连续函数的类）

$F^{**}:\{f\mid f:\subseteq\Sigma^{*}\to\Sigma^{*}\}$；$F^{*\omega}:\{f\mid f:\subseteq\Sigma^{*}\to\Sigma^{\omega}\}$；

$F^{\omega*}:\{f\mid f:\subseteq\Sigma^{\omega}\to\Sigma^{*}\text{是连续的且}dom(f)\text{是开集}\}$；

$F^{\omega\omega}:\{f\mid f:\subseteq\Sigma^{\omega}\to\Sigma^{\omega}\text{是连续的并且}dom(f)\text{是}G_{\delta}\text{集}\}$。

定理 1 （连续扩充）1）每个连续的部分函数 $f:\subseteq\Sigma^{\omega}\to\Sigma^{*}$ 在 $F^{\omega*}$ 上有一个扩充；2）每个连续的部分函数 $f:\subseteq\Sigma^{\omega}\to\Sigma^{\omega}$ 在 $F^{\omega\omega}$ 上有一个扩充。

定义 7 先定义“卷函数”：$\iota:\Sigma^{*}\to\Sigma^{*}$，$\iota(a_1a_2\ldots a_n):=110a_10a_2\ldots a_n011$。其中 $n\in\mathbf{N},a_1\ldots a_n\in\Sigma$。

$\forall x,x_3,x_1,\cdots\in\Sigma^{*},p,p_2,p_1,\cdots\in\Sigma^{\omega},i,j,k\in\mathbf{N},k\geq 1$，定义元组函数$<x_1,\cdots,x_k>:=\iota(x_1)\cdots\iota(x_k)\in\Sigma^{*}$，

$<x,p>:\iota(x)p\in\Sigma^{\omega}$，$<p,x>:\iota(x)p\in\Sigma^{\omega}$，

$<p_2,\cdots,p_k>=p_1(0)\cdots p_k(0)p_1(1)\cdots p_k(1)\cdots\in\Sigma^{\omega}$，

$<x_0,x_1,\cdots>:=\iota(x_0)\iota(x_1)\cdots\in\Sigma^{\omega}$，

$<p_0,p_1,\cdots><i,j>=p_i(j)$ $(<p_0,p_1,\cdots>\in\Sigma^{\omega})$。

对以上每个元组函数，使用同样的符号<>。如果变元属于 Σ^{ω}，那么这些函数是单射甚至双射的。“卷函数”ι 的定义保证了 ω 的子字 $\iota(u)\iota(v)$ 只能平凡地重叠，即如果 $\iota(u)$ 是 $\iota(v)$ 的子字，那么 $\iota(u)=\iota(v)$，同样地，如果 x 是 $\iota(u)$ 的后缀和 $\iota(v)$ 的前缀，那么 $\iota(u)=\iota(v)=x$ 或者 $x\in\{\lambda,11,\lambda\}$。

定义 8 对所有的 $a,b\in\{*,\omega\}$，定义 F^{ab} 的标准表示 $\eta^{ab}:\Sigma^{\omega}\to F^{ab}$，即 $\eta^{ab}(<x,p>)(y):=\xi^{ab}(x)<p,y>$；

对所有的 $x\in\Sigma^{*}$，$p\in\Sigma^{\omega}$，$y\in\Sigma^{\omega}$，如果对任意的 $x\in\Sigma^{*}$，$\iota(x)$ 不是 q 的前缀，$\eta^{ab}(q)$ 没有定义，简记 $\eta^{ab}(q)$ 为 η^{ab}_{q}。

根据定义 8，$\eta^{ab}_{<e,x>}(y)$ 是型 2 图灵机 M，它的哥德尔数或编码为 x，该式表示该机器应用于输入 $<p,y>$ 得到的结果。可认为 M 是一个问答器，对于任意给定的 $p\in\Sigma^{\omega}$ 计算连续函数 $\eta^{ab}_{<e,x>}$，有时称 q 为函数 η^{ab}_{q} 的程序。

引理 1[1] 对所有的 $a,b\in\{*,\omega\}$，η^{ab} 是 F^{ab} 的表示。

证明 由定义 8，$p\in\Sigma^{\omega}$ 是 $Q\in\Sigma^{\omega}$ 的一个列表，当且仅当 $Q=\{x\in\Sigma^{*}\mid\iota(x)<p\}$。

