第三章

有穷状态自动机

问题

- 文法给出语言的描述，提供生成语言的手段。
- 如何机械的判断一个字符串是不是属于一个语言？

  ——语言的识别问题

- 不同种类的语言有不同的判断方法。

  ——正则语言用有穷状态自动机（FA）判断

问题

S→aA|aB，A→aA|c，B→aB|d，句子aaac是该文法定义语言的句子吗？

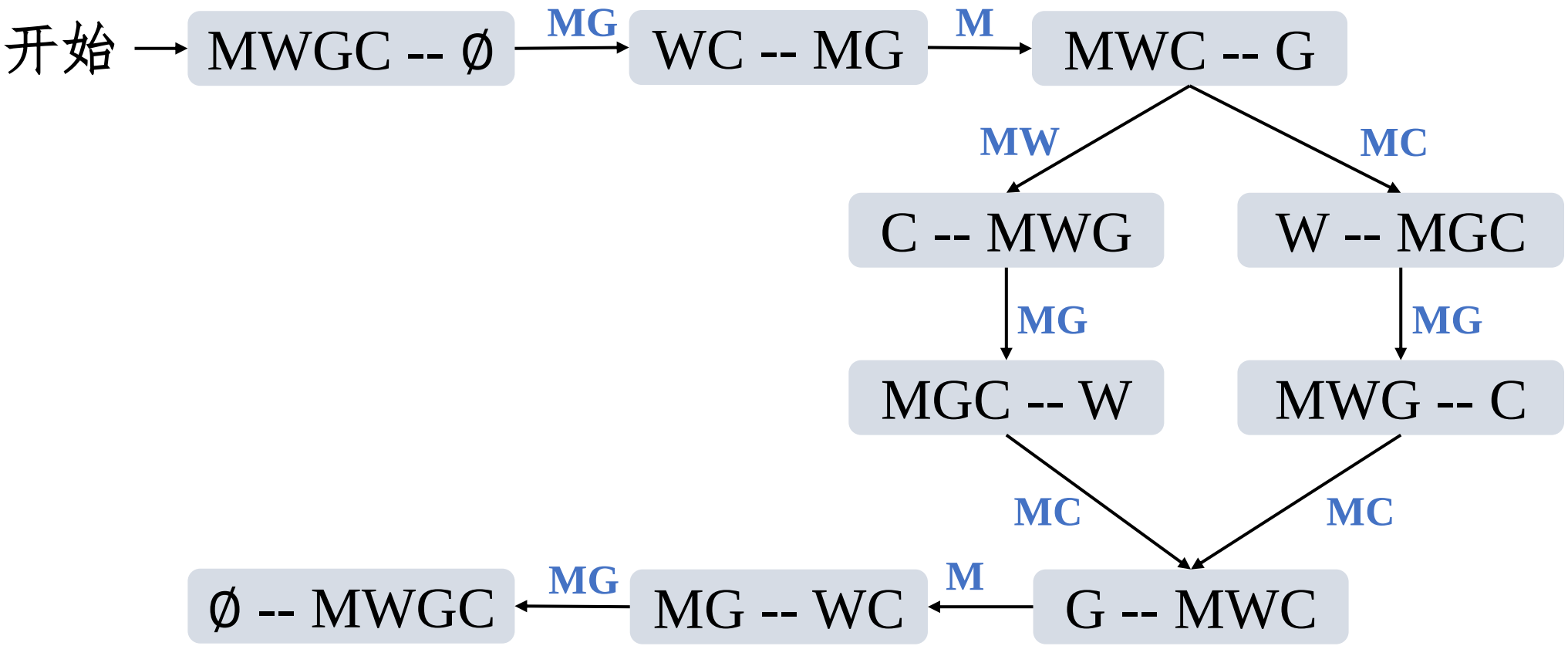
- **问题：推导和归约中的回溯问题将对系统的效率产生极大的影响**
- **解决方法：从识别的角度去考虑问题，设计有穷自动机系统来识别语言。**

过河问题

- 一个人带着一头狼、一头羊和一颗青菜准备过河；
- 人可以单独过河或带三者之一过河；
- 狼吃羊、羊吃菜；
- 如何在保证羊和菜不被吃掉的情况下过河？

过河问题模型化

人：M（Men）
狼：W（Wolf）
羊：G（Goat）
菜：C（Cabbage）



章节目录

识别系统（模型）

1. 系统具有有穷个状态，不同的状态代表不同的意义。按照实际的需要，系统可以在不同的状态下完成规定的任务。
2. 我们可以将输入字符串中出现的字符汇集在一起构成一个字母表。系统处理的所有字符串都是这个字母表上的字符串。

识别系统（模型）

3. 系统在任何一个状态下，从输入字符串中读入一个字符，根据当前状态和读入的这个字符转到新的状态。当前状态和新的状态可以是同一个状态，也可以是不同的状态。

当系统从输入字符串中读入一个字符后，它下一次再读时，会读入下一个字符。这就是说，相当于系统维持有一个读写指针，该指针在系统读入一个字符后指向输入串的下一个字符。

识别系统（模型）

4. 系统中有一个状态（开始状态），系统在这个状态下开始进行某个给定句子的处理。
5. 系统中还有一些状态（终止状态），把所有将系统从开始状态引导到这种状态的字符串放在一起构成一个语言，该语言就是系统所能识别的语言。

相应的物理模型

- 一个右端无穷的输入带。
- 一个有穷状态控制器(Finite State Control, FSC) 。
- 一个读头。

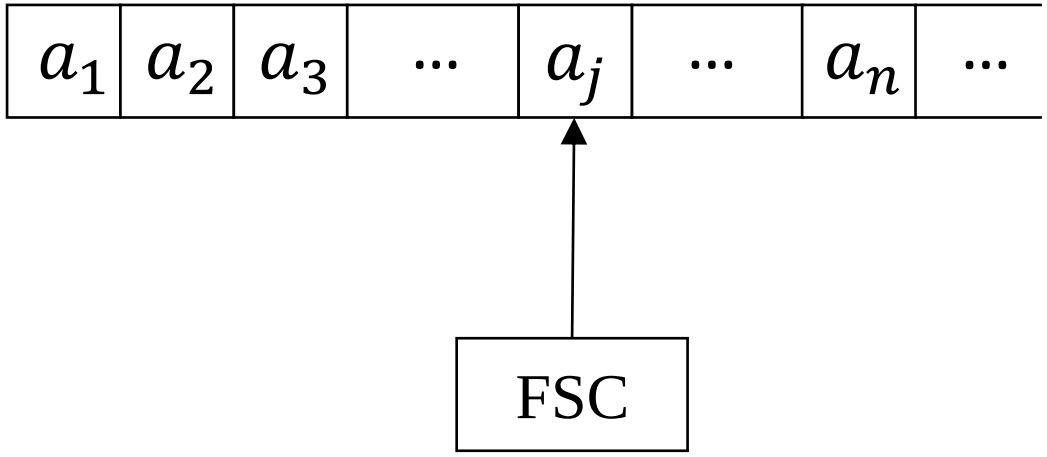


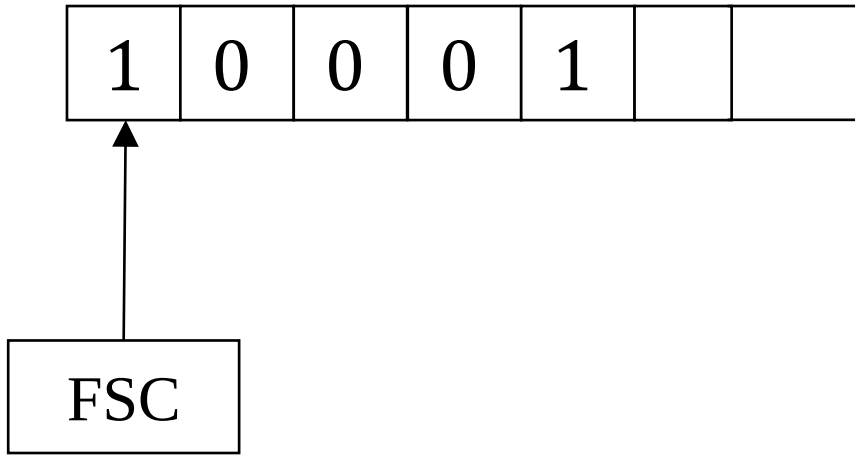
图3-1 有穷状态自动机的物理模型

相应的物理模型

系统的每一个动作由三个节拍构成：

- 读入读头正注视的字符；
- 根据当前状态和读入的字符改变有穷控制器的状态；
- 将读头向右移动一格。

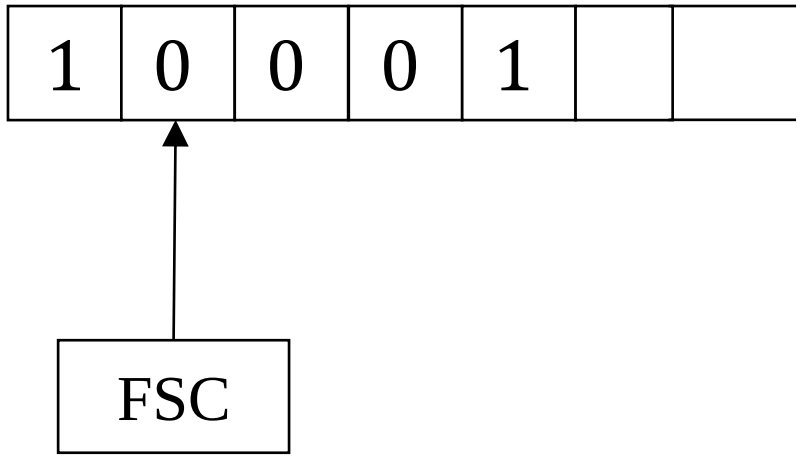
例3-1 用有穷自动机识别$\{\omega \in \{0,1\}^* | \omega$的长度是偶数$\}$。



1. 输入带中字符
2. 有穷控制器中存储的状态
3. 读头读入字符后状态的改变
4. 最初的状态
5. 接收的状态

偶数　　奇数

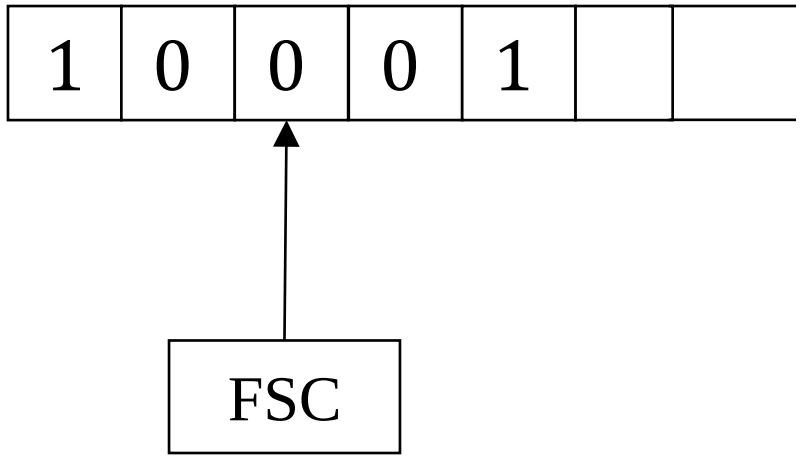
例3-1 用有穷自动机识别$\{\omega \in \{0,1\}^* | \omega$的长度是偶数$\}$。



偶数

1. 输入带中字符
2. 有穷控制器中存储的状态
3. 读头读入字符后状态的改变
4. 最初的状态
5. 接收的状态

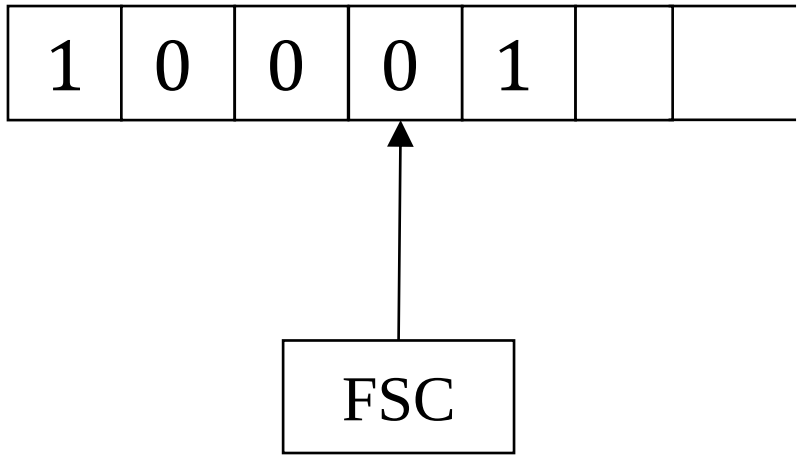
例3-1 用有穷自动机识别$\{\omega \in \{0,1\}^* | \omega$的长度是偶数$\}$。



奇数

1. 输入带中字符
2. 有穷控制器中存储的状态
3. 读头读入字符后状态的改变
4. 最初的状态
5. 接收的状态

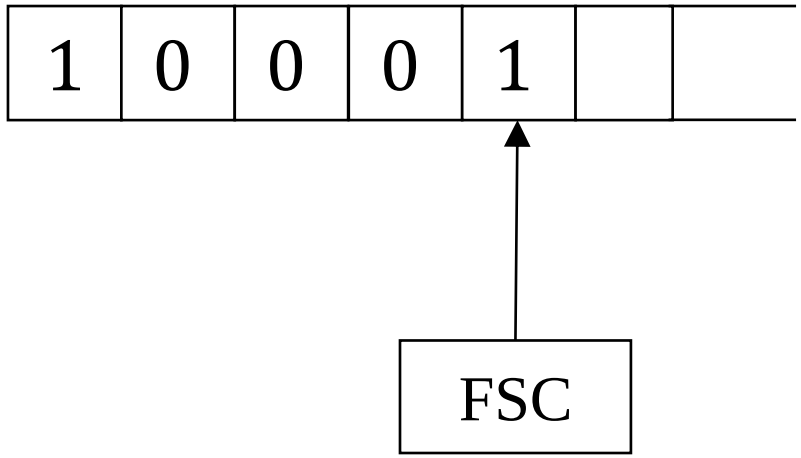
例3-1 用有穷自动机识别$\{\omega \in \{0,1\}^* | \omega$的长度是偶数$\}$。



偶数

1. 输入带中字符
2. 有穷控制器中存储的状态
3. 读头读入字符后状态的改变
4. 最初的状态
5. 接收的状态

例3-1 用有穷自动机识别$\{\omega \in \{0,1\}^* | \omega$的长度是偶数$\}$。



奇数

1. 输入带中字符
2. 有穷控制器中存储的状态
3. 读头读入字符后状态的改变
4. 最初的状态（偶数）
5. 接收的状态（偶数）

章节目录

定义3-1 有穷状态自动机(Finite Automaton, FA)

$$\mathbf{M} = (\boldsymbol{Q}, \boldsymbol{\Sigma}, \boldsymbol{\delta}, \boldsymbol{q_0}, \boldsymbol{F})$$

1. $\boldsymbol{Q}$：状态的非空有穷集合，$\forall \boldsymbol{q} \in \boldsymbol{Q}$，$\boldsymbol{q}$ 称为M的一个状态(state)。
2. $\boldsymbol{\Sigma}$：输入字母表，输入字符串都是$\boldsymbol{\Sigma}$上的字符串。

定义3-1 有穷状态自动机(Finite Automaton, FA)

$$\mathbf{M} = (\boldsymbol{Q}, \boldsymbol{\Sigma}, \boldsymbol{\delta}, \boldsymbol{q_0}, \boldsymbol{F})$$

3. $\boldsymbol{\delta}$：转移函数(transition function)，有时候又叫做状态转换函数或者移动函数。$\boldsymbol{\delta}$：$\boldsymbol{Q \times \Sigma \to Q}$，对$\forall(\boldsymbol{q}, \boldsymbol{a}) \in \boldsymbol{Q \times \Sigma}, \boldsymbol{\delta}(\boldsymbol{q}, \boldsymbol{a}) = \boldsymbol{p}$，表示：M在状态$\boldsymbol{q}$读入字符$\boldsymbol{a}$，将状态变成$\boldsymbol{p}$，并将读头向右移动，指向输入字符串的下一个字符。

定义3-1 有穷状态自动机(Finite Automaton, FA)

$$\mathbf{M} = (Q, \Sigma, \delta, q_0, F)$$

4. q_0：$q_0 \in Q$，是M的开始状态(initial state)，也可叫做初始状态或者启动状态。

5. F：$F \subseteq Q$，是M的终止状态(final state)集合。$\forall q \in F$，q称为M的终止状态，又称为接受状态(accept state)。

例3-2 下面是一个有穷状态自动机

$$M=(\{q_0,q_1,q_2\},\{0\},\delta_1,q_0,\{q_2\})$$

其中，$\delta_1(q_0,0)=q_1,\ \delta_1(q_1,0)=q_2,\ \delta_1(q_2,0)=q_1$。

0	q_1	✘
00	q_2	✓
000	q_1	✘
0000	q_2	✓

状态转移图（transition diagram）

■$q \in Q$： q是该有向图中的一个顶点；

■$\delta(q, a) = p$：有一条从顶点q到顶点p的标记为a的弧；

■$q \in F$： 标记为q的顶点被用双层圈标出；

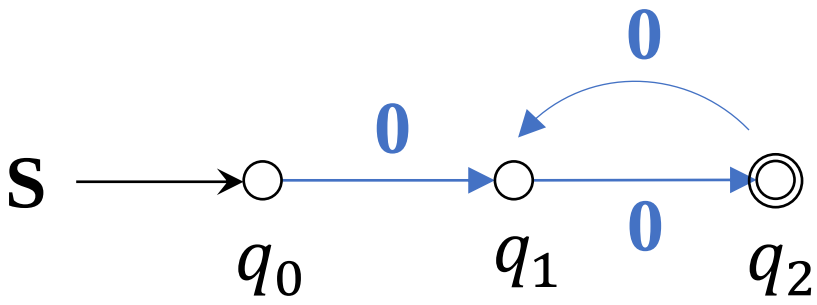
■用标有S的箭头指出开始状态。

续例3-2 下面是一个有穷状态自动机

$$M=(\{q_0,q_1,q_2\},\{0\},\delta_1,q_0,\{q_2\})$$

其中，$\delta_1(q_0,0)=q_1,\delta_1(q_1,0)=q_2,\delta_1(q_2,0)=q_1$。

画出其对应的状态转移图。



0	q_1	✘
00	q_2	✓
000	q_1	✘
0000	q_2	✓

状态转移表

- 第一列中标出开始状态和终止状态；
- 第二列列出有穷状态自动机的所有状态；
- 第一行列出所有的输入字符；
- 中间的方格内是该行所对应状态下输入该列所对应字符后转换到的新状态。

续例3-2 下面是一个有穷状态自动机

$$M=(\{q_0,q_1,q_2\},\{0\},\delta_1,q_0,\{q_2\})$$

其中，$\delta_1(q_0,0)=q_1,\delta_1(q_1,0)=q_2,\delta_1(q_2,0)=q_1$。

给出其对应的状态转移表。

状态说明	状态	输入字符
		0
开始状态	q_0	q_1
	q_1	q_2
终止状态	q_2	q_1

例3-3 下面是一个有穷状态自动机

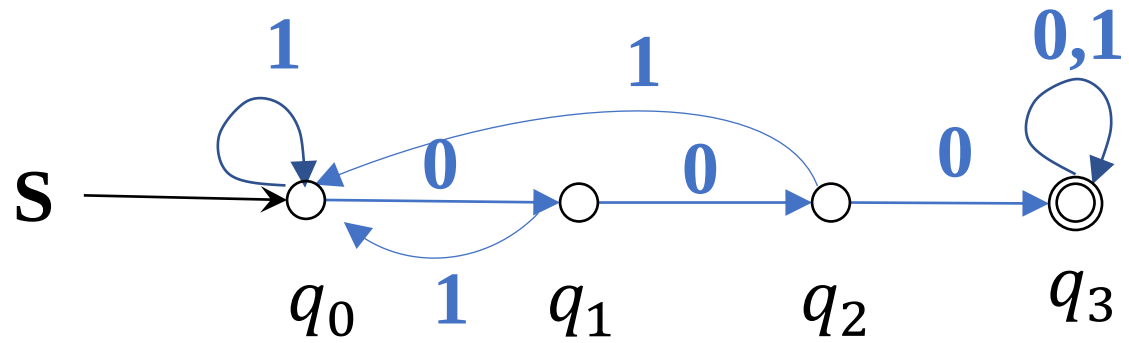
$$M=(\{q_0,q_1,q_2,q_3\},\{0,1\},\delta_2,q_0,\{q_3\})$$

其中，$\delta_2(q_0,0)=q_1$，$\delta_2(q_0,1)=q_0$，$\delta_2(q_1,0)=q_2$，

$\delta_2(q_1,1)=q_0$，$\delta_2(q_2,0)=q_3$，$\delta_2(q_2,1)=q_0$，

$\delta_2(q_3,0)=q_3$，$\delta_2(q_3,1)=q_3$。

给出其对应的状态转移图与状态转移表。



例3-3 下面是一个有穷状态自动机

$$M=(\{q_0, q_1, q_2, q_3\}, \{0, 1\}, \delta_2, q_0, \{q_3\})$$

状态说明	状态	输入字符	
		0	1
开始状态	q_0	q_1	q_0
	q_1	q_2	q_0
	q_2	q_3	q_0
终止状态	q_3	q_3	q_3

扩充定义域

■定义FA的目的是用它来识别语言（字符串），而不是单个字符，因此需要将δ的定义域从$Q \times \Sigma$扩充到$Q \times \Sigma^*$上。

扩充定义域

对任意的$q \in Q, \omega \in \Sigma^*, a \in \Sigma$:

■$\hat{\delta}(q, \varepsilon) = q$;

■$\hat{\delta}(q, \omega a) = \delta(\hat{\delta}(q, \omega), a)$。

由于δ的定义域$Q \times \Sigma$是$\hat{\delta}$的定义域$Q \times \Sigma^*$的真子集，即$Q \times \Sigma \subset Q \times \Sigma^*$，所以对于任意的$(q, a) \in Q \times \Sigma$，如果$\hat{\delta}$和$\delta$都有相同的值，就不用区分这两个符号了。

扩充定义域

事实上，对于任意的$q \in Q, a \in \Sigma$，有

$$\blacksquare \hat{\delta}(q, a) = \hat{\delta}(q, \varepsilon a)$$

$$= \delta(\hat{\delta}(q, \varepsilon), a)$$

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

由于对于任意的$q \in Q, a \in \Sigma, \delta(q, a)$均有确定的值，所以将这种FA称为确定的有穷状态自动机(Deterministic Finite Automaton, DFA)

