关于Neumann自动机与Turing机相似问题

侯 广 坤
（计算机科学系）

Neumann自动机是由 Neumann. J. Ven. 提出来的一个确定性计算模型，本文的主要结果是：引出确定性计算模型的相似性概念，证明Neumann自动机与Turing 机在可计算性的意义上有相同的功能；而在计算复杂性的意义上是相似的。

设P为某字母表Σ上为机器M所接受的字，用$t_M(P)$来描述输入字P到结束停机所需要的步数。

令 $$T_M(n)=\max_{|P|=n} t_M(P)$$

其中n是字P的长$|P|$。

定义 两个确定性计算模型类**A**，**B**称为相似的，如果

(1)任意用模型$A\epsilon$**A**可计算的函数，存在$B\epsilon$**B**，此函数在B上可计算；反之亦然。

(2)若对于任意用模型$A\epsilon$**A**可计算的函数φ，以长度为n的字P作为输入字时的时间耗费为$T_A(n)$；由模型A可以构造模型$B\epsilon$**B**，并且存在着多项式$h(x)$，$g(x)$，$f(x)$，使B计算同一函数从P开始的时间耗费为$T_B(n)$：

若$T_A(n)<k$，则$T_B(n)\leqslant h(n)$

若$T_A(n)\geqslant k'n$，则$g(T_A(n))\leqslant T_B(n)\leqslant f(T_A(n))$。其中$k,k'$为不依赖于$n$的常数。

相似关系具有反身性，对称性和传递性。

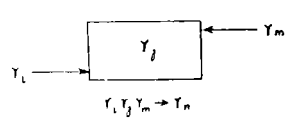
Neumann自动机是由Neumann单元组成的计算装置。

定义 Neumann单元是满足如下条件的器件：

(1)在每一步$t=1,2,\cdots\cdots$中，单元的状态是有穷非空字母表$\Sigma=\{r_1,r_2,\cdots r_v\}$中的一个字母。

(2)单元有两个输入：左输入和右输入，它们每一步在向单元输入Σ中的一个字母。

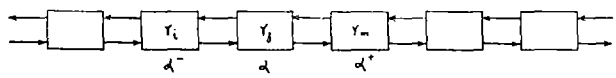
(3)单元在$t+1$步的状态不仅依赖于本单元t步的状态，而且还依赖于左、右输入在t步的状态。单元实现了函数$\phi(p,r,q)$，自变量和函数值都是Σ中的字母。



图一 Neumann单元

对于一个Neumann单元有着这样的一个特殊状态称为静止状态，若$\phi(r,r,r)=r$或$rrr\to r$；所有其它情况为运行状态。因此在字母表Σ中包含静止字符ϕ，单元的静止状态为$\phi\phi\phi\to\phi$。

定义 假若指定某个Neumann单元η，Neumann自动机是从η出发两面延长的有穷个Neumann单元的粘合，使得每一个Neumann单元的两个输入状态分别是左、右相邻的Neumann单元的状态。



图二 Neumann自动机

定义 某步骤t中所有Neumann单元的状态构成Neumann自动机在t步的步局，用$K(t)$来表示。

任一个Neumann单元α，在t步中用$\alpha(t)$来表示它在t时的状态，在步局$K(t)$中用$\alpha^+(t)$和$\alpha^-(t)$分别来表示单元α在t步的右输入通道和左输入通道的状态。

由于Neumann机不像Turing机一样有内部状态，从而需指出初始字和结束字。为此，我们将字母表Σ变为两层字母表，并作如下规定，不失一般性，令原字母表Σ只有两个元素$\{0, 1\}$。

(1)我们研究Neumann单元的状态是两层字母表中的字母，且规定：字母表中$\begin{matrix}\wedge\\\wedge\end{matrix}$为静止字符；而$\begin{matrix}0&1\\q&q\end{matrix}$为初始字的第一个字符，$\begin{matrix}0&1\\!&!\end{matrix}$为结束字的第一个字符，而$\begin{matrix}0&1\\\wedge&\wedge\end{matrix}$是除初始字和结束字外的其它字的第一个字符。

(2)我们还规定，对于指令$\alpha^-(t)\alpha(t)\alpha^+(t)\to\alpha(t+1)$的左边只碰到$\begin{matrix}\wedge&0&1&0&1\\\wedge&\wedge&\wedge&!&!\end{matrix}$时，即左边的三个状态中没有一个形如$\begin{matrix}0&1\\q&q\end{matrix}$的字符，则此时就有$\alpha(t+1)=\alpha(t)$。