1）若 $a=b=*$，假定有一台型 2 图灵机，它有一条输入 $<p,y>$ 的纸带，其中 $p\in\Sigma^{\omega},y\in\Sigma^{*}$。其工作方式为：型 2 图灵机 M 搜寻 p 的子字 $\iota(v,z)$，其中 $z\in\Sigma^{*}$，若找到这样一个子字，那么 M 输出 z 并停止；否则函数无定义。考虑 $f\in F^{**}$，令 p 为序列 $\{<y,z>\mid f(y)=z\}$，则对所有的 $y\in\Sigma^{*},f_{\omega}<p,y>=f(y)$，令 x 为型 2 图灵机的编码 $f_{M}=\xi^{\omega*}_{e}$，则对所有的 $y\in\Sigma^{*},\eta^{**}_{<e,x>}(y)=\xi^{\omega*}_{x}<p,y>=f_{M}<p,y>=f(y)$，所以 $\eta^{**}_{<e,x>}=f$。因此 $F^{**}=range(\eta^{**})$。

2）若 $a=*$，$b=\omega$ 对所有的 $q\in\Sigma^{\omega},\eta^{*\omega}(q)\in F^{*\omega}$，假定有一台型 2 图灵机，它有一条输入 $<p,y>$ 的纸带，其中 $p\in\Sigma^{\omega},y\in\Sigma^{*}$，它在第 n（$n=0,1,\cdots$）步的工作方式为：在第 n 步前，在输出纸带上输出 z_n，在第 n 步，型 2 图灵机 M 搜寻 p 的子字 $\iota(y,z)$，其中 $z\in\Sigma^{*},z_n\subseteq z$，如果找到了这样一个子字（否则在第 n 步永久停留而不再改写），那么，在输出纸带上，它将 z_n 拓展为 z，从

而完成第n步。考虑$f\subset F^{\Sigma^*\to\Sigma^*}$，令$p$为序列$<y,z>$，其中$y\in dom(f), z\subseteq f(y)$，则对于所有的$y\in\Sigma^*, f_M<p,y>=f(y)$，从而得到$\eta^{\omega\omega}_{<x,p>}=f$，其中$x$是型2图灵机$M$的编码。因此$F^{\omega\to\omega}=range(\eta^{\omega\omega})$。

3）若$a=\omega, b=*$，对于固定的$x\in\Sigma^*$，$p\in\Sigma^{\omega}$，$\eta^{*\omega}_{<x,p>}(q)=\xi^{\omega^*}_x<p,q>=\xi^{\omega^*}_x<p,q>\cdot g(q)$，其中$g(q):=<p,q>$。因为$g\in F^{\omega\omega}, dom(g)\in\Sigma^{\omega}, \eta^{\omega^*}_{<x,p>}\in F^{\omega^*}$，所以$\xi^{\omega^*}_x\in F^{\omega^*}$。因此$range(\eta^{\omega^*})\subseteq F^{\omega^*}$。若找到了这样一个子字，那么$M$输出$z$并停止；否则函数无定义。

考虑$f\subset F^{\omega^*}$，那么对于某一前缀无限制的函数$h:\subseteq\Sigma^*\to\Sigma^*$，$f=h_*$，且存在$\{<y,z>|h(y)=z\}$的一个序列$p\in\Sigma^{\omega}$。则对任何$q\in\Sigma^{\omega}$，$f_M<p,q>=h_*(q)$，如果$x$是型2图灵机$M$的编码，假定有一台型2图灵机，它有一条输入$<p,y>$的纸带，其中$p,q\in\Sigma^{\omega}$，则图灵机工作方式为：型2图灵机$M$搜寻$p$的子字$t(y,z)$，其中$y,z\in\Sigma^*$且$y\subseteq q$。则对于任何$q\in\Sigma^{\omega}$，$\eta^{\omega^*}_{<x,p>}(q)=\xi^{\omega^*}_x<p,q>=f_M<p,q>=h_*(q)=f(q)$，所以$\eta^{\omega^*}_{<x,p>}=f$。因此$F^{\omega^*}=range(\eta^{\omega^*})$。

