



# 第4章 图灵机

许桂靖 杨 莹

# Overview

- 图灵机 (Turing Machine, TM), 是计算机的一种简单的数学模型。
- 历史上, 冯·诺曼计算机的产生就是以图灵机为理论基础的。 TM的介绍。
- 丘奇—图灵论题: 一切合理的计算模型都等同于图灵机。(一切算法都可以用Turing机实现)。

# Overview

- 图灵机所定义的语言类---递归可枚举集合
- 图灵机所计算的整数函数类---部分递归函数
- 以图灵机为模型,研究问题的可计算性,即确定该问题是可计算的、部分可计算的,还是不可计算的。

# Overview

4.1 图灵机模型

4.2 图灵机的变化和组合

4.3 通用图灵机

4.4 图灵机可计算性

4.5 部分递归函数及其与图灵机  
的等价性（简介）

## 4.1 图灵机模型

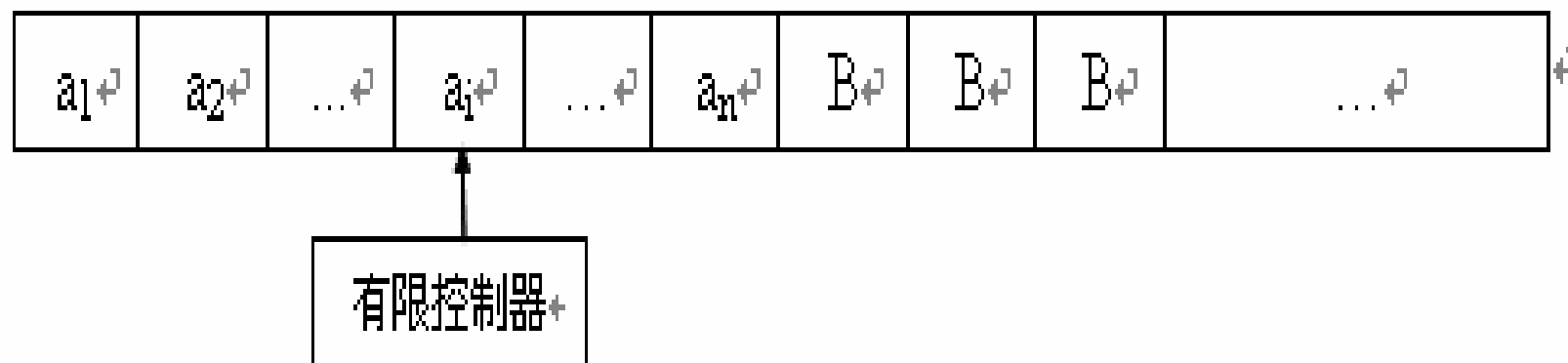


图 4-1 图灵机基本模型

## 4.1 图灵机模型

定义4-1 图灵机  $M = (K, \Sigma, \Gamma, \delta, q_0, B, F)$ , 其中

$K$  是有穷的状态集合;

$\Gamma$  是所允许的带符号集合;

$B \in \Gamma$ , 是空白符;

$\Sigma \subseteq \Gamma$ ,  $B \notin \Sigma$ , 是输入字符集合;

$F \subseteq K$ , 是终止状态集合。

$q_0 \in K$ , 是初始状态;

## 4.1 图灵机模型

$$\delta : K \times \Gamma \rightarrow K \times \Gamma \times \{L, R, S\}$$

是图灵机的动作(状态转移)函数, 这里

L表示读头左移一格;

R表示读头右移一格;

S表示读头不动;

$$\delta(q, a) = (p, b, z)$$

表示状态 $q$ 下读头所读符号为 $a$ 时, 状态转移为 $p$ , 读头符号变为 $b$ , 同时读头位置变化为 $z$ .

## 4.1 图灵机模型

定义4-2 设当前带上字符串为 $x_1x_2 \dots x_n$ ,  
当前状态为 $q$ , 读头正在读 $x_i$ , 图灵机的瞬  
时描述ID 为

$$x_1x_2 \dots x_{i-1} q x_i \dots x_n$$

带格局为:

B111B....  
↑  
q1

(Machine's configuration)



## 4.1 图灵机模型

■ 定义4-3 设当前的瞬时描述

$$ID_1 = x_1 x_2 \dots x_{i-1} q x_i \dots x_n$$

若有  $\delta(q, x_i) = (p, y, L)$ , 则图灵机瞬时描述变为

$$ID_2 = x_1 x_2 \dots x_{i-2} p x_{i-1} y x_{i+1} \dots x_n;$$

若有  $\delta(q, x_i) = (p, y, R)$ , 则图灵机瞬时描述变为

$$ID_2 = x_1 x_2 \dots x_{i-1} y p x_{i+1} \dots x_n.$$

## 4.1 图灵机模型

### ■ 定义4-3

瞬时描述 $ID_1$ 经过一步变为瞬时描述 $ID_2$ , 称 $ID_1$ 与 $ID_2$ 具有一步变化关系, 表示为

$$ID_1 \vdash ID_2$$



若 $ID_1$ 经过 $n$ 步变为 $ID_2$  ( $n \geq 0$ ), 即有

$$ID_1 \vdash ID \vdash \dots \vdash ID_2$$

称 $ID_1$ 与 $ID_2$ 具有多步变化关系, 简记为

$$ID_1 \vdash^* ID_2$$

【例子1】设计一个图灵机实现一进制数加1功能。计算的函数是 $f(x)=x+1$ ,即所谓后继函数,“ $V \leftarrow V+1$ ”。

(  $q_1, B, q_2, B, R$ ),  
( $q_2, 1, q_2, 1, R$ ),  
( $q_2, B, q_3, 1, S$ ).

\* 事实上,五元组序列就是后来的程序(软件).

$K=\{q_1, q_2, q_3\}$ ,  $\Sigma=\{1\}$ ,  $\Gamma=\{1, B\}$ ,  $F=\{q_3\}$ , 初始状态  
 $=q_1$ .  $B=B$ .

初始带格局为:       $B111B....$   
                           $\uparrow$   
                           $q_1$

【例子2】设计一个图灵机实现一进制数减1功能。计算的函数是 $f(x)=x-1$ ,即所谓前驱函数,“ $V \leftarrow V - 1$ ”。

(q1, B, q2,B,R),  
(q2, 1, q3,1, S),  
(q2, B, q5,B, S), /\* q2遇到B停机 \*/  
(q3, 1, q3,1,R),  
(q3, B, q4,B, S),  
(q4, B, q4, B,L),  
(q4, 1, q5,B, S).

$K=\{q1, q2, q3, q4, q5\}$ ,  $\Sigma=\{1\}$ ,  $\Gamma=\{1, B\}$ ,  $F=\{q5\}$ , 初始状态=q1. B=B.

初始带格局为:      B111B....  
                          ↑  
                          q1

**【例子3】** 设计一个图灵机实现二进制数加1功能。计算的函数是 $f(x)=x+1$ ,即所谓后继函数, 其一次典型的运行如图1所示。

( q,0,q,0,R),  
(q,1,q,1,R),  
(q,B,p,B,L),  
(p,0,q<sub>f</sub>,1,S),  
(p,1,p,0,L),  
(p,B, q<sub>f</sub>,1,S)。

$K=\{p, q, q_f\}$ ,  $\Sigma=\{0,1\}$ ,  $\Gamma=\{0,1,B\}$ ,  $F=\{q_f\}$ ,  $q_0=q$ .  $B=B$ .

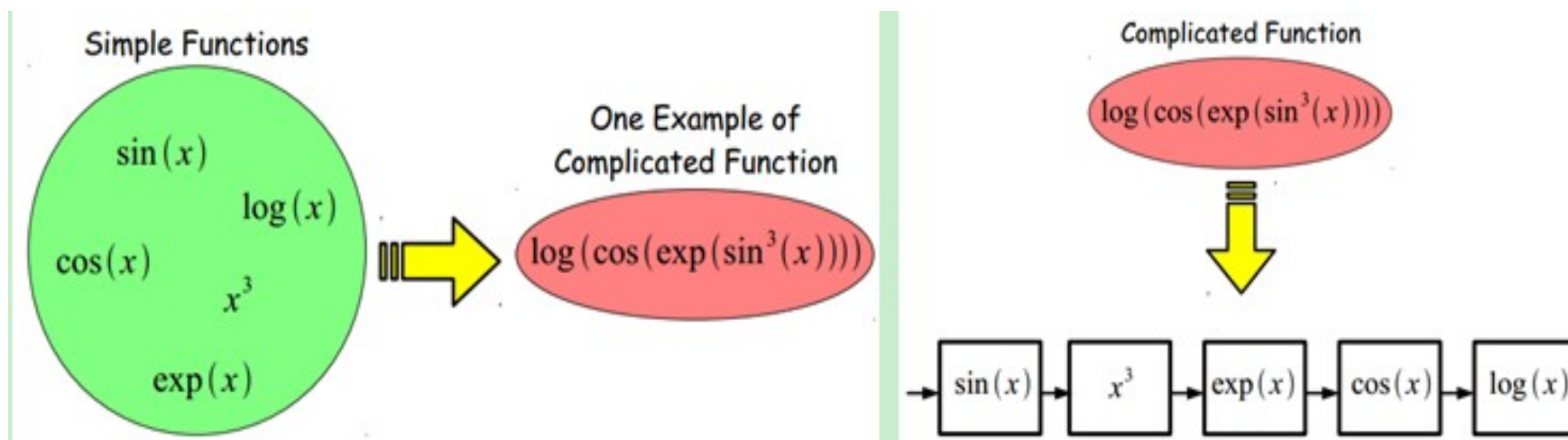
<b>(q,0)</b>	<b>1</b>	<b>0</b>	<b>1</b>	<b>1</b>	<b>B</b>
<b>0</b>	<b>(q,1)</b>	<b>0</b>	<b>1</b>	<b>1</b>	<b>B</b>
<b>0</b>	<b>1</b>	<b>(q,0)</b>	<b>1</b>	<b>1</b>	<b>B</b>
<b>0</b>	<b>1</b>	<b>0</b>	<b>(q,1)</b>	<b>1</b>	<b>B</b>
<b>0</b>	<b>1</b>	<b>0</b>	<b>1</b>	<b>(q,1)</b>	<b>B</b>
<b>0</b>	<b>1</b>	<b>0</b>	<b>1</b>	<b>1</b>	<b>(q, B)</b>
<b>0</b>	<b>1</b>	<b>0</b>	<b>1</b>	<b>(p,1)</b>	<b>B</b>
<b>0</b>	<b>1</b>	<b>0</b>	<b>(p,1)</b>	<b>0</b>	<b>B</b>
<b>0</b>	<b>1</b>	<b>(p,0)</b>	<b>0</b>	<b>0</b>	<b>B</b>
<b>0</b>	<b>1</b>	<b>1</b>	<b>0</b>	<b>0</b>	<b>B</b>

有了语句“ $V \leftarrow V + 1$ ”，“ $V \leftarrow V - 1$ ”和  
“IF  $V \neq 0$  GOTO L”，即可实现S语言编程。

几个程序例子。 ([LINK](#))

S语言从编程的视角研究TM，使我们看到TM的计算能力是如何展现出来的。宏指令macro不是原始指令，而是（具有特定功能）原始指令程序段的缩写。可以大大简化程序设计。宏指令是展示TM强大计算能力的直观方式之一。

**Deep Learning:** 可通过学习一种深层非线性网络结构，实现复杂函数逼近，表征输入数据分布式表示。





## 4.1 图灵机模型

- **定义4-4** 对于图灵机  $M = (K, \Sigma, \Gamma, \delta, q_0, B, F)$ , 定义图灵机接受的语言集  $L(M)$  为

$$L(M) = \{x \mid x \in \Sigma^* \ \& \ q_0 x \vdash_M^* \alpha_1 q \alpha_2 \ \& \ q \in F \ \& \ \alpha_1, \alpha_2 \in \Gamma^*\}$$