定义3-2 设$\mathbf{M}=(Q,\Sigma,\delta,q_0,F)$是一个DFA，对于$\forall x\in\Sigma^*$，如果$\delta(q_0,x)\in F$，则称$x$被M接受，如果$\delta(q_0,x)\notin F$，则称M不接受$x$。

$$L(\mathbf{M})=\{x|x\in\Sigma^*\text{且}\delta(q_0,x)\in F\}$$

称为由M接受（识别）的语言。

如果$L(M_1)=L(M_2)$，则称M_1与M_2等价。

例3-4 构造一个DFA，它接受的语言是$\{x000y|x,y\in\{0,1\}^*\}$。

M共有几种状态？

- q_0：M的启动状态。
- q_1：M读到了一个0，可能是子串“000”的第一个0。
- q_2：M在q_1后又读到了一个0，可能是子串“000”的第二个0。
- q_3：M在q_2后紧接着又读到了一个0，发现输入的字符串含有子串“000”。因此q_3是终止状态。

例3-4 构造一个DFA，它接受的语言是$\{x000y|x,y\in\{0,1\}^*\}$。

状态转移函数如何定义？

1. $\delta(q_0,0)=q_1$：M读到了一个0，可能是子串“000”的第一个0，等待第二个0；
2. $\delta(q_0,1)=q_0$：M读到了一个1，需要继续等待可能是子串“000”的第一个0；
3. $\delta(q_1,0)=q_2$：M又读到了一个0，可能是子串“000”的第二个0，等待第三个0；
4. $\delta(q_1,1)=q_0$：M刚读了一个0后，读到了一个1，表明之前的0不是子串“000”的第一个0，重新回到初始状态q_0，以寻找子串“000”的第一个0。

例3-4 构造一个DFA，它接受的语言是$\{x000y|x,y\in\{0,1\}^*\}$。

状态转移函数如何定义？

5. $\delta(q_2,0)=q_3$：M又读到一个0，找到了子串“000”；
6. $\delta(q_2,1)=q_0$：M在刚读00后，读到了一个1，表明之前的00不是子串“000”的前两个0，重新回到初始状态q_0，以寻找子串“000”的第一个0。
7. $\delta(q_3,1)=q_3$：M找到了子串“000”，该串的剩余部分直接读完即可。
8. $\delta(q_3,0)=q_3$：M找到了子串“000”，该串的剩余部分直接读完即可。

例3-4 构造一个DFA，它接受的语言是$\{x000y|x,y\in\{0,1\}^*\}$。

到此，得到接受语言$\{x000y|x,y\in\{0,1\}^*$的**DFA：**

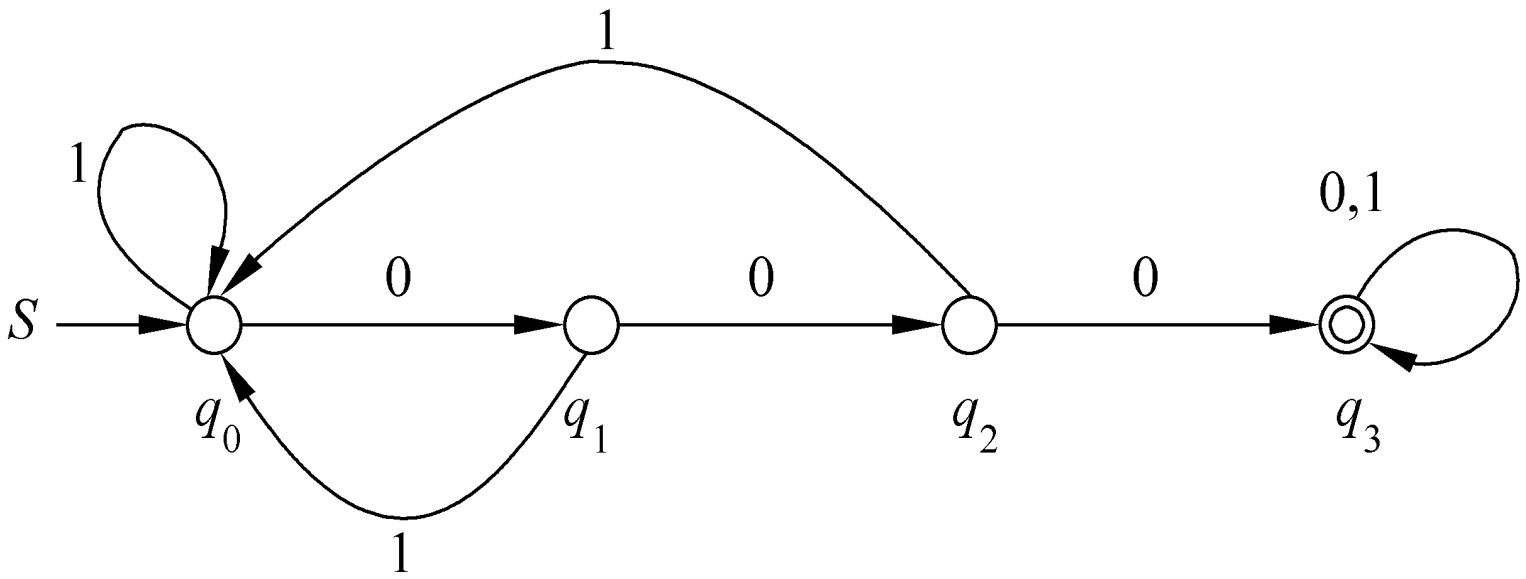
1. 按定义给出

$$\mathrm{M}=(\{q_0,q_1,q_2,q_3\},\{0,1\},\{\delta(q_0,0)=q_1,\delta(q_0,1)$$
$$=q_0,\delta(q_1,0)=q_2,\delta(q_1,1)=q_0,\delta(q_2,0)=q_3,\delta(q_2,1)$$
$$=q_0,\delta(q_3,0)=q_3,\delta(q_3,1)=q_3\},q_0,\{q_3\})$$

例3-4 构造一个DFA，它接受的语言是$\{x000y|x,y\in\{0,1\}^*\}$。

到此，得到接受语言$\{x000y|x,y\in\{0,1\}^*$的DFA：

2. 状态转移图



例3-4 构造一个DFA，它接受的语言是$\{x000y|x,y\in\{0,1\}^*\}$。

到此，得到接受语言$\{x000y|x,y\in\{0,1\}^*$的DFA：

3. 状态转移表

状态说明	状态	输入字符	
		0	1
开始状态	q_0	q_1	q_0
	q_1	q_2	q_0
	q_2	q_3	q_0
终止状态	q_3	q_3	q_3

例3-4’ 构造一个DFA，它接受的语言是$\{x000|x \in \{0,1\}^*\}$。

语言$\{x000|x \in \{0,1\}^*$ 与$\{x000y|x,y \in \{0,1\}^*$ 的不同之处是什么？

- 前者要求每个句子都是以“000”结尾的，后者要求句子中含有子串“000”即可。
- 查到“000”并不表明此串可以被接受，而需要看这3个连续的0是否为输入串的最后3个字符。

例3-4’ 构造一个DFA，它接受的语言是$\{x000|x \in \{0,1\}^*\}$。

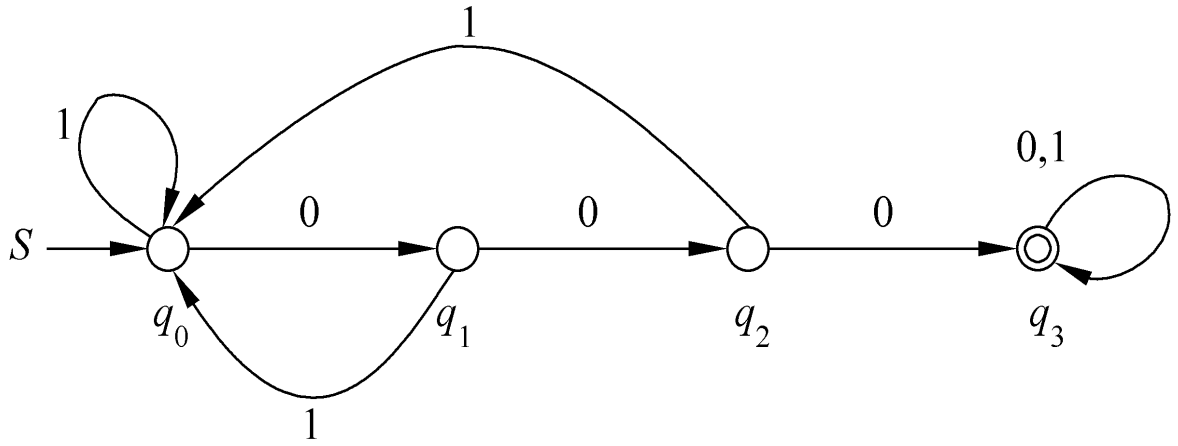
M共有几种状态？

- q_0：M的启动状态。
- q_1：M读到了一个0，可能是最后3个0的第一个0。
- q_2：M在q_1后又读到了一个0，可能是最后3个0的第二个0。
- q_3：M在q_2后紧接着又读到了一个0，发现输入的字符串以“000”结尾。因此q_3是终止状态。

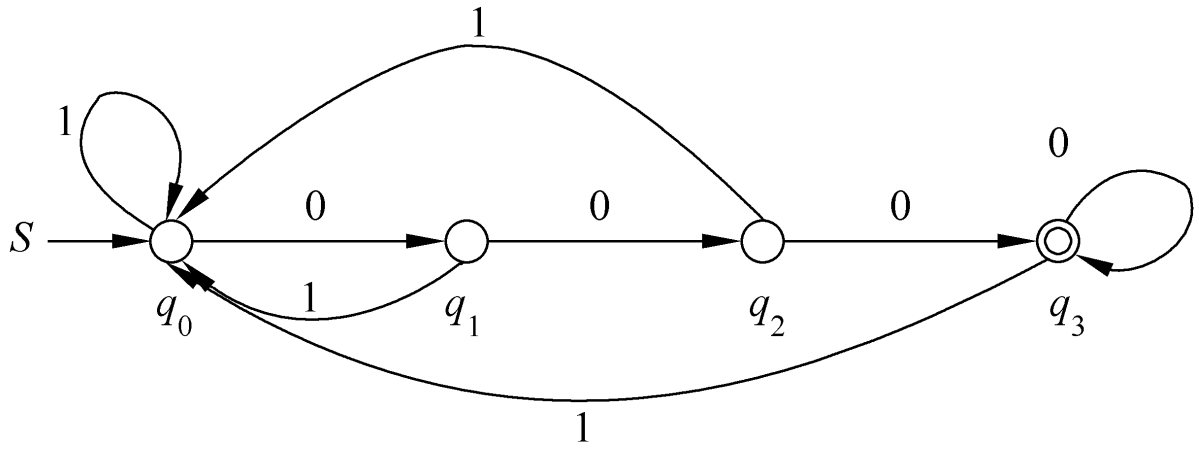
  如果在q_3后紧接着读到了一个0，则输入串仍旧以“000”结尾；如果在q_3后紧接着读到了一个1，则要重新检查是否以“000”结尾。

例3-4’ 构造一个DFA，它接受的语言是$\{x000|x \in \{0,1\}^*\}$。

$\{x000y|x,y \in \{0,1\}^*\}$:



$\{x000|x \in \{0,1\}^*\}$:



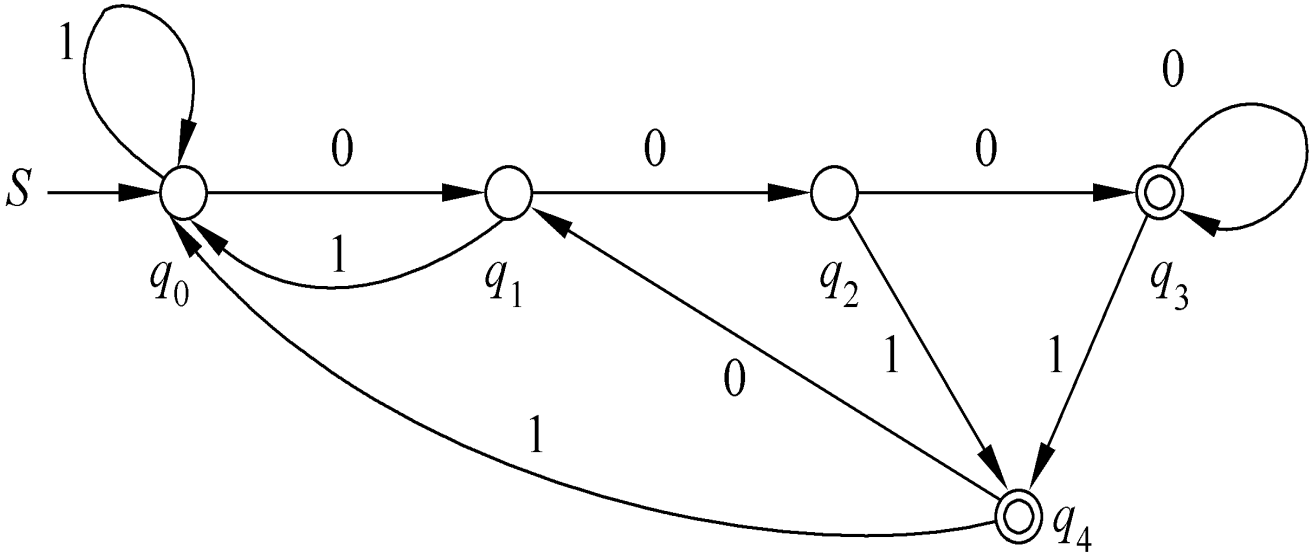
几点注意：

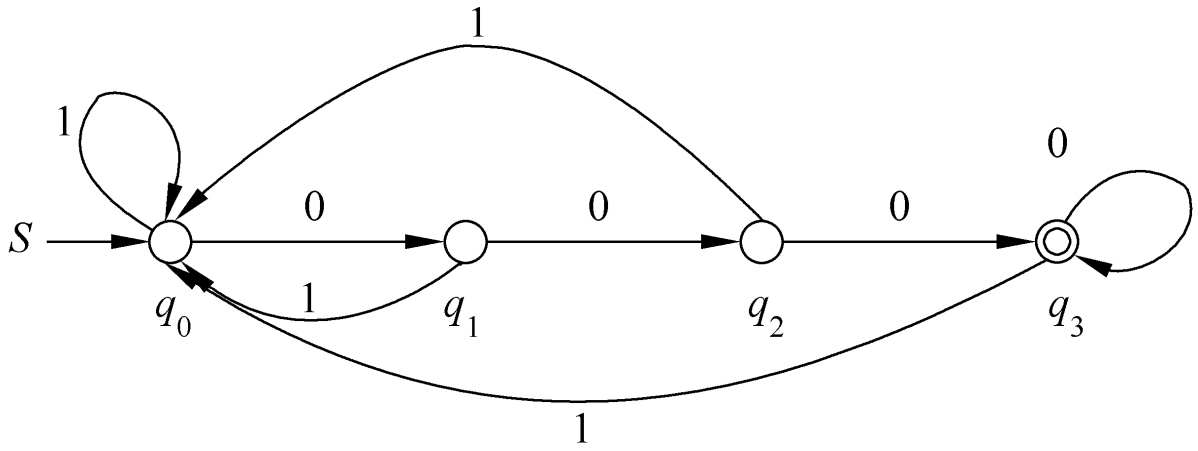
1. 定义FA时，常常只给出FA相应的状态转移图就可以了；
2. 一个FA有且只能有一个初始状态；
3. 一个FA可以有多于1个的终止状态；
4. 对于DFA来说，每个顶点的出度恰好等于输入字母表中所含的字符的个数。
5. 字符串x被FA M接受的充分必要条件是：在M的状态转移图中存在一条从开始状态到某一个终止状态的有向路，该有向路上从第一条边到最后一条边的标记依次并置构成了字符串x。简称此路的标记为x。

例3-5 构造一个DFA，它接受的语言是$\{x000|x \in \{0,1\}^*\} \cup \{x001|x \in \{0,1\}^*\}$。

该语言有几个终止状态？

$\{x000|x \in \{0,1\}^*$：

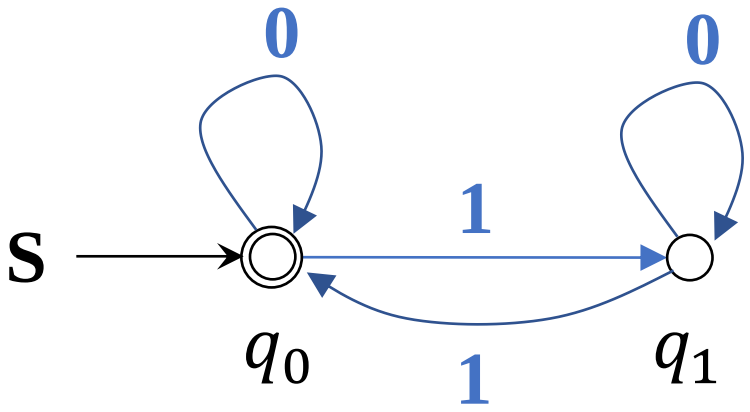




例3-6 构造一个DFA M，它接受的语言是$\{\omega|\omega \in \{0,1\}^*$,且$\omega$含有偶数个1$\}$。

q_0：ω有偶数个1

q_1：ω有奇数个1



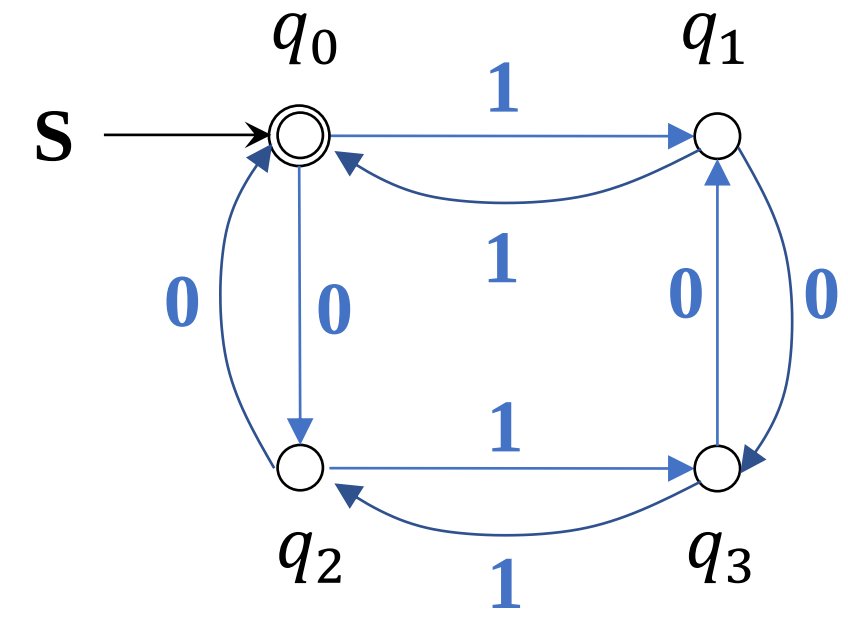
例3-7 构造一个DFA M，它接受的语言是$\{\omega|\omega \in \{0,1\}^*$, 且$\omega$含有偶数个0和偶数个1$\}$。

q_0：偶0偶1

q_1：偶0奇1

q_2：奇0偶1

q_3：奇0奇1



即时描述：

定义3-3 设$M=(Q,\Sigma,\delta,q_0,F)$是一个FA，$\forall x,y\in\Sigma^*$，$\delta(q_0,x)=q$，$xqy$称为M的一个即时描述(instantaneous description, ID)，它表示xy是M正在处理的一个字符串，x引导M从q_0启动并达到状态q，M的读头当前正指向y的首字母。

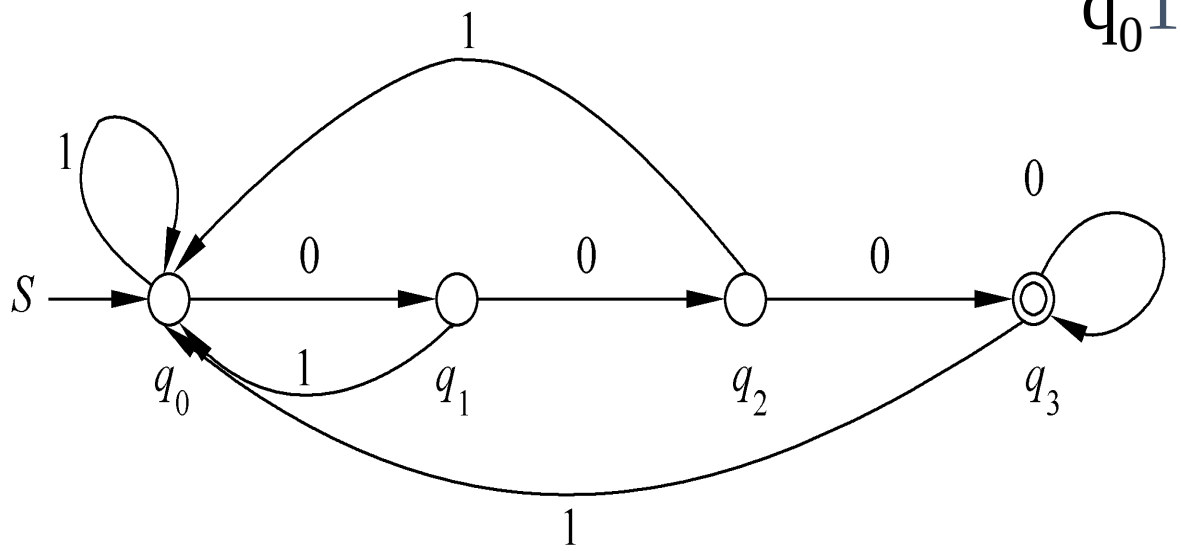
- 如果$xqay$是M的一个即时描述，且$\delta(q,a)=p$，则$xqay\vdash_M xapy$。
- 表示M在状态q时已经处理完x并且读头正指向输入字符a，此时它读入a并转入状态p，将读头向右移动一格指向y的首字母。
- $\vdash_M$表示M的一次移动。

- $\alpha \vdash_M^n \beta$：表示M从即时描述α经过n次移动到达即时描述β。M存在即时描述$\alpha_1,\alpha_2,\ldots\ldots,\alpha_{n-1}$，使得$\alpha \vdash_M \alpha_1,\alpha_1 \vdash_M \alpha_2,\ldots,\alpha_{n-1} \vdash_M \beta$。

- 当n=0时,有α=β。即$\alpha \vdash_M^0 \alpha$。
- $\alpha \vdash_M^+ \beta$：表示M从即时描述α经过至少1次移动到达即时描述β。
- $\alpha \vdash_M^* \beta$：表示M从即时描述α经过若干步移动到达即时描述β。

- 当意义清楚时,我们将符号$\vdash_M$、$\vdash_M^n$、$\vdash_M^*$、$\vdash_M^+$中的M省去,分别用$\vdash$、$\vdash^n$、$\vdash^*$、$\vdash^+$表示。

