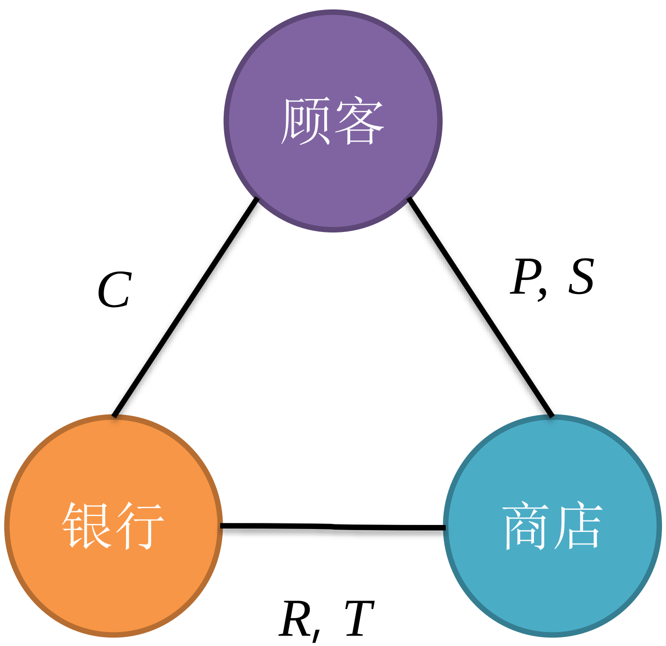
第二章（DFA）2025

本章内容

- 依次介绍DFA、NFA和ε-NFA的完整内容，证明它们三者在枚举语言的能力上等价。
- 具体介绍DFA定义及表示、DFA如何判定输入串为接受还是拒绝、扩展DFA转移函数以表示DFA的连续转移，并用于表示DFA的语言。
- 类似的叙述方式用于NFA和ε-NFA的介绍，并引入ε闭包和ε闭集概念，以及将NFA转换为DFA的子集构造法。
- 三方面知识点：语言识别器；判定性质；等价性质。
- 分为两个PPT文件。