4）若$a=b=\omega$，仿照情况3）证明$range(\eta^{\omega\omega})\subseteq F^{\omega\omega}$性质。假定有一台型2图灵机，它有一条输入$<p,y>$的纸带，其中$p,q\in\Sigma^{\omega}$，它在第$n(n=0,1,\cdots)$步的工作方式为：第$n$步前，在输出纸带上输出$z_n$，在第$n$步，型2图灵机$M$搜寻$p$的子字$t(y,z)$，其中$y,z\in\Sigma^*, y\subseteq q$，$z_n\subseteq z$，如果找到了这样的子字，那么它将$z_n$拓展为$z$，从而完成第$n$步。现在，仿照情况3），可证明$F^{\omega\omega}=range(\eta^{\omega\omega})$。

引理2　函数$f:\subseteq\Sigma^{\omega}\to\Sigma^{\delta}$是可计算的，当且仅当存在某一可计算的$p\in\Sigma^{\omega}$，使得$f=\eta^{\omega\delta}_p$。[1]

证明　如果f是可计算的，那么存在编码x的图灵机，使得$f(y)=\xi^{\omega\delta}_x<p,y>$，那么令$p=<x,0^{\omega}>$；如果存在某一可计算的$p\in\Sigma^{\omega}$，使得$f=\eta^{\omega\delta}_{<x,p>}$，那么$f(y)=\xi^{\omega\delta}_x<p,y>$，因此$f$是可计算的。

定义9　令$a,b,c\in\{0,\omega\}$，G^{ab}是函数$g:\subseteq\Sigma^{a}\to\Sigma^{\delta}$的集合，而$\zeta:\subseteq\Sigma^{c}\to G^{ab}$是$G^{ab}$的命名系统。定义通用的$utm-$和$smn-$性质如下：

$utm(\zeta)$：对所有的$x\in dom(\zeta)$和$y\in\Sigma^{a}$，存在一个（“通用”）可计算函数$u:\subseteq\Sigma^{c}\times\Sigma^{a}\to\Sigma^{b}$，其中$\zeta_x(y)=u(x,y)$。

$smn(\zeta)$：对所有的可计算函数$f:\subseteq\Sigma^{c}\times\Sigma^{a}\to\Sigma^{\delta}$，存在一个完全可计算函数$s:\Sigma^{c}\to\Sigma^{c}$，使得对所有的$x\in\Sigma^{c}, s(x)\in dom(\zeta)$，并且对所有的$x\in\Sigma^{c}$和$y\in\Sigma^{a}$，$f(x,y)=\zeta_{s(x)}(y)$成立。

定理2　η^{ab}的$utm-$和$smn-$定理：对任意$a,b\in\{0,\omega\}$，有$utm(\eta^{ab})$和$smn(\eta^{ab})$。

证明　令$v:\subseteq\Sigma^{c}\times\Sigma^{a}\to\Sigma^{c}$为$\xi^{\omega b}$中的一个可计算通用函数。定义$u\subset\Sigma^{\omega}\times\Sigma^{a}\to\Sigma^{b}$，即$u(<x,p>,y):=v(x,<p,y>)$，对所有$x\in\Sigma^*, p\in\Sigma^{\omega}, y\in\Sigma^{a}, u(q,y)=div$。

如果$\forall x\in\Sigma^*, t(x)$是$q$的前缀，那么函数$u$是可计算的。因为$\eta^{ab}_{<x,p>}(y)=\xi^{\omega}_{kx}<p,y>=v(x,<p,y>)=u(<x,p>,y)$，所以$\eta^{ab}$有$utm-$性质。

令$g:\subseteq\Sigma^{\omega}\times\Sigma^{a}\to\Sigma^{\delta}$是可计算的，则存在某一$x\subset TC$对所有的$p\in\Sigma^{\omega}, y\in\Sigma^{a}$。定义$s:\Sigma^{\omega}\to\Sigma^{\omega}$，其中$s(p):<x,p>$，则$s$是可计算的，且对所有的$p\in\Sigma^{\omega}, y\in\Sigma^{a}$，$\eta^{ab}_{s(p)}(y)=\eta^{ab}_{<x,p>}(y)=g(p,y)$，因此$\eta^{ab}$有$smn-$性质。