- TM接受的语言叫做递归可枚举语言 (recursively enumerable language, r.e.)。

## 4.1 图灵机模型

- 【例4-1】设计一个图灵机，使得
$$L(M) = \{0^n 1^n \mid n \geq 1\}.$$

设计思路：在带上每当将一个0变为X，就把一个1变为Y。当将所有0变为X时，恰将所有1变为Y，这个串就是合法的，最后将X、Y分别还原为0、1。

## 4.1 图灵机模型

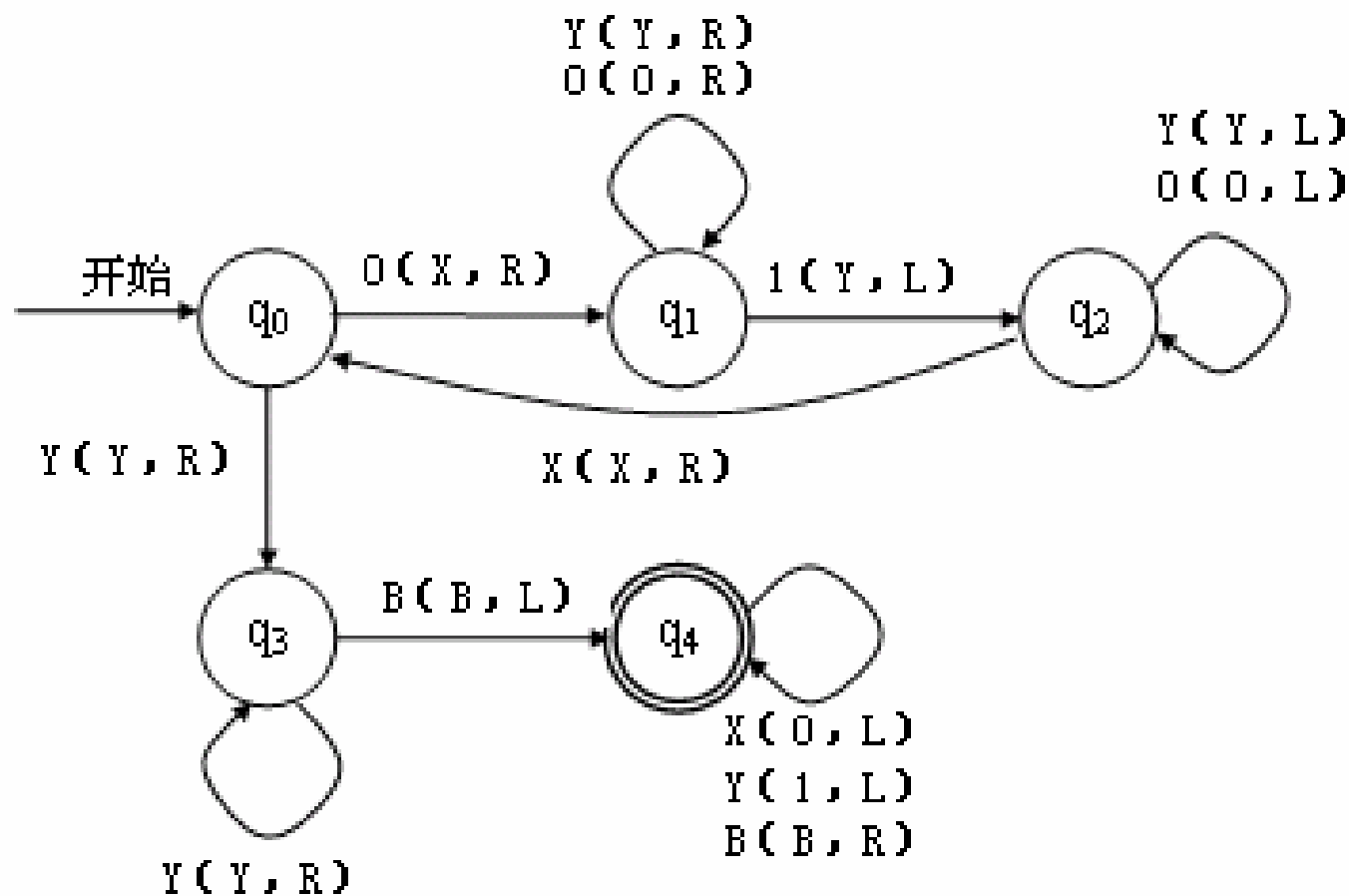


图 4-2 图灵机  $M_1$  的状态转移图

## 4.1 图灵机模型

**【例4-2】** 设计一个图灵机，使之接受

$$L(M) = \{wcw \mid w \in \{a,b\}^*\}$$

设计思路：在C左侧，从左至右逐一处理字符，用状态记下它并标志该符号为已处理符号(X或Y)，移至C右侧对应位置后，判断是否是相同符号。若相同，再返回C左侧循环，直至所有符号比较完毕。最后将标志符号修改回原符号。在设计时，特别注意用状态存贮符号的方法，这是图灵机设计的重要方法之一。

## 4.1 图灵机模型

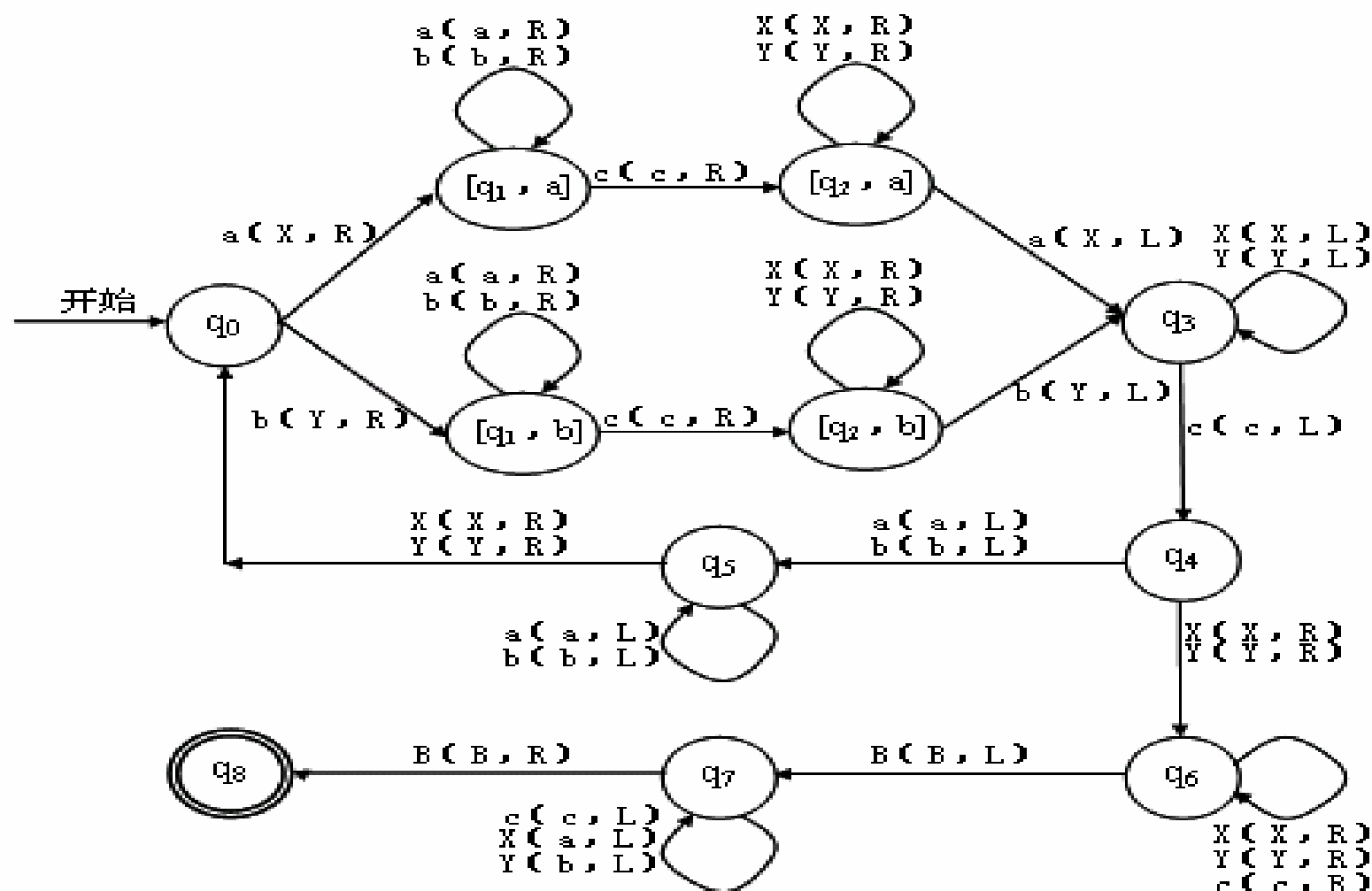


图 4-3 图灵机  $M_2$  的状态转移图

## 4.1 图灵机模型

**【例4-3】** 设计一个图灵机，计算自然数 $n$ 的以2为底的对数。

用一进制表示输入和输出值。 $a^n$ 表示输入 $n$ ， $b^m$ 表示输出 $m$ 。

设计思路：从左到右扫描带，把所碰到的 $a$ 划掉一个，留一个，并将计数器加1。重复此过程，直至 $a$ 不复存在。这里，用字符 $c$ 表示划掉的字符。

# 4.1 图灵机模型

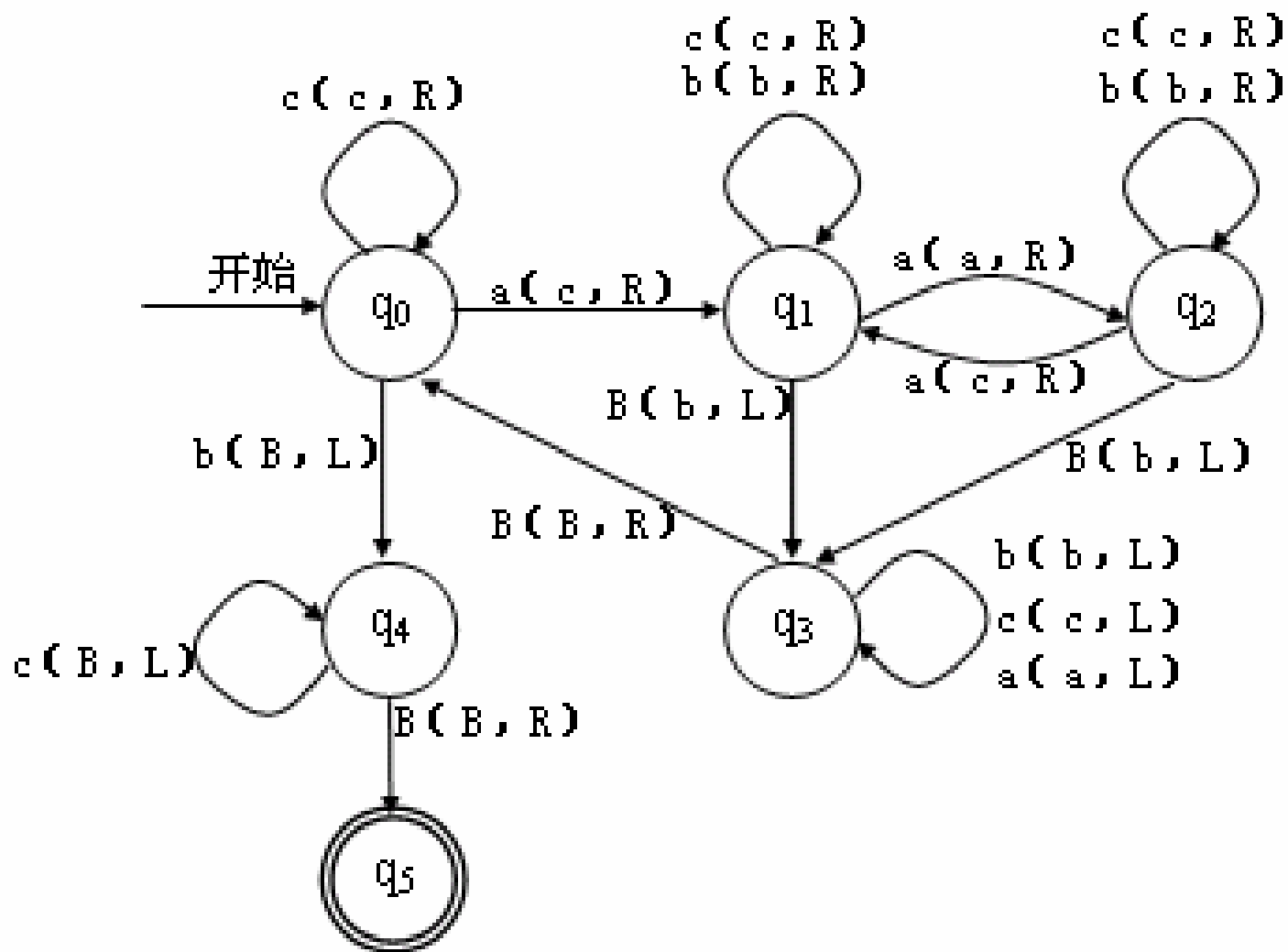


图 4-4 图灵机 M3 的状态转移图

## 4.1 图灵机模型

【例4-4】设计一个图灵机，计算二个自然数 $m$ 、 $n$ 的减法：

$$m-n=\begin{cases} m-n & \text{若 } m \geq n \\ 0 & \text{否则} \end{cases}$$

设计时，整数 $n$ 用 $0^n$ 表示。开始时，带上符号为 $0^m10^n$ ，结束时，带上符号为 $0^{m-n}$ 。每当在1的左边将一个0改变为B，就在1的右边将一个0改为1，若1的右边无0时，再将左边改为B的0恢复回来(只恢复一个多改的B)，结果为带上0串。