定义 设A为Neumann自动机，f为某字函数，如果对于每一个此函数定义域中的字P，自动机A以P为输入字在有穷步内显示$R=f(P)$，则称为Neumann自动机计算函数f。

定义 所有能由某Neumann自动机计算的函数，称为Neumann可计算函数。

定理1 任意在Turing M上可计算的函数f，都存在着Neumann自动机A，使f在A上可计算。

证明 设Turing M的外字母表是$\Sigma=\{\sigma_1,\sigma_2\cdots\sigma_k\}$；内字母表$\theta=\{q_1,q_2,\cdots q_m\}$。$f$在$M$上可计算意味着任意在$f$定义域中的字$P$可由初始步局$K_0$给出，Turing$M$通过有穷步达到步局$K$，使$K$时带上所表示的内容是字$R=f(P)$。

首先，我们将任何一个Turing步局都可以用Neumann步局来描述。在Turing步局K的带中的α格可以描述为$\begin{matrix}x\\q\end{matrix}$，其中$x$是此方格中属于$\Sigma$中的字符。而$q$由如下条件确定：若磁头不位于方格$\alpha$，则$q$为$\wedge$，这里$\wedge$不属于Turing机的内字母表中。若磁头位于方格$\alpha$之中，则$q$为Turing的内部状态。这样，我们建立的Neumann自动机的状态字母表是共有$(m+1)\cdot k$个元素的两层的字母表R。

现在我们要证明三个事实：(1)对于所有的Turing机的初始字，可以用Neumann机的初始字来描述；(2)所有Turing机的结束字可以用Neumann机的结束字来描述；(3)设Neumann步局$\widetilde{K}$描述Turing步局K，且Turing步局K变换到K'(通过一个步骤)，

则Neumann步局$\widetilde{K}$通过一个步骤变换到$\widetilde{K'}$，而步局$\widetilde{K'}$正好是描述Turing步局K'。

其实只要我们做到(3)的对应，而(1),(2)是显然的。在字母表R中有着这样的状态$\overset{x}{\wedge}$，它在Turing步局中表明这样的意义：在带中是空方格，而磁头不在此位置上。在一个Turiug机中除有穷多个方格外都是这种方格。此时，$\overset{x}{\wedge}$可以认为是Neumann机的静止状态。根据上面的方法，将Turing步局K,用Neumann步局$\widetilde{K}$描述，Turing机从步局K到步局K'的一步的变换时，只在磁头所指的方格中交换信息。在步局$\widetilde{K}$中，除Turing机磁头都指向的方格外，其它都是形如$\overset{x}{\wedge}$的状态。根据Neumann自动机定义时的规定(2)，只有一个Neumann单元交换信息，其余的单元都是$\alpha(t+1)=\alpha(t)$。此正是Turing机的一个步骤做的工作。

因此，结论(3)是正确的，而对于任意一个Turing可计算函数，我们都可以找到一个Neumann机去计算此函数。〔证毕〕。

定理 2　对于任意一个Neumann可计算函数，我们都可以找到一架Turing机来计算它。

证明　现在我们从任一Neumann机出发，证明用图来描述：

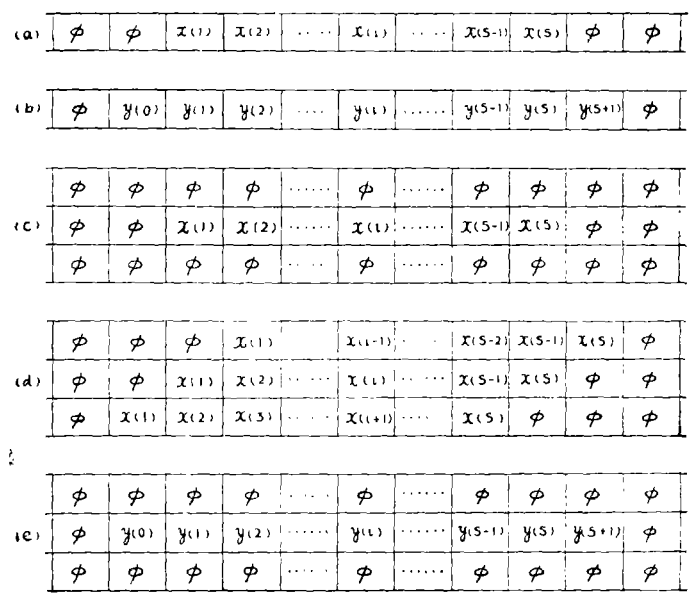


图 三　Turing 机和 Neumann 机对应

假定有一Neumann自动机，它的一个步局$\widetilde{K}$由图三中的(a)所示。经过一个步骤的变换为图三(b)所示，即步局$\widetilde{K'}$。