4　应用举例：改进的单向多带图灵机

对一般的单向多带图灵机进行改进，可用来计算任意有限多位小数的乘法。改进的单向多带图灵机如下：

1)此图灵机有两条输入纸带，一条工作纸带及一条输出纸带；

2)输入纸带首格及数据左端的格子均写入字母H，数据表示为小数形式（如3表示为3.0）；

3)字母表Σ：{0, 1, ⋯, 9, •, H}；初始状态s:start；停机状态H:halt；控制规则σ具体见表1。

表1　改进图灵机的控制规则

Table 1　The controlling instructions of improved Turing machine

输入					响应			
当前状态	纸带1当前符号	纸带2当前符号	纸带3当前符号	纸带4当前符号	纸带3新符号	纸带4新符号	读写头移动	新状态
start	H	H	any	any			left	
start	0,⋯,9	0,⋯,9	any	any	2		left	
start	0,⋯,9	•	any	any	1		left	$1p_2$
start	•	0,⋯,9	any	any	1		left	$1p_1$
start	•	•	any	any	H		right	2p
$1p_1$	any	0,⋯,9	any	any	1		left	
$1p_1$	any	•	any	any	H		right	2p
$1p_1$	0,⋯9	any	any	any	1		left	
$1p_1$	•	any	any	any	H		right	2p
2p	0,⋯9,	0,⋯9,	any	any			right	
2p	•	•	any	any			left	point
	H	H						
point	any	any	2	any	H		left	plus1
point	any	any	H	any			left	POINT
point	any	any	1	any			right	
plus1	any	any	1	any			left	
plus1	any	any	H	any	1		left	$plus1_1$
$plus1_1$	any	any	any	any	H		right	point