## 4.1 图灵机模型

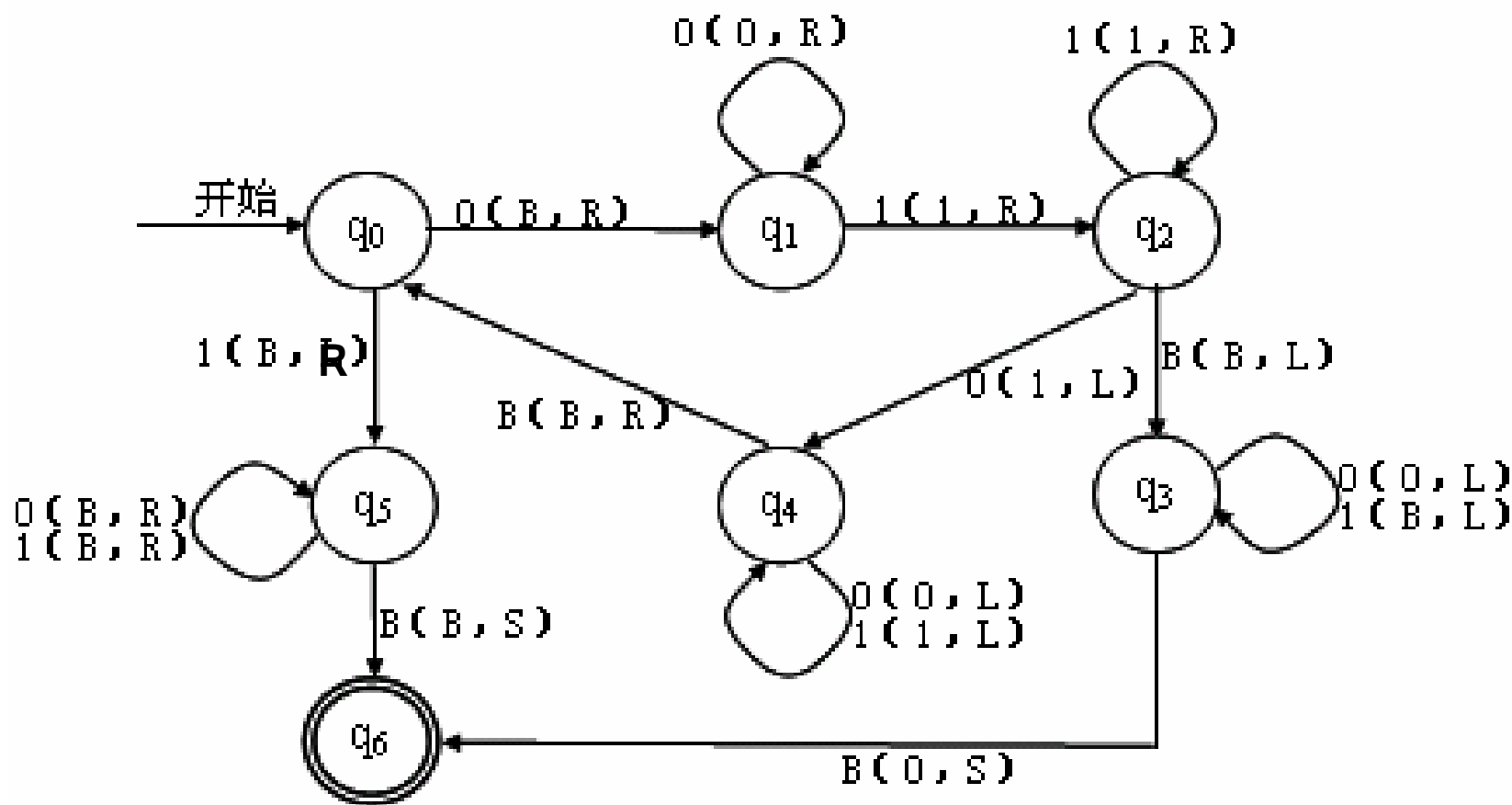


图 4-5 图灵机  $M_4$  的状态转移图

## 4.1 图灵机模型

**【例4-5】** 设计图灵机实现数字从一进制表示到二进制表示的转换。

这个图灵机的设计可以仿例4-3，不同在于每次循环时，要保留除以2的余数作为当前二进制位的值。注意这里首先计算出的是二进制的低位值，所以要将结果不断右移以插入新生成的位，生成的结果是低位在右端。初始时，整数 $n$ 用 $a^n$ 表示，结束时，带上是0、1构成的二进制数。

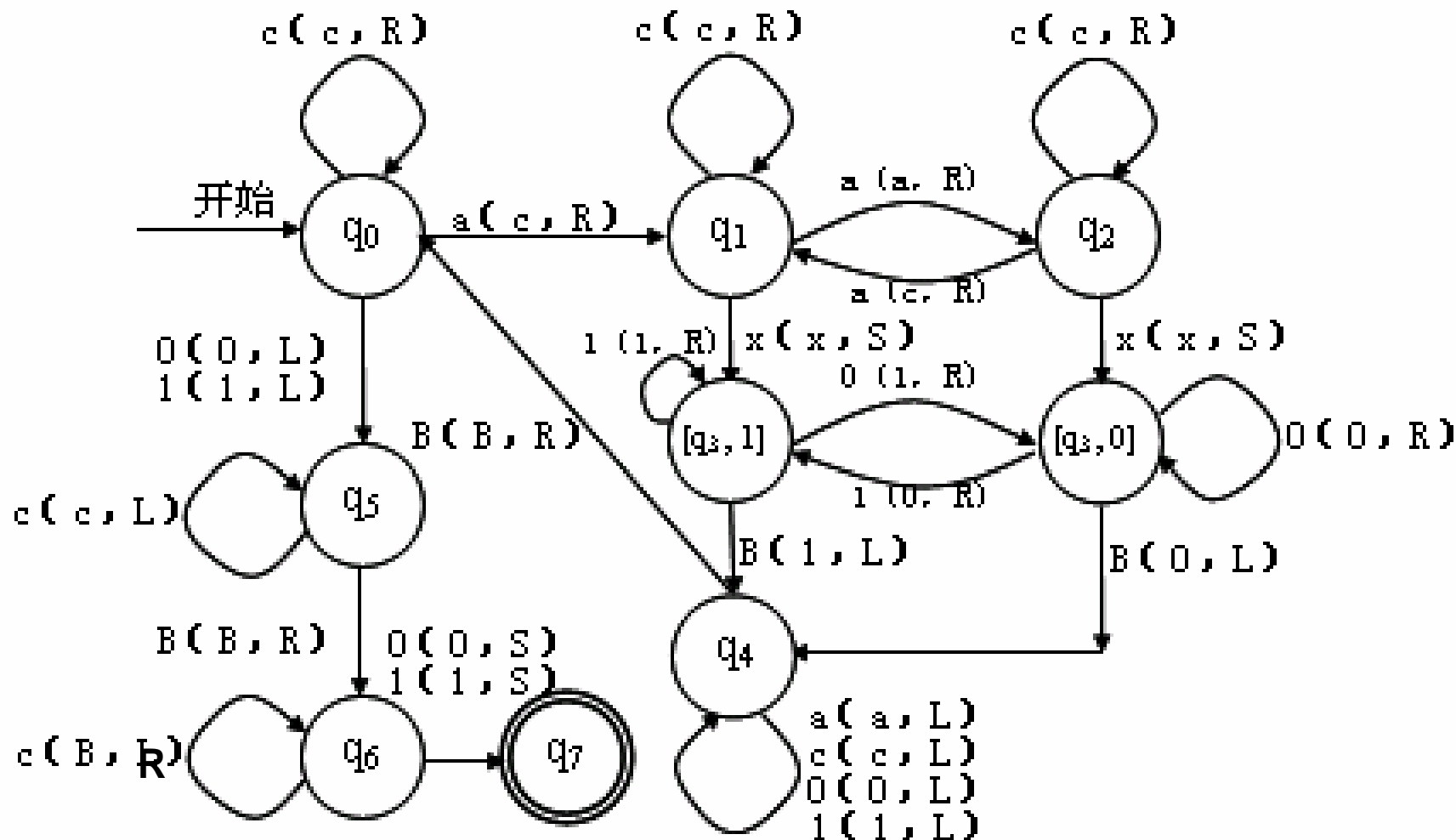


图 4-6 图灵机  $M_5$  的状态转移图

## 4.2 图灵机的变化和组合

- 4.2.1 双向无穷带图灵机
- 4.2.2 多带图灵机
- 4.2.3 非确定图灵机
- 4.2.4 多头图灵机
- 4.2.5 多维图灵机
- 4.2.6 离线图灵机
- 4.2.7 图灵机的组合
- 4.2.8 枚举器

## 4.2.1 双向无穷带图灵机

**定理4-1**  $L$ 被一个具有双向无穷带的图灵机识别，当且仅当它被一个单向无限带的图灵机识别。

**证明：**单向无限TM模拟双向无限TM，采用多道技术。

$a_0$	$a_1$	$a_2$	$a_3$	$\dots$	$\dots$	
$\Phi$	$a_{-1}$	$a_{-2}$	$a_{-3}$	$\dots$	$\dots$	

图 4-7 双道单向无穷带

## 4.2.2 多带图灵机

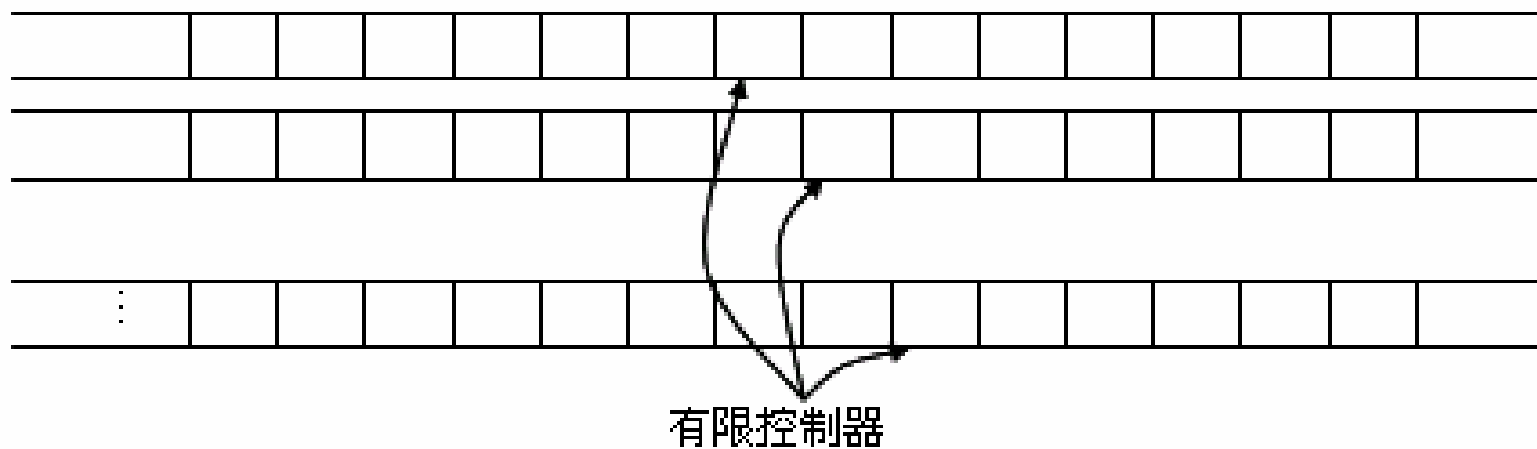


图 4-8 多带图灵机

## 4.2.2 多带图灵机

- **【例4-6】** 设计一个二带图灵机，使得  $T(M) = \{ \omega \omega \mid \omega \in \{0,1\}^* \}$ 。
- 这个问题的关键是比较字符串前后两个部分，为此，首先要对带上字符串计数：每二元素计数加1，按计数值将字符串分为前后两个部分，并将它们分别存放于不同带上，然后进行比较。