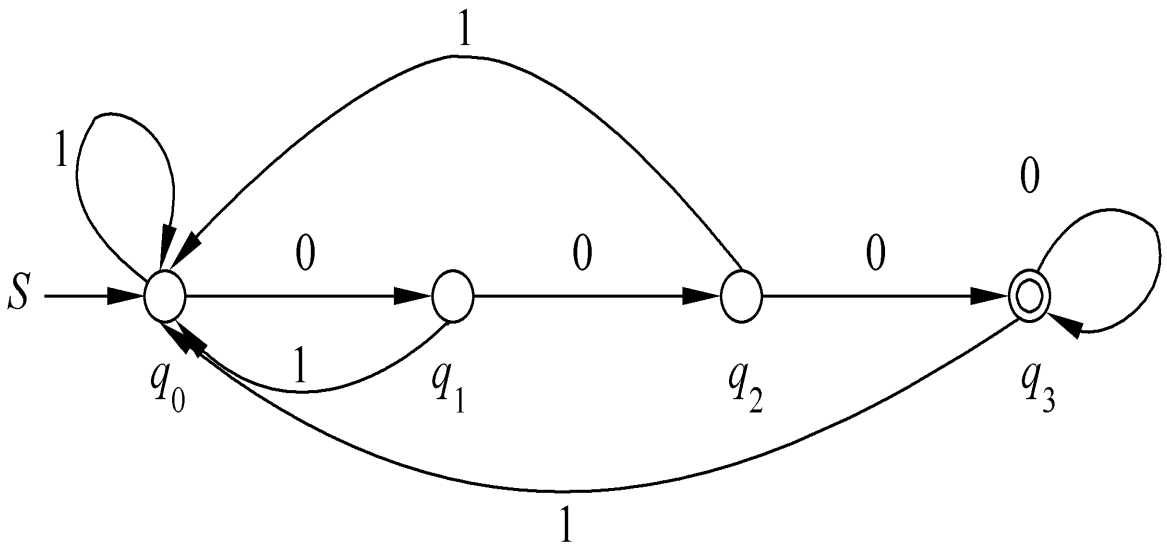
例3-8 用例3-4’所定义的M处理输入串**1010010001**。



$q_0$1010010001 ⊢ 1$q_0$010010001
⊢ 10$q_1$10010001
⊢ 101$q_0$0010001
⊢ 1010$q_1$010001
⊢ 10100$q_2$10001
⊢ 101001$q_0$0001
⊢ 1010010$q_1$001
⊢ 10100100$q_2$01
⊢ 101001000$q_3$1
⊢ 1010010001q_0

由于M在处理完1010010001后达到状态q_0，不是终止状态q_3，所以1010010001不是M接受的语言。

续例3-8 用例3-4’所定义的M处理输入串**1001011**。

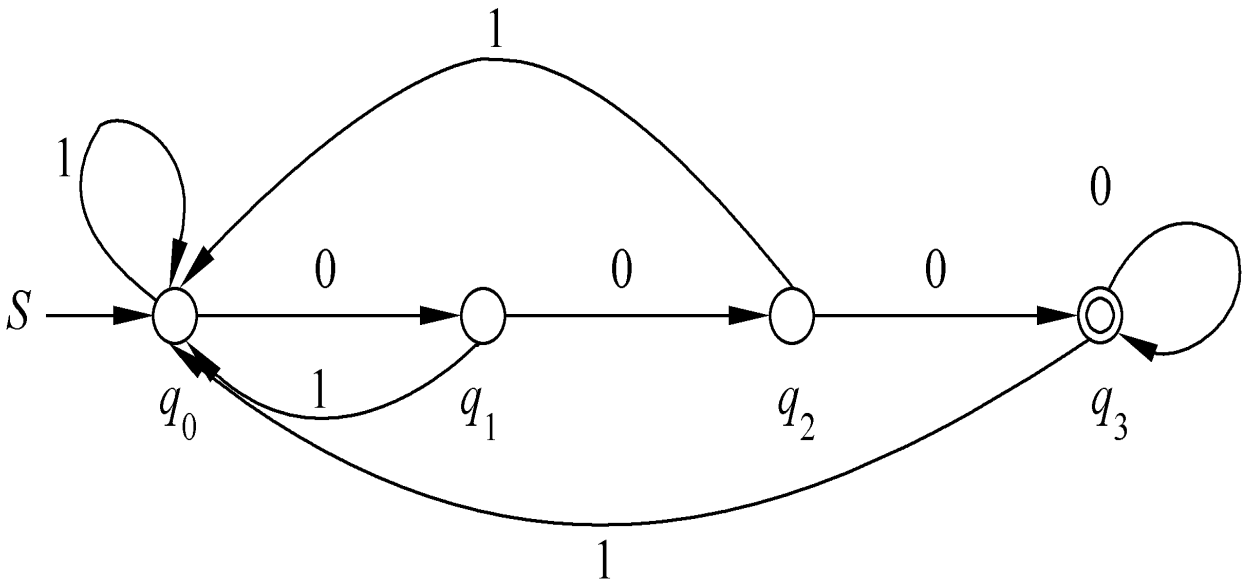


$q_0$1001011 ⊢ 1$q_0$001011
⊢ 10$q_1$01011
⊢ 100$q_2$1011
⊢ 1001$q_0$011
⊢ 10010$q_1$11
⊢ 100101$q_0$1
⊢ 1001011q_0

由于M在处理完1001011后达到状态q_0，不是终止状态q_3，所以1001011不是M接受的语言。

对例3-4’中DFA而言，$x \in \Sigma^*$，不难证明：

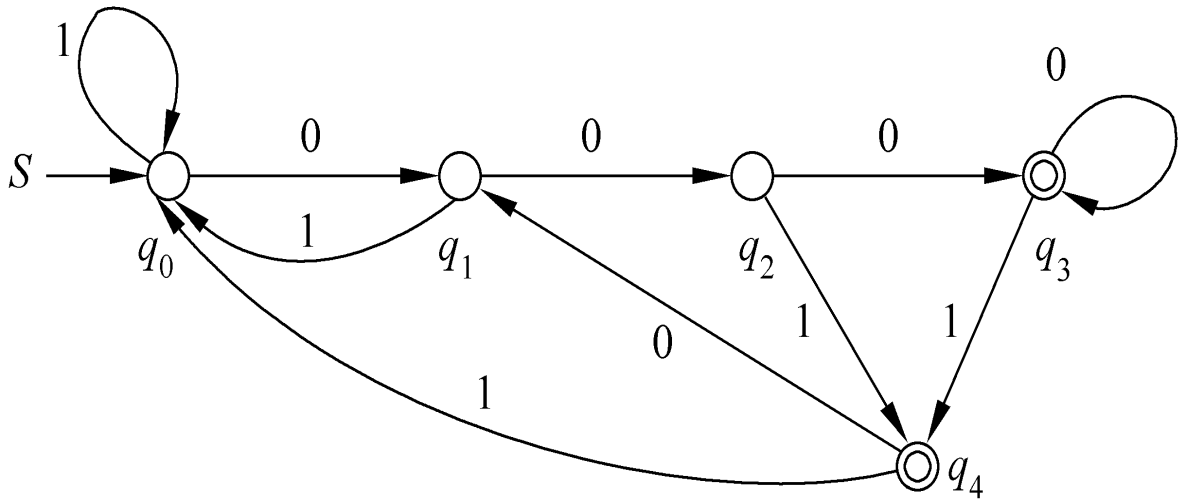
- $q_0x1 \vdash^+ x1q_0$
- $q_0x10 \vdash^+ x10q_1$
- $q_0x100 \vdash^+ x100q_2$
- $q_0x000 \vdash^+ x000q_3$



定义3-4 设M $= (Q, \Sigma, \delta, q_0, F)$是一个FA，对$\forall q \in Q$，能引导FA从开始状态到达$q$的字符串集合为

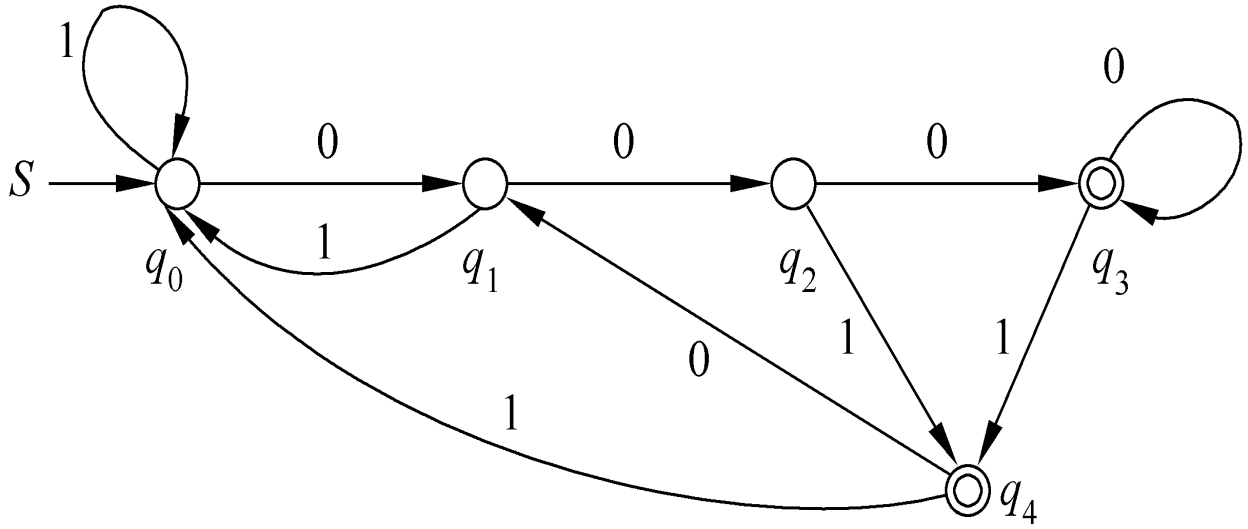
$$set(q) = \{x | x \in \Sigma^*, \delta(q_0, x) = q\}$$

对例3-5中的DFA，有



- set(q_3) $=\{x | x \in \Sigma^*,$ x以000结尾$\}$
- set(q_4) $=\{x | x \in \Sigma^*,$ x以001结尾$\}$

对例3-5中的DFA，有



- set(q_0) ={x|$x \in \Sigma^*$， x =ε或者x以1结尾（非001）}
- set(q_1) ={x|$x \in \Sigma^*$， x =0或者x以10结尾}
- set(q_2) ={x|$x \in \Sigma^*$， x =00或者x以100结尾}

这5个集合是两两互不相交，而且5个集合的并正好是{0，1}*。

DFA中的关系与等价关系：

一般地，对于任意一个DFA $M=(Q,\Sigma,\delta,q_0,F)$，按照如下方式定义关系$R_M$：

对 $\forall x,y\in\Sigma^*$，$xR_My\Leftrightarrow\exists q\in Q$，使 $x\in set(q)$ 和 $y\in set(q)$同时成立。

■按照这个定义所得到的关系实际上是Σ^*上的一个等价关系。利用这个关系，可以将Σ^*划分成不多于|Q|个等价类。

■进一步地讨论将在后面进行。

例3-9 构造一个DFA，它接受的语言为 $\{0^n1^m2^k|n,m,k\geq 1\}$。

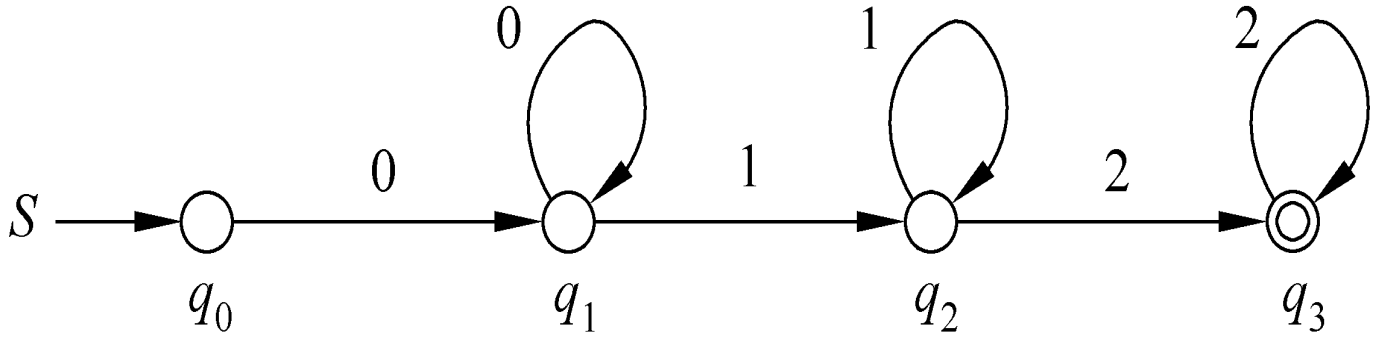
- 该语言句子的特点是：0在最前面，1在中间，2在最后。它们不可以交叉，也不可以颠倒顺序，字符0，1，2的个数均不可少于1。

定义4个状态：

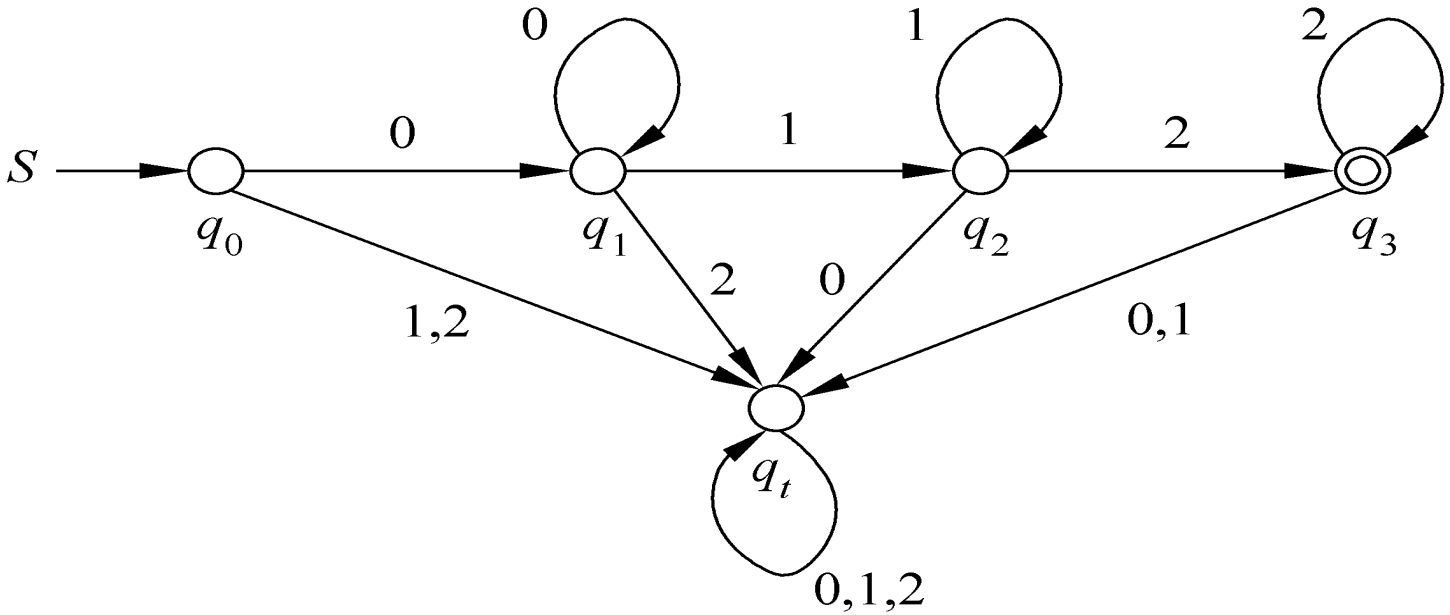
- **q_0：M的启动状态；**
- **q_1：M读到至少一个0，并等待更多的0；**
- **q_2：M读到至少一个0和至少一个1后，并等待读更多的1；**
- **q_3：M读到至少一个0、至少一个1后，读到了至少一个2。**

例3-9 构造一个DFA，它接受的语言为 $\{0^n1^m2^k|n,m,k\geq 1\}$。

（1）先设计“主体框架”



（2）再补充细节



几点注意：

1. 当FA一旦进入状态q_t，它就无法离开此状态。所以，q_t相当于一个陷阱状态(trap)。
2. 在构造一个识别给定语言的FA时，用画图的方式比较方便、直观。我们可以先根据语言的主要特征画出该FA的“主体框架”，然后再去考虑画出一些细节要求的内容。

几点注意：

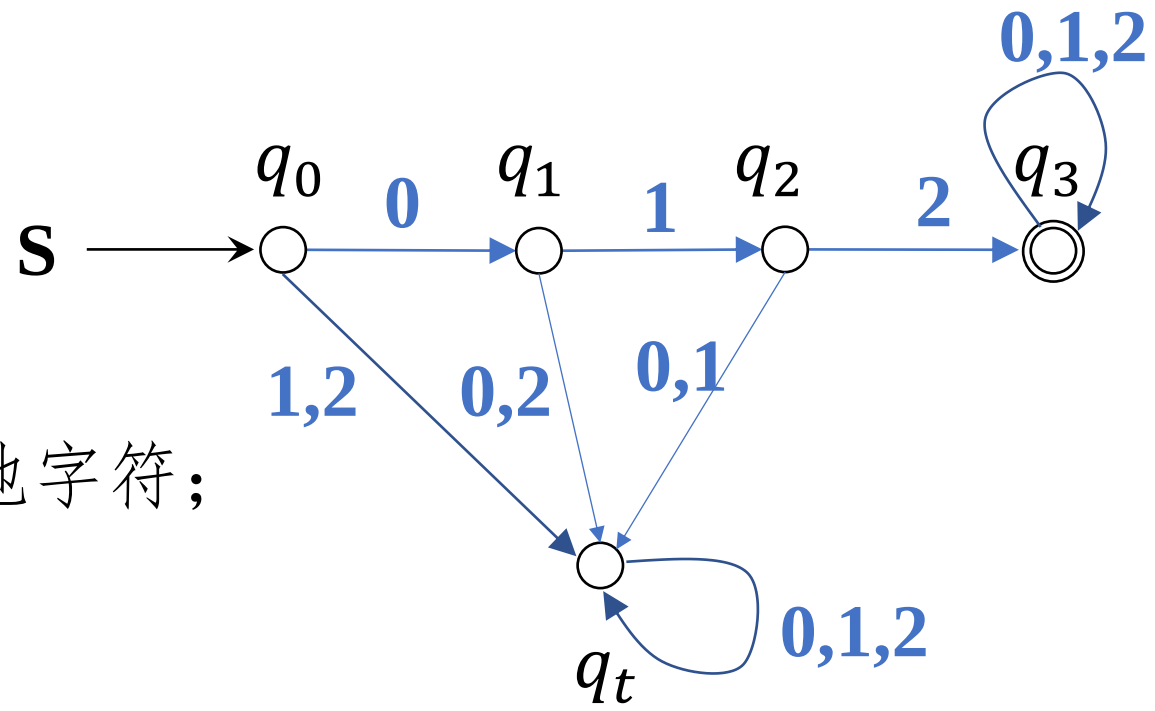
3. FA的状态具有一定的记忆功能：由于FA只有有穷个状态，在识别一个语言的过程中，如果有无穷种情况需要记忆，是无法构造出相应的FA的。

- 如对语言$\{0^n1^n|n \geq 1\}$，当扫描到n个0时，需要去寻找n个1，由于n有无穷多个，所以需要记忆的0的个数有无穷多种，因此无法构造出接受该语言的FA。也就是说该语言不属于FA可以接受的语言类。
- 相应的证明留到研究FA接受的语言类的性质时进行。

例3-10 构造一个DFA，它接受的语言是$\{012\omega|\omega \in \{0,1,2\}^*\}$。

M共有几种状态？

- q_0：M的启动状态，等待0；
- q_1：M读到了一个0，等待1；
- q_2：M又读到了一个1，等待2；
- q_3：M读到了一个2，继续读入其他字符；
- q_t：陷阱状态



例3-11 构造一个DFA，它接受的语言为$\{x|x \in \{0,1\}^*$，且当把x看成十进制数时，x模3与0同余$\}$。

定义4个状态：

- q_s：M的开始状态；
- q_0：对应除以3余数为0的x组成的等价类；
- q_1：对应除以3余数为1的x组成的等价类；
- q_2：对应除以3余数为2的x组成的等价类。

例3-11 构造一个DFA，它接受的语言为$\{x|x \in \{0,1\}^*$，且当把x看成十进制数时，x模3与0同余$\}$。

- q_s：在此状态下读入0时，有x =0，所以应该进入状态q_0；读入1时，有x =1，所以应该进入状态q_1。
  即：$\delta(q_s,0)=q_0$；$\delta(q_s,1)=q_1$。
- q_0：能引导M到达此状态的x除以3余0，所以有：x =3*n+0。
  （1）读入0时，引导M到达下一个状态的字符串为x0，
  x0=2*(3*n+0)=3*2*n+0。所以$\delta(q_0,0)=q_0$。
  （2）读入1时，引导M到达下一个状态的字符串为x1，
  x1=2*(3*n+0)+1=3*2*n+1。所以$\delta(q_0,1)=q_1$。

例3-11 构造一个DFA，它接受的语言为$\{x|x \in \{0,1\}^*$，且当把x看成十进制数时，x模3与0同余$\}$。

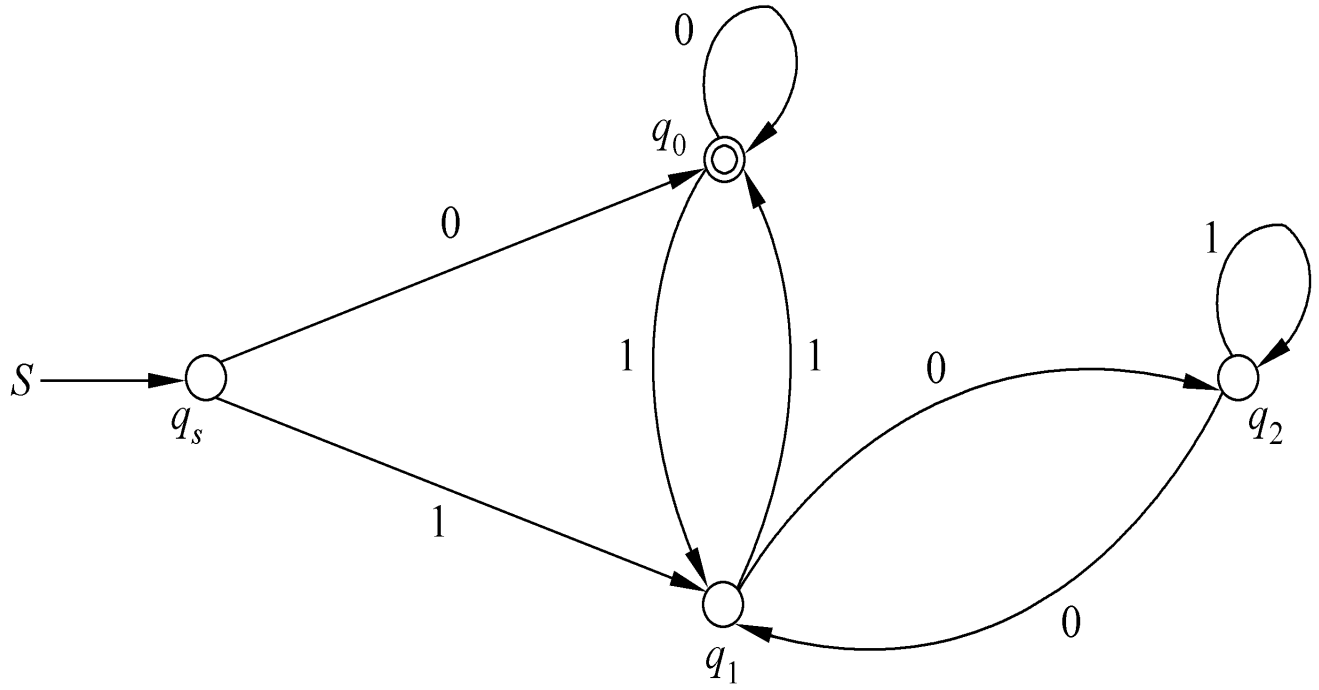
- q_1：能引导M到达此状态的x除以3余1，所以有：x =3*n+1。
  （1）读入0时，引导M到达下一个状态的字符串为$x0$, x $x0$=2*(3*n+1)=3*2*n+2。所以$\delta(q_1,0)=q_2$。
  （2）读入1时，引导M到达下一个状态的字符串为$x1$, $x1$=2*(3*n+1)+1=3*2*n+2+1= 3*(2*n+1)。所以$\delta(q_1,1)=q_0$。