一个简单购物系统的DFA建模示例



P 付款：顾客把电子货币发给商店，商店收到。

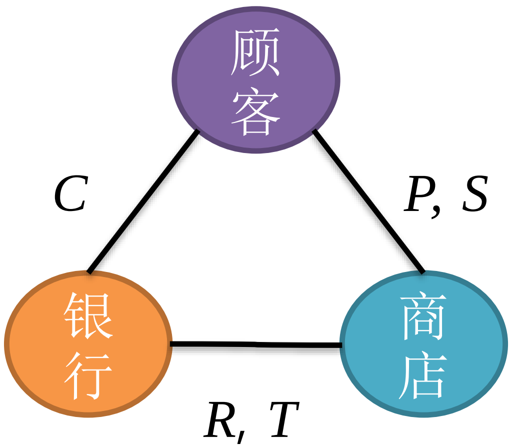
S 送货：商店送货给顾客，顾客收到。

C 取消：顾客让银行取消电子货币，银行取消。

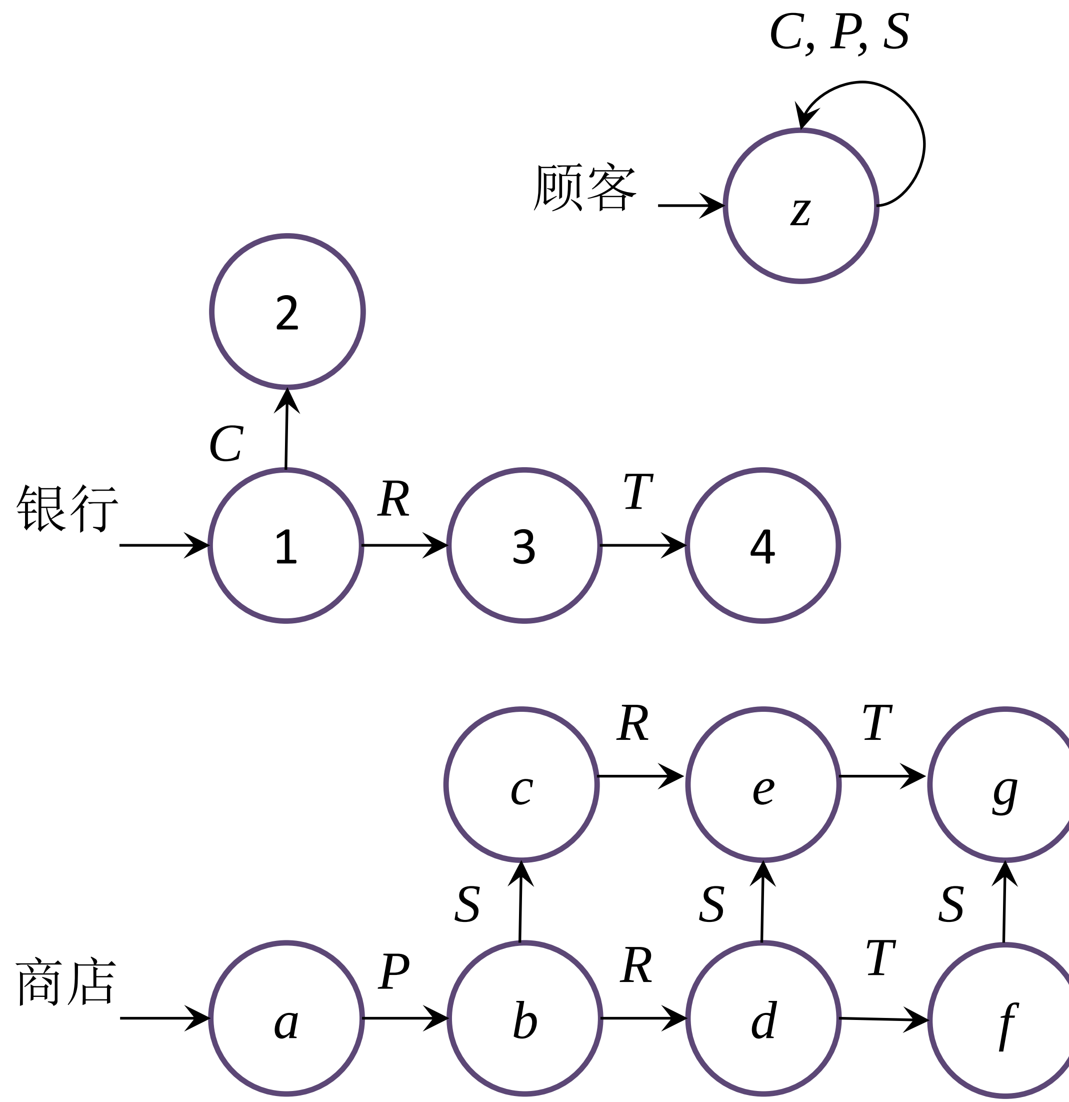
R 兑换：商店将电子货币发给银行，银行收到并予以认证。

T 转帐：银行签发给商店电子货币，商店收到。

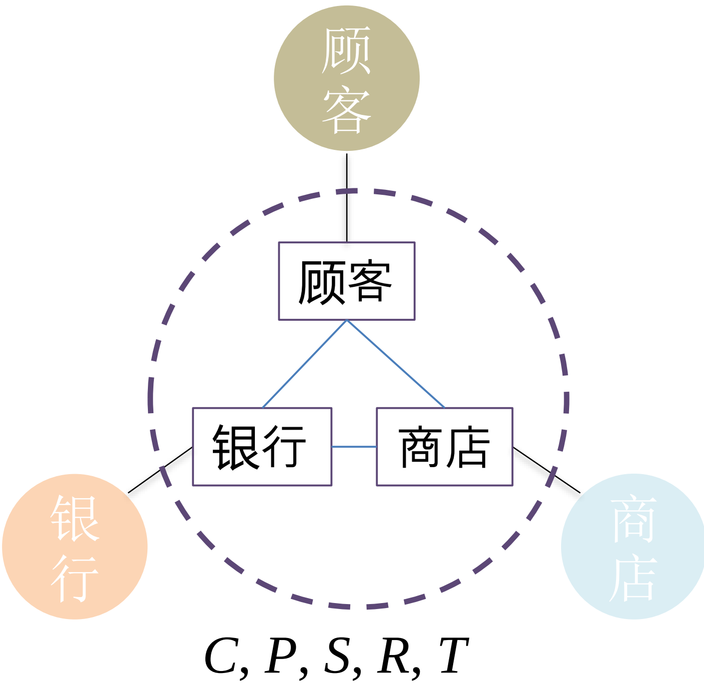
顾客、商店、银行的自动机模型

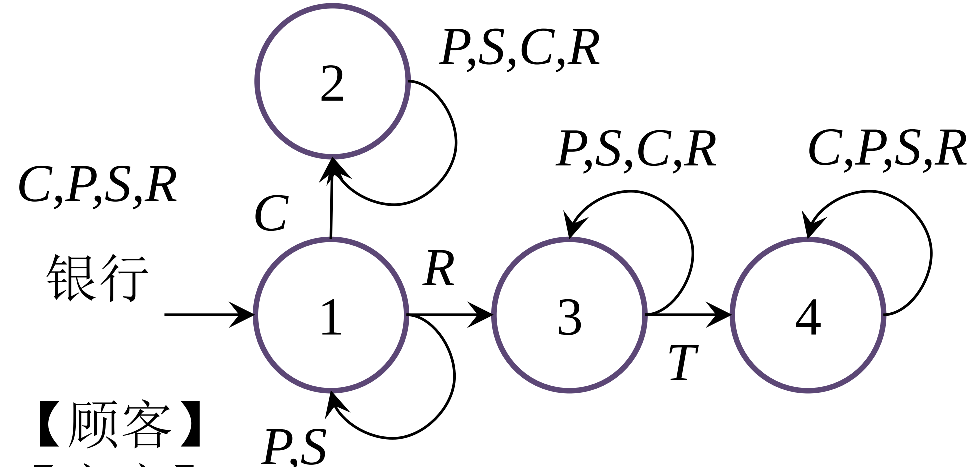


P 付款：顾客把电子货币发给商店，商店收到。
S 送货：商店送货给顾客，顾客收到。
C 取消：顾客让银行取消电子货币，银行取消。
R 兑换：商店将电子货币发给银行，银行收到并予以认证。
T 转帐：银行签发给商店电子货币，商店收到。