续表

输入					响应			
当前状态	纸带1当前符号	纸带2当前符号	纸带3当前符号	纸带4当前符号	纸带3新符号	纸带4新符号	读写头移动	新状态
POINT	any	any	1,2	any			left	
POINT	any	any	H	any			left	printp
printp	any	any	any	any		●	right	head
head	0,…9,	0,…9,	any	any		0	right	
head	●	●	any	any			left	multi
	H	H						
multi	any	●,0	any	any			left	
multi	any	1	any	any				$multi_1$
…	…	…	…	…				…
multi	any	9	any	any				$multi_9$
multi	any	H	any	any				halt
$multi_1$	1	any	0,1,H	any	●			m_1p_1
$multi_1$	1	any	9	any				m_1p_1
$multi_1$	2	any	0,1,H	any	●			m_1p_2
$multi_1$	2	any	9	any				m_1p_2
…	…	…	…	…	…	…	…	…
$multi_1$	9	any	0,1,H	any	●			m_1p_9
$multi_1$	9	any	9	any				m_1p_9
$multi_1$	0	any	0,1,H	any	●			m_1p_0
$multi_1$	0	any	9	any				m_1p_0
$multi_1$	H	any	any	any			right	nextm
…	…	…	…	…	…	…	…	…
$multi_9$	1	any	0,1,H	any	●			m_9p_9
$multi_9$	1	any	9	any				m_9p_9
$multi_9$	2	any	0,1,H	any	●			m_9p_{18}
$multi_9$	2	any	9	any				m_9p_{18}
…	…	…	…	…	…			…
$multi_9$	9	any	0,1,H	any	●			m_9p_{81}
$multi_9$	9	any	9	any				m_9p_{81}
$multi_9$	0	any	0,1,H	any	●			m_9p_0
$multi_9$	0	any	9	any				m_9p_0
$multi_9$	H	any	any	any			right	nextm
nextm	any	any	0,1,…, 9,H	any	0		right	
nextm	any	any	●	any			left	multi
m_1p_0	any	any	any	any	9		left	$multi_1$
m_1p_1	any	any	any	0	9	1	left	$multi_1$
m_1p_1	any	any	any	1	9	2	left	$multi_1$
…	…	…	…	…	…	…	…	…
m_1p_1	any	any	any	9	9	0	left	$1plus_1$
m_1p_2	any	any	any	0	9	2	left	$multi_1$
m_1p_2	any	any	any	1	9	3	left	$multi_1$
…	…	…	…	…	…	…	…	…
m_1p_2	any	any	any	8	9	0	left	$1plus_1$
m_1p_2	any	any	any	9	9	1	left	$1plus_1$
…	…	…	…	…	…	…	…	…
m_1p_9	any	any	any	0	9	0	left	$multi_1$
m_1p_9	any	any	any	1	9	1	left	$1plus_1$
…	…	…	…	…	…	…	…	…
m_1p_9	any	any	any	9	9	8	left	$1plus_1$
…	…	…	…	…	…	0	…	…
m_9p_0	any	any	any	any	9		left	$multi_9$
m_9p_9	any	any	any	0	9	9	left	$9plus_1$
m_9p_9	any	any	any	1	9	0	left	$9plus_1$
…	…	…	…	…	…	…	…	…
m_9p_9	any	any	any	9	9	8	left	$9plus_1$
…	…	…	…	…	…	…	…	…
m_9p_{81}	any	any	any	0	9	1	left	$9plus_8$
m_9p_{81}	any	any	any	1	9	2	left	$9plus_8$
…	…	…	…	…	…	…	…	…
m_9p_{81}	any	any	any	9	9	0	left	$9plus_9$
$1plus_1$	any	any	any	0		1	right	1find9
…	…	…	…	…	…	…	…	…
$9plus_1$	any	any	any	0		1	right	9find9
$1plus_1$	any	any	any	1		2	right	1find9
…	…	…	…	…	…	…	…	…
$9plus_1$	any	any	any	1		2	right	1find9
…	…	…	…	…	…	…	…	…
$1plus_1$	any	any	any	9		0	right	$1plus_1$
…	…	…	…	…	…	…	…	…
$9plus_1$	any	any	any	9		0	right	$9plus_9$
…	…	…	…	…	…	…	…	…
$1plus_9$	any	any	any	0		9	right	1find9
…	…	…	…	…	…	…	…	…
$9plus_9$	any	any	any	0		9	right	9find9
$1plus_9$	any	any	any	1		0	right	$1plus_1$
…	…	…	…	…	…	…	…	…
$9plus_9$	any	any	any	1		0	right	$9plus_1$
…	…	…	…	…	…	…	…	…
$1plus_9$	any	any	any	9		8	right	$1plus_1$
…	…	…	…	…	…	…	…	…
$9plus_9$	any	any	any	9		8	right	$9plus_1$
1find9	any	any	0,…, 8,H,	any			right	
…	…	…	●	…	…	…	…	…
9find9	any	any	…	any			right	
1find9	any	any	0,…, 8,H,	any			left	$multi_1$
…	…	…	●	…	…	…	…	…
9find9	any	any	9	any			left	$multi_9$
			…					
			9					

参考文献：

[1] Klaus Weihrauch. Computable Analysis:An Introduction[M]. New York：Springer，2000.

[2] Oliver Aberth. Computable Analysis[M]. New York：Mcgraw-Hill，1980.

[3] Lenore Blum，Felipe Cucker，Michael Schub，et al. Complexity and Real Computation[M]. New York：Springer，1998.

[4] Ning Zhong. Recursively enumerable subsets of $\Re_q$ in two computing models: Blum-Shub-Smale machine and Turing machine [J]. Theoretical Computer Science，1998，197：79-94.

[5] 蒋宗礼，姜守旭. 形式语言与自动机理论[M]. 北京：清华大学出版社，2003.

[6] 赵正平. 图灵机及其构造研究[J]. 电脑知识与技术：学术交流，2006(26)：192-194.

[7] 王 强. 四则运算图灵机的构造[J]. 内蒙古师范大学学报:自然科学汉文版，2004 (3)：275-277.

[8] 蒋东海，卢殿臣. 用图灵机计算Klein—Gordon方程[J]. 佳木斯大学学报：自然科学版，2007(2)：268-269.

[9] 胡宝洁，赵忠文，曾 峦，等. 图灵机和图灵测试[J]. 电脑知识与技术：学术交流，2006(23)：132-133.

[10] 刘 义，田 静，李 帅. 基于图灵机的计算问题[J]. 黑龙江科技信息，2008(14)：81-82.

（责任编辑：廖友媛）