例3-11 构造一个DFA，它接受的语言为$\{x|x \in \{0,1\}^*$，且当把x看成十进制数时，x模3与0同余$\}$。

- q_2：能引导M到达此状态的x除以3余2，所以有：x =3*n+2。
  （1）读入0时，引导M到达下一个状态的字符串为$x0$,
  $x0$=2*(3*n+2)=3*2*n+4= 3*(2*n+1)+1。所以$\delta(q_2,0) = q_1$。
  （2）读入1时，引导M到达下一个状态的字符串为$x1$,
  $x1$=2*(3*n+2)+1=3*2*n+4+1= 3*(2*n+1)+2。所以$\delta(q_2,1) = q_2$。

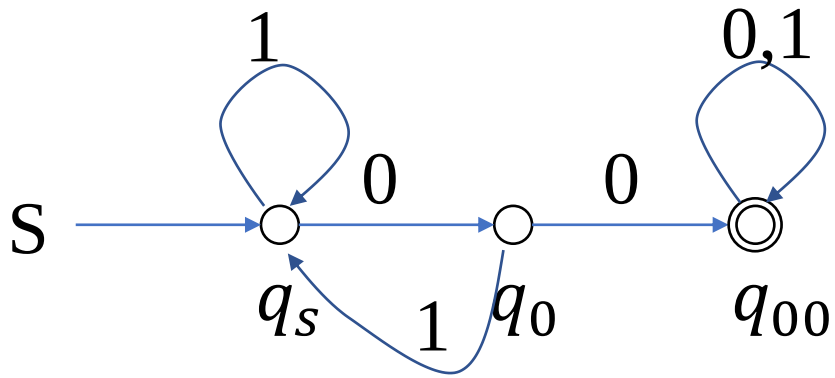
例3-11 构造一个DFA，它接受的语言为$\{x|x \in \{0,1\}^*$，且当把x看成十进制数时，x模3与0同余$\}$。

$$\delta(q_s,0)=q_0;\ \ \delta(q_s,1)=q_1$$
$$\delta(q_0,0)=q_0;\ \ \delta(q_0,1)=q_1$$
$$\delta(q_1,0)=q_2;\ \ \delta(q_1,1)=q_0$$
$$\delta(q_2,0)=q_1;\ \ \delta(q_2,1)=q_2$$

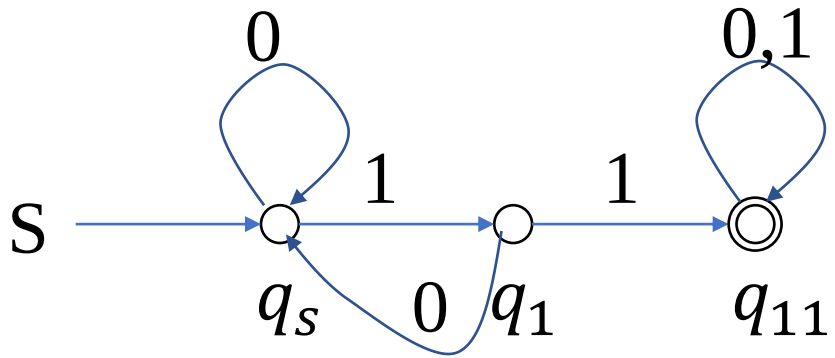


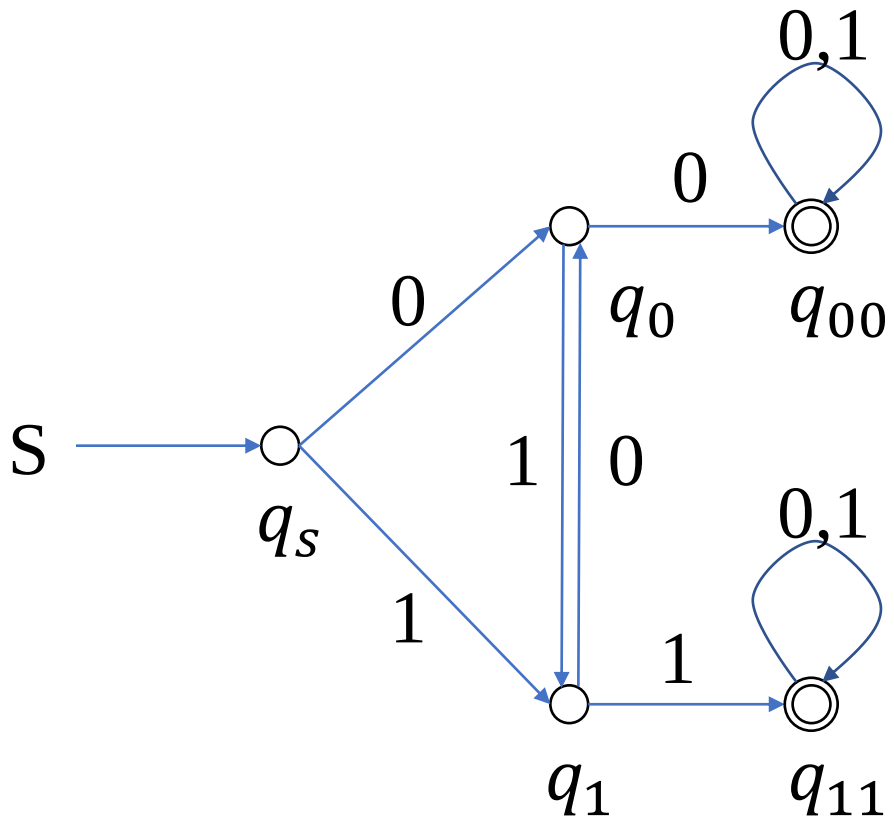
例3-12 构造接受 $L=\{x|x\in\{0,1\}^*$, 且 x 含有子串00或11}的DFA。

含有子串00的DFA：



含有子串11的DFA：

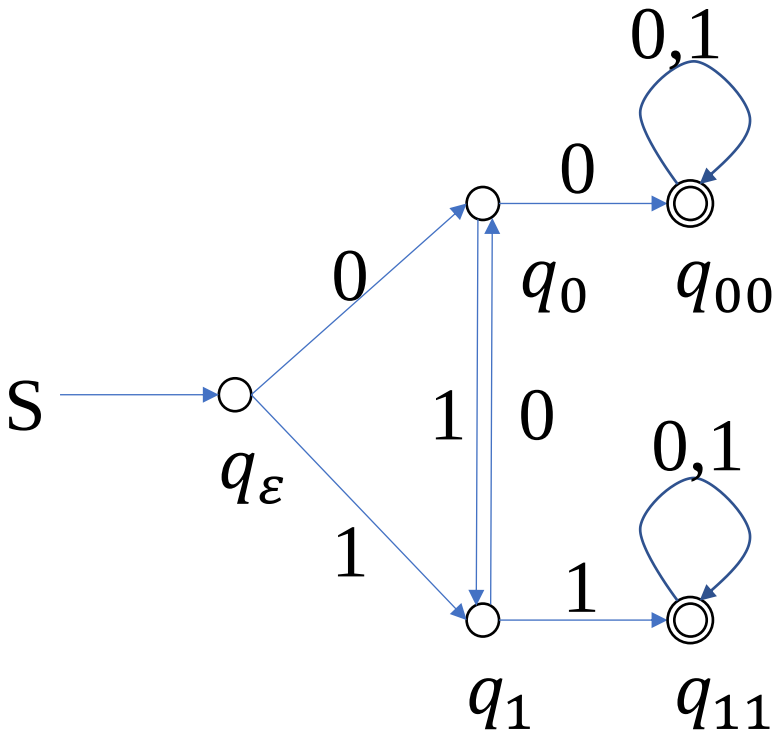


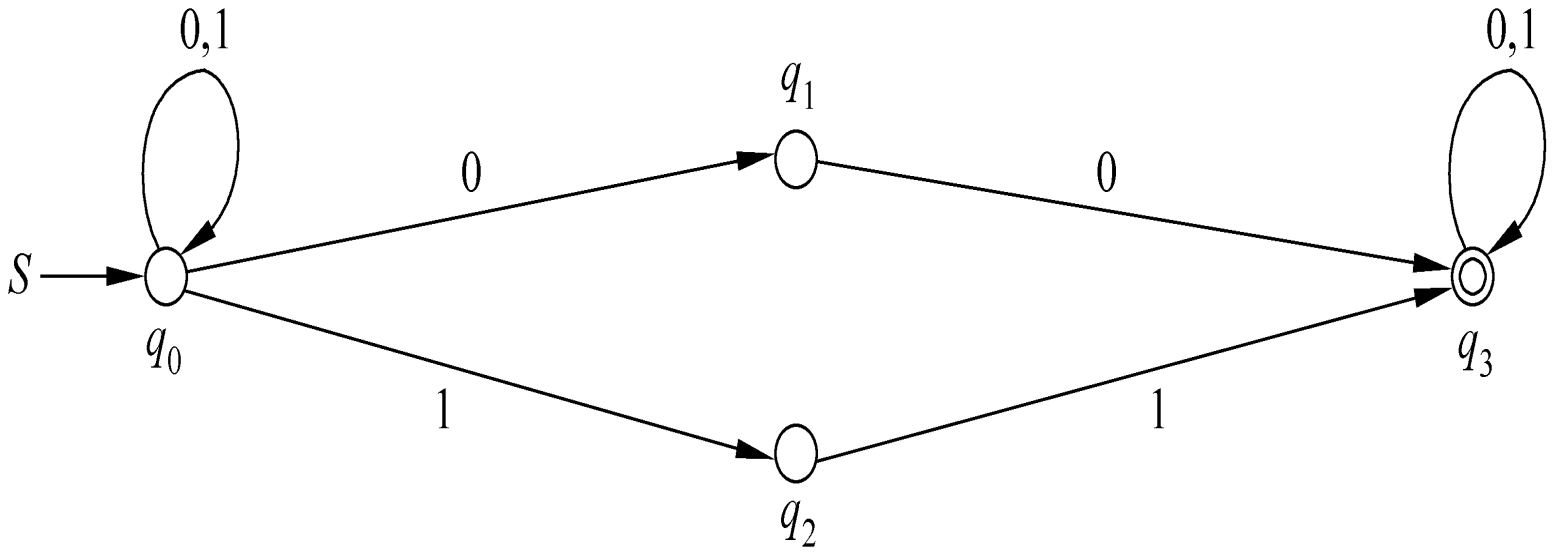


章节目录

3.3 不确定的有穷状态自动机

例3-12’ 构造接受$L=\{x|x\in\{0,1\}^*$,且x含有子串00或11}的FA。

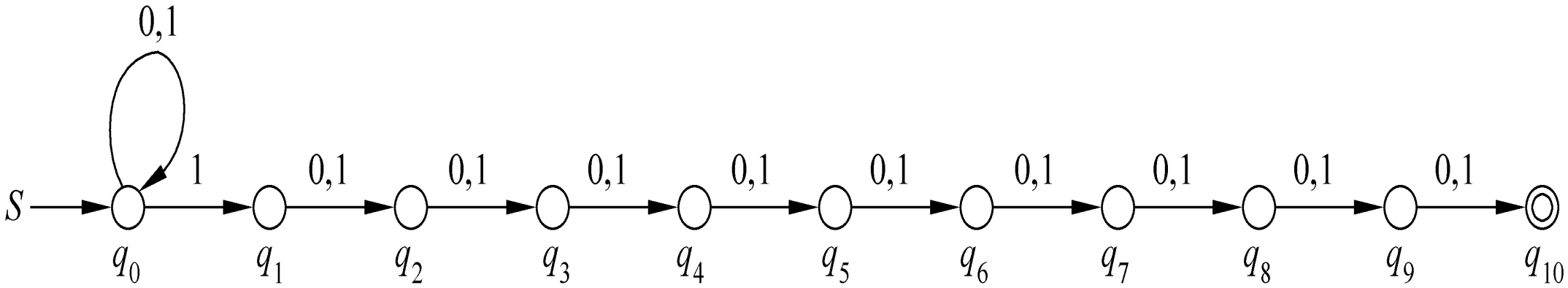




3.3.1 作为对DFA的修改

例3-13 构造接受$L=\{x|x\in\{0,1\}^*$,且x的倒数第十个字符为1}的FA。

困难：$x=\cdots a_{10}a_9a_8a_7a_6a_5a_4a_3a_2a_1$，需要记住最后10位，记录的状态太多。

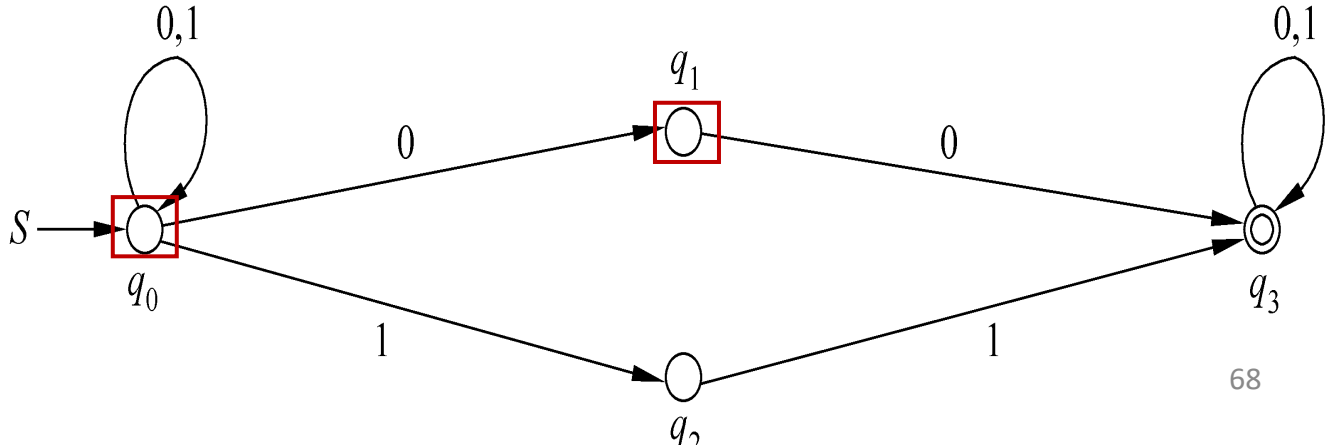


3.3.1 作为对DFA的修改

“FA”与DFA的区别在于：

1. 并不是对于所有的$(q,a) \in Q \times \Sigma$，$\delta(q,a)$ 都有状态与它对应；
2. 并不是对于所有的$(q,a) \in Q \times \Sigma$，$\delta(q,a)$只对应一个状态。

- “FA”在任意时刻可以处于有穷多个状态。
- “FA”具有“智能”。



3.3 不确定的有穷状态自动机

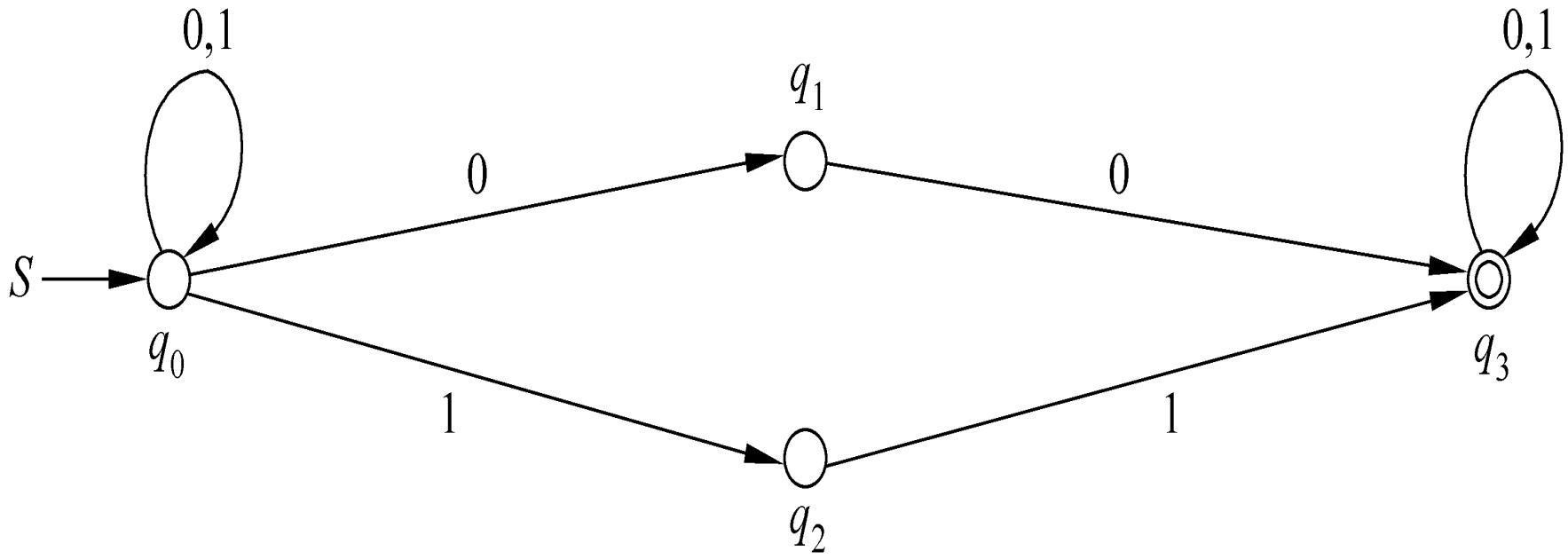
不确定的有穷状态自动机NFA：

定义3-4 不确定的有穷状态自动机（Non-deterministic Finite Automaton, NFA）M是一个五元组$M = (Q, \Sigma, \delta, q_0, F)$

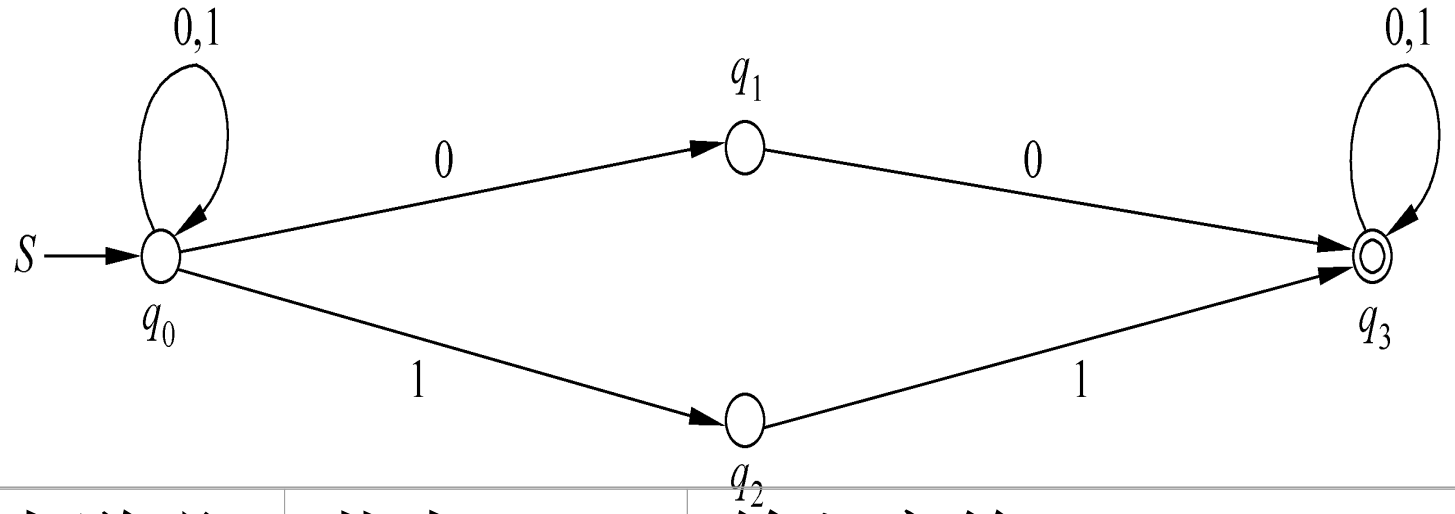
- 其中Q, Σ, q_0, F 的意义同DFA；
- δ：$Q \times \Sigma \to 2^Q$，对$\forall(q,a) \in Q \times \Sigma$，$\delta(q,a) = \{p_1, p_2, \cdots, p_m\}$表示M在状态$q$读入字符$a$，可以选择将状态变成$p_1$、或$p_2$、…、或$p_m$，并将读头向右移动一格，指向输入字符串的下一个字符。