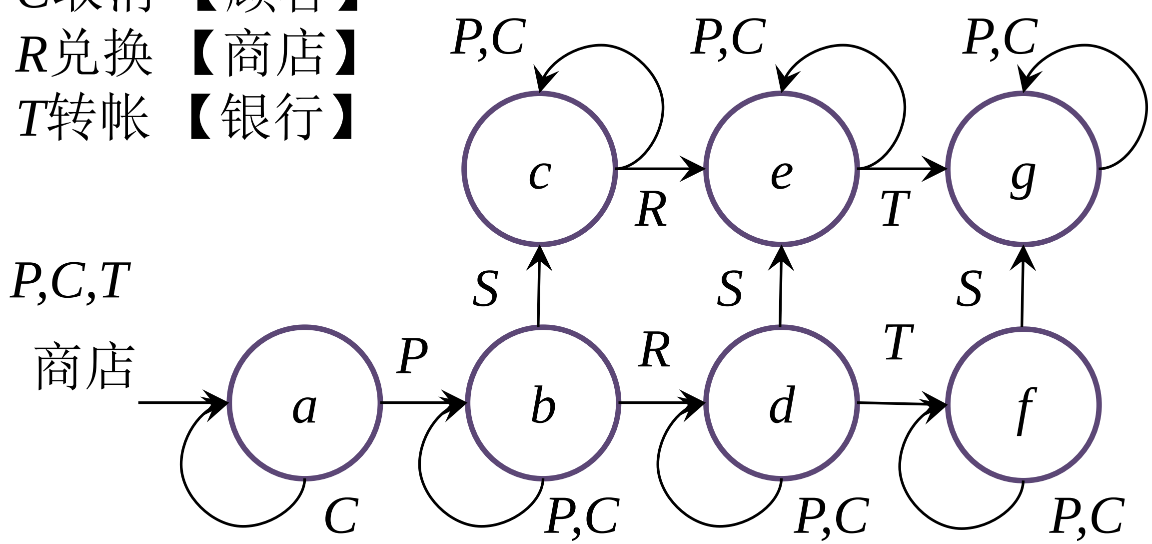


一个系统：各自动机对于无关交互保持迁移不变





*P*付款 【顾客】
*S*送货 【商店】
*C*取消 【顾客】
*R*兑换 【商店】
*T*转帐 【银行】



三个各自的状态对于由另外两方发起的交互都要有迁移弧射出；银行收到R后才*T*；客户随意*C*和*P*；商店收到*P*后会*S*也会R

乘积自动机（用于验证没有漏洞）

*P*付款 【顾客】
*S*送货 【商店】
*C*取消 【顾客】
*R*兑换 【商店】
*T*转帐 【银行】

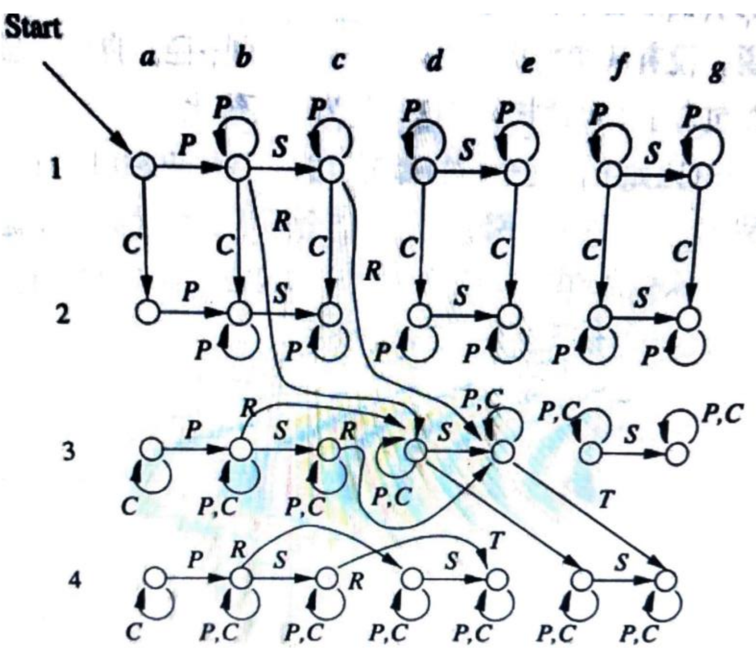
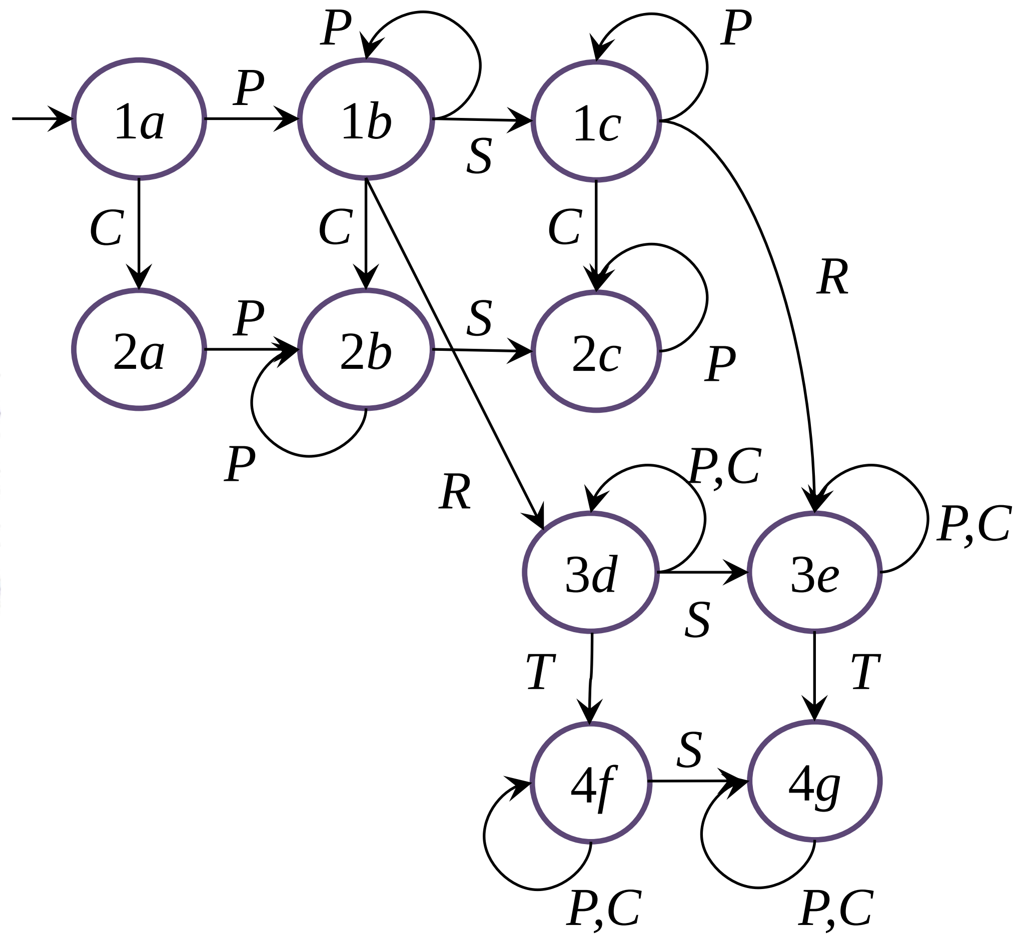
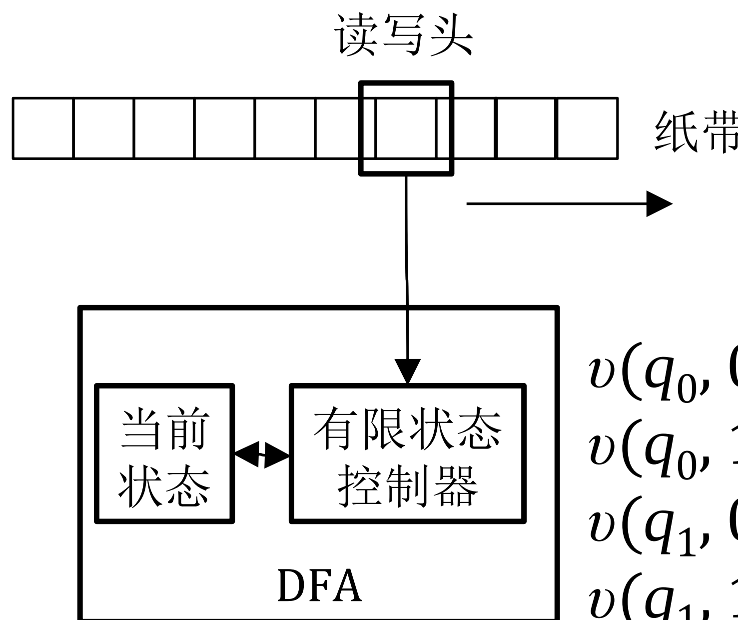


图2-3 商店和银行的乘积自动机



2.1 确定型有穷自动机DFA

- DFA $A = (Q, \Sigma, \upsilon, q_0, F)$
- 有限状态集合Q
- 字母表Σ为有限符号集合
- 状态转移函数υ: $Q \times \Sigma \rightarrow Q$
- $q_0 \in Q$ 初始状态
- $F \subseteq Q$ 接受状态



$\upsilon(q_0, 0) = q_0$
$\upsilon(q_0, 1) = q_1$
$\upsilon(q_1, 0) = q_2$
$\upsilon(q_1, 1) = q_1$
$\upsilon(q_2, 0) = q_2$
$\upsilon(q_2, 1) = q_2$

0 … 0	1 … 1

$\{0^m 1^n | m \geq 0, n \geq 1\}$

q_0 初始状态 q_1 接受状态 q_2

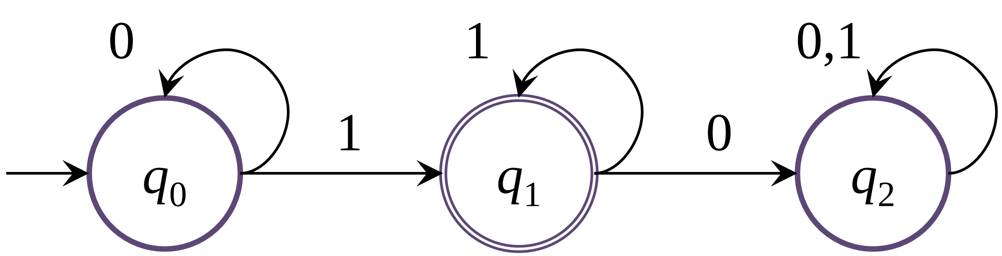
DFA $(\{q_0, q_1, q_2\}, \{0, 1\}, \upsilon, q_0, \{q_1\})$

DFA表示：代数、转移图、转移表

DFA $A = (\{q_0, q_1, q_2\}, \{0, 1\}, \upsilon, q_0, \{q_1\})$

$\upsilon = \{((q_0, 0), q_0), ((q_0, 1), q_1),$
$((q_1, 0), q_2), ((q_1, 1), q_1), ((q_2, 0), q_2),$
$((q_2, 0), q_2)\}$

DFA $A = (Q, \Sigma, \upsilon, q_0, F)$
$Q = \{q_0, q_1, q_2\}$,
$\Sigma = \{0, 1\}$,
$F = \{q_1\}$, $\upsilon = \ldots$



	0	1
$\rightarrow q_0$	q_0	q_1
$^*q_1$	q_2	q_1
q_2	q_2	q_2

$\mathcal{G}(A) = (Q, E, \Sigma)$
$E = \{(p, q, a) \mid p, q \in Q, a \in \Sigma, q = \upsilon(p, a)\}$

$\mathbb{T}[\text{ind}(q), 0]$
$\mathbb{T}[0, \text{ind}(a)]$
$\mathbb{T}[\text{ind}(q), \text{ind}(a)]$