## 4.2.2 多带图灵机

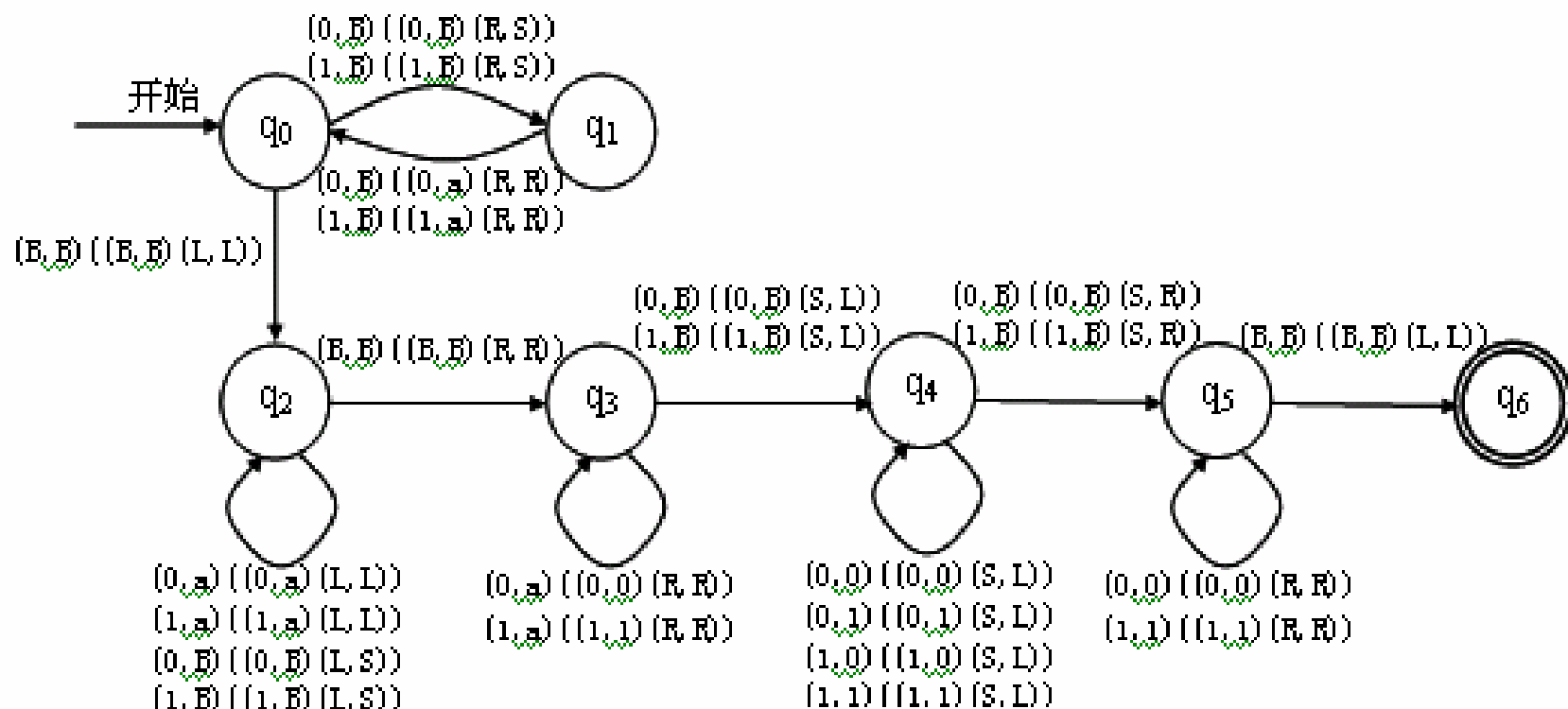


图 4-9 图灵机  $M_6$  的状态转移图



## 4.2.2 多带图灵机

**【例4-7】** 设计二带图灵机，实现二进制到一进制的转换。

设这个图灵机为M7，其第一带用作输入带，第二带用作输出带。设计思路是从左到右扫描输入带上的二进制字符，并使用公式 $r*2+b$ 生成输出带上一进制数，其中 $r$ 是当前输出带上一进制数， $b$ 是当前输入带上扫描的字符，这里的 $r*2$ 就是将原输出带上一进制数 $r$ 复制一遍。例如：1001的一进制数计算过程。

## 4.2.2 多带图灵机

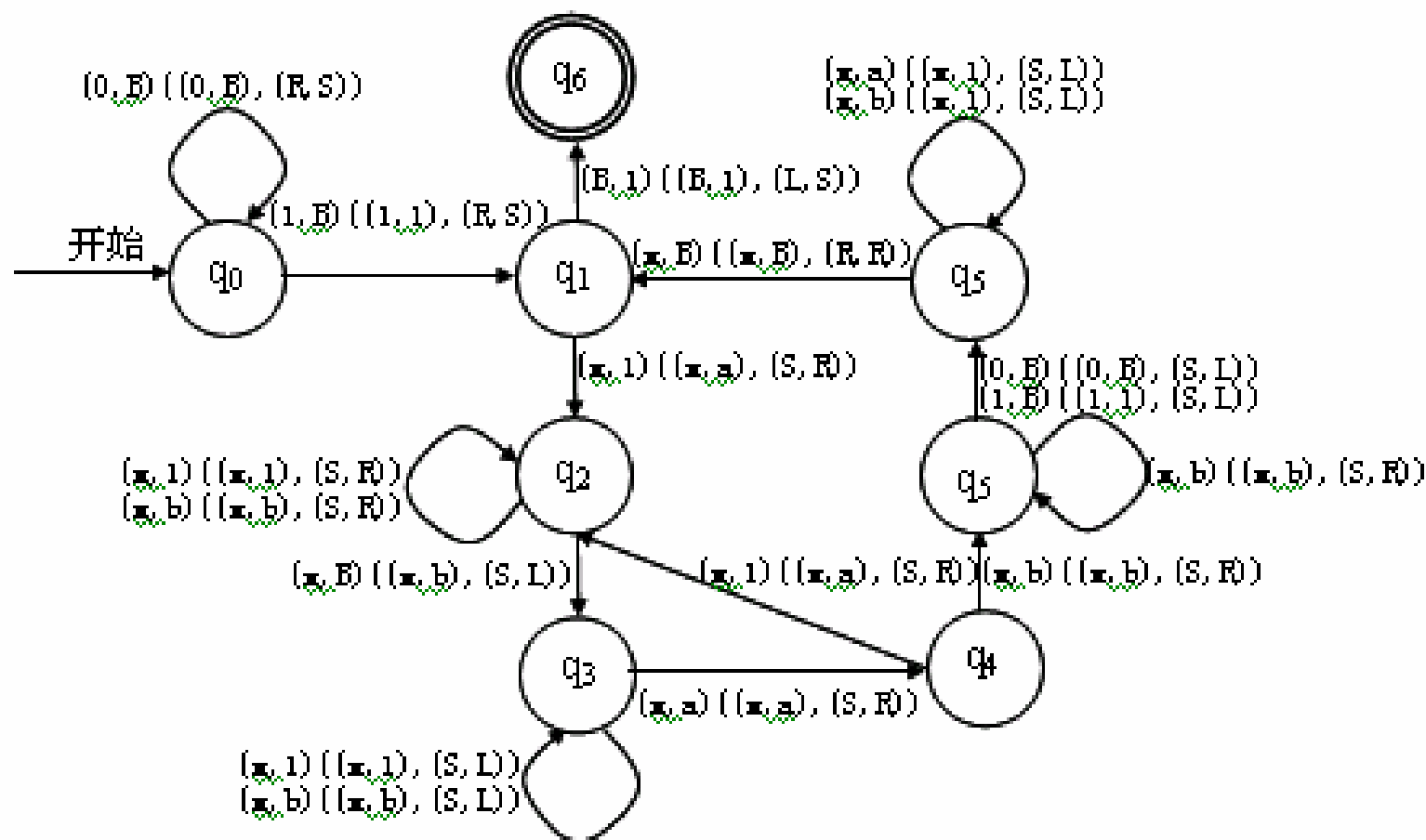


图 4-10 图灵机 M7 的状态转移图 (其中  $x$  表示 0 或 1)

## 4.2.2 多带图灵机

**定理4-2** 如果一个语言 $L$ 被一个多带图灵机接受，它就能被一个单带图灵机接受。

## 4.2.3 非确定图灵机

- 非确定图灵机由一个有穷控制器、一条带和一个读头组成。对于一个给定的状态和被读头扫描的带符号，机器的下一个动作将有有穷个选择。
- 设一个非确定图灵机  $M1=(K, \Sigma, \Gamma, \delta, q_0, B, F)$ ，除转移函数  $\delta$  外，其它同一般图灵机的定义。转移函数  $\delta$  仍是从  $K \times \Gamma$  到  $K \times \Gamma \times \{L, R, S\}$  上的映射，但它可能有多个映射的像，即存在  $q \in K, a \in \Sigma$ ，
$$\delta(q, a) = (p_1, b_1, c_1)$$
$$\delta(q, a) = (p_2, b_2, c_2)$$
$$\dots\dots$$
$$\delta(q, a) = (p_r, b_r, c_r)$$

## 4.2.3 非确定图灵机

**【例4-8】** 下面的图灵机就是不确定图灵机。无向图 $G$ 中, 从 $a$ 出发合法路径判定的不确定型图灵机。

$q_0$ 是初态,  $q_2$ 是终止状态.  
图中“a”表示一个字符.e.g. 字符串abe就是一条路径, 而bd则不是.  
这个例子主要说明: 该图灵机是不确定的.

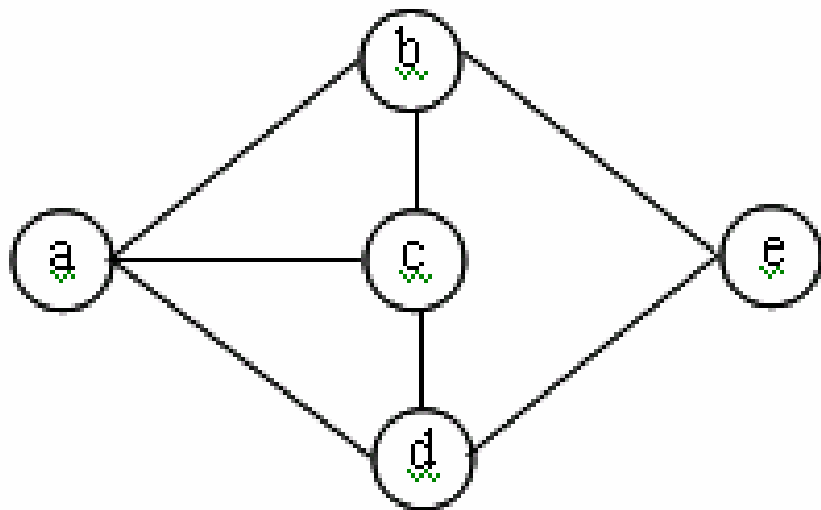


图 4-11 无向图  $G$

$$\delta(q_0, a) = (q_a, a, S)$$

$$\delta(q_0, b) = (q_b, b, S)$$

$$\delta(q_0, c) = (q_c, c, S)$$

$$\delta(q_0, d) = (q_d, d, S)$$

$$\delta(q_0, e) = (q_e, e, S)$$

$$\delta(q_a, a) = (q_b, a, R)$$

$$\delta(q_a, a) = (q_c, a, R)$$

$$\delta(q_a, a) = (q_d, a, R)$$

$$\delta(q_a, B) = (q_1, B, L)$$

$$\delta(q_b, b) = (q_a, b, R)$$

$$\delta(q_b, b) = (q_c, b, R)$$

$$\delta(q_b, b) = (q_e, b, R)$$

$$\delta(q_b, B) = (q_1, B, L)$$

$$\delta(q_c, c) = (q_a, c, R)$$

$$\delta(q_c, c) = (q_b, c, R)$$

$$\delta(q_c, c) = (q_e, c, R)$$

$$\delta(q_c, B) = (q_1, B, L)$$

$$\delta(q_d, d) = (q_a, d, R)$$

$$\delta(q_d, d) = (q_c, d, R)$$

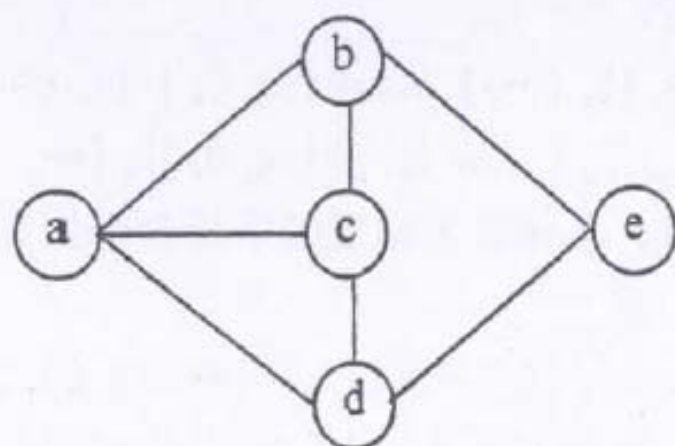


图 4-11 无向图 G

$$\delta(q_d, B) = (q_1, B, L)$$

$$\delta(q_e, e) = (q_b, e, R)$$

$$\delta(q_e, e) = (q_d, e, R)$$

$$\delta(q_e, B) = (q_1, B, L)$$

$$\delta(q_1, a) = (q_1, a, L)$$

$$\delta(q_1, b) = (q_1, b, L)$$

$$\delta(q_1, c) = (q_1, c, L)$$

$$\delta(q_1, d) = (q_1, d, L)$$

$$\delta(q_1, e) = (q_1, e, L)$$

$$\delta(q_1, B) = (q_2, B, R)$$

## 4.2.3 非确定图灵机

**定理4-3** 如果语言 $L$ 被一个非确定图灵机 $M_1$ 接受, 则 $L$ 将被某个确定图灵机 $M_2$ 接受。

## 4.2.4 多头图灵机

- 一个 $k$ 头图灵机有 $k$ 个读头，一个控制器和一条带，读头由1到 $k$ 编号，图灵机的一个动作由当前状态和被每个读头所扫描的符号。在一个动作中，每个读头独立地左移、右移或不动。
- **定理4-4** 如果 $L$ 被某个 $k$ 个读头的图灵机接受，则它能被一个单头图灵机接受。