- FA的状态转移图、状态对应的等价类、即时描述对NFA都有效。
- DFA是NFA的特例。

续例3-12 接受$L=\{x|x\in\{0,1\}^*$,且x含有子串00或11}的NFA对应的状态转移表。

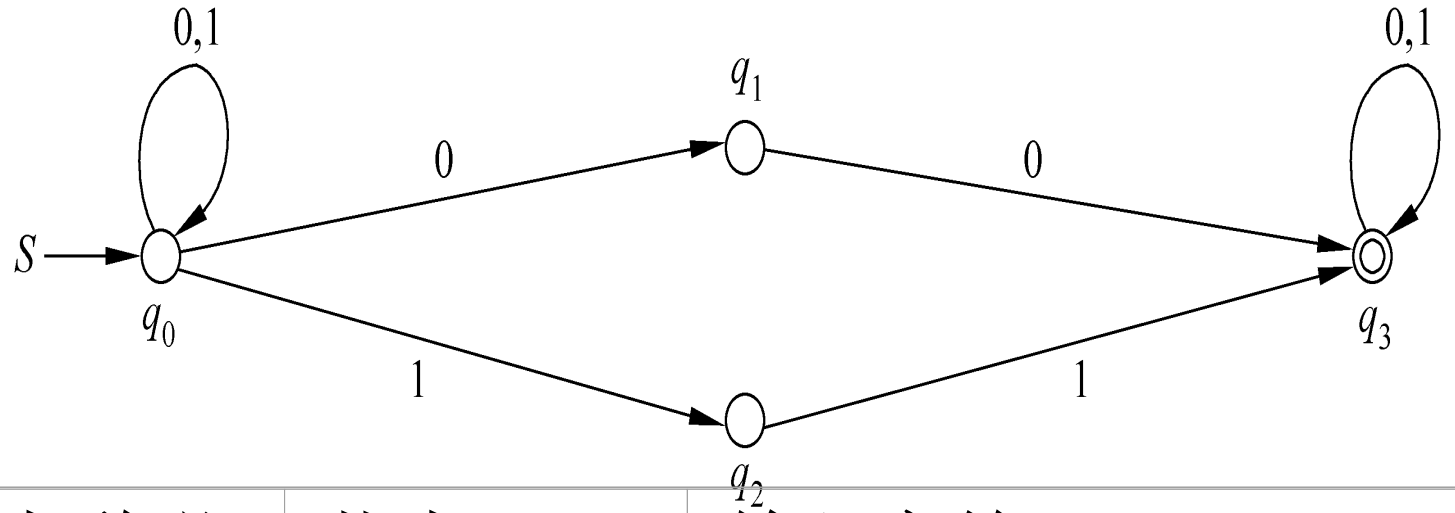


3.3.2 NFA的形式定义



状态说明	状态	输入字符	
		0	**1**
启动状态	q_0		
	q_1		
	q_2		
终止状态	q_3		

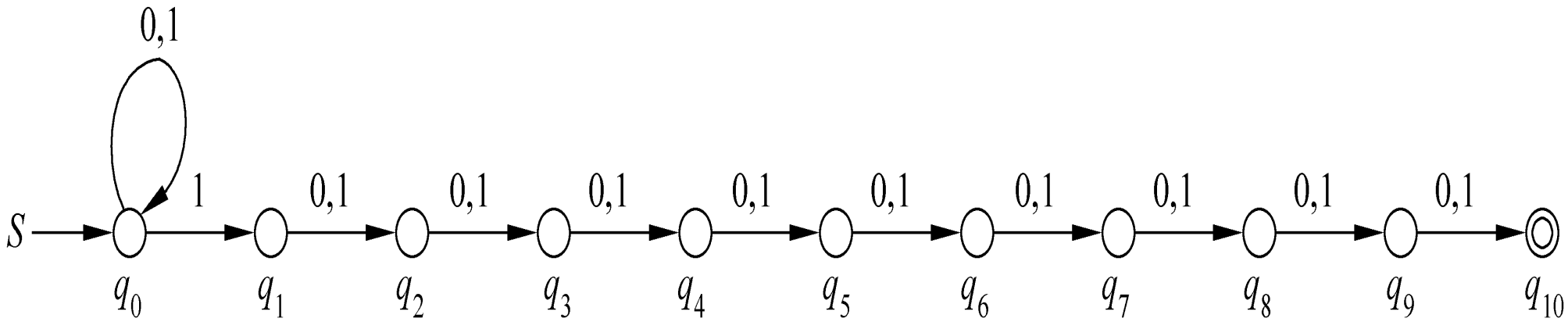
3.3.2 NFA的形式定义



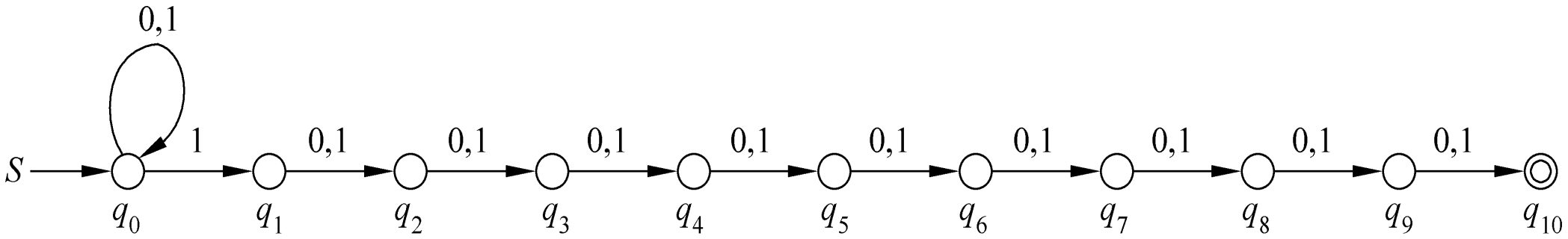
状态说明	状态	输入字符	
		0	1
启动状态	q_0	$\{q_0,q_1\}$	$\{q_0,q_2\}$
	q_1	$\{q_3\}$	Φ
	q_2	Φ	$\{q_3\}$
终止状态	q_3	$\{q_3\}$	$\{q_3\}$

3.3.2 NFA的形式定义

续例3-13 构造接受$L=\{x|x\in\{0,1\}^*$,且x的倒数第十个字符为1}的NFA对应的状态转移表。



3.3.2 NFA的形式定义



状态说明	状态	输入字符	
		0	1
启动状态	q_0	$\{q_0\}$	$\{q_0, q_1\}$
	q_1	$\{q_2\}$	$\{q_2\}$
	q_2	$\{q_3\}$	$\{q_3\}$
	q_3	$\{q_4\}$	$\{q_4\}$
	q_4	$\{q_5\}$	$\{q_5\}$
	q_5	$\{q_6\}$	$\{q_6\}$
	q_6	$\{q_7\}$	$\{q_7\}$
	q_7	$\{q_8\}$	$\{q_8\}$
	q_8	$\{q_9\}$	$\{q_9\}$
	q_9	$\{q_{10}\}$	$\{q_{10}\}$
终止状态	q_{10}	Φ	Φ

扩充定义域

在NFA中，同样需要将δ扩充为

$$\hat{\delta}: Q \times \Sigma^* \to 2^Q$$

对任意的$q \in Q, \omega \in \Sigma^*, a \in \Sigma$，定义

1. $\hat{\delta}(q, \varepsilon) = \{q\}$;
2. $\hat{\delta}(q, \omega a) = \{p | \exists r \in \hat{\delta}(q, \omega)\}$，使得$p \in \delta(r, a)\}$

扩充定义域

$$\blacksquare \hat{\delta}(q,a) = \hat{\delta}(q,\varepsilon a)$$

$$= \{p \mid \exists r \in \hat{\delta}(q,\varepsilon), p \in \delta(r,a)\}$$

$$= \{p \mid \exists r \in \{q\}, p \in \delta(r,a)\}$$

$$= \{p \mid p \in \delta(q,a)\}$$

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

■ 和DFA的结论一样，两值相同，也不用区分$\hat{\delta}$和δ这两个符号。

扩充定义域

■ 进一步扩充δ的定义域：

$$\delta: 2^Q \times \Sigma^* \to 2^Q$$

对任意的$\boldsymbol{P \subseteq Q, \omega \in \Sigma^*}$，

$$\delta(P, \omega) = \bigcup_{q \in P} \delta(q, \omega)$$

■ 由于对$\forall(\boldsymbol{q, \omega}) \in \boldsymbol{Q \times \Sigma^*}$

$$\delta(\{q\}, \omega) = \bigcup_{q \in \{q\}} \delta(q, \omega) = \delta(q, \omega)$$

所以，不一定严格地区分δ的第1个分量是一个状态还是一个含有一个元素的集合。

3.3.2 NFA的形式定义

NFA M接受(识别)的语言：

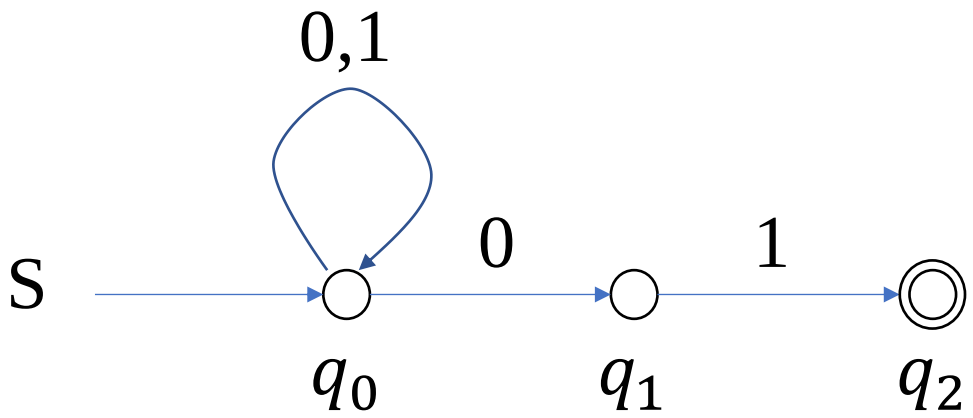
■对于$\forall x \in \Sigma^*$，

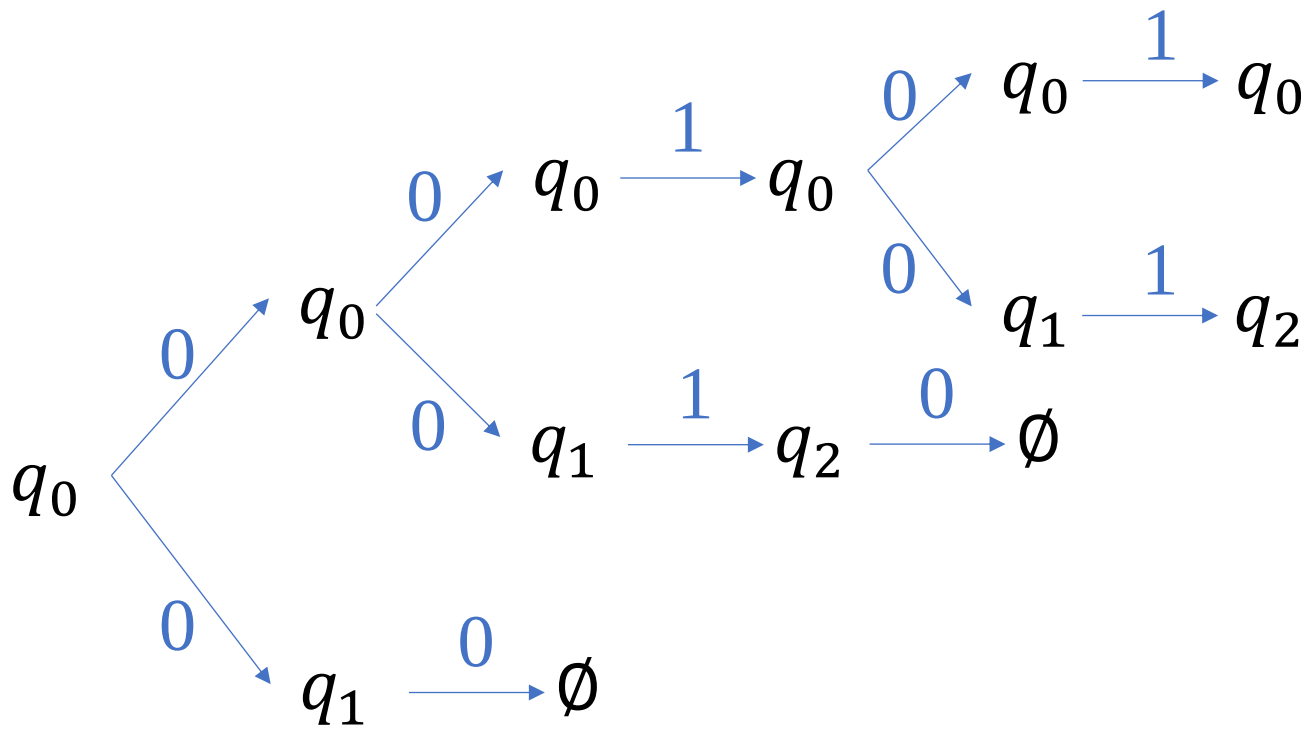
如果$\delta(q_0, x) \cap F \neq \emptyset$，则称$x$被M接受；

如果$\delta(q_0, x) \cap F = \emptyset$，则称M不接受$x$。

■$\mathrm{L(M)} = \{x | x \in \Sigma^*$且$\delta(q_0, x) \cap F \neq \emptyset\}$，称为由M接受(识别)的语言。

例3-14 接受全部以01结尾的字符串的NFA，识别字符串00101的过程。



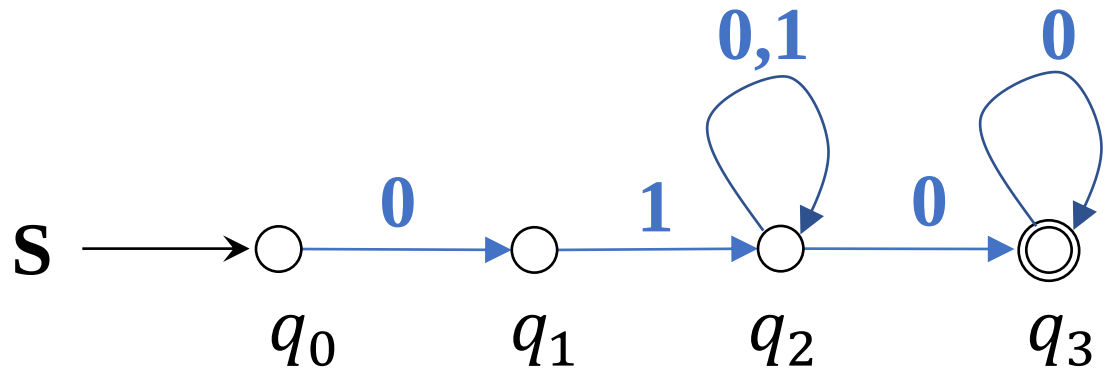


$\{q_0, q_2\} \cap F \neq \emptyset$，则00101为可由M识别的语言。

例3-15 构造NFA M，使它接受的语言为$\{01\omega 0^m | \omega \in \{0,1\}^*, m \geq 1\}$。

每个字符串可写为$01\omega 0 0^{m-1}, m-1 \geq 0$

- q_0：M的启动状态，等待0；
- q_1：M读到了一个0，等待1；
- q_2：M读到了一个1，可能读入0、1串，等待0；
- q_3：M读到了一个0，可能读入多个0



3.3 不确定的有穷状态自动机

3.3.3 NFA与DFA等价

■对于一个输入字符，NFA与DFA的差异是前者可以进入若干个状态，而后者只能进入一个唯一状态。

■虽然NFA在某一时刻同时进入若干个状态，但是，这若干个状态合在一起的“总效果”相当于它处于这些状态对应的一个“综合状态”。因此，我们考虑让DFA用一个状态去对应NFA的一组状态。

NFA 与DFA的对应关系：

$$\mathbf{NFA:}\ \mathbf{M_1} = (Q, \Sigma, \delta_1, q_0, F_1)$$

$$\mathbf{DFA:}\ \mathbf{M_2} = (Q_2, \Sigma, \delta_2, q_0', F_2)$$

- **NFA**从开始状态 q_0 启动，我们就让相应的**DFA**从状态 $[q_0]$ 启动。所以$q_0' = [q_0]$。（用[]表示DFA的一个状态）
- 对于**NFA**的一个状态组$\{q_1, q_2, \cdots, q_n\}$, 如果**NFA**在此状态组时读入字符$a$后可以进入状态组$\{p_1, p_2, \cdots, p_m\}$, 则让相应的**DFA**在状态$[q_1, q_2, \cdots, q_n]$读入字符$a$时，进入状态$[p_1, p_2, \cdots, p_m]$。

3.3.3 NFA与DFA等价

定理 3-1 NFA与DFA等价。

- **NFA: $\mathbf{M_1} = (Q, \Sigma, \delta_1, q_0, F_1)$**
- **DFA: $\mathbf{M_2} = (Q_2, \Sigma, \delta_2, q_0', F_2)$**

证明：

（1）构造与NFA $\mathbf{M_1}$等价的DFA $\mathbf{M_2}$。

- $Q_2 = 2^Q$,
- $F_2 = \{[p_1, p_2, \cdots, p_m] | \{p_1, p_2, \cdots, p_m\} \subseteq \mathrm{Q} \,\&\, \{p_1, p_2, \cdots, p_m\} \cap F_1 \neq \emptyset\}$
- $\delta_2([q_1, q_2, \cdots, q_n], a) = [p_1, p_2, \cdots, p_m] \Leftrightarrow \delta_1(\{q_1, q_2, \cdots, q_n\}, a) = \{p_1, p_2, \cdots, p_m\}$

定理 3-1 NFA与DFA等价。

- **NFA: $\mathbf{M_1} = (Q, \Sigma, \delta_1, q_0, F_1)$**
- **DFA: $\mathbf{M_2} = (Q_2, \Sigma, \delta_2, q_0', F_2)$**

证明：

（2）证明$\delta_1(q_0, x) = \{p_1, p_2, \cdots, p_m\} \Leftrightarrow$

$\delta_2([q_0], x) = [p_1, p_2, \cdots, p_m]$，设 $x \in \Sigma^*$，施归纳于$|x|$：

- $x = \varepsilon$，$\delta_1(q_0, \varepsilon) = \{q_0\}$，$\delta_2([q_0], \varepsilon) = [q_0]$
- 若当 $|x| = n$ 时结论成立，下面证明当 $|x| = n + 1$ 时结论也成立。
- 不妨设$x = \omega a, |\omega| = n, a \in \Sigma$，则

$$\delta_1(q_0, \omega a) = \delta_1(\delta_1(q_0, \omega), a) = \delta_1(\{q_1, q_2, \cdots, q_n\}, a)$$
$$= \{p_1, p_2, \cdots, p_m\}$$

定理 3-1 NFA与DFA等价。

- **NFA: $\mathbf{M_1} = (Q, \Sigma, \delta_1, q_0, F_1)$**
- **DFA: $\mathbf{M_2} = (Q_2, \Sigma, \delta_2, q_0', F_2)$**

证明：

- 由归纳假设，$\delta_1(q_0, \omega) = \{q_1, q_2, \cdots, q_n\} \Leftrightarrow \delta_2([q_0], \omega) = [q_1, q_2, \cdots, q_n]$
- 根据δ_2的定义：
  $\delta_2([q_1, q_2, \cdots, q_n], a) = [p_1, p_2, \cdots, p_m] \Leftrightarrow \delta_1(\{q_1, q_2, \cdots, q_m\}, a) = \{p_1, p_2, \cdots, p_m\}$
- 所以，$\delta_2([q_0], \omega a) = \delta_2(\delta_2([q_0], \omega), a) = \delta_2([q_1, q_2, \cdots, q_n], a) = [p_1, p_2, \cdots, p_m]$

3.3.3 NFA与DFA等价

定理 3-1 NFA与DFA等价。

- **NFA: $\mathbf{M_1} = (Q, \Sigma, \delta_1, q_0, F_1)$**
- **DFA: $\mathbf{M_2} = (Q_2, \Sigma, \delta_2, q_0', F_2)$**

证明：

- 故，如果$\delta_1(q_0, \omega a) = \{p_1, p_2, \cdots, p_m\}$，则必有$\delta_2([q_0], \omega a) = [p_1, p_2, \cdots, p_m]$。
- 由上述推导可知，反向的推导也成立。这就是说，结论对$|x| = n + 1$也成立。
- 由归纳法原理，结论对$\forall x \in \Sigma^*$成立。

定理 3-1 NFA与DFA等价。

- **NFA: $\mathbf{M_1} = (Q, \Sigma, \delta_1, q_0, F_1)$**
- **DFA: $\mathbf{M_2} = (Q_2, \Sigma, \delta_2, q_0', F_2)$**

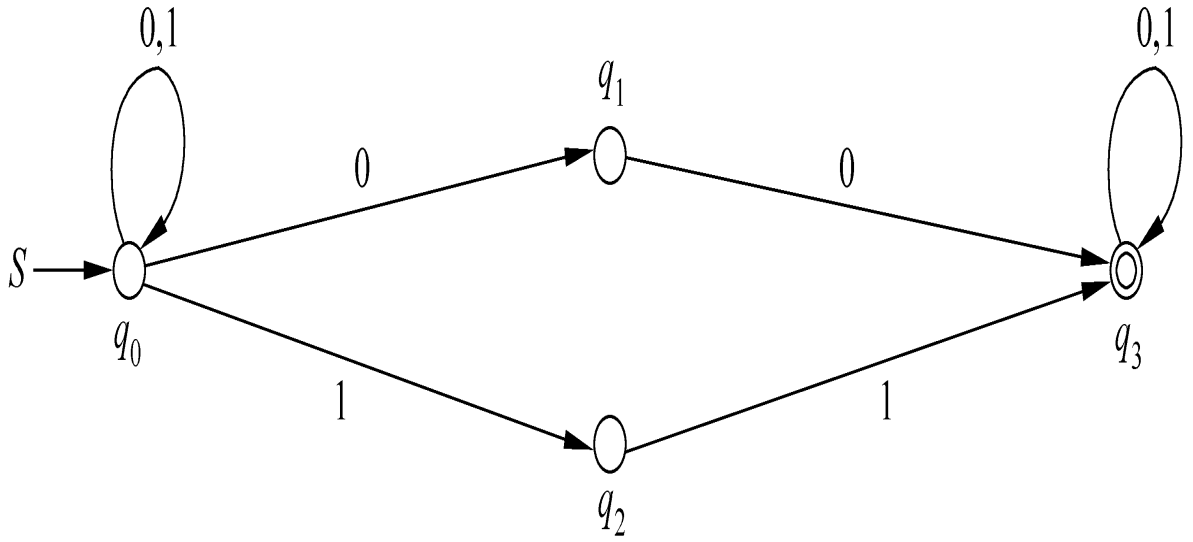
证明：

（3）证明$\mathbf{L(M_1) = L(M_2)}$

- 设$x \in \mathbf{L(M_1)}$，且$\delta_1(q_0, x) = \{p_1, p_2, \cdots, p_m\}$，从而$\delta_1(q_0, x) \cap F_1 \neq \emptyset$，即$\{p_1, p_2, \cdots, p_m\} \cap F_1 \neq \emptyset$。由$F_2$的定义，$[p_1, p_2, \cdots, p_m] \in F_2$，即$\delta_2([q_0], x) = [p_1, p_2, \cdots, p_m] \in F_2$。所以，$x \in \mathbf{L(M_2)}$，故$\mathbf{L(M_1) \subseteq L(M_2)}$。
- 类似可得$\mathbf{L(M_2) \subseteq L(M_1)}$。即$\mathbf{L(M_1) = L(M_2)}$得证。