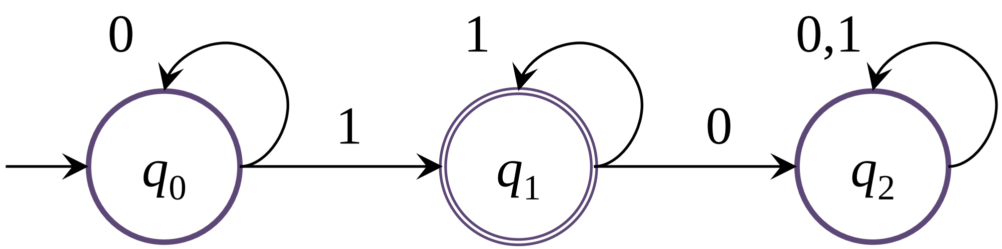
DFA的三种表示都彼此等价的

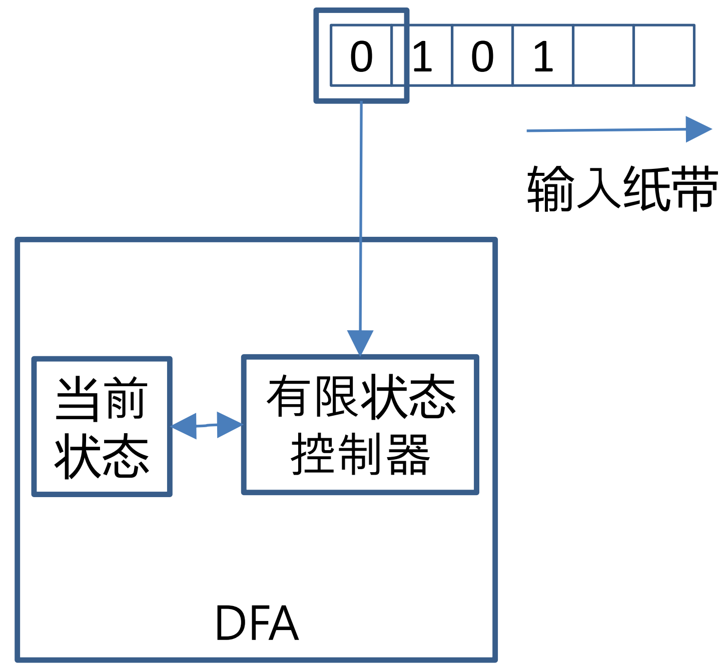
代数 A	转移图$\mathcal{G}(A)$	转移表$\mathbb{T}$
Q 元素 q	顶点（圆圈）	$\mathbb{T}[\mathrm{ind}(q), 0]$
Σ 元素 a	权（弧标记）	$\mathbb{T}[0, \mathrm{ind}(a)]$
υ 元素	边（转移弧）	$\mathbb{T}[i, j\,], i, j\neq 0$
q_0	箭头标记顶点	箭头标记$\mathbb{T}[1, 0]$
F 元素	双圈顶点	*标记$\mathbb{T}[i, 0], i\neq 0$

DFA的判定性质: DFA接受、拒绝输入串

- 0、1、00、01、000、001、010、0101

	0	1
$\rightarrow q_0$	q_0	q_1
$*q_1$	q_2	q_1
q_2	q_2	q_2





(<状态>, <输入符号>) -> <状态>

形式描述DFA接受串w（DFA的判定性质）

- DFA (Q, Σ, υ, q_0, F) **接受** w，如果对于$r_0, r_1, \ldots, r_n \in Q$ **满足**:
  - $r_0 = q_0$
  - $\upsilon(r_i, w_{i+1}) = r_{i+1}$ 其中 $i = 0, \ldots, n\text{-}1$
  - $r_n \in F$

- 为方便描述，使用路径术语：
  - 路径起点（**始端**）r_0，终点（**末端**）r_n，路径**标记**w
  - 可用谓词表示为PATH[r_0, r_n, w]
  - 路径隐含$\upsilon(r_i, w_{i+1}) = r_{i+1}$ 其中 $i = 0, \ldots, n\text{-}1$
- 何时不接受（**拒绝**）？
  - 不存在**末端为接受状态的路径**（路径长度为$|w|$)
  - 即不存在$r_n \in F$**使得**PATH[r_0, r_n, w]成立

用算法描述DFA的判定性质(运行过程)

输入：DFA $A = (Q, \Sigma, \upsilon, q_0, F)$；输入串$w \in \Sigma^*$

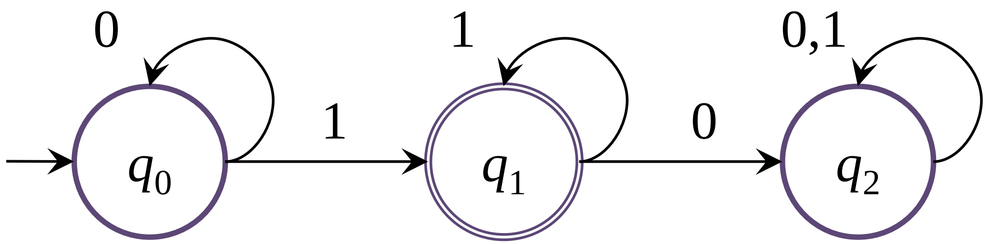
输出：dfa(w)为真表示接受w，否则拒绝它

```
int dfa(w) {
q = q0;                    //当前状态
x = w;                     //剩余串
while (x != ε) {
        x = ay;            //当前输入符号a
        q = υ(q, a);       //状态转移
        x = y;             //修改剩余串
        }
return q∈F;                //为真表示接受
}
```

习惯约定：$w, x, y, \ldots$ 是串， $a, b, c, \ldots$ 是符号。

DFA的几种形式

完全形



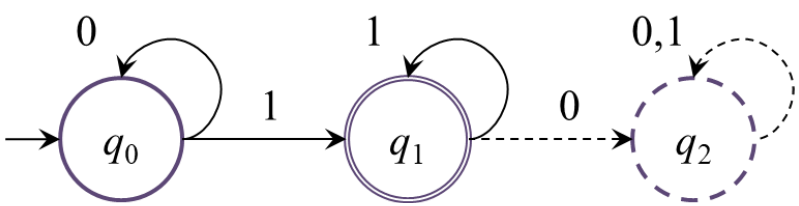
DFA $A=(Q, \Sigma, \upsilon, q_0, F)$

$\mathrm{dom}(\upsilon)= Q\times\Sigma$

$\forall w\in\Sigma^* \cdot \mathrm{PATH}[q_0, q, w]$

若$q\in F$那么$w\in L(A)$

简化型

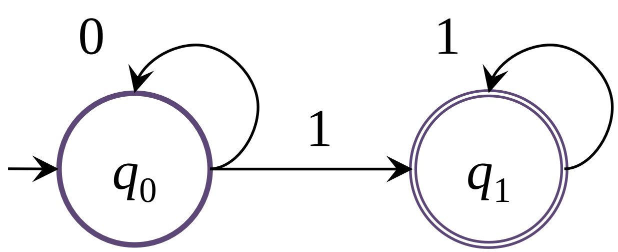


DFA $A=(Q, \Sigma, \upsilon, q_0, F)$

$\mathrm{dom}(\upsilon)\subset Q\times\Sigma$

A 突然死亡当$\upsilon(q, \mathrm{c})=\perp$，其中，$q$ 是当前状态，c 为读入符号

最简形



$\forall w\in L(A) \cdot \mathrm{PATH}[q_0, q, w]\wedge q\in F$

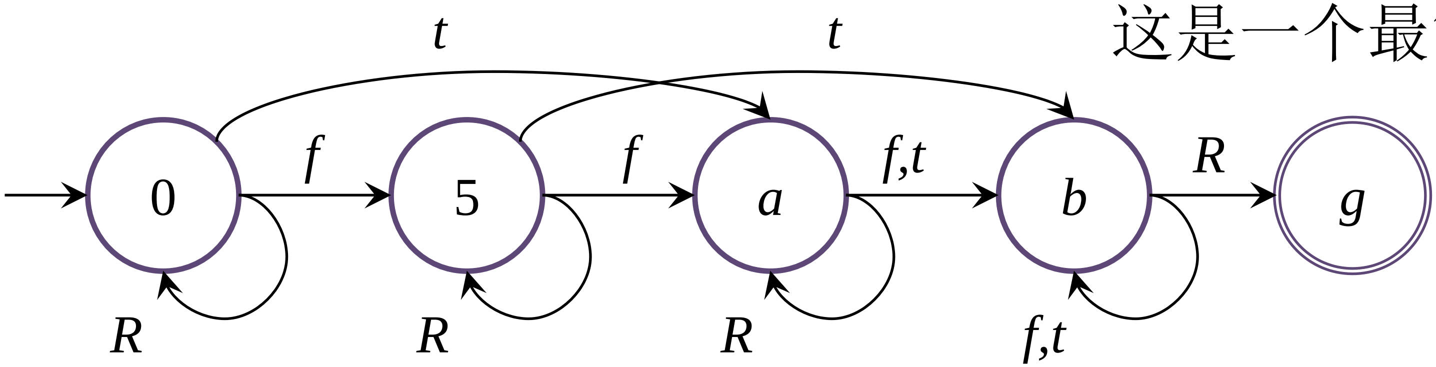
若 $w\notin L(A)$那么

$\mathrm{PATH}[q_0, q, \pi(w, i)], 0\leqslant \mathrm{i}<|w|$； $\forall w\in L(A) \cdot \mathrm{PATH}[q_0, q, w]\wedge q\notin F$