现在我们定义一架3-带Turing机M，它的中间那一条带正好是Neumann步局$\widetilde{K}$。在图三中的(c)，第一条带和第三条带的相应方格均为ϕ。Turing机是这样工作的：第一条带向右移一格，然后将$x(1),x(2),\cdots x(s)$写在带中；第三条带向左移一格，将$x(1),x(2),\cdots x(s)$写在带中。这样得出如图三(d)所示的步局。在每条带工作时，其它带保

持原来的状态。我们把这样所得到的(d)的每一列作为一架Turing的指令，正好是(a)中所示的每个Neumann单元的工作状况，而得到的结果记在中间的一条带中，第一条带和第三条带为ϕ。如图三(e)所示。

这样我们证明了：对于任意一架Neumann自动机A，可以找到一架3-带Turing机M，并且，若这架Neumann机A从步局$\widetilde{K}$加工到$\widetilde{K'}$，则这架Turiug机从K到K'，在K'中中间那条带的内容正好是Neumann机反应段在$\widetilde{K'}$时的内容。即任意一个 Neumann可计算函数，都可以找到一个Turing机来计算它。〔证毕〕。

现在我们考察字对称性识别问题：给出一个字母表上长度为n的字(不失一般性，令此字母表是$\{0, 1\}$)$P(1)P(2)\cdots P(n)$是否是对称的？也就是说是否等于它的镜反射$P(n)P(n-1)\cdots P(2)P(1)$。显然这个问题是可计算的，因为任意字是有穷的字长。我们有下列的引理。

引理1 若用任一Turing机来解字P对称性识别问题，则$T_M(n)\geqslant C'_M n^2$，其中n是输入字P的长，C'_M只是依赖于M而不依赖于n的数。

证明 假定Turing机M的工作是从第一步局开始，在第一步局中是安排字P。开始时，这架Turing机注视并记住$P(1)$，然后右移n次到$P(n)$，若$P(n)\neq P(1)$，则Turing机停机输出0(否定)。若$P(n)=P(1)$，则Turing机磁头移到$P(n-1)$，记住$P(n-1)$向左移一直到$P(2)$，比较$P(2)$和$P(n-1)$，若$P(2)\neq P(n-1)$，则停机输出0。否则又从$P(2)$开始向右移，这样延续下去一直到对应的位不等为止，而输出0。若P是对称的，则比较所有的$P(i)(i=1,2,\cdots n)$，需要的步数是：

$$n+(n-1)+\cdots\cdots+2+1=\frac{n(n+1)}{2}\approx\frac{n^2}{2}.$$

假若Turing机在每次记忆时记住不止一个方格的内容，而是k个方格的内容，这样的Turing机记作M_k，此时对称性识别的时间耗费大约是$n^2/4k$。在所有字长为n(不失一般性令n为偶数)的字中，对称性识别的时间耗费就必须是$\geqslant n^2/4k$，也就是说$T_M(n)\geqslant C'_M n^2$，C'_M依赖于Turing M，不依赖于n。〔证毕〕。

引理2 存在着这样的Neumann自动机A_0，以线性时间耗费识别二进制字P的对称性，也就是说不超过$C''_{A_0}|P|$；$|P|$表示字P的长度；C''_{A_0}是不依赖于P的常数。

证明 不失一般性，令P的长度为偶数$2r$。若P的长度为$2r+1$，则可以用一条指令使这架Neumann机在不影响识别P的步数的阶内变为长度为$2r$的字。这样的变换不影响我们所证的结论。

在$P(r)$和$P(r+1)$所在的两个相邻的Neumann单元中间用一条线隔开(这条线只是人为的记号，不在Neumann字母表之内，并且也不输入任Neumann单元之中)，如图四所示，此时。我们按上面所说的规则建立两层的Neumann字母表R_P。字母表R_P是在给出P后所建立的字母表，使得在线的左边的所有反应段域内的下方加上一个左字，而在线的右边的反应段域内的下方加上一个右字，其它地方同前面一样用$\wedge$来表示。

此时，我们给出三组八条指令构成Neumann程序：指令形如$\alpha^-(t)\alpha(t)\alpha^+(t)\to\alpha(t+1)$的形式

Ⅰ

(1)$\alpha^-(t)$：任意，$\alpha(t):\overset{x}{\underset{右}{}}$，$\alpha^+(t):\overset{y}{\underset{q}{}}$，$\alpha(t+1):\overset{y}{\underset{q}{}}$。