综上所述，定理成立。

例3-16 下图的NFA对应的DFA的状态转移函数如表3-7所示。



3.3.3 NFA与DFA等价

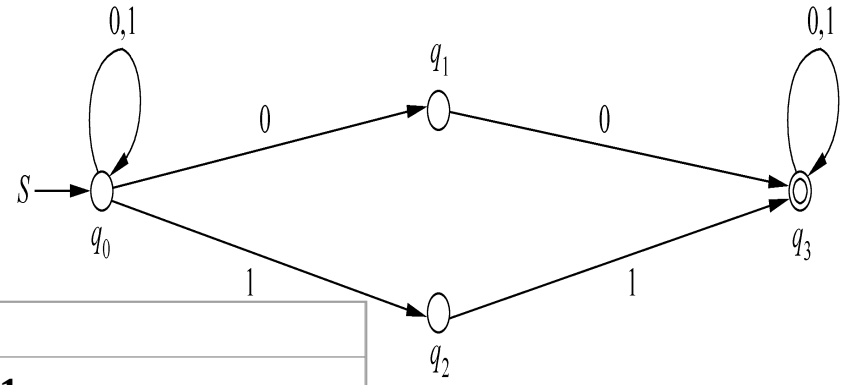


表3-7　状态转移函数

状态说明		状态	输入字符	
			0	1
启动	√	$[q_0]$	$[q_0,q_1]$	$[q_0,q_2]$
		$[q_1]$	$[q_3]$	$[\Phi]$
		$[q_2]$	$[\Phi]$	$[q_3]$
终止		$[q_3]$	$[q_3]$	$[q_3]$
	√	$[q_0,q_1]$	$[q_0,q_1,q_3]$	$[q_0,q_2]$
	√	$[q_0,q_2]$	$[q_0,q_1]$	$[q_0,q_2,q_3]$
终止		$[q_0,q_3]$	$[q_0,q_1,q_3]$	$[q_0,q_2,q_3]$
		$[q_1,q_2]$	$[q_3]$	$[q_3]$
终止		$[q_1,q_3]$	$[q_3]$	$[q_3]$
终止		$[q_2,q_3]$	$[q_3]$	$[q_3]$
		$[q_0,q_1,q_2]$	$[q_0,q_1,q_3]$	$[q_0,q_2,q_3]$
终止	√	$[q_0,q_1,q_3]$	$[q_0,q_1,q_3]$	$[q_0,q_2,q_3]$
终止	√	$[q_0,q_2,q_3]$	$[q_0,q_1,q_3]$	$[q_0,q_2,q_3]$
终止		$[q_1,q_2,q_3]$	$[q_3]$	$[q_3]$
终止		$[q_0,q_1,q_2,q_3]$	$[q_0,q_1,q_3]$	$[q_0,q_2,q_3]$
		$[\Phi]$	$[\Phi]$	$[\Phi]$

3.3.3 NFA与DFA等价

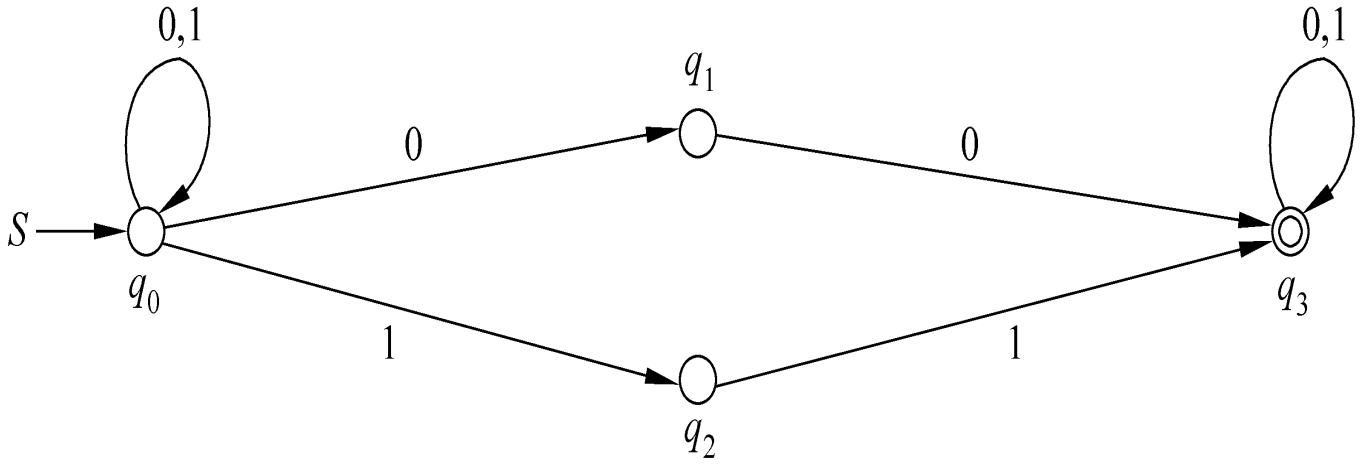
- 不可达状态(inaccessible state)：不存在从$[q_0]$对应的顶点出发，到达该状态对应的顶点的路。我们称此状态从开始状态是不可达的。
- 表3-7中，所有标记“√”的状态是从开始状态可达的，其他是不可达的——无用的。

上页表格包含太多不可达状态，无意义。

子集构造法：构造与给定NFA等价的DFA

1. 先只把开始状态$[q_0]$填入表的状态列中；
2. 如果表中所列的状态列有未处理的，则任选一个未处理的状态$[q_1, q_2, \cdots, q_n]$，对Σ中的每个字符a，计算$\delta([q_1, q_2, \cdots, q_n], a)$，并填入相应的表项中；
3. 如果$\delta([q_1, q_2, \cdots, q_n], a)$在表的状态列未出现过，则将它填入表的状态列；
4. 如此重复下去，直到表的状态列中不存在未处理的状态。

续例3-16 利用子集构造法将下图的NFA转化为DFA。



1. NFA的状态转移表：

	NFA状态	0	1
开始	q_0	$\{q_0, q_1\}$	$\{q_0, q_2\}$
	q_1	$\{q_3\}$	$\emptyset$
	q_2	$\emptyset$	$\{q_3\}$
终止	q_3	$\{q_3\}$	$\{q_3\}$

3.3.3 NFA与DFA等价

1. NFA状态转移表：

	NFA状态	0	1
开始	q_0	$\{q_0, q_1\}$	$\{q_0, q_2\}$
	q_1	$\{q_3\}$	$\emptyset$
	q_2	$\emptyset$	$\{q_3\}$
终止	q_3	$\{q_3\}$	$\{q_3\}$

2. DFA状态转移表：

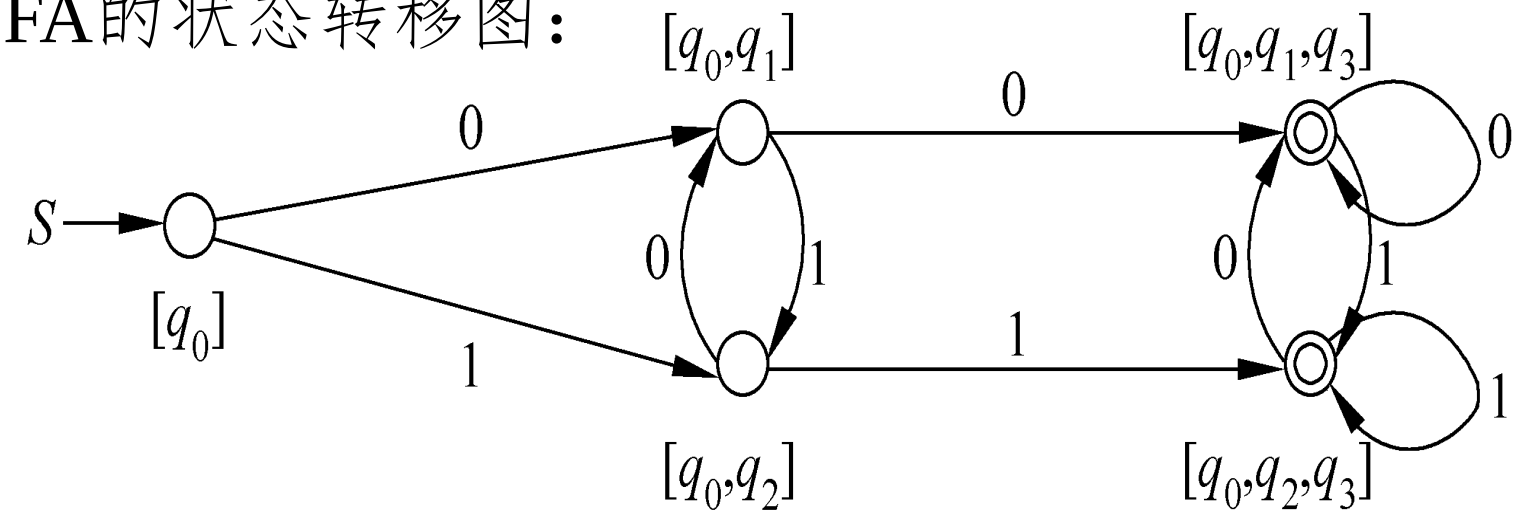
	DFA状态	0	1
开始	$[q_0]$		

3.3.3 NFA与DFA等价

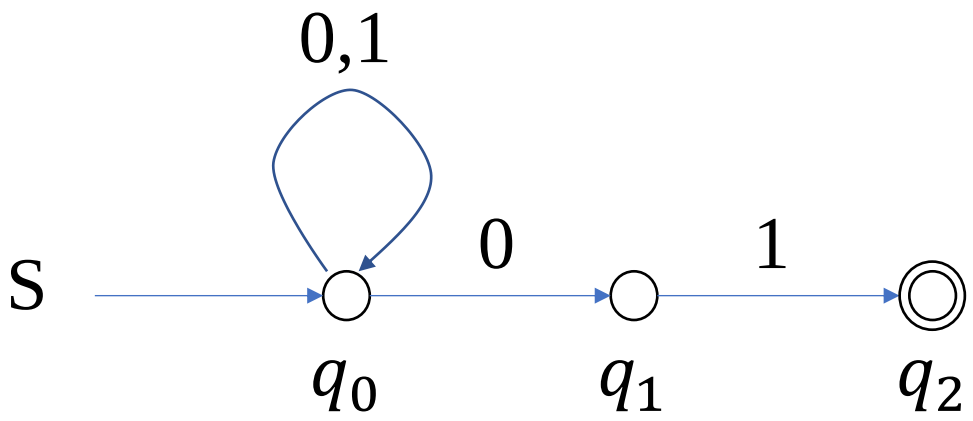
2. DFA的状态转移表：

	DFA状态	0	1
开始	$[q_0]$	$[q_0, q_1]$	$[q_0, q_2]$
	$[q_0, q_1]$	$[q_0, q_1, q_3]$	$[q_0, q_2]$
	$[q_0, q_2]$	$[q_0, q_1]$	$[q_0, q_2, q_3]$
终止	$[q_0, q_1, q_3]$	$[q_0, q_1, q_3]$	$[q_0, q_2, q_3]$
终止	$[q_0, q_2, q_3]$	$[q_0, q_1, q_3]$	$[q_0, q_2, q_3]$

3. DFA的状态转移图：



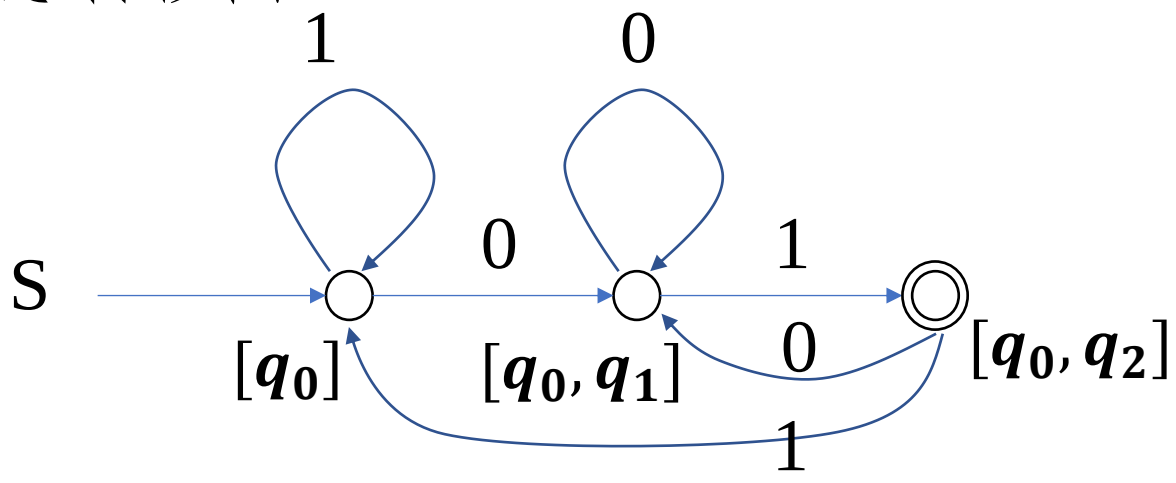
例3-17 利用子集构造法将下图的NFA转化为DFA。



1. DFA状态转移表：

	DFA状态	0	1
开始	$[q_0]$		

2. DFA状态转移图

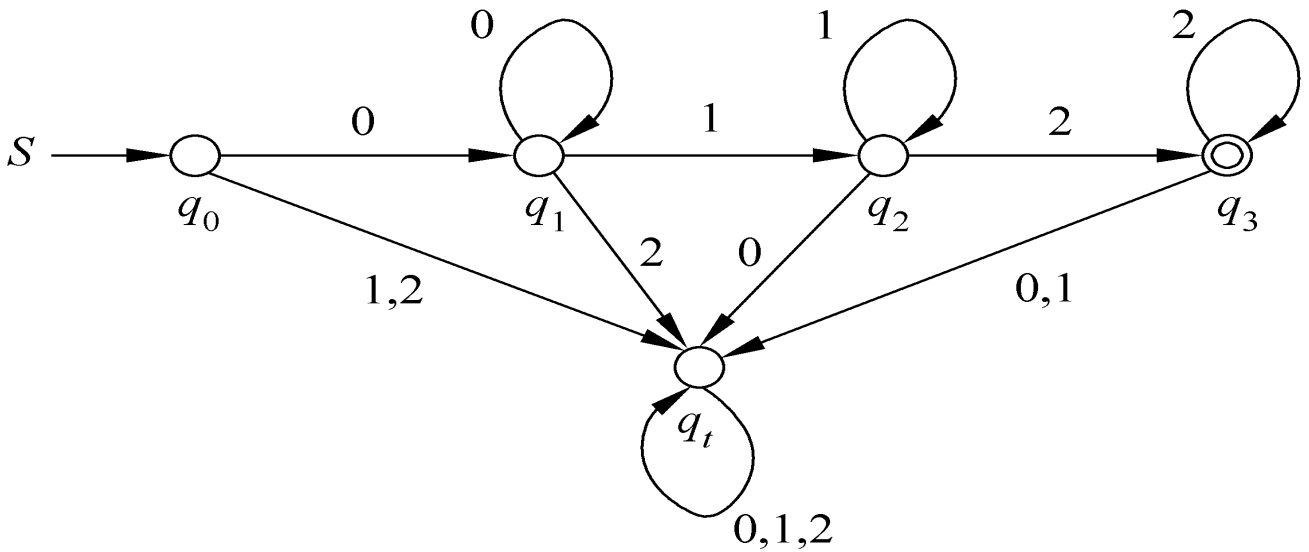


章节目录

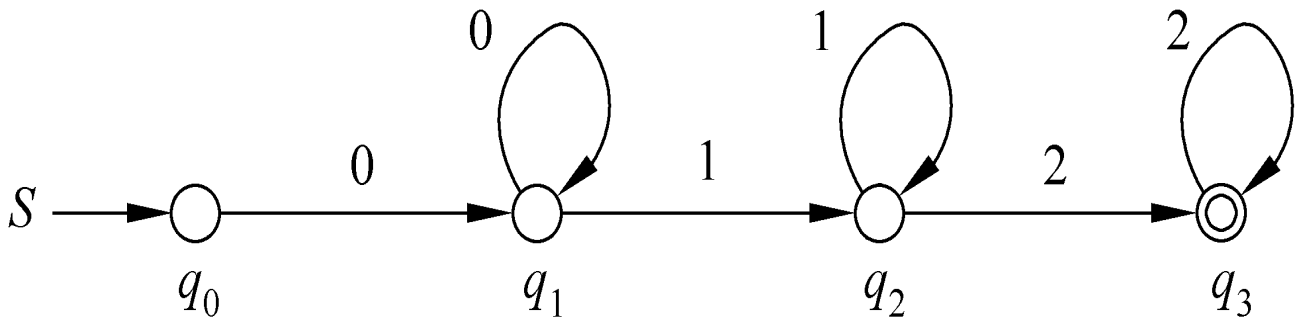
续例3-9 构造接受语言$L(M)=\{0^n1^m2^k, n,m,k\geq 1\}$的DFA和NFA。

- **DFA:**

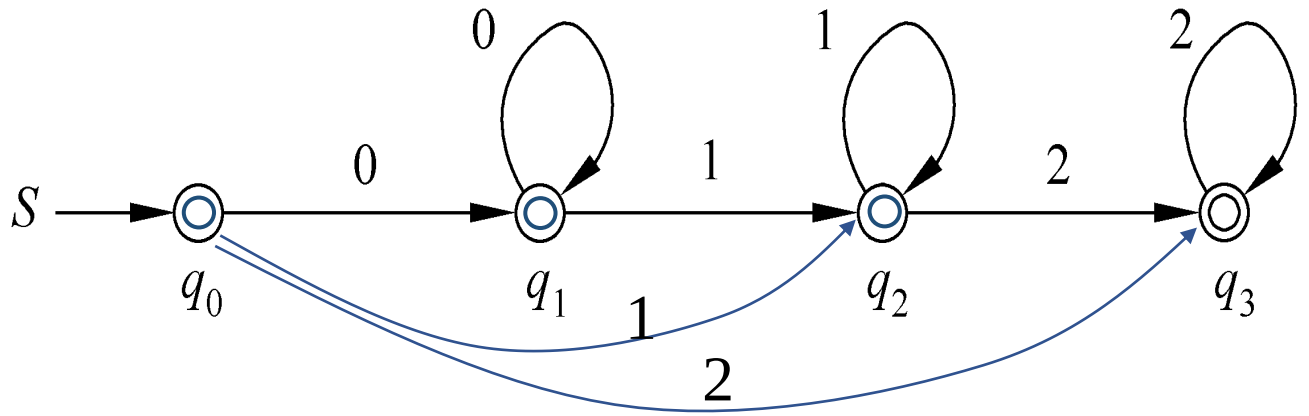
  例3-7



- **NFA:**

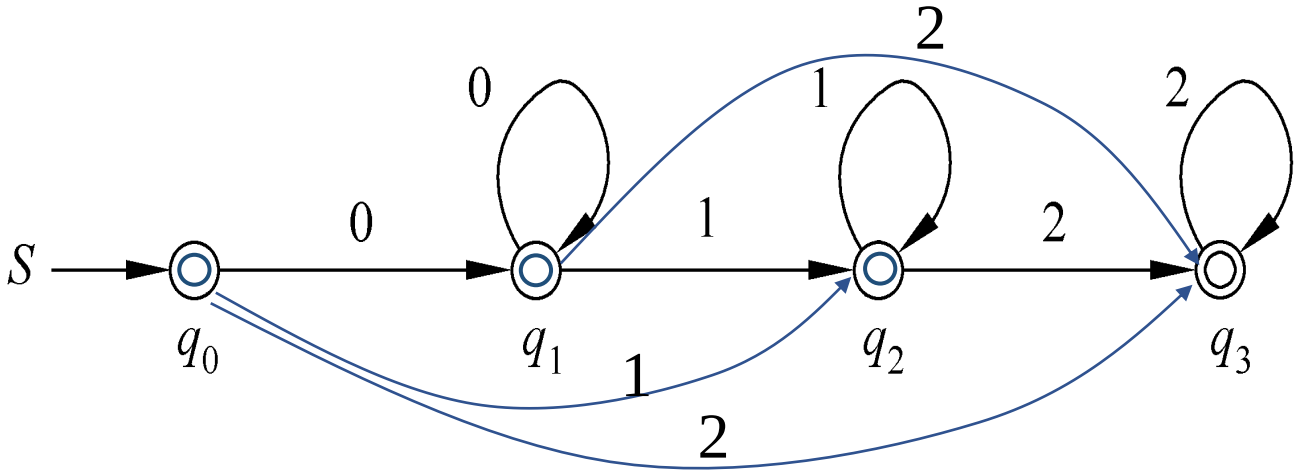


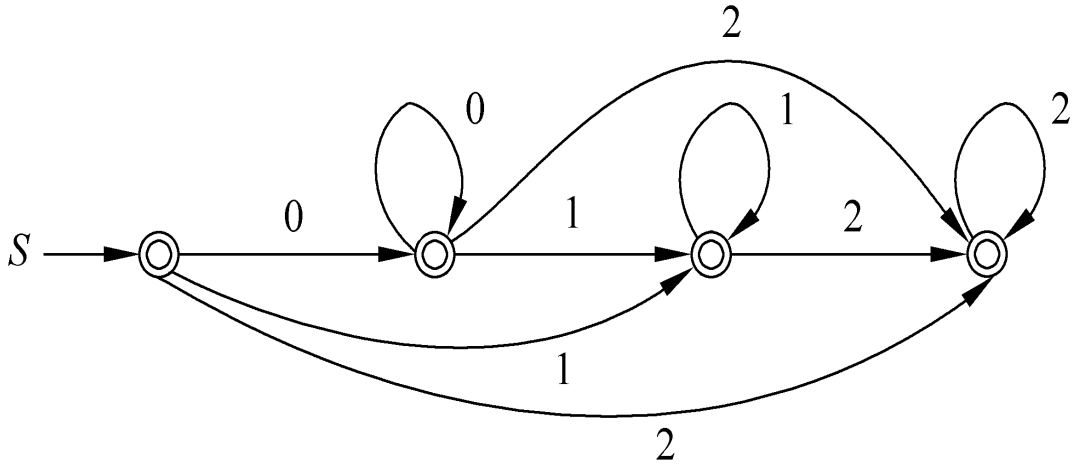
例3-18 构造接受语言$L(M)=\{0^n1^m2^k, n,m,k\geq 0\}$的NFA。



$n=m=k=0$	ε	q_0为终止态
$k\geq 1, n=m=0$	2^k	$q_0\rightarrow q_3$的边2
$m\geq 1, n=k=0$	1^m	$q_0\rightarrow q_2$的边1， q_2为终止态
$n\geq 1, m=k=0$	0^n	q_1为终止态

例3-18 构造接受语言$L(M)=\{0^n1^m2^k, n,m,k\geq 0\}$的NFA。



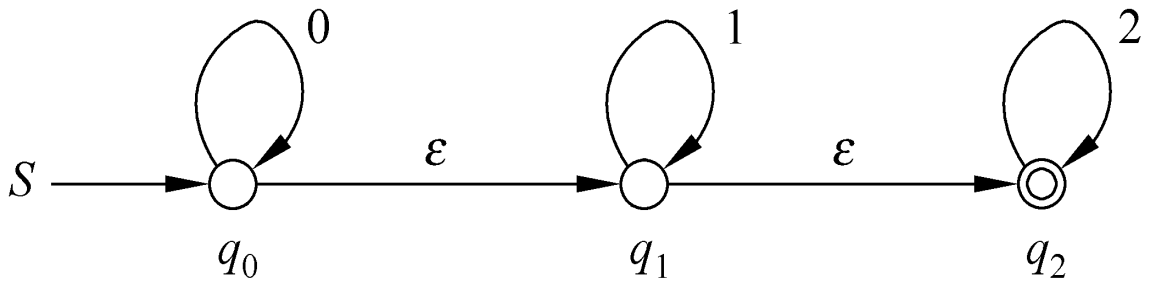


$n=0,m,k\geq 1$ 1^m2^k $q_0\rightarrow q_2$的边1

$m=0,n,k\geq 1$ 0^n2^k $q_1\rightarrow q_3$的边2

$k=0,n,m\geq 1$ 0^n1^m q_2为终止态

- 接受语言$L(M)=\{0^n1^m2^k, n,m,k\geq 0\}$的NFA是否可以构造成下图所示的“ε-NFA ”?