完全形DFA及其简化型（口香糖机）

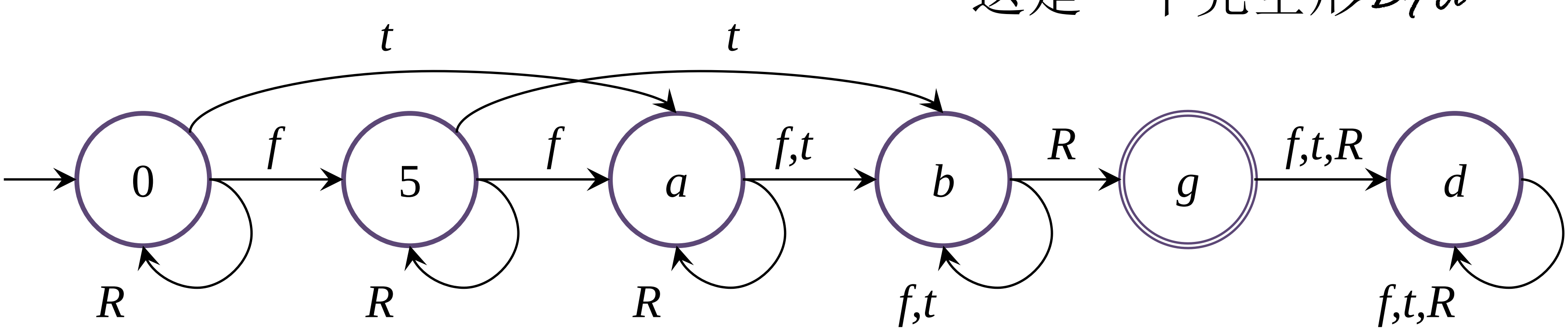
允许箭弧有缺失

这是一个最简形DFA



箭弧是完全的

这是一个完全形DFA



如果缺失与*d*节点邻接的一条或两条弧，

这是该DFA的一个简化型

状态*d*为一个陷阱状态

DFA的一般形式

对任意DFA $A = (Q, \Sigma, \upsilon, q_0, F)$，有：

$\mathrm{dom}(\upsilon) \subseteq Q\times\Sigma$ （完全形、最简形、简化型）

对每个$(q, c)\in Q\times\Sigma \setminus \mathrm{dom}(\upsilon)$，令$\upsilon' (q, c) = \perp$，且

$\forall c\in\Sigma \cdot \upsilon' (\perp, c) = \perp$，

那么DFA $A' = (Q\cup\{\perp\}, \Sigma, \upsilon\cup\upsilon', q_0, F)$是一个与$A$等价的完全形

将转移到状态⊥解释为自动机突然死亡。

一般常用$q_\perp$作为状态⊥

DFA的通用判定算法

- 算法：完全形、最简形、简化型DFA通用的判定性质

输入：DFA $A = (Q, \Sigma, \upsilon, q_0, F)$；输入串$w\in\Sigma^*$。

输出：dfa(w)为真表示接受w，否则拒绝它。

```
int dfa(w) {
q = q0;                         //当前状态q
x = w;                          //剩余串x
while (x != ε) {
        x = ay;                              //当前输入符号a
        if((q = υ(q, a)) ∉Q)return 0;        //A突然死亡
        x = y;                               //修改剩余串
        }
return q∈F;                     //为真表示接受
}。
```

扩展转移函数

- 为了方便表示连续的转移，扩展转移函数υ 为 $\tilde{\upsilon}$
- 归纳定义$\tilde{\upsilon}$ 为

基础： $\tilde{\upsilon}(q, \varepsilon) = q$

归纳： $\tilde{\upsilon}(q, wa) = \upsilon(\tilde{\upsilon}(q, w), a)$

- $\tilde{\upsilon}(q , w)$表示DFA中存在唯一一条始端为q、末端为$\tilde{\upsilon}(q, w)$、标记为w的路径，即有PATH$[q, \tilde{\upsilon}(q, w), w]$
- 其中允许状态⊥

扩展转移函数举例

	0	1
$\rightarrow * q_0$	q_0	q_1
$* q_1$	q_2	q_1
q_2	q_2	q_2

$$
\begin{aligned}
&\tilde{\upsilon}(q_0, 011) \\
&= \upsilon(\tilde{\upsilon}(q_0, 01), 1) \\
&= \upsilon(\upsilon(\tilde{\upsilon}(q_0, 0), 1), 1) \\
&= \upsilon(\upsilon(\upsilon(\tilde{\upsilon}(q_0, \varepsilon), 0), 1), 1) \\
&= \upsilon(\upsilon(\upsilon(q_0, 0), 1), 1) \\
&= \upsilon(\upsilon(q_0, 1), 1) \\
&= \upsilon(q_1, 1) \\
&= q_1
\end{aligned}
$$

$$
\begin{aligned}
&\tilde{\upsilon}(q_1, 011) \\
&= \upsilon(\tilde{\upsilon}(q_1, 01), 1) \\
&= \upsilon(\upsilon(\tilde{\upsilon}(q_1, 0), 1), 1) \\
&= \upsilon(\upsilon(q_2, 1), 1) \\
&= \upsilon(q_2\ 1) \\
&= q_2
\end{aligned}
$$

	0	1
$\rightarrow * q_0$	q_0	q_1
$* q_1$	$\perp$	q_1

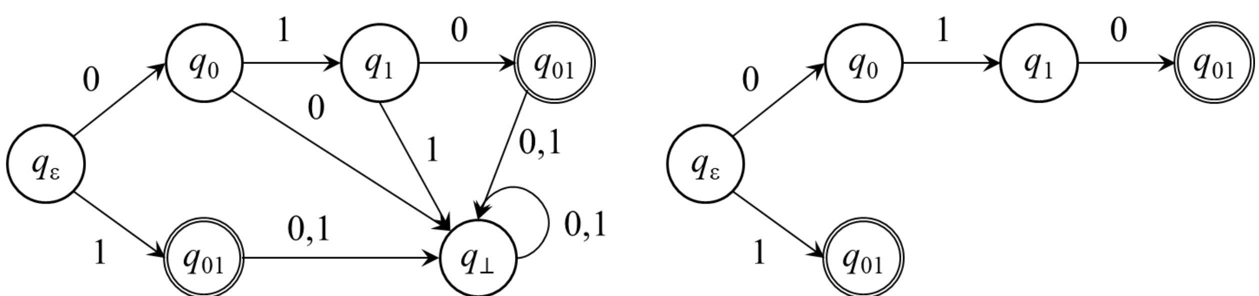
$$
\begin{aligned}
&\tilde{\upsilon}(q_0, 101) = \\
&= \upsilon(\tilde{\upsilon}(q_0, 10), 1) \\
&= \upsilon(\upsilon(\tilde{\upsilon}(q_0, 1), 0), 1) \\
&= \upsilon(\upsilon(q_1, 0), 1) \\
&= \upsilon(\perp, 1) \\
&= \perp
\end{aligned}
$$

DFA $(Q, \Sigma, \upsilon, q_0, F)$状态$q\in Q$的可达性

情形①：对任意$w\in\Sigma^*$，PATH$[q_0,q,w]$都不成立。即q是从初始状态不可达的。如果DFA有这样的状态q，则称其有该情形。

情形②：对于任意$q_1\in F$和$w\in\Sigma^*$，PATH$[q,q_1,w]$都不成立。即从q不能到达接受状态。如果DFA有此q则称其有该情形。

情形③：如果存在$q_1\in F$和$x,y\in\Sigma^*$，PATH$[q_0,q_1,x]\wedge$PATH$[q,q_1,y]$成立。即存在始端为q_0末端为q的路径、始端为q末端为q_1的路径。如果DFA有这样的状态q，则称其有可达性情形③。