解释：x,y,q,z——被任意分隔在右半步局。

Ⅱ

(2)$\alpha^-(t):\overset{z}{\underset{q}{}}$，$\alpha(t):\overset{x}{\underset{左}{}}$，$\alpha^+(t):\overset{y}{\underset{左}{}}$，$\alpha(t+1)\overset{z}{\underset{q}{}}$。

(3) $\alpha^-(t):\overset{z}{\underset{左}{}}$，$\alpha(t):\overset{x}{\underset{左}{}}$，$\alpha^+(t):\overset{x}{\underset{右}{}}$，$\alpha(t+1):\overset{z}{\underset{左}{}}$。

解释：q,x,y,z——发现非对称前的任意被分隔在左半步局。

(4) $\alpha^-(t):\overset{z}{\underset{左}{}}$，$\alpha(t):\overset{x}{\underset{左}{}}$，$\alpha^+(t):\overset{y}{\underset{右}{}}$，$\alpha(t+1):\overset{x}{\underset{左}{}}$。

(5) $\alpha^-(t):\overset{z}{\underset{左}{}}$，$\alpha(t):\overset{x}{\underset{左}{}}$，$\alpha^+(t)$：任意，$\alpha(t+1):\overset{z}{\underset{左}{}}$。

解释：x,y,z——任意$x\neq y$在发现非对称后的左半步局内。

Ⅲ

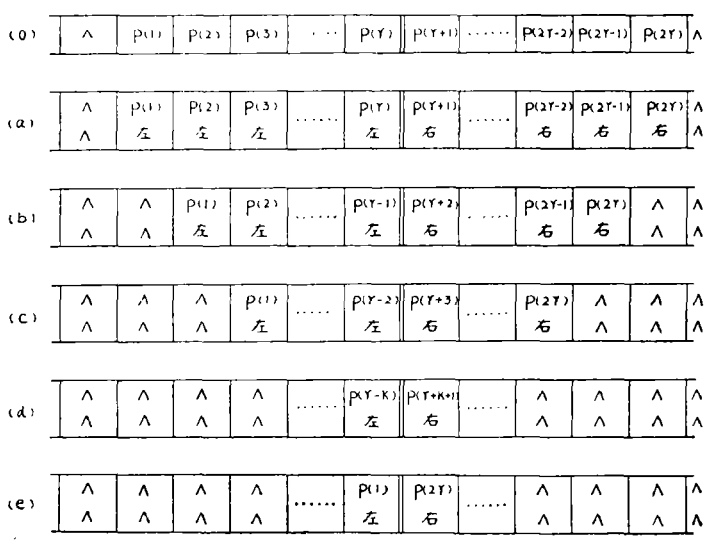
(6)$\alpha^-(t):\wedge$，$\alpha(t):\overset{x}{\underset{左}{}}$，$\alpha^+(t):\overset{x}{\underset{右}{}}$，$\alpha(t+1):\overset{1}{\underset{!}{}}$。

(7) $\alpha^-(t):\wedge$，$\alpha(t):\overset{x}{\underset{左}{}}$，$\alpha(t):\overset{y}{\underset{右}{}}$，$\alpha(t+):\overset{0}{\underset{!}{}}$。

(8) $\alpha^-(t):\wedge$，$\alpha(t):\overset{x}{\underset{左}{}}$，$\alpha+(t)\overset{z}{\underset{右}{}}$，$\alpha(t+1):\overset{0}{\underset{!}{}}$。

其中，!为停机符号。

解释：x,y,z——任意的，但$x\neq y$时给出结果。



图四 Neumann机A_0的步局

我们要建立Neumann机A_0，首先要建立Neumann机A_1和A_2，然后将A_1和A_2串联起来。$A_0=A_1\cdot A_2$：A_1中的指令里表明开始状态的字符，作为A_0的开始状态。A_1中表明停机状态的字符改为表明A_2开始状态的字符。而A_2中的结束状态作为A_0的结束状态。

现在我们先建立自动机A_1。A_1是将一个单层的步局改变为双层的步局，即在图四中将步局(0)，加工为步局(a)。即执行形如$xyz\to\overset{y}{\underset{右}{}}$或$xyz\to\overset{y}{\underset{左}{}}$的指令，完全决定于$\alpha(t)$位于右或左半步局。

至于Neumann机A_2是由上述三组八条指令所组成的Neumann程序，使得：

(1)A_1工作的时间不超过C_A，事实上，从步局(0)到步局(a)是执行一次指令。

(2)A_1同A_2的状态除$\wedge$，$\overset{0}{\underset{左}{}}$，$\overset{1}{\underset{左}{}}$，$\overset{1}{\underset{右}{}}$，$\overset{1}{\underset{右}{}}$外没有共同的状态。