- 其构造显然比例3-18中NFA更容易，当然还希望他们是等价的。

定义3-5 带空移动的不确定有穷状态自动机（Non-deterministic Finite Automaton with ε-moves, ε-NFA）M是一个五元组$\mathrm{M} = (Q, \Sigma, \delta, q_0, F)$

- 其中Q, Σ, q_0, F 的意义同NFA；
- δ：状态转移函数，$Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$。

非空移动

■$\forall(q,a) \in Q \times \Sigma$

■$\delta(q,a) = \{p_1, p_2, \cdots, p_m\}$

■表示M在状态q读入字符a，可以选择地将状态变成p_1、或者p_2、…、或者p_m，并将读头向右移动一格而指向输入字符串的下一个字符。

空移动

- $\forall q \in Q$
- $\delta(q, \varepsilon) = \{p_1, p_2, \cdots, p_m\}$
- 表示M在状态q不读入任何字符，可以选择地将状态变成p_1、或者p_2、…、或者p_m，读头不动。

Callback 定义3-5

δ：$Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$。

扩充定义域

进一步扩充$\boldsymbol{\delta}$的定义域：$\hat{\boldsymbol{\delta}}: \boldsymbol{Q} \times \boldsymbol{\Sigma}^* \to \boldsymbol{2^Q}$，对任意的$\boldsymbol{P} \subseteq \boldsymbol{Q}, \boldsymbol{q} \in \boldsymbol{Q}, \boldsymbol{\omega} \in \boldsymbol{\Sigma}^*, \boldsymbol{a} \in \boldsymbol{\Sigma}$，定义：

$\boldsymbol{\varepsilon}$-CLOSURE$(\boldsymbol{q}) = \{\boldsymbol{p}|$从$\boldsymbol{q}$到$\boldsymbol{p}$有一条标记为$\boldsymbol{\varepsilon}$的路$\}$

$\boldsymbol{\varepsilon}$-CLOSURE$(\boldsymbol{P}) = \bigcup_{\boldsymbol{p} \in \boldsymbol{P}} \boldsymbol{\varepsilon}$-CLOSURE$(\boldsymbol{p})$

1. $\hat{\boldsymbol{\delta}}(\boldsymbol{q}, \boldsymbol{\varepsilon}) = \boldsymbol{\varepsilon}$-CLOSURE$(\boldsymbol{q})$
2. $\hat{\boldsymbol{\delta}}(\boldsymbol{q}, \boldsymbol{\omega a}) = \boldsymbol{\varepsilon}$-CLOSURE$(\boldsymbol{P})$

其中，$\boldsymbol{P} = \{\boldsymbol{p}|\exists \boldsymbol{r} \in \hat{\boldsymbol{\delta}}(\boldsymbol{q}, \boldsymbol{\omega})$使得$\boldsymbol{p} \in \boldsymbol{\delta}(\boldsymbol{r}, \boldsymbol{a})\}$

$= \bigcup_{\boldsymbol{r} \in \hat{\boldsymbol{\delta}}(\boldsymbol{q}, \boldsymbol{\omega})} \boldsymbol{\delta}(\boldsymbol{r}, \boldsymbol{a})$

Callback 定义3-5

δ：$Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$。

扩展转移函数

■进一步扩展转移函数：$2^Q \times \Sigma \to 2^Q$，对任意的$(P, a) \in 2^Q \times \Sigma$：

$$\delta(P, a) = \bigcup_{q \in P} \delta(q, a)$$

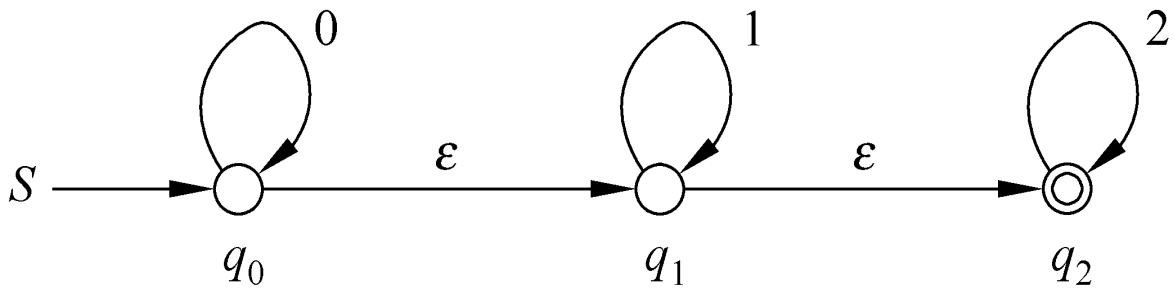
■再进一步扩展转移函数：$2^Q \times \Sigma^* \to 2^Q$，对任意的$(P, \omega) \in 2^Q \times \Sigma^*$：

$$\hat{\delta}(P, \omega) = \bigcup_{q \in P} \hat{\delta}(q, \omega)$$

- 在ε-NFA中，对任意$a \in \Sigma, q \in Q$，$\hat{\delta}(q, \varepsilon) \neq \delta(q, \varepsilon)$，$\hat{\delta}(q, a) \neq \delta(q, a)$，$\hat{\delta}$与$\delta$是不同的，需要区分。

例3-19 ε-NFA如右图，

观察$\hat{\delta}$与δ的区别。



状态	δ $\Sigma \cup \{\varepsilon\}$				$\hat{\delta}$ Σ^*			
	ε	0	1	2	ε	0	1	2
q_0	$\{q_1\}$	$\{q_0\}$	Φ	Φ	$\{q_0,q_1,q_2\}$	$\{q_0,q_1,q_2\}$	$\{q_1,q_2\}$	$\{q_2\}$
q_1	$\{q_2\}$	Φ	$\{q_1\}$	Φ	$\{q_1,q_2\}$	Φ	$\{q_1,q_2\}$	$\{q_2\}$
q_2	Φ	Φ	Φ	$\{q_2\}$	$\{q_2\}$	Φ	Φ	$\{q_2\}$

ε-NFA接受的语言

设$M=(Q,\Sigma,\delta,q_0,F)$是一个ε-NFA，对于$\forall x\in\Sigma^*$，

如果$\hat{\delta}(q_0,x)\cap F\neq\emptyset$，则称$x$被M接受；

如果$\hat{\delta}(q_0,x)\cap F=\emptyset$，则称M不接受$x$。

$$L(M)=\{x|x\in\Sigma^*\text{且}\hat{\delta}(q_0,x)\cap F\neq\emptyset\}$$

- 不难看出，NFA是一种特殊的ε-NFA 。

定理 3-2 ε-NFA与NFA等价。

思路：

- 设有**ε-NFA $\mathbf{M_1} = (Q, \Sigma, \delta_1, q_0, F)$**

（1）构造与之等价的**NFA $\mathbf{M_2} = (Q, \Sigma, \delta_2, q_0, F_2)$**。

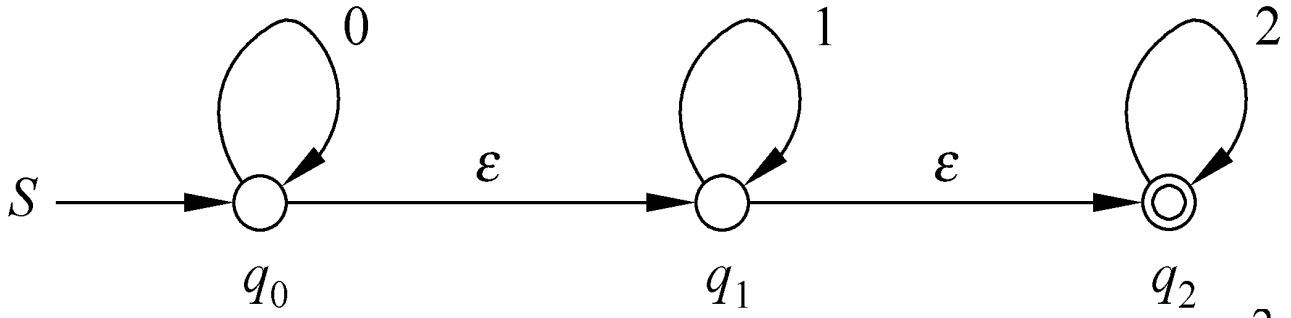
（2）证明对$\forall x \in \Sigma^+, \delta_2(q_0, x) = \widehat{\delta_1}(q_0, x)$

（3）证明对$\forall x \in \Sigma^+$,

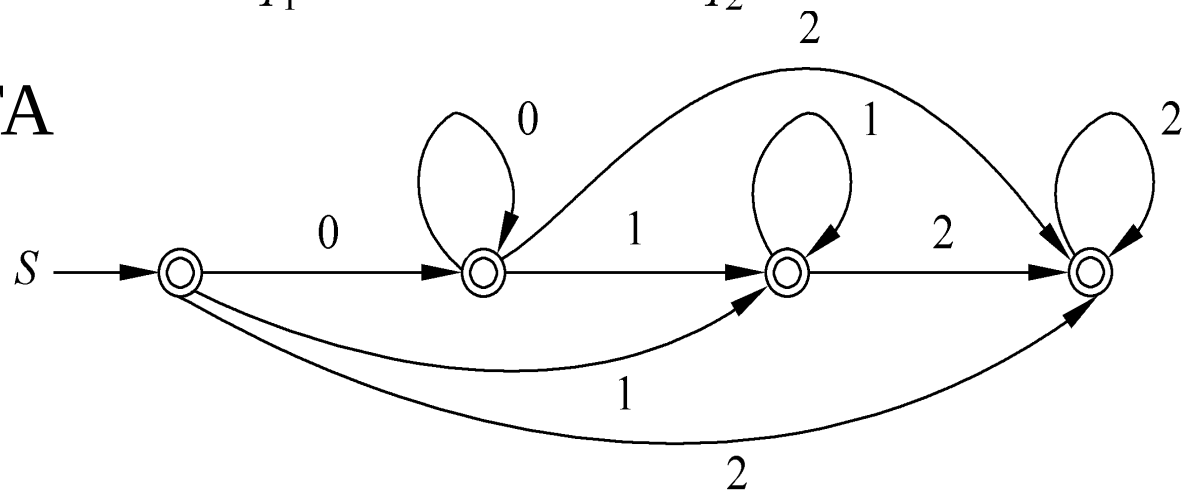
$$\delta_2(q_0, x) \cap F_2 \neq \emptyset \iff \widehat{\delta_1}(q_0, x) \cap F \neq \emptyset。$$

（4）证明$\varepsilon \in L(M_1) \Leftrightarrow \varepsilon \in L(M_2)$

例3-20 求与接受语言$L(M)=\{0^n1^m2^k, n,m,k\geq 0\}$的ε-NFA 等价的NFA。

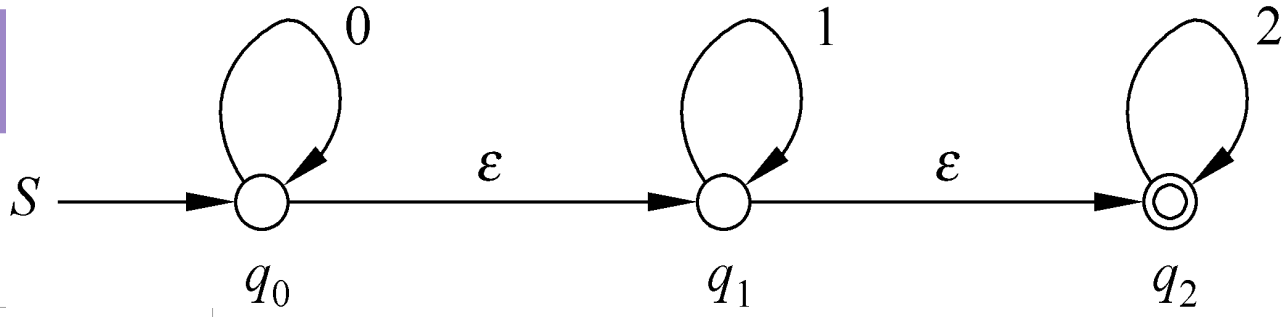


例3-18已给出一个NFA



是否能给出从**ε-NFA**转化为**NFA**的普适性方法？

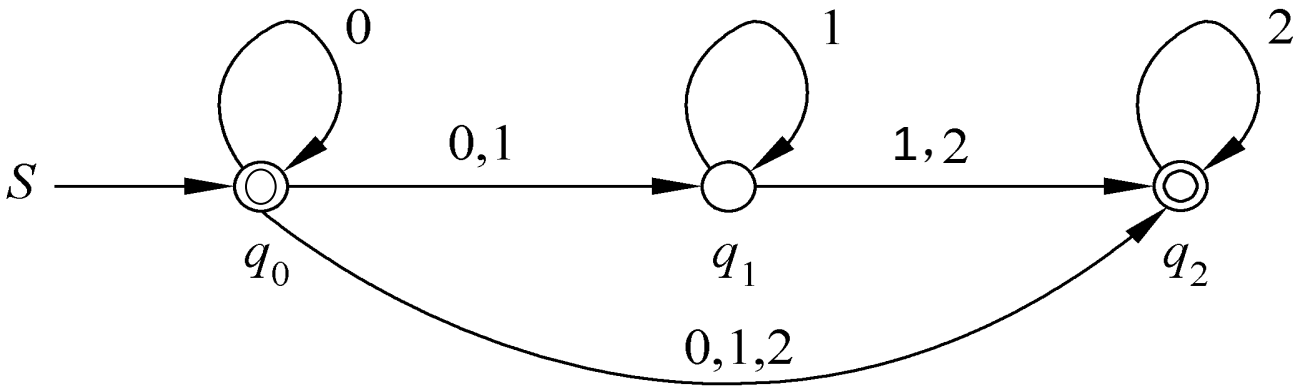
3.4 带空移动的有穷状态自动机



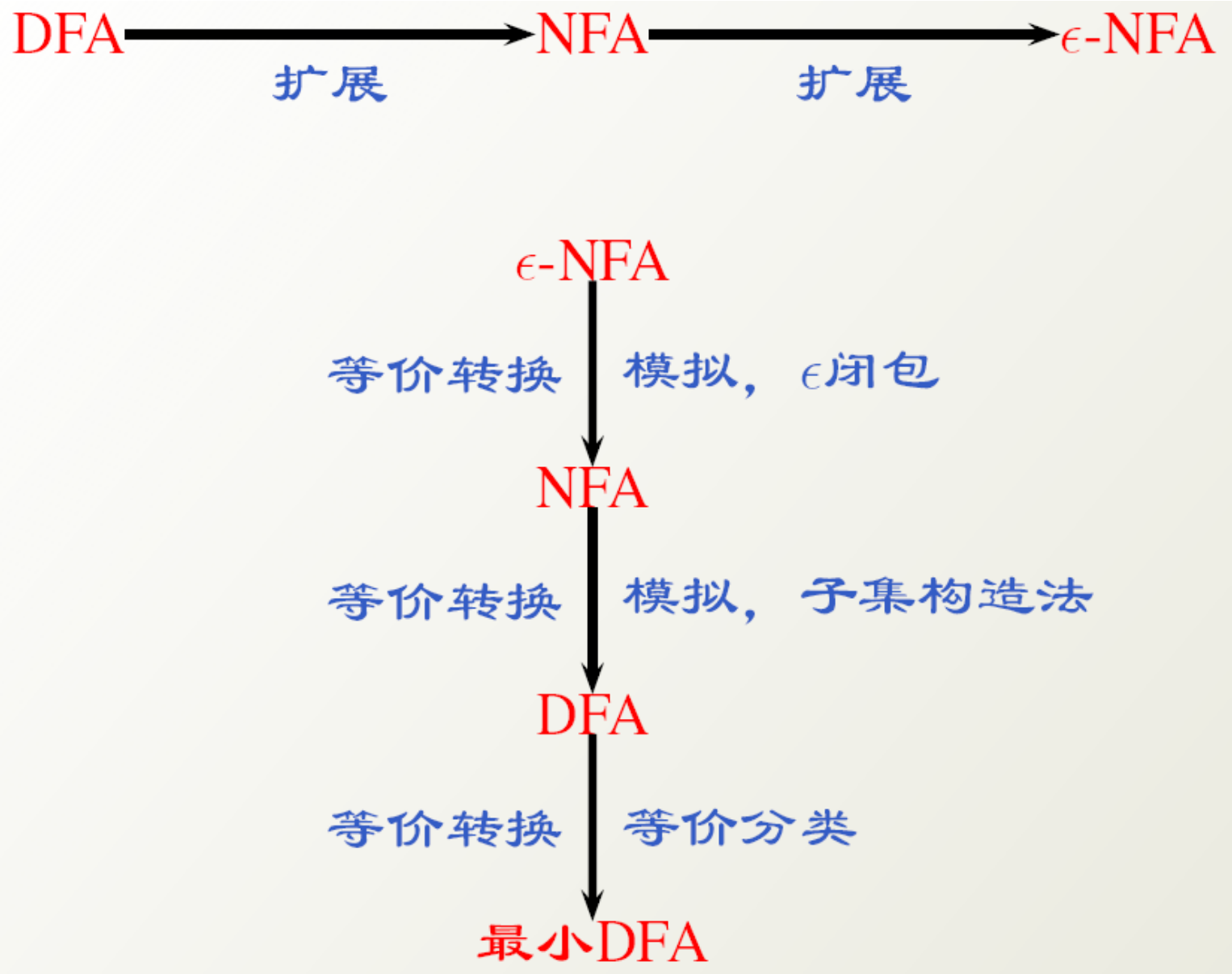
例3-19中已给出$\hat{\delta}$

状态	$\hat{\delta}$		
	0	**1**	**2**
q_0	$\{q_0,q_1,q_2\}$	$\{q_1,q_2\}$	$\{q_2\}$
q_1	Φ	$\{q_1,q_2\}$	$\{q_2\}$
q_2	Φ	Φ	$\{q_2\}$

原状态	后状态	可输入的字符
q_0	q_0	0
	q_1	0，1
	q_2	0，1，2
q_1		
	q_1	1
	q_2	1，2
q_2		
	q_2	2

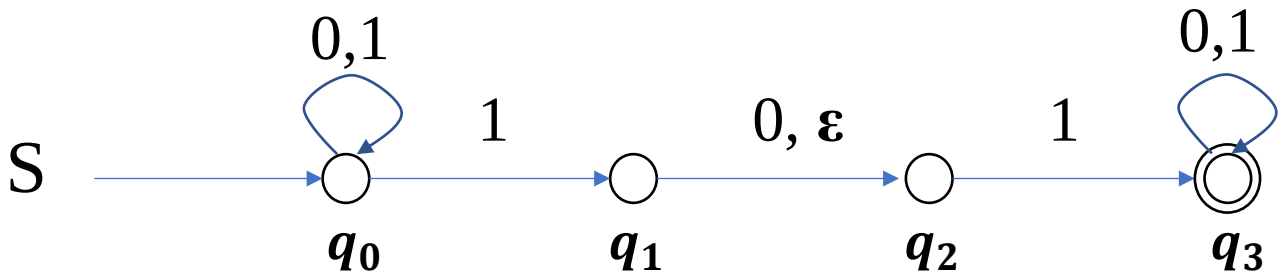


3.4 带空移动的有穷状态自动机



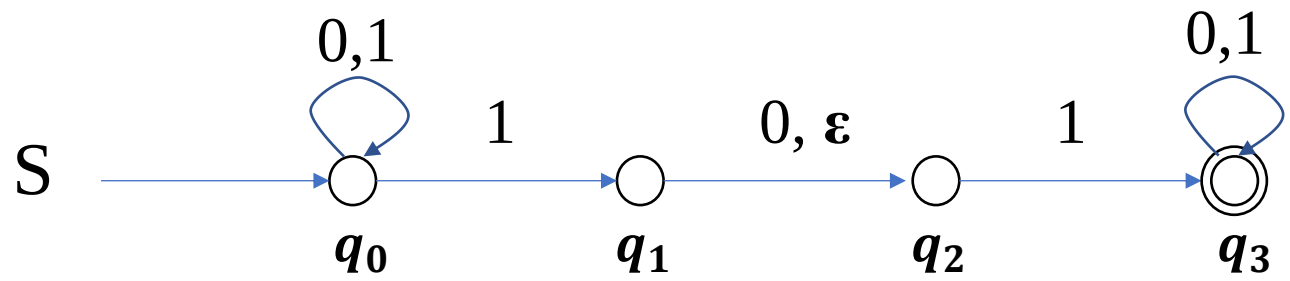
课堂练习

课堂练习：求与下方ε-NFA 等价的DFA。



课堂练习

课堂练习：求与下方ε-NFA 等价的DFA。



ε-NFA的状态转移图 → NFA的状态转移表→ DFA的状态转移表

NFA的状态转移表

	状态	0	1
开始	q_0	$\{q_0\}$	$\{q_0, q_1, q_2\}$
	q_1	$\{q_2\}$	$\{q_3\}$
	q_2	$\emptyset$	$\{q_3\}$
终止	q_3	$\{q_3\}$	$\{q_3\}$

课堂练习

NFA的状态转移表

	状态	0	1
开始	q_0	$\{q_0\}$	$\{q_0, q_1, q_2\}$
	q_1	$\{q_2\}$	$\{q_3\}$
	q_2	$\emptyset$	$\{q_3\}$
终止	q_3	$\{q_3\}$	$\{q_3\}$

DFA的状态转移表

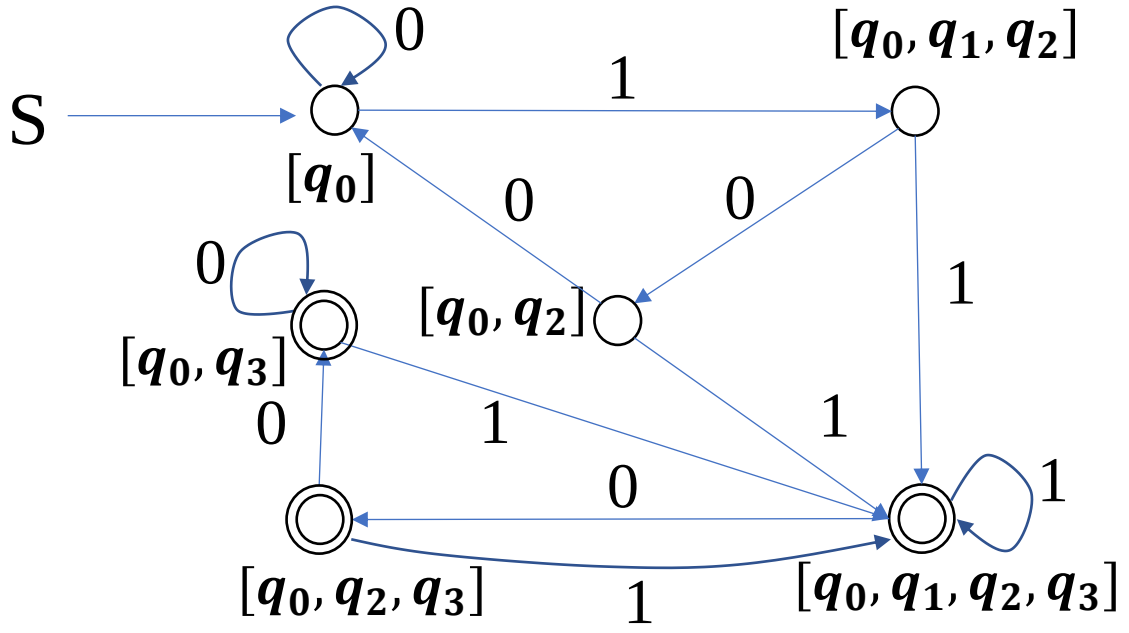
	状态	0	1
开始	$[q_0]$	$[q_0]$	$[q_0, q_1, q_2]$
	$[q_0, q_1, q_2]$	$[q_0, q_2]$	$[q_0, q_1, q_2, q_3]$
	$[q_0, q_2]$	$[q_0]$	$[q_0, q_1, q_2, q_3]$
终止	$[q_0, q_1, q_2, q_3]$	$[q_0, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$
终止	$[q_0, q_2, q_3]$	$[q_0, q_3]$	$[q_0, q_1, q_2, q_3]$
终止	$[q_0, q_3]$	$[q_0, q_3]$	$[q_0, q_1, q_2, q_3]$