(a)完全形　(b)最简形

①$\forall w\in\Sigma^* \cdot q\neq\tilde{\upsilon}(q_0, w)$

②$\forall p\in F, w\in\Sigma^* \cdot p\neq\tilde{\upsilon}(q, w)$

③$\exists p\in F, x,y\in\Sigma^* \cdot q=\tilde{\upsilon}(q_0, x)\wedge p=\tilde{\upsilon}(q, y)$

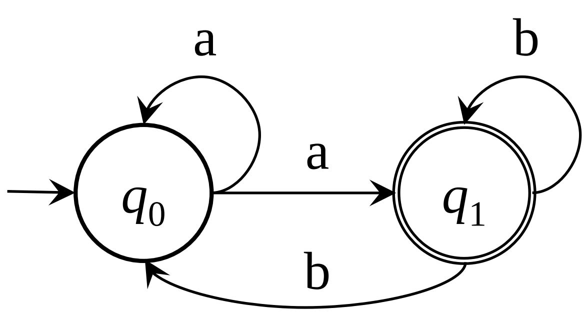
DFA定义的语言

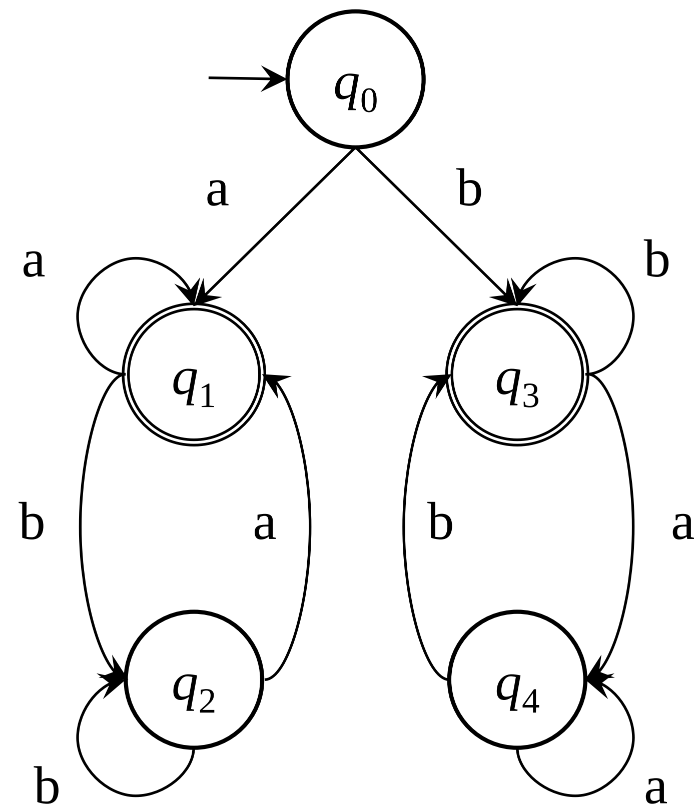
- DFA $M = (Q, \Sigma, \upsilon, q_0, F)$
- $L(M) = \{w \in \Sigma^* \mid \tilde{\upsilon}(q_0, w) \in F\}$

- DFA定义的语言是正则语言
- 断言某语言是正则的，如果它是某个DFA的语言

DFA的语言示例

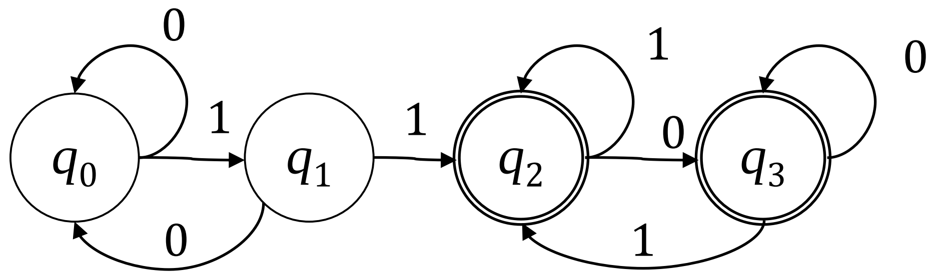
- DFA $M = (Q, \Sigma, \upsilon, q_0, F\ )$
- $L(M) = \{w \in \Sigma^* \mid \tilde{\upsilon}(q_0, w) \in F\ \}$
- $\Sigma = \{a, b\}$



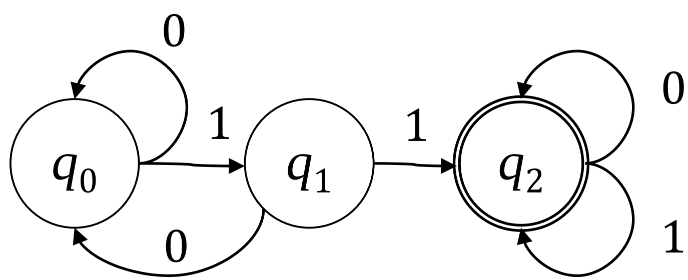


识别{0,1}上的含有子串11的串

	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_0	q_2
$*q_2$	q_3	q_2
$*q_3$	q_3	q_2