(3)在A_1工作的过程中在出现步局(a)之前不出现状态$\overset{0}{\underset{左}{}}$，$\overset{1}{\underset{左}{}}$，$\overset{0}{\underset{右}{}}$，$\overset{1}{\underset{右}{}}$。

Neumann机A_2的工作关键在与分隔线相邻的左半步局的单元的工作。这样一个单元所执行的是第(3)或(4)条指令取决于分隔线相邻的两边的单元的状态的上层字母是否相同。其它单元执行第(1)或(2)或(5)条指令。假若A_2一直到图四的步局(e)，仍然没有发现非对称性单元，则执行第(6)或(7)来判定其对称性。若在(e)之前已发现非对称单元，则从发现的那步开始到(e)执行(5)条指令，(e)之后执行第(8)条指令。

因此，Neumann机A_2的工作步数至多是r步。因此$T_{A_0}(n)\leqslant C''_{A_0}n$，即不超过$C''_{A_0}|P|$。〔证毕〕。

定理3 设任一架Turing机解字对称性识别问题的时间耗费为$T_M(n)$，则可以找到一架Neumann自动机A_0，使$T_M(n)\geqslant C_M T^2_{A_0}(n)$这里$C_M$是不依赖于$n$的常数。

证明 由引理1和引理2很容易推出所需的结论。

定理4 对于任意用Neumann自动机A可计算的函数f，可以找到这样一架Turing机M计算f，使得对任意输入字P：

$$t_M(P)\leqslant D'_A\, t^2_A(P)+D''_A|P|t_A(P)$$

证明 我们在定理2的证明的基础上来估计所建立的对应于A的Turing机M计算f的复杂性。

令Neumann机A以长度为d的字开始工作到t步，当然这个字已属于可计算函数f的定义域中。此时对应的Turing机做以下的工作：当Neumann机工作一步时，Turing机M必须在第一条带上相应于第二条带向右移一格写上对应第二条带的内容；在第三条带上相应于第二条带向左移一格写上对应于第二条带的内容。然后按照列分别执行列中所描述的指令，最后将第一条带和第三条带都冲空(ϕ)。也就是说每一个Neumann自动机步，Turing机就依照字的长度来回做了四次，又Neumann机每一步向左右各扩展一个单元，所以当Neumann机A由长度为d的字开始工作到t步时，相应的Turing机已做了如下步数的工作：

$$4(d+2)+4(d+4)+\cdots+4(d+2t)=4(d+t+1)t.$$

现令Neumann机计算f是以字P开始工作到结束停机所需的步数是$t_A(P)$，从上述的结论，相应的Turing机已做了如下步数：

$$4(|P|+t_A(P)+1)t_A(P).$$

又Turing机又要回到最左边来给出结束字，所要花的步数是$|P|+2t_A(P)$，因此

$$t_M(P)\leqslant 4(|P|+t_A(P)+1)t_A(P)+2(|P|+2t_A(P)).$$

$$t_M(P)\leqslant 4t'_A(P)+2(2t_A(P)+1)|P|+8t_A(P)$$

$$t_M(P)\leqslant D'_A\ t_A^2(P)+D''_A|P|t_A(P).$$〔证毕〕

推论　若$T_A(n)<k$，则$T_M(n)\leqslant D_An^2$.

若 $T_A(n)\geqslant k'n$，则$T_M(n)\leqslant D_A^*\ T_A^2(n)$.

基本定理　Neumann自动机与Turing机是相似的.

根据相似性定义，由定理1,2,3,4及推论直接得到.

我们可以用类似的方法证明其它一些计算模型类的相似性. 当然，也可能找到其它更简单的方法. 在证明其它一些确定性计算模型类相似之后，我们得出了如下两个'想法'：

(一)所有在Church论题下可计算性的意义上有相同功能的确定性计算模型，在计算复杂性的意义上是相似的.

(二)若以是否存在多项式时间算法把所有可计算性问题划分为P类问题和算法难解的问题两大类，则某个问题的复杂性(对于上述两大类的从属性)只是问题本身的属性，与程序无关，即不依赖于计算它所选择的确定性计算模型.

On the Similarity of Neumann Automata with Turing Machine

Hou Guangkun

Abstract

This paper discusses the simility concept of computational model, and shows that Neumann automata has the similarity with Turiug machine.

注，文中所述的计算模型都是指确定性计算模型类