## 4.2.5 多维图灵机

多维图灵机具有通常的有限控制器，但带却由 $k$ 维单元阵列组成。这里，在所有 $2k$ 个方向上（ $k$ 个轴，每轴正、负两个方向），都是无限的，根据状态和扫视的符号，该装置改变状态，打印一个新的符号，在 $2k$ 个方向上移动它的读头，开始时，输入沿着一个轴排列，读头在输入的左端。

## 4.2.6 离线图灵机

**定理4-5** 如果 $L$ 被一个二维图灵机 $M_1$ 接受，  
那么 $L$ 将被一个一维图灵机 $M_2$ 接受。

## 4.2.7 图灵机的组合

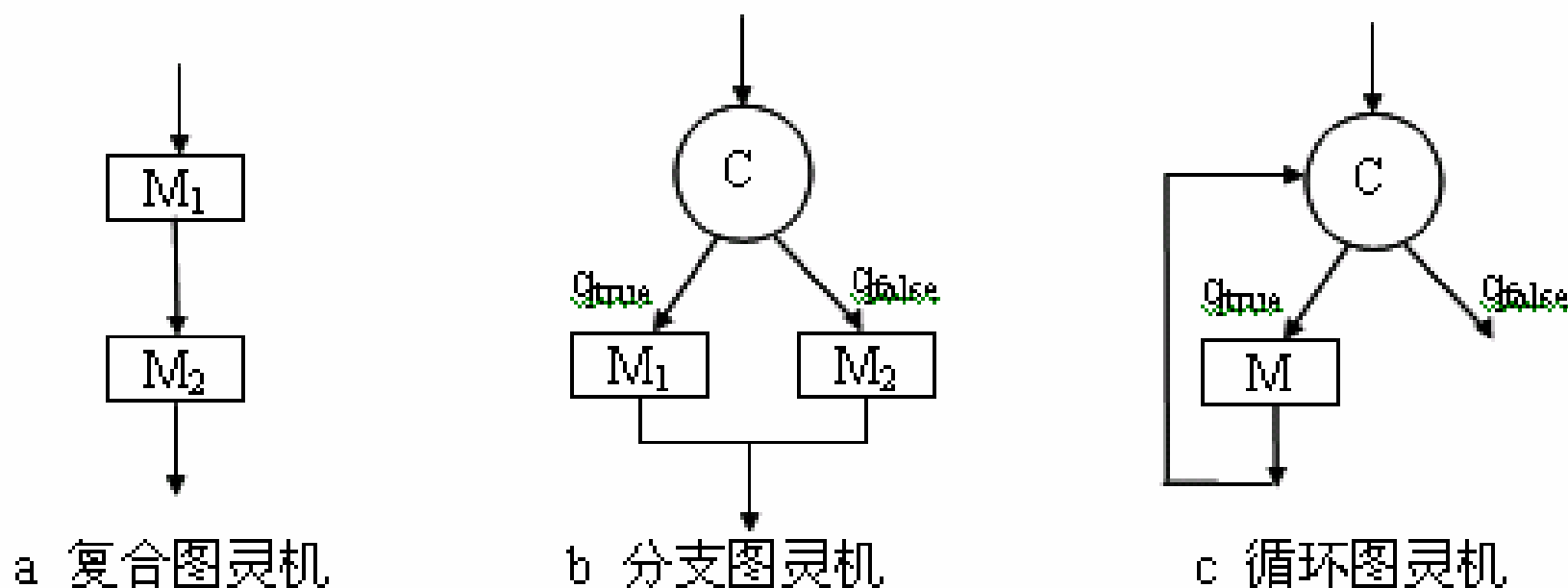


图 4-15 多个图灵机的组合

写个分支语句的图灵程序: if ( $V \geq 1$ ) then  $M_1$  else  $M_2$ . (笔记本4)

## 4.2.7 图灵机的组合

【例4-9】设计一个TM，完成乘法运算 $m \times n$ 。开始时带上符号为： $0^m 1 0^n 1$ ，结束时带上符号为 $0^{mn}$ ，用子程序调用的方式完成。

设计思想是：每当抹去左边一个0，就在第二个1后面拷贝n个0。当第一个1的左边全变为B时，带上就为 $10^n 10^{mn}$ ，再抹去 $10^n 1$ ，带上就剩 $0^{mn}$ ，即为所求。

设计Copy子程序 这个子程序完成在第二个1后面拷贝n个0的操作。

第1次被调用

开始ID:  $B 0^{m-1} 1 q_1 0^n 1$

结束ID:  $B 0^{m-1} 1 q_5 0^n 10^n$

第i次被调用

开始ID:  $B^i 0^{m-i} 1 q_1 0^n 10^{(i-1)n}$

结束ID:  $B^i 0^{m-i} 1 q_5 0^n 10^{in}$

在拷贝时，每当将一个0变成2，就在末尾写个0。当 $0^n$ 变为 $2^n$ 时，就已在右边加了 $0^n$ 。再将2变为 $0^n$ 。

## 4.2.7 Copy子程序

在拷贝时，每当将一个0变成2，就在末尾写个0。当 $0^n$ 变为 $2^n$ 时，就已在右边加了 $0^n$ 。再将2变为 $0^n$ 。

Copy=({q1,q2,q3,q4,q5},{0,1},{0,1,2,B},q1, $\delta$ ,{q5},B)

(q1,0,q2,2,R);

(q1,1,q4,1,L);

(q2,0,q2,0,R);

(q2,1,q2,1,R);

(q2,B,q3,0,L);

(q3,0,q3,0,L)

(q3,1,q3,1,L);

(q3,2,q1,2,R);

(q4,1,q5,1,R);

(q4,2,q4,0,L);

## 4.2.7 Copy子程序

在拷贝时，每当将一个 0 变成 2，就在末尾写个 0。当  $0^n$  变为  $2^n$  时，就已在右边加了  $0^n$ 。再将  $2^n$  变为  $0^n$ 。子程序设计为：

$$\text{Copy} = (\{q_1, q_2, q_3, q_4\}, \{0, 1\}, \{0, 1, 2, B\}, \delta, q_1, B, \{q_5\})$$

表 4-5 图灵机 Copy 的转移函数定义

	0	1	2	B
$Q_1$	$(q_2, 2, R)$	$(q_4, 1, L)$		
$Q_2$	$(q_2, 0, R)$	$(q_2, 1, R)$		$(q_3, 0, L)$
$Q_3$	$(q_3, 0, L)$	$(q_3, 1, L)$	$(q_1, 2, R)$	
$Q_4$		$(q_5, 1, R)$	$(q_4, 0, L)$	
$Q_5$				

## 4.2.7 图灵机的组合

设计主  $T \cup M$

$$M = \{q_0, q_6, q_7, q_8, q_9, q_{10}, q_{11}, q_{12}, q_{13}\}, \{0, 1\}, \{0, 1, 2, B\}, \delta, q_0, B, \{q_{13}\}$$

开始ID为  $q_0 0^m 1 0^n 1$ , 进入Copy入口ID为  $B 0^{m-1} 1 q_1 0^n 1$ , 为此,

$$\delta(q_0, 0) = (q_6, B, R)$$

$$\delta(q_6, 0) = (q_6, 0, R)$$

$$\delta(q_6, 1) = (q_1, 1, R)$$

从Copy结束时的ID, 进入主TM的开始ID

$$(B 0^{m-1} 1 q_5 0^n 1 0^n \vdash B q_0 0^{m-1} 1 0^n 1 0^n)$$

$$\delta(q_5, 0) = (q_7, 0, L)$$

$$\delta(q_7, 1) = (q_8, 1, L)$$

$$\delta(q_8, 0) = (q_9, 0, L) \quad /* (q_8, 0) \text{ 是正常乘法, } (q_8, B) \text{ 则转入善后} */$$

$$\delta(q_9, 0) = (q_9, 0, L) \quad /* q_9 \text{ 的作用是 } (q_9, B) \text{ 可回到主程序开始} */$$

$$\delta(q_9, B) = (q_0, B, R)$$

善后处理:  $B^m 1 q_5 0^n 1 0^{mn}$

善后处理:  $B^m 1 q_5 0^n 1 0^{mn} \xrightarrow{*} 0^{m+n}$

$$\delta(q_8, B) = (q_{10}, B, R)$$

$$\delta(q_{10}, 1) = (q_{11}, B, R)$$

$$\delta(q_{11}, 0) = (q_{11}, B, R)$$

$$\delta(q_{11}, 1) = (q_{12}, B, R)$$

$$\delta(q_{12}, 0) = (q_{12}, 0, R)$$

$$\delta(q_{12}, B) = (q_{13}, B, L)$$



## 4.2.8 枚举器

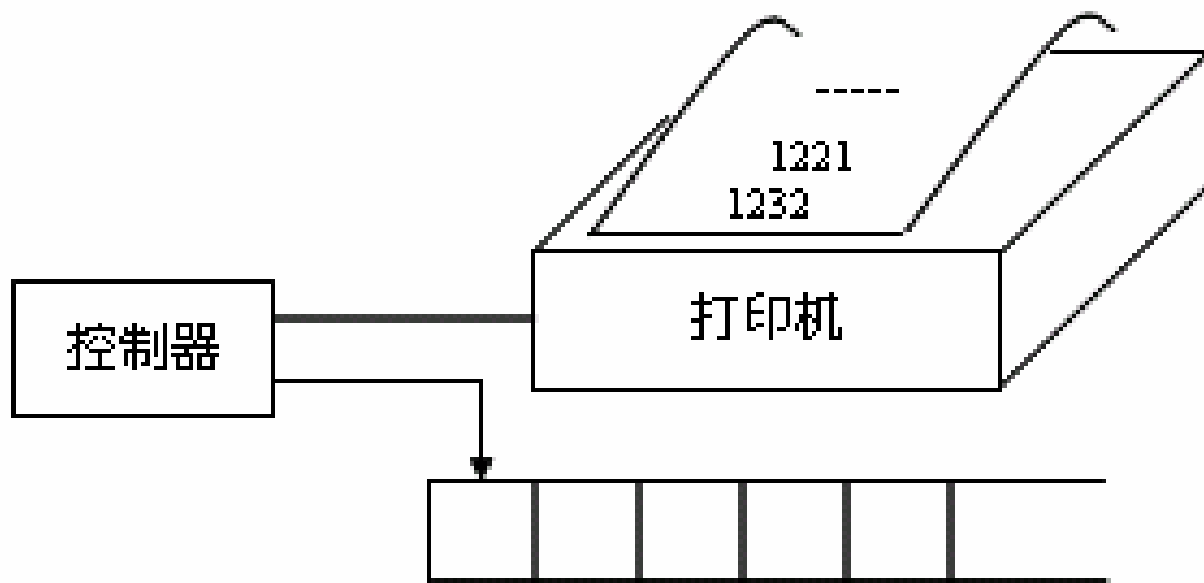


图 4- 16 枚举器构造

## 4.2.8 枚举器

**定理4-6** 一个语言是图灵可识别的，当且仅当有枚举器枚举它。

证明：首先证明如果有枚举器 $E$ 枚举语言 $L$ ，则存在图灵机 $M$ 识别 $L$ 。构造 $M$ 如下：

对于任意输入串 $w$ ，运行 $E$ 。每当 $E$ 输出一个串时，与 $w$ 比较，若相等，接受 $w$ ，并停机。

显然， $M$ 接受在 $E$ 输出序列中出现过的那些串。

现在证明若有图灵机 $M$ 识别语言 $L$ ，则有枚举器 $E$ 枚举 $L$ 。

设 $L=\{w_1, w_2, w_3, \dots\}$ ，构造 $E$ 如下：

对 $i=1,2,3,\dots$ 执行如下步骤

(1) 对 $w_1, w_2, w_3, \dots, w_i$ 中的每一串，让 $M$ 以其作为输入运行 $i$ 步。

(2) 若在这个计算中， $M$ 接受 $w_j$ ，则打印 $w_j$ ，

$M$ 接受 $w$ ，它最终将出现在 $E$ 的枚举打印中。事实上，它可能在 $E$ 的列表中出现无限多次，因为每一次重复运行， $M$ 在每一个串上都从头开始运行。

# 通用图灵机U

- 可以构造通用图灵机U，它可以模拟任意一个图灵机P，只要把P的五元组集合和输入值告知U即可。
- 即有： $U("x", "P") = P("x")$ 。

原始通用图灵机

## 4.3 通用图灵机

(1) 缓冲域 带的最左面是标记符  $A$ ,  $A$  的右边是  $|K|+|\Gamma|+2$  个单元构成的缓冲 ( $|K|$ 、 $|\Gamma|$  分别是状态集和字母集的元素数目)。此域放当前状态和当前字符。

(2)  $M$  的描述字域 缓冲区域右边紧接的是  $M$  的描述字  $d_M$ , 以  $B$  为开始标志, 以 3 个 0 结束。对于转移函数

$$\delta(q,a)=(q',a',d),$$

我们用五元组表示为  $(q, a, q', a', d)$ , 将顺序调整为  $(q, a, a', d, q')$ 。 $d_M$  就是由这样的五元组组成的序列。对于每个五元组中的状态和字符, 我们用其序号的一进制表示, 其间用 0 分隔, 五元组间用 2 个 0 分隔。例如: 若有  $\delta(q_2, 0)=(q_3, 2, L)$ , 表示成上面定义的五元组是  $(q_2, 0, 2, L, q_3)$ , 再用其序号表示为  $(3, 1, 3, 1, 4)$ , 在  $U_2$  的带上表示为

00111010111010111100.....