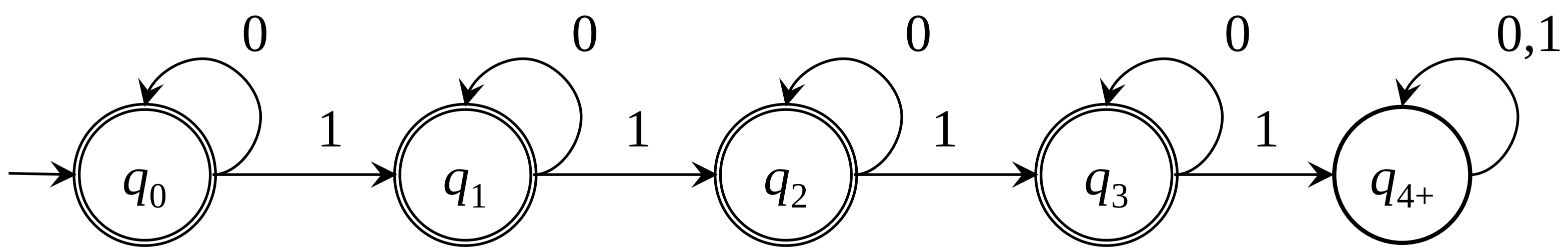


	0	1
$\rightarrow q_0$	q_0	q_1
q_1	q_0	q_2
$*q_2$	q_2	q_2



{0,1}上最多有三个1的串

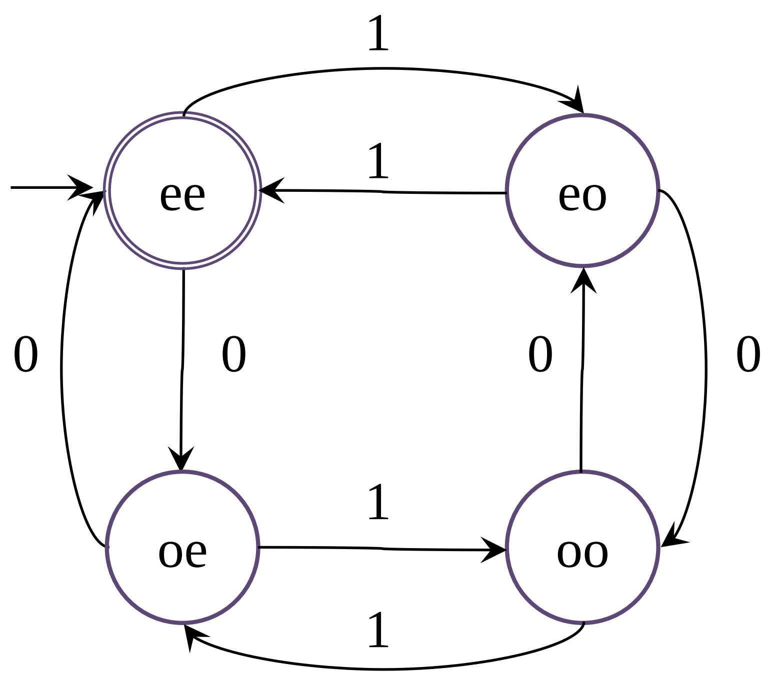
- 构造字母表{0, 1}上的DFA 接受所有最多含有三个1的串



例2.6

- {0,1}上的，含有偶数个0和偶数个1 的串的全集，是正则语言。

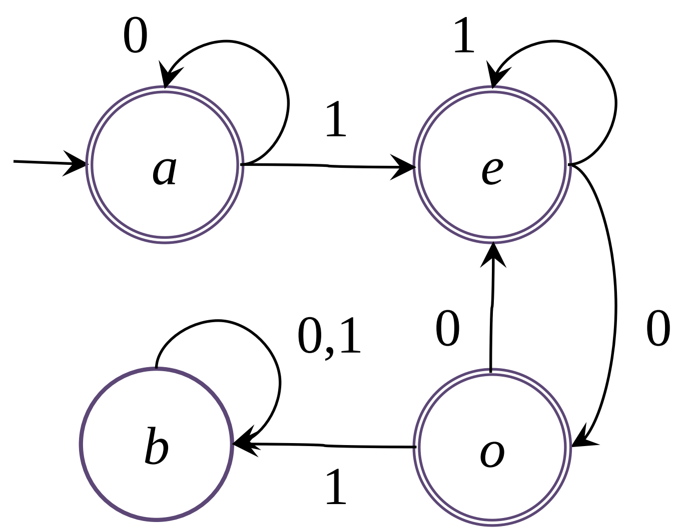
思路：输入串已消耗掉的部分只有四种情况，分别用状态来记住，继续读入符号将继续发生状态转移，当输入串消耗完以后，所达到的状态就决定了原输入串是什么情况，比如偶数个0和偶数个1



例2.7

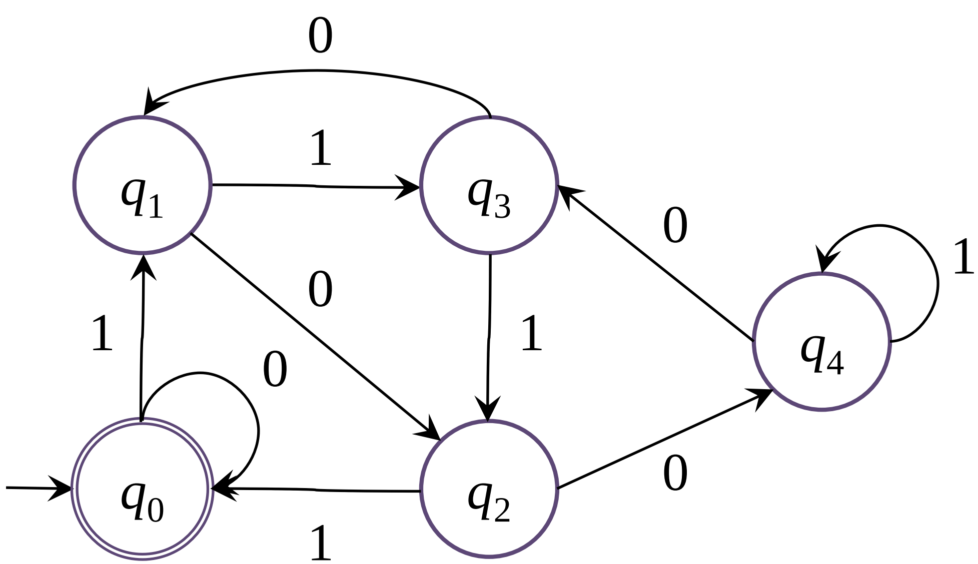
- {0,1}上不包含一对1且中间被奇数个0隔开的任何串。

  思路：排除包含模式10…01的串，该模式中的0为奇数个



例2.8

- 令$L=\{w\in\{0,1\}^*|w$ 看作二进制数的话能被5整除$\}$



分析：一个数除以5，其余数只能是0，…，4五种，因此我们以q_0，…，q_4分别表示这五种状态。因为接受能被5整除的数，故状态0既为初始状态，又为终结状态。接着，考虑二进制数在其串后增添0或1时，状态的转化情况。在二进制串后添1位，即可理解为将先前的串值乘以二再加上所添的数值。那么，串尾添数后新的数值模5的余数便可以计算出来。即可以得到添0或1后的新的状态。

归纳{0,1}上语言的DFA设计题目

- 前缀有这个、没这个、同时有这两个、或者有这个或者那个；
- 子串同上；
- 后缀同上；
- 汇总性：有n个1、没有n个1、有n个1和m个0、没有n个1同时也没有m个0；
- 计算性：被5整除、是素数、是奇数、是偶数
- 整数、定点数
- 对题目要求：典型且简单