课堂练习

DFA的状态转移表

	状态	0	1
开始	$[q_0]$	$[q_0]$	$[q_0, q_1, q_2]$
	$[q_0, q_1, q_2]$	$[q_0, q_2]$	$[q_0, q_1, q_2, q_3]$
	$[q_0, q_2]$	$[q_0]$	$[q_0, q_1, q_2, q_3]$
终止	$[q_0, q_1, q_2, q_3]$	$[q_0, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$
终止	$[q_0, q_2, q_3]$	$[q_0, q_3]$	$[q_0, q_1, q_2, q_3]$
终止	$[q_0, q_3]$	$[q_0, q_3]$	$[q_0, q_1, q_2, q_3]$

DFA的状态转移图



章节目录

3.5 FA是正则语言的识别器

3.5.1 FA与右线性文法

3.5.2 FA与左线性文法

DFA处理句子的特性：

DFA M $= (Q, \Sigma, \delta, q_0, F)$，处理句子$a_1 a_2 \cdots a_n$

① **M**按照句子$a_1 a_2 \cdots a_n$中字符的出现顺序，从开始状态q_0开始，依次处理字符a_1、a_2、…、a_n，在这个处理过程中，每处理一个字符进入一个状态，最后停止在某个终止状态。

② 它每次处理且仅处理一个字符：第i步处理输入字符a_i。

③ 对应于使用**δ(q，a)=p**的状态转移函数的处理，相当于在状态**q**完成对**a**的处理，接下来从**p**开始接着实现对后续字符的处理。

④ 当**δ(q，a)=p∈F**，且**a**是输入串的最后一个字符时，**M**完成对此输入串的处理。

- 满足定理2-1的文法推导句子的过程可以与一个“适当的”DFA处理该句子的过程相对应。暂时A_0用表示S。

Callback:

定理2-1 L是RL的充要条件是存在一个文法，该文法产生语言L，并且它的产生式要么是形如：A→a的产生式，要么是形如A→aB的产生式。其中A、B为语法变量，a为终极符号。

文法产生句子的过程如下：

$A_0 \Rightarrow a_1A_1$	对应产生式$A_0 \rightarrow a_1A_1$	A_0变到A_1
$\Rightarrow a_1a_2A_2$	对应产生式$A_1 \rightarrow a_2A_2$	A_1变到A_2
…	…	…
$\Rightarrow a_1a_2\ldots a_{n-1}A_{n-1}$	对应产生式$A_{n-2} \rightarrow a_{n-1}A_{n-1}$	A_{n-2}变到A_{n-1}
$\Rightarrow a_1a_2\ldots a_{n-1}a_n$	对应产生式$A_{n-1} \rightarrow a_n$	A_{n-1}变到字符a_n

FA识别句子的过程如下：

$q_0\ a_1a_2\ldots a_{n-1}a_n$

$\vdash a_1q_1\ a_2\ldots a_{n-1}a_n$　对应$\delta(q_0,a_1)=q_1$　　q_0变到q_1

$\vdash a_1a_2q_2\ldots a_{n-1}a_n$　对应$\delta(q_1,a_2)=q_2$　　q_1变到q_2

……　　…

$\vdash a_1a_2\ldots a_{n-1}q_{n-1}a_n$　对应$\delta(q_{n-2},a_{n-1})=q_{n-1}$　　q_{n-2}变到q_{n-1}

$\vdash a_1a_2\ldots a_{n-1}a_nq_n$　对应$\delta(q_{n-1},a_n)=q_n$　　q_{n-1}变到接收态q_n

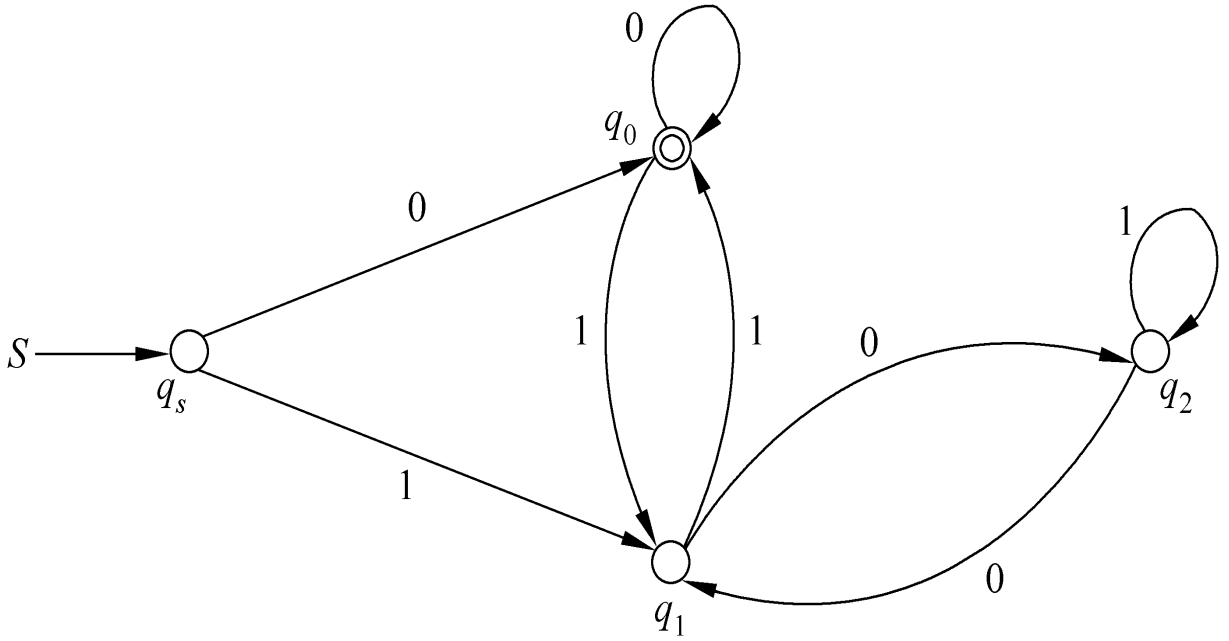
- 其中q_n为M的终止状态。考虑根据$a_1, a_2,\ldots, a_n$，让A_0与q_0对应，A_1与q_1对应，A_2与q_2对应，…，A_{n-2}与q_{n-2}对应，A_{n-1}与q_{n-1}对应。这样，就有希望得到满足定理2-1的正则文法的推导与DFA的移动互相模拟的方式。

定理 3-3 **FA接受的语言是正则语言。**

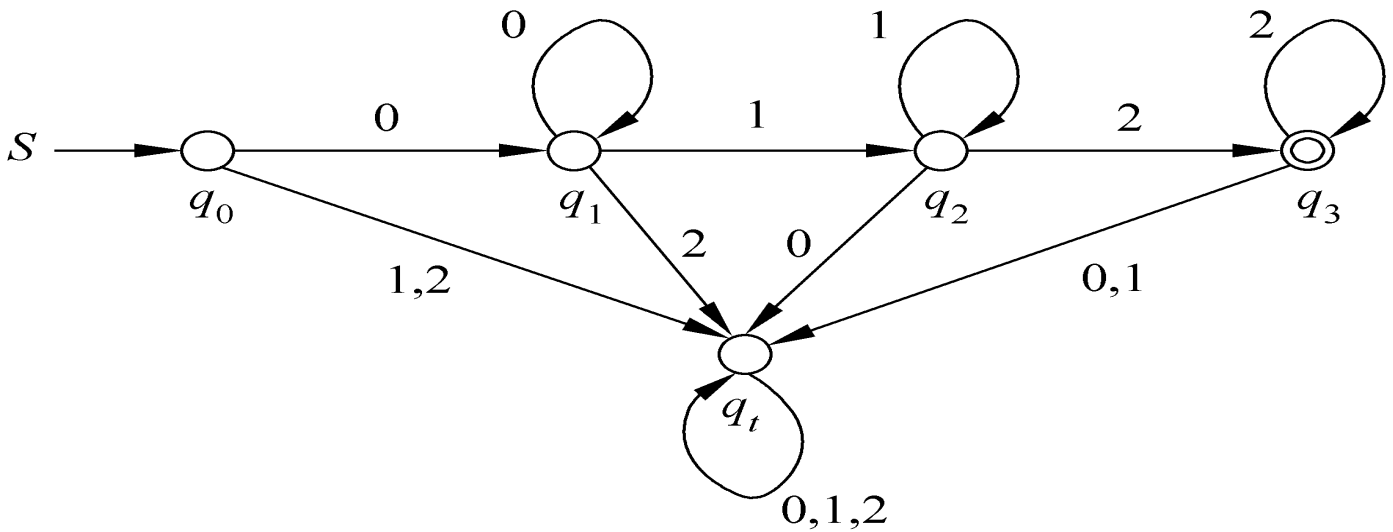
基本思想：让RG的派生对应DFA的移动。

例3-21 求与所给DFA等价的正则文法。

- $q_s \rightarrow 0q_0|0|1q_1$
- $q_0 \rightarrow 0q_0|0|1q_1$
- $q_1 \rightarrow 0q_2|1|1q_0$
- $q_2 \rightarrow 0q_1|1q_2$



例3-22 求与所给DFA等价的正则文法。



- $q_0 \rightarrow 0q_1|1q_t|2q_t$
- $q_1 \rightarrow 0q_1|1q_2|2q_t$
- $q_2 \rightarrow 0q_t|1q_2|2q_3|2$
- $q_3 \rightarrow 0q_t|1q_t|2q_3|2$
- $q_t \rightarrow 0q_t|1q_t|2q_t$

文法简化为：

- $q_0 \rightarrow 0q_1$
- $q_1 \rightarrow 0q_1|1q_2$
- $q_2 \rightarrow 1q_2|2q_3|2$
- $q_3 \rightarrow 2q_3|2$

从上述产生式看出，一旦q_t在句型中出现就永远不会消失，所以含有这样变量的产生式是不会对语言有什么贡献的，可以删除。实际上，对DFA中的陷阱状态，在构造与该DFA等价的正则文法时，是可以不考虑的。

3.5.1 FA与右线性文法

定理 3-4 正则语言可以由FA接受。

基本思想：让FA模拟RG的派生。

推论 3-1 FA与正则文法等价。

根据定理3-3和定理3-4即可得到。

例3-23 构造与如下所给正则文法等价的FA。

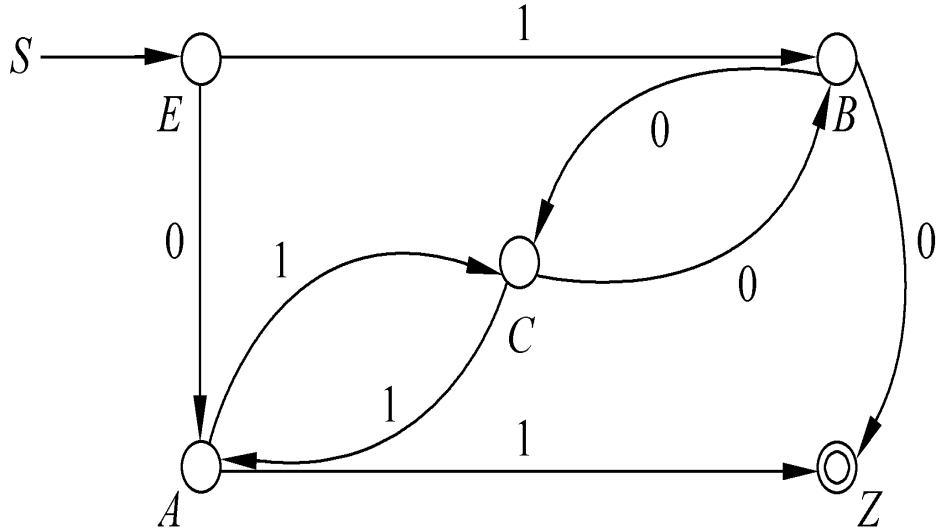
- G_1：E→0A|1B

  A→1|1C

  B→0|0C

  C→0B|1A

引入一个终止状态：Z。



- 转换函数

δ(E,0)={A} 对应E→0A

δ(E,1)={B} 对应E→1B

δ(A,1)={Z,C} 对应A→1|1C

δ(B,0)={Z,C} 对应B→0|0C

δ(C,0)={B} 对应C→0B

δ(C,1)={A} 对应 C→1A

3.5 FA是正则语言的识别器

3.5.1 FA与右线性文法

3.5.2 FA与左线性文法

如何根据左线性文法构造等价的FA

1. 句子$a_1a_2\ldots a_n$中的字符被推导出的顺序与出现的顺序相反，a_n，a_{n-1}，…，a_2，a_1；而按照归约来看，它们被归约成语法变量的顺序则正好与它们在句子中出现的顺序相同。可见，归约过程与FA处理句子字符的顺序是一致的，所以，可以考虑依照“归约”来研究FA的构造。

G_1：
$S_1 \rightarrow A_1 6$
$A_1 \rightarrow B_1 5$
$B_1 \rightarrow C_1 4$
$C_1 \rightarrow D_1 3$
$D_1 \rightarrow E_1 2$
$E_1 \rightarrow F_1 1$
$F_1 \rightarrow 0$

如何根据左线性文法构造等价的FA

2. 对于形如A→a的产生式， 在一个句子的推导中一旦使用了它举行就变成了句子，而且a是该句子的第一个字符，是根据形如A→a的产生式进行归约的。对应到FA中，FA从开始状态出发，读到句子的第一个字符a，应将它归约为A。如果考虑用语法变量对应FA的状态，那么，此时我们需要引入一个开始状态，如Z。这样，对应形如A→a的产生式，可以定义A∈δ(Z，a)。

如何根据左线性文法构造等价的FA

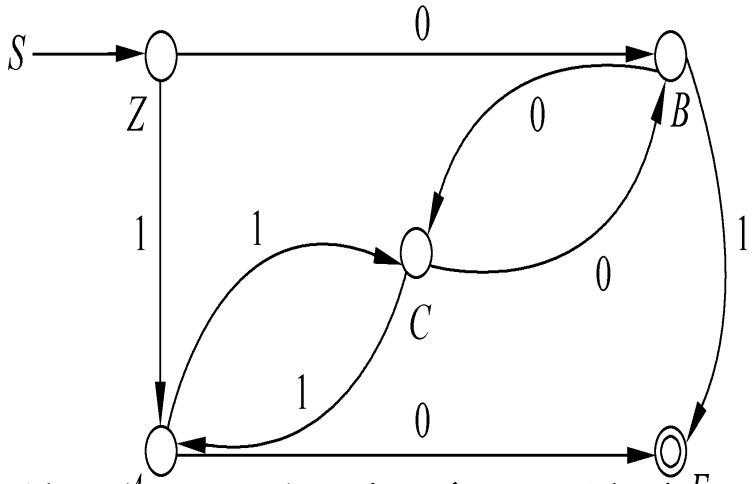
3. 按照上面的分析，对应于形如A→Ba的产生式，FA应该在状态B读入a时，将状态转换到A。也可以理解为：在状态B，FA已经将当前句子处理过的前缀“归约”成了B，在此时它读入a时，要将Ba归约成A，因此，它进入状态A。

4. 按照“归约”的说法，如果一个字符串是文法G产生语言的句子，它最终应该被归约成文法G的开始符号。所以，G的开始符号对应的状态就是相应的FA的终止状态。

例3-24 求与所给左线性文法等价的FA。

- G_2：E→A0|B1
  A→1|C1
  B→0|C0
  C→B0|A1

按照之前分析，E为文法的开始符号，故状态E为FA的终止状态，还需引入一个开始状态Z。



与文法等价的FA的移动函数如下：

δ(A,0)=E　　对应E→A0
δ(B,1)=E　　对应E→B1
δ(Z,1)=A　　对应A→1
δ(C,1)=A　　对应A→C1
δ(Z,0)=B　　对应B→0
δ(C,0)=B　　对应B→C0
δ(B,0)=C　　对应C→B0
δ(A,1)=C　　对应 C→A1

3.5.2 FA与左线性文法

知道了如何根据左线性文法构造FA后，很容易弄清楚如何构造一个给定DFA等价的左线性文法：

- 如果δ(A,a)=B，则“A”后紧跟一个a应该“归约”成B（而转换成右线性文法时，则是A产生aB）。
- 如果δ(A,a)=B为终止状态，则“A”后紧跟一个a不仅应该“归约”成B，还应该“归约”成文法的开始符号；
- 如果δ(A,a)=B时A还是DFA的开始状态，则a应该“归约”成B。

3.5.2 FA与左线性文法

为方便起见，在构造给定的DFA等价的左线性文法之前，先对该DFA（的状态转移图）做如下“预处理”：

（1）删除DFA的陷阱状态（包括与之相关的弧）；

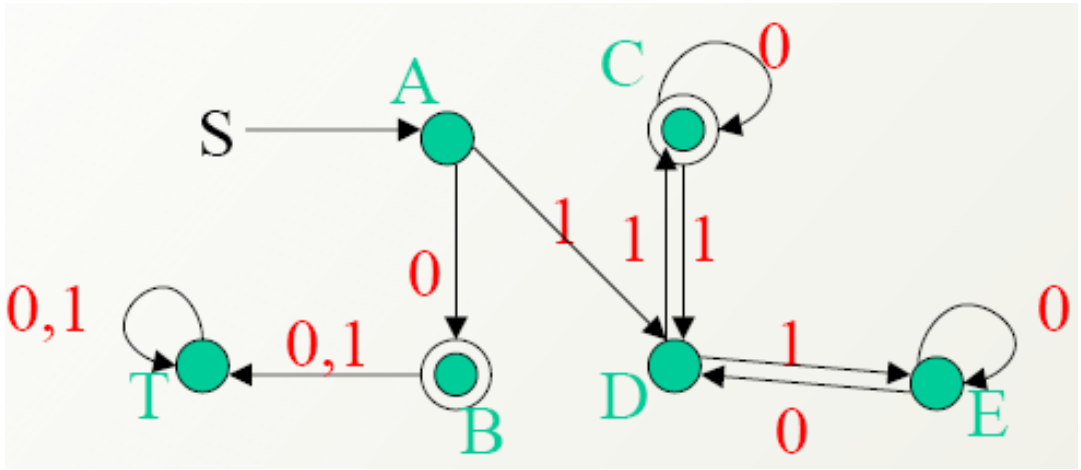
（2）在图中加一个识别状态Z，“复制”一条原来到达终止状态的弧，使它从原来的起点出发，到达新添加的识别状态；

（3）如果开始状态也是终止状态，则增加产生式$Z \rightarrow \varepsilon$ 。

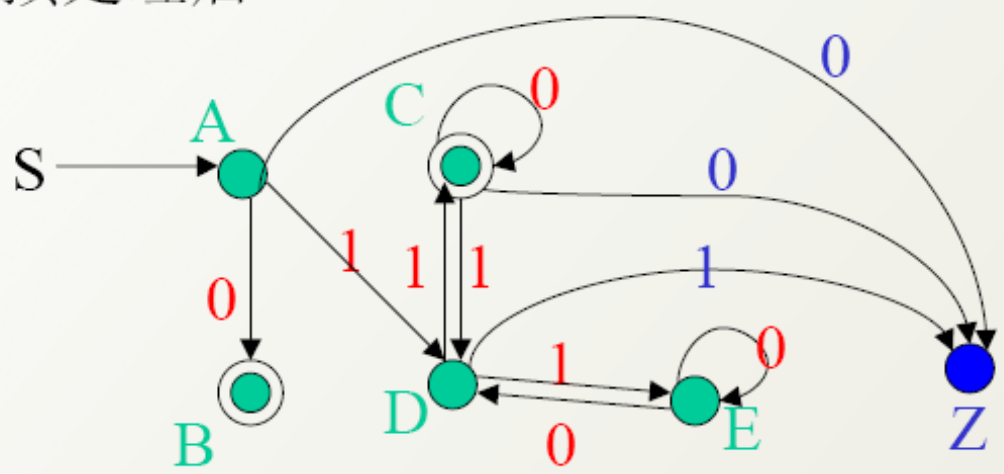
通过这样的“预处理”，就不用再给与原终止状态特殊的考虑，相应的文法构造可依照如下进行：

- 如果$\delta(A,a)=B$, 则有产生式$B \rightarrow Aa$；
- 如果$\delta(A,a)=B$，且A是开始状态，则有产生式$B \rightarrow a$。

3.5.2 FA与左线性文法



预处理后



得到文法：

Z→ 0|C0|D1

B→0 （没有用，最后删掉）

C→D1|C0

D→1|C1|E0

E→E0|D1

<u>定理 3-5</u> 左线性文法与FA等价。

章节目录

章节目录

3.7 本章小结

本章讨论正则语言的识别器FA。包括DFA、NFA、ε-NFA, RG与FA的等价性, 带输出的FA和双向FA。

- FA M是一个五元组，$\mathrm{M}=(Q,\Sigma,\delta,q_0,F)$，它可以用状态转移图表示。
- M接受的语言为$\mathbf{L(M)}=\{x|x\in\Sigma^*$且$\delta(q,x)\in F\}$。如果$\mathbf{L(M_1)=L(M_2)}$，则称$\mathbf{M_1}$与$\mathbf{M_2}$等价。
- FA的状态具有有穷的存储功能。这一特性可以用来构造接受一个给定语言的FA。
- NFA允许M在一个状态下读入一个字符时选择地进入某一个状态，$\mathbf{L(M)}=\{x|x\in\Sigma^*$ 且 $\delta(q,x)\cap F\neq\emptyset\}$。

3.7 本章小结

- ε-NFA在状态q不读入任何字符，可以选择地将状态变成p_1、p_2、…或者p_m。这叫做M在状态q做一个空移动。
- NFA与DFA等价，ε-NFA与NFA等价，统称为FA。
- 根据需要，可以在FA中设置特殊的陷阱状态。
- FA是正则语言的识别模型，可以证明FA和左线性文法、右线性文法等价。
- 2DFA不仅允许读头向前移动，还允许读头向后移动。通过这种扩充，2DFA仍然与FA等价。
- Moore机和Mealy机是两种等价的带输出的FA，Moore机根据状态决定输出字符，Mealy机根据移动决定输出字符。

Thanks!