(3)  $M$  的输入描述域  $M$  的描述字域的右边存放的是输入串的编码: 用字母的序号表示该字母, 并用 0 分隔。该域以  $C$  为起始标志。如输入为 111, 则该串表示为:

110110110

## 4.3 通用图灵机

UTM U2模拟  $T M$   $M$  具体步骤如下:

- 步0 将读头置于描述字区域的开始符号  $B$  上。此时缓冲区域是  $0$ ;
- 步1 U2复写标记  $B$  或  $E$  后面的状态到缓冲域的初始位置, 然后在其后面加一个辅助符号  $D$ , 读写头右移到符号  $C$ ;
- 步2 U2复写  $C$  后面的初始带符号复制到缓冲区部分, 即在  $D$  之后;
- 步3 U2将  $D$  改为  $0$ , 现缓冲区中包含当前状态和扫描的字母;
- 步4 对描述字域进行搜索, 查看前两项匹配的元组。若所有的五元组都不匹配, 则表明  $M$  停机,  $U2$  也停机。若找到一个匹配的五元组, 标记  $D$ 、 $E$  表示正在比较的状态或符号;

## 4.3 通用图灵机

- 步5 (找到匹配的五元组)删去缓冲区的内容, 并把标记  $E$  右移一个符号, 使  $E$  处于该五元组的新符号的前面;
- 步6 用新符号代替标记  $C$  后面的符号(可能需扩充或压缩原输入描述域),  $U2$ 将标记  $E$  右移至五元组的方向分量前;
- 步7  $U2$ 计数方向分量中  $1$  的个数, 然后按要求向左或向右移动标记  $C$ 。若  $C$  向左离开了输入串, 则  $U2$  停机; 若  $C$  向右离开了输入串,  $U2$  必须添加一个表示下一个方格的新的“0”串。  $U2$ 将标记  $E$  右移至五元组新的状态分量前, 当前  $U2$  描述字域标记  $E$  后面不是结束状态, 则转至步1; 否则接受输入串并停机。

## 4.3 通用图灵机

[BACK](#)

表 4.7 通用图灵机 U2 模拟 M 的过程

S0	A00000000	<u>B</u> 1010101101100101110111010111001101101101101000	C10110
S1	A1D000000	B1010101101100101110111010111001101101101101000	<u>C</u> 10110
S2	A1D100000	B1010101101100101110111010111001101101101101000	C10110
S3	A10100000	B1010101101100101110111010111001101101101101000	C10110
S4	A1D100000	B <u>1E</u> 10101101100101110111010111001101101101101000	C10110
S5	A00000000	B101 <u>E</u> 101101100101110111010111001101101101101000	C10110
S6	A00000000	B10101 <u>E</u> 1101100101110111010111001101101101101000	C10110
S7	A00000000	B10101011 <u>E</u> 1100101110111010111001101101101101000	01C110
S1	A11D00000	B1010101101100101110111010111001101101101101000	01 <u>C</u> 110
S2	A11D11000	B1010101101100101110111010111001101101101101000	01C110
S3	A11011000	B1010101101100101110111010111001101101101101000	01C110
S4	A11D11000	B10101011011001011101110101110011 <u>E</u> 1101101101101000	01C110
S5	A00000000	B10101011011001011101110101110011011 <u>E</u> 1101101000	01C110
S6	A00000000	B10101011011001011101110101110011011011 <u>E</u> 1101000	01C110
S7	A00000000	B10101011011001011101110101110011011011011 <u>E</u> 1000	01011C

【例 4-10】U2 模拟 TM  $M = (\{q_0, q_1, q_2, q_3\}, \{0, 1\}, \{0, 1, B\}, \delta, q_0, B, \{q_3\})$ , 其中  $\delta$  定义见表 4-6。

表 4-6 图灵机 M 的转移函数定义

	0	1	B
$q_0$	$(q_1, 0, R)$		$(q_2, B, L)$
$q_1$		$(q_0, 1, R)$	
$q_2$	$(q_2, 0, L)$	$(q_2, 0, L)$	$(q_3, B, R)$

- 该图灵机的功能是接收语言  $(01)^*$ 。
- 通用图灵机 U2 可以模拟任何一个图灵机，也就是说，U2 可以模拟所有算法。
- 通用图灵机描述了计算机的最根本的计算能力。



- 对于任何一个图灵机 $M$ ，都有一个确定的描述字 $d_M$ ，如在例4-10中

$d_M = 1010101101100101110111010111001$   
 $101101101101000$

- 将它看作一个二进制数，实质上，它是一个整数，因此按这种表示，任何一个图灵机都和一个整数相对应，这个整数，称为图灵机的标记（编码）。

## 4.4 图灵可计算性

- 4.4.1 图灵可计算性函数
- 4.4.2 图灵机的枚举定理
- 4.4.3 图灵机的计算限界

## 4.4.1 图灵可计算性函数

- 定义4-5 函数 $f(x_1, x_2, \dots, x_n)$ 是部分图灵可计算的，若有一图灵程序 $P$ ，使得

$$\psi_P(x_1, x_2, \dots, x_n) = f(x_1, x_2, \dots, x_n)$$

- 其中 $\psi_P(x_1, x_2, \dots, x_n)$ 是 $P$ 对应的函数，“=”表示若有定义，则值相等；否则，均无定义。

## 4.4.1 图灵可计算性函数

**定义4-6** 函数 $f(x_1, x_2, \dots, x_n)$ 是全函数，若它对一切 $x_1, x_2, \dots, x_n$ 都有定义。

**定义4-7** 函数 $f(x_1, x_2, \dots, x_n)$ 是图灵可计算函数，若它是部分图灵可计算的而且是全函数。

## 4.4.1 图灵可计算性函数

**引理 4-1** 如果  $g$  是  $n$  元图灵可计算函数, 则存在图灵机  $M$ , 它把带模式

$$0\bar{x}_1 0\bar{x}_2 0 \dots 0\bar{x}_n 0$$

变为

$$0\bar{x}_1 0\bar{x}_2 0 \dots 0\bar{x}_n 0 \overline{g(x_1, x_2, \dots, x_n)} 0$$

而且读头不会移到初始方格的左边。

**引理 4-2** 如果  $g_1, g_2, \dots, g_m$  是  $n$  元图灵机, 那么存在一个图灵机, 它把带模式

$$0\bar{x}_1 0\bar{x}_2 0 \dots 0\bar{x}_n 0$$

变成

$$0\bar{x}_1 0\bar{x}_2 0 \dots 0\bar{x}_n 0 \overline{g_1(x_1, x_2, \dots, x_n)} 0 \overline{g_2(x_1, x_2, \dots, x_n)} 0 \dots 0 \overline{g_m(x_1, x_2, \dots, x_n)} 0$$

而且读写头不会移到初始扫描方格的左边。

## 4.4.1 图灵可计算性函数

**定理4-7** 设 $g_1, g_2, \dots, g_m$ 是 $n$ 元图灵可计算函数,  $h$ 是 $m$ 元图灵可计算函数, 则复合函数

$$f = h \cdot (g_1, g_2, \dots, g_m)$$

也是图灵可计算的。

# 通用图灵机U

- 可以构造通用图灵机U，它可以模拟任意一个图灵机P，只要把P的五元组集合和输入值告知U即可。
- 即有： $U("x", "P") = P("x")$ 。

原始通用图灵机

## 4.4.2 图灵机的枚举定理

1. 图灵机的标记(编码)
2. 图灵可计算的**重要定理**

**定义4-8** 若 $z$ 是一个图灵机 $M$ 的标记, 并且 $M$ 计算部分函数 $g(x_1, x_2, \dots, x_n)$ , 则定义函数 $\Phi^{(n)}$ 为

$$\Phi^{(n)}(x_1, x_2, \dots, x_n, z) = g(x_1, x_2, \dots, x_n)$$

函数 $\Phi$ 称为**通用函数**, 它就是通用图灵机所计算的函数.



## 4.4.2 图灵机的枚举定理

**定理4-8 [枚举定理]** 对于每个自然数 $y$ , 部分函数 $\Phi^{(n)}(x_1, x_2, \dots, x_n, y)$ 是(部分)图灵可计算函数。

[证明]

- 它能枚举(一一列举)所有部分可计算函数:
- $\Phi^{(n)}(x_1, x_2, \dots, x_n, 0),$   
 $\Phi^{(n)}(x_1, x_2, \dots, x_n, 1),$   
 $\Phi^{(n)}(x_1, x_2, \dots, x_n, 2),$   
 $\Phi^{(n)}(x_1, x_2, \dots, x_n, 3), \dots$

证明结束。

r.e.集合的定义:  $B = \{ x \in \mathbb{N} \mid g(x) \downarrow \}$ , 其中  $g(x)$  是一个部分可计算函数。

记  $W_n = \{ x \in \mathbb{N} \mid \Phi(x, n) \downarrow \}$

则  $W_0, W_1, W_2, W_3, \dots$ , 一一列举了所有的r.e.集。

## 4.4.2 图灵机的枚举定理

**定理4-9 [迭代定理]** 对一切 $n$ , 有图灵可计算函数,  $S^{(n)}(z, x_1, x_2, \dots, x_n)$ , 使得对一切 $m$ , 有

$$\begin{aligned} \Phi^{(n+m)}(z, x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_m) = \\ \Phi^{(m)}(S^{(n)}(z, x_1, x_2, \dots, x_n), y_1, y_2, \dots, y_m) \end{aligned}$$

[证明]

## 4.4.2 图灵机的枚举定理

定义4-9 计步谓词 $STP^{(n)}$ 定义如下:

$STP^{(n)}(z, x_1, x_2, \dots, x_n, t)$  描述字为 $z$ 的图灵机程序 $M$ 对于输入 $x_1, x_2, \dots, x_n$ 在 $t$ 步内停机。其中“一步”定义为完成图灵机 $M$ 的状态转移函数的一个变换。

定理4-10[计步定理] 对任意 $n$ , 谓词 $STP^{(n)}(z, x_1, x_2, \dots, x_n, t)$ 是图灵可计算的。

[证明]

## 4.4.3 图灵机的计算限界

$$g(x) = \begin{cases} \Phi^{(n)}(x, x) + 1 & \text{如果 } \Phi^{(n)}(x, x) \downarrow \\ 0 & \text{如果 } \Phi^{(n)}(x, x) \uparrow \end{cases}$$

	0	1	2	3
0	<u><math>\Phi^{(1)}(0, 0)</math></u> $\Phi^{(1)}(0, 1)$ $\Phi^{(1)}(0, 2)$		$\Phi^{(1)}(0, 3) \dots$	
1	$\Phi^{(1)}(1, 0)$ <u><math>\Phi^{(1)}(1, 1)</math></u> $\Phi^{(1)}(1, 2)$		$\Phi^{(1)}(1, 3) \dots$	
2	$\Phi^{(1)}(2, 0)$ $\Phi^{(1)}(2, 1)$ <u><math>\Phi^{(1)}(2, 2)</math></u>		$\Phi^{(1)}(2, 3) \dots$	
3	$\Phi^{(1)}(3, 0)$ $\Phi^{(1)}(3, 1)$ $\Phi^{(1)}(3, 2)$		<u><math>\Phi^{(1)}(3, 3)</math></u> $\dots$	
	$\dots$	$\dots$		

表 4-8 图灵机类中一元函数阵列

例： 函数 **$g(x)$** 不是图灵可计算的。

证明： 采用反证法：

假设  $g$  可以被图灵机类的一个一元函数计算，例如  $\Phi^{(1)}(z, x)$ ，则有

$$g(x) = \Phi^{(1)}(z, x)$$

当  $x=z$  时，若  $\Phi^{(1)}(z, z)$  有定义，有  $g(z) = \Phi^{(1)}(z, z) = \Phi^{(1)}(z, z) + 1$ ，矛盾；若  $\Phi^{(1)}(z, z)$  无定义，则  $g(z) = 0$ ，与  $\Phi^{(1)}(z, z)$  不同，矛盾。

由反证法可知，函数 **$g(x)$** 是没有图灵机来计算的，故得到  
 **$g(x)$ 不是图灵可计算**的结论。

## 4.4.3 图灵机的计算限界

0. 停机函数  $\text{Halt}(x,y)$  是不可计算的. ([link](#))

1. 对角线域函数

$$d(x) = \begin{cases} 1 & \text{如果 } \Phi^{(1)}(x, x) \downarrow \\ 0 & \text{如果 } \Phi^{(1)}(x, x) \uparrow \end{cases}$$

铺砖问题(Domino Problem, Wang Tile Problem) [>>](#)

## 4.4.3 图灵机的计算限界

**定理 4-10** 对角线域函数  $d$  不是图灵可计算的。

证明[方法 1]: 假设函数  $d$  是图灵可以计算的, 那么函数  $\hat{d}$  也可应该是可计算的, 其中

$$\hat{d}(x) = \begin{cases} 0 & \text{如果 } d(x)=0 \\ \uparrow & \text{如果 } d(x) \neq 0 \end{cases}$$

令  $z$  是函数  $\hat{d}$  的标记, 则有  $\hat{d}(z) = \Phi^{(1)}(z, z)$ , 而

$$\hat{d}(z) = \begin{cases} 0 & \text{如果 } d(z)=0 \\ \uparrow & \text{如果 } d(z) \neq 0 \end{cases} = \begin{cases} 0 & \text{如果 } \Phi^{(1)}(z, z) \uparrow \\ \uparrow & \text{如果 } \Phi^{(1)}(z, z) \downarrow \end{cases}$$

因此  $\Phi^{(1)}(z, z) \downarrow$  当且仅当  $\Phi^{(1)}(z, z) \uparrow$ , 即  $\hat{d}(z)$  有定义当且仅当  $\hat{d}(z)$  无定义。由此矛盾推得  $d$  不可计算。



## 4.4.3 图灵机的计算限界

证明[方法 2]: 假设函数  $d$  是图灵可计算的, 且计算  $d$  的图灵机为  $M_1$ 。修改  $M_1$  成为新的计算  $\hat{d}$  的图灵机  $M_2$ , 具体动作如下:  $M_2$  开始时象  $M_1$  一样动作, 如果  $M_1$  输出 0, 那么  $M_2$  也同样输出 0 并停机; 但如果  $M_1$  输出 1, 则  $M_2$  向右移动不停机。显然  $M_2$  计算函数  $\hat{d}$ 。现设  $z$  是  $M_2$  的一个标记, 并考虑输入带模式为  $\bar{z}$  时的情况。由  $M_2$  的设计可知:  $M_2$  停机并输出 0 当且仅当  $M_1$  输出 0, 当且仅当图灵机  $M_2$  未能输出一个 0 (由假设), 因此输入为  $\bar{z}$  时  $M_2$  停机并输出 0, 当且仅当输入为  $\bar{z}$  时  $M_2$  不输出 0, 矛盾, 所以函数  $d$  不可能是图灵可计算的。

### 4.4.3 图灵机的计算限界

$$h(x) = \begin{cases} 1 & \text{如果输入为 } \bar{x} \text{ 时, 图灵机 } M_x \text{ 停机} \\ 0 & \text{如果输入为 } \bar{x} \text{ 时, 图灵机 } M_x \text{ 不停机} \end{cases}$$

## 4.4.3 图灵机的计算限界

**定理4-11** 对角线停机函数 $h$ 不是图灵可计算的。

[证明]

**定理4-12** 空带停机函数 $h_0$ 不是图灵可计算的，其中

$$h_0(x) = \begin{cases} 1 & \text{如果图灵机 } M_x \text{ 始于空带时停机} \\ 0 & \text{否则} \end{cases}$$

[证明]

## 4.4.3 图灵机的计算限界

证明：对于每个自然数  $x$ ，构造图灵机  $W_x$ ，它的初始带是空的， $W_x$  的动作如下：

步 1 在空带上，写入  $x+1$  个 1；

步 2 执行  $M_x$  动作。

其中步 S1，可用  $x+1$  个状态生成  $x+1$  个 1。

假设  $h_0$  可由图灵机  $M1$  计算，设计图灵机  $M2$ ，其操作如下：当输入为  $\bar{x}$  时， $M2$  完成

步 1 计算  $W_x$  的单带通用机上五元组的标记  $z$ ，并把它转变为一进制形式  $\bar{z}$ ；

步 2 执行  $M1$ ，此时带上的输入是  $W_x$  的标记。

由上面构造的图灵机，我们有：当输入带模式为  $\bar{x}$  时， $M2$  在带上输出 1，当且仅当  $M1$  在输入带模式为  $W_x$  标记时，输出 1，当且仅当  $W_x$  始于空带时停机，当且仅当输入带模式为  $\bar{x}$  时， $M_x$  停机，即当输入带模式为  $\bar{x}$  时， $M2$  在带上输出 1，当且仅当  $M_x$  停机。又因为  $M2$  是计算对角线停机函数  $h$ ，而已知  $h$  不是图灵可计算的，故产生矛盾，由此推得  $h_0$  不是图灵可计算的。

## 4.4.3 图灵机的计算限界

### 2. 完全性函数

各种完全函数是不可计算的，例如考虑一元完全函数  $t$ ， $t$  定义为

$$t(x) = \begin{cases} 1 & \text{如果以 } x \text{ 为标记的一元图灵部分可计} \\ & \text{算函数是完全的} \\ 0 & \text{否则} \end{cases}$$

这个函数就不是图灵可计算的。

**定理 4-13** 一元完全函数  $t(x)$  不是图灵可计算的。

证明：假设  $t$  是图灵可计算的，那么对于任意  $x$ ，当将  $x$  作为标记时，对于任意  $y$ ， $\Phi^{(1)}(x, y)$  是否有定义是可以判定的，这也就意味着  $\Phi^{(1)}(x, x)$  是否有定义是可以判定的，即对角线域函数是可以计算的，这与对角线域函数不是图灵可计算的结论相悖，所以  $t$  不可能是图灵可计算的。

## 4.5 部分递归函数及其与图灵机的等价性

## 4.5.1 函数的复合算子、递归算子、取极小值算子

■ 定义1 设 $f$ 是 $m$ 元函数,  $g_1, g_2, \dots, g_m$ 是 $n$ 元函数, 函数 $h$ 定义为

$$y = h(x_1, x_2, \dots, x_n) = f(g_1(x_1, x_2, \dots, x_n), g_2(x_1, x_2, \dots, x_n), \dots, g_m(x_1, x_2, \dots, x_n))$$

则称函数 $h$ 是函数 $f$ 和 $g_1, g_2, \dots, g_m$ 的复合函数, 记为 $h = f \cdot (g_1, g_2, \dots, g_m)$ 。

## 4.5.1 函数的复合算子、递归算子、取极小值算子

- 定义2 设 $g$ 、 $h$ 、 $f$ 分别是 $n$ 元、 $n+1$ 元、 $n+2$ 元函数，函数 $h$ 的定义为

$$\begin{cases} h(0, x_1, x_2, \dots, x_n) = g(x_1, x_2, \dots, x_n) \\ h(z', x_1, x_2, \dots, x_n) = f(h(z, x_1, x_2, \dots, x_n), z, x_1, x_2, \dots, x_n) \end{cases}$$

- 则称函数 $h$ 是原始递归于 $g$ 、 $f$ 的原始递归函数。又称 $h$ 是依据 $z$ 用原始递归运算得到的， $z$ 是递归变量， $x_1, x_2, \dots, x_n$ 是原始递归参数。



## 4.5.1 函数的复合算子、递归算子、取极小值算子

■ **定义3** 设 $f$ 、 $h$ 是 $n+1$ 元、 $n$ 元函数，且 $f$ 全函数，函数 $h$ 的定义为

$$y = h(x_1, x_2, \dots, x_n) = \min\{z \mid f(x_1, x_2, \dots, x_n, z) = 0\} \\ z \geq 0$$

则称函数 $h$ 是取极小算子作用于函数 $f$ 的结果。

## 4.5.1 函数的复合算子、递归算子、取极小值算子

- **定义4** 函数  $f(x_1, x_2, \dots, x_{n+1})$  是正则函数，若对于任何  $x_1, x_2, \dots, x_n$ ，恒存在  $z$  ( $z \geq 0$ )，使得

$$f(x_1, x_2, \dots, x_n, z) = 0$$

## 4.5.2 原始递归函数

### ■ 三个基本函数

常值函数:  $C_m^{(n)}(x_1, x_2, \dots, x_n) = m$

投影函数:  $I_m^{(n)}(x_1, x_2, \dots, x_n) = x_m$

后继函数:  $S(x) = x + 1$

## 4.5.2 原始递归函数

**定义 5** 原始递归函数定义是:

(i) 常值函数、投影函数和后继函数是初始的原始递归函数;

(ii) 如果  $\psi, \xi$  是原始递归函数,  $\varphi$  原始递归于  $\psi$  与  $\xi$ , 则  $\varphi$  是原始递归函数;

(iii) 如果  $\psi(x_1, x_2, \dots, x_n), x_1(y_1, y_2, \dots, y_m), x_2(y_1, y_2, \dots, y_m), \dots, x_n(y_1, y_2, \dots, y_m)$  是原始递归函数, 则

$$\varphi(x_1, x_2, \dots, x_n) = \psi(x_1(y_1, y_2, \dots, y_m), x_2(y_1, y_2, \dots, y_m), \dots, x_n(y_1, y_2, \dots, y_m))$$

是原始递归函数。

(iv) 有限次使用 (i) ~ (iii) 得到的函数, 是原始递归函数。

## 4.5.2 原始递归函数

**【例 1】** 求和函数  $\text{add}(x, y) = x + y$  是原始递归函数  
因为  $\text{add}(x, y)$  可以写成

$$\begin{cases} \text{add}(x, 0) = I_1^{(1)}(x) = x \\ \text{add}(x, y+1) = S(I_1^{(3)}(\text{add}(x, y), y, x)) = \text{add}(x, y) + 1 \end{cases}$$

## 4.5.2 原始递归函数

【例 2】 乘积函数  $\text{mul}(x, y) = x * y$  是原始递归函数  
函数  $\text{mul}(x, y)$  的递归式可写成

$$\begin{cases} \text{mul}(x, 0) = C_0^{(1)}(x) = 0 \\ \text{mul}(x, y+1) = \text{add}(I_1^{(3)}(\text{mul}(x, y), y, x), I_3^{(3)}(\text{mul}(x, y), y, x)) = \text{mul}(x, y) + x \end{cases}$$

## 4.5.2 原始递归函数

**【例 3】**  $\text{fac}(x) = x!$  是原始递归函数  
因为函数  $\text{fac}(x)$  可以写成如下递归式：

$$\begin{cases} \text{fac}(0) = C_1^{(0)}(0) = 1 \\ \text{fac}(x+1) = \text{mul}(\text{fac}(x), S(x)) = \text{fac}(x, y) * (x+1) \end{cases}$$

所以函数  $\text{fac}(x)$  是原始递归函数。

## 4.5.4 部分递归函数及其与图灵机的等价性

**定理4-9** 部分图灵可计算的函数是部分递归函数。

证明：首先将这个过程的数字编码。

若  $|\Gamma|=l$ ，设  $\Gamma=\{0,1,\dots,l-1\}$ 。

$$C(c_0c_1\dots c_k)=\sum_{i=0}^n c_i l^i$$

$$Q(T)=\{0,1,2,\dots,k-1\},$$

其中0表示初始状态，1表示接受状态，2是一新补的特殊的状态：永不停机，即当在原T中， $(q,a)$ 无定义时，进入2状态。



## 4.5.4 部分递归函数及其与图灵机的等价性

状态转移函数表示为

$$\delta(q,a) = (q(q,a), \text{char}(q,a), d(q,a))$$

其中 $q(q,a)$ ,  $\text{char}(q,a)$ ,  $d(q,a)$ 都是有限点有定义的函数。现将它们扩充为全函数。

若对于 $q,a$ 无定义, 则  $\delta(q,a)=2$ ,  $\text{char}(q,a)=a$ ,  $d(q,a)=S$ 。易见扩充后, 它们都是原始递归的。

# 部分递归函数及其与图灵机的等价性

其次考虑瞬时描述及其变化。

当运行至第  $t$  步时， $T$  的瞬时描述（见图 5-1）为：

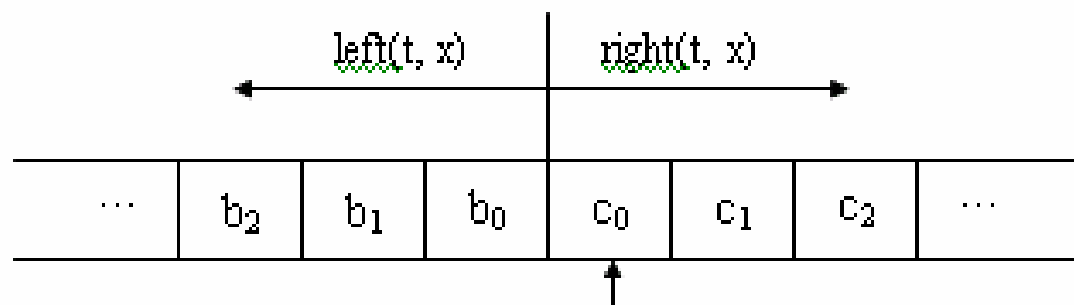


图 5-2 图灵机  $T$  在第  $t$  步时的带状态

$$\text{left}(t, x) q(t, x) \text{right}(t, x)$$

其中

$\text{left}(t, x) = \sum b_i \cdot i^t$ ，表示第  $t$  步时带头左面的字符串的编码；

$\text{right}(t, x) = \sum c_i \cdot i^t$ ，表示第  $t$  步时带头开始向右的字符串

编码；

$q(t, x)$  表示第  $t$  步时的状态，它初始时为 0，即  $q(0, x) = 0$ 。第  $t$  步时带头字母为  $c_0 = r(\text{right}(t, x), 1)$ ，其中  $r(m, n)$  表示  $m$  除以  $n$  的余数；带头右边的字母  $b_0 = r(\text{left}(t, x), 1)$ 。

至  $t+1$  步时，有

$$q(t+1, x) = Q(q(t, x), c_0)$$

$$\text{left}(t+1, x) = \begin{cases} [\text{left}(t, x) / l] \end{cases}$$

$$\text{left}(t+1, x) = \begin{cases} \text{left}(t, x) \end{cases}$$

$$\text{left}(t+1, x) = \begin{cases} \text{left}(t, x) \cdot l + \text{char}(q(t, x), c_0) \end{cases}$$

若带头左移

若带头不动

若带头右移

1.2.9

$$\text{right}(t+1, x) = \begin{cases} ([\text{right}(t, x) / l] \cdot l + \text{char}(q(t, x), c_0)) \cdot l + b_0 \\ [\text{right}(t, x) / l] \cdot l + \text{char}(q(t, x), c_0) \\ [\text{right}(t, x) / l] \end{cases}$$

若带头左移

若带头不动

若带头右移

若在第  $t'$  步时,  $T$  进入接受状态, 即  $q(t', x)=1$ , 则  $t'=\min_{t \geq 0} (q(t, x)=1)$ 。此时输出为  $\text{right}(t', x)$ , 即

$$y=T(x)=\text{right}(\min_{t \geq 0} (q(t, x)=1), x)$$

显然这是一个部分递归函数。这表明图灵机  $T$  计算的函数是部分递归的, 定理得证。

- **引理4-1** 设 $C$ 是 $\Sigma(T)^*$ 上图灵可实现的编码, 则存在图灵机 $T'$ ,  $\Sigma(T')=\{a\}$ ,  $T'$ 完成译码:  $\forall a \in N$ ,  $T'(a^x) = C^{-1}(x)$ 。
- **定理4-10** 部分递归函数都是部分图灵可计算的函数。

- **定理4-11** 对任意正整数 $n$ , 都存在 $n+1$ 元部分递归函数 $\varphi(x_1, x_2, \dots, x_n, y)$ , 使得对于任意一个 $n$ 元部分递归函数 $f(x_1, x_2, \dots, x_n)$ , 都存在一个自然数 $e$ , 满足

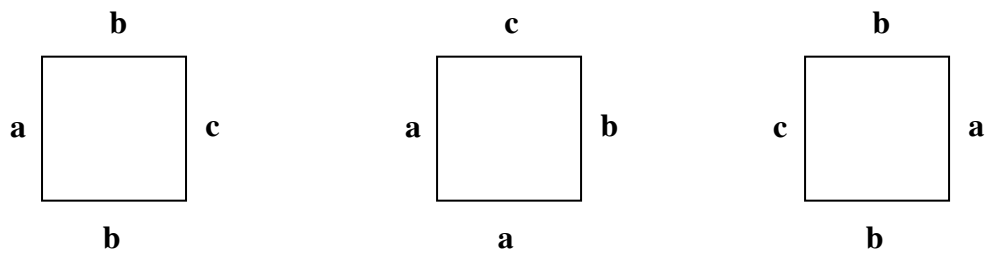
$$f(x_1, x_2, \dots, x_n) = \varphi(x_1, x_2, \dots, x_n, e)$$

## 2.2 Domino problem ( Wang tiles ) 铺砖问题

Kahr, Moore, Wang的“惊人的”结果，来源于Wang H 1961年的思路，他提出了著名的Domino Problem （Robinson称之为Wang Tiles[Robinson 1978]，中文叫“铺砖问题”或“骨牌游戏”，见Wang H, 1961; Wang H, 1983）。后来的发展证明，**Domino Problem**与**Turing机**等价，并且已经成为将逻辑公式与图灵机联系起来的方便工具（Borger,2001书[5]），Borger, Gradel, Gurevich的著名著作《The Classical Decision Problem》通篇使用Domino Problem的方法证明逻辑公式类的规约性和给出公式类可满足性判断的复杂度上下界。

原始的 **Domino Problem** 定义如下：

给定了一个由方型骨牌组成的有穷集合，这些骨牌大小相同(比如，全都具有单位面积)而且不同的方块的边上以不同的方式着了色。再假定每一骨牌（骨牌型）都有**无穷多个样品**。不允许旋转或翻转一骨牌，这样，例如下面的三个骨牌是不同的类型：

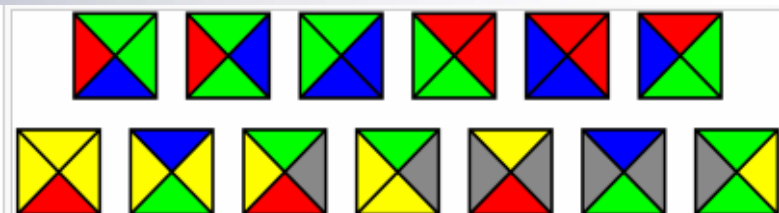



一骨牌集称为(在无穷平面上)可解的，当且仅当使用骨牌集中的骨牌能铺盖整个平面（铺砖的原则是相邻的边有相同的颜色）。

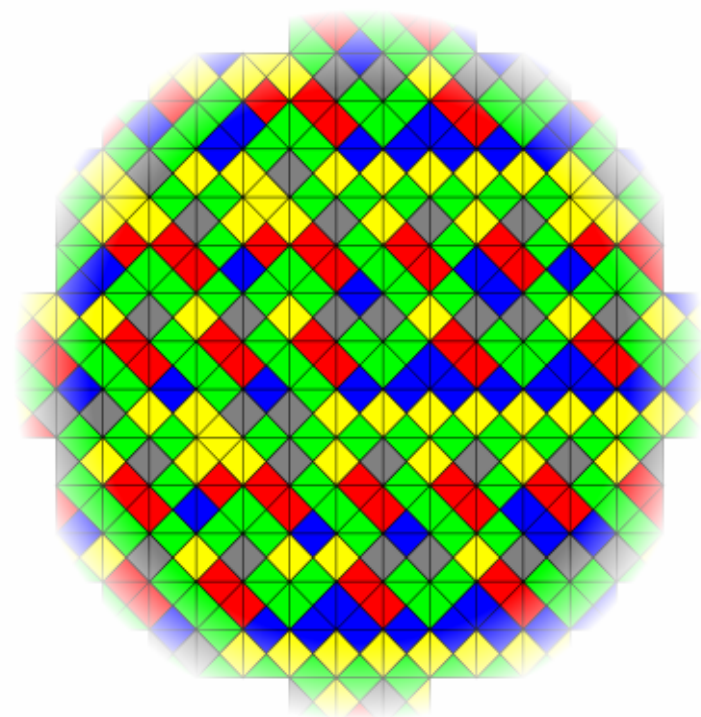
(Example)



**Wang tiles** (or **Wang dominoes**), first proposed by mathematician, logician, and philosopher **Hao Wang** in 1961, are a class of **formal systems**. They are modelled visually by equal-sized squares with a color on each edge which can be arranged side by side (on a regular square grid) so that abutting edges of adjacent tiles have the same color; the tiles cannot be rotated or reflected. The basic question about a set of Wang tiles is whether it can **tile** the plane or not, i.e., whether



This set of 13 Wang tiles will  tile the plane but only **aperiodically**.



一个Domino Problem模拟Turing Machine的例子。取自(Boas,1996[10])。

$(q,0)$	1	0	1	1	$\square$
0	$(q,1)$	0	1	1	$\square$
0	1	$(q,0)$	1	1	$\square$
0	1	0	$(q,1)$	1	$\square$
0	1	0	1	$(q,1)$	$\square$
0	1	0	1	1	$(q,\square)$
0	1	0	1	$(p,1)$	$\square$
0	1	0	$(p,1)$	0	$\square$
0	1	$(p,0)$	0	0	$\square$
0	1	1	0	0	$\square$

Figure 1: Time Space diagram of a computation by the successor machine

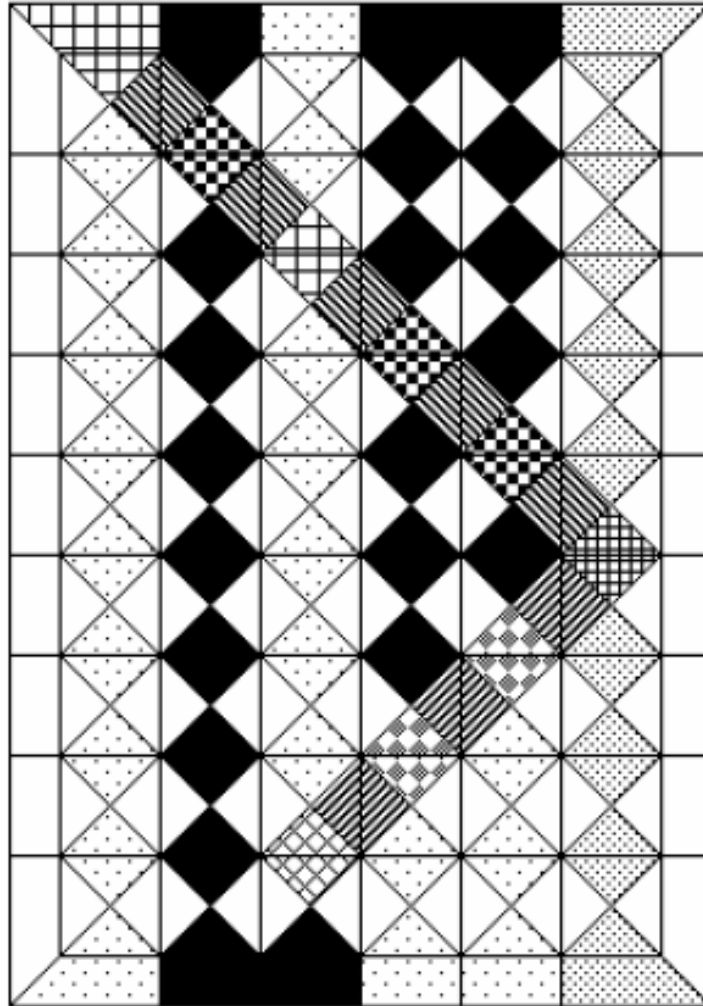


Figure 6: A tiling representing the computation  $11 + 1 = 12$

[BACK](#)

## 4.5 习题

1. 设计一图灵机，计算整数函数  $y=\max(x_1, x_2, \dots, x_n)$ ，其中函数  $\max$  是求最大值函数。
2. 设计一个三带图灵机，进行两个二进制自然数的加法和乘法。
3. 证明：存在一完全的一元图灵可计算函数  $k$ ，使得对所有  $x$ ,

$$\Phi^{(1)}(k(x), y) = C_x^{(1)}(y)$$

其中常值函数  $C_x^{(1)}(y) = x$ .

4. 证明关于图灵机的递归定理：设  $f$  是任意的完全图灵可计算的一元整数函数， $n$  是任意的正整数，则存在自然数  $m$ ，使得  $M_m$  与  $M_{f(m)}$  计算相同的  $n$  元整数函数。
5. 简述并举例说明对角化技术在证明中的应用。
6. 通用图灵机的模拟。（选做）