用状态作记忆（例2.9）

- {0, 1}上以01结尾的串

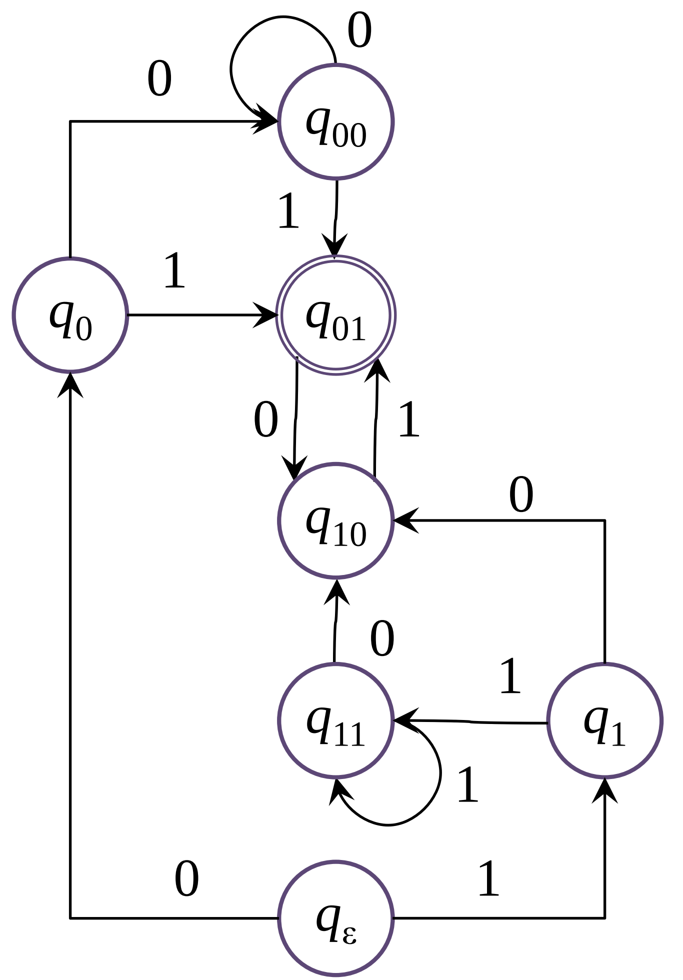
用四个状态记住输入串的前缀的最后两位（已消耗串的最后两位）。

q_{00}

q_{01}

q_{10}

q_{11}



用状态作记忆的局限性（例2.10）

- {0, 1}上以101结尾的串

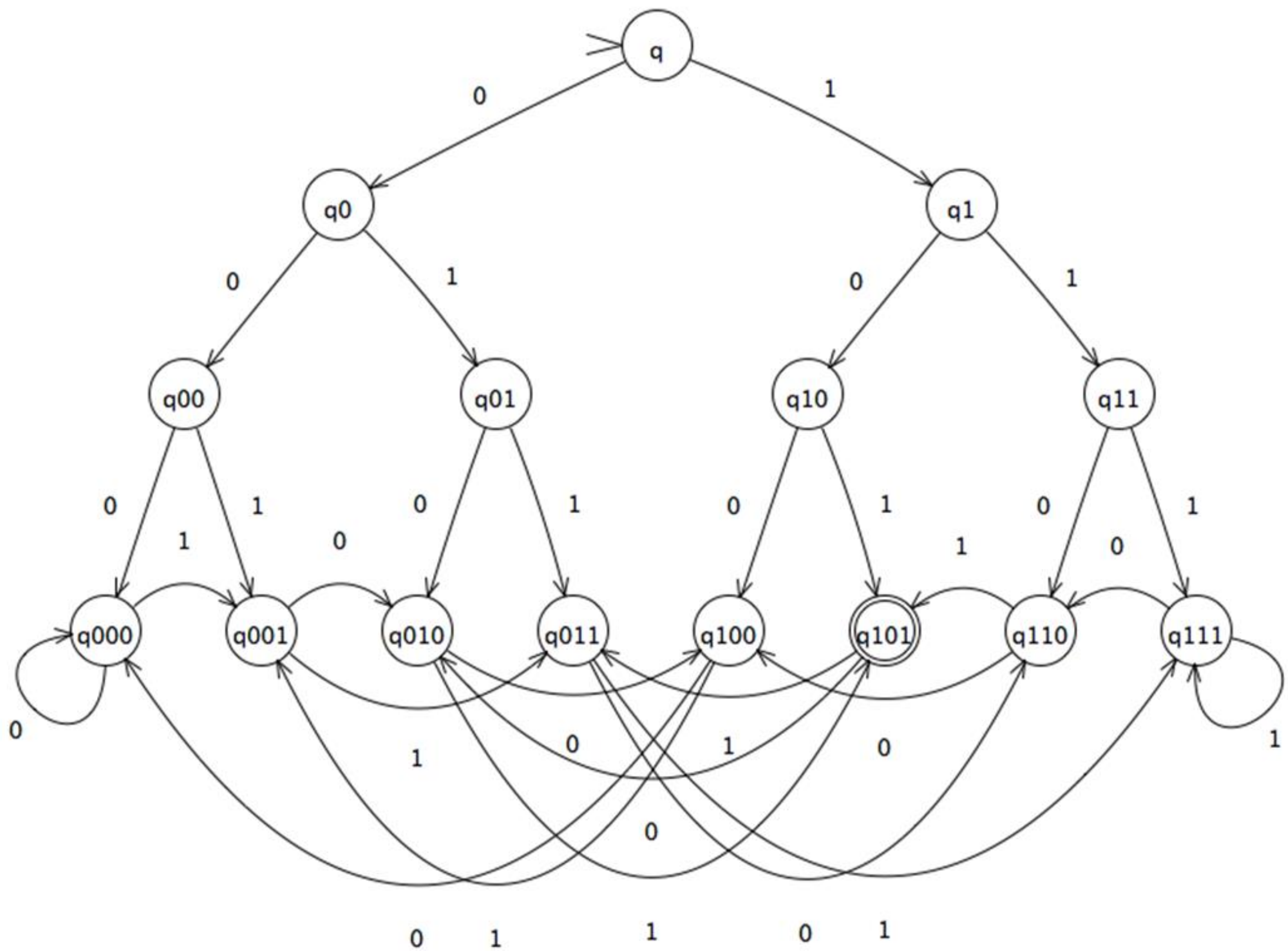
用八个状态记住已消耗串的最后三位。

q_{000}

q_{001}

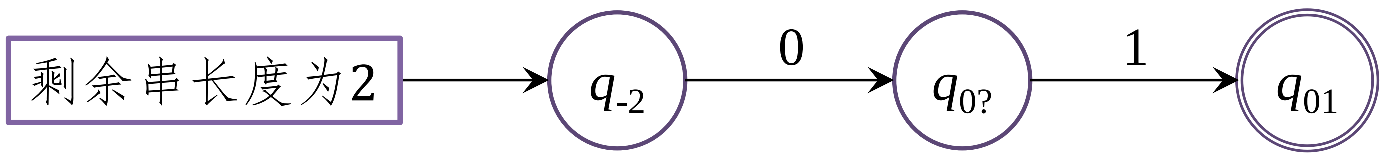
…

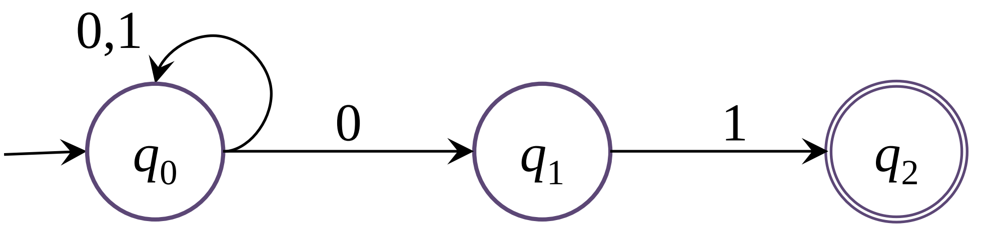
q_{111}



引入不确定性来避免状态爆炸

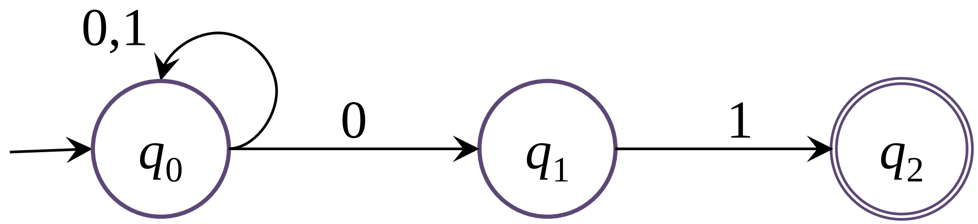
- 猜测

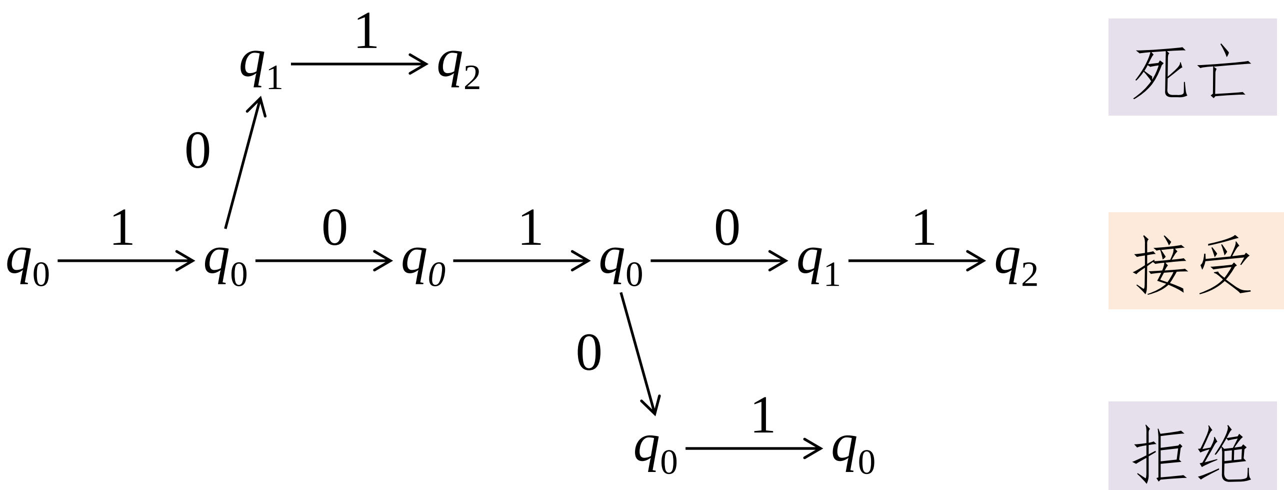




穷举与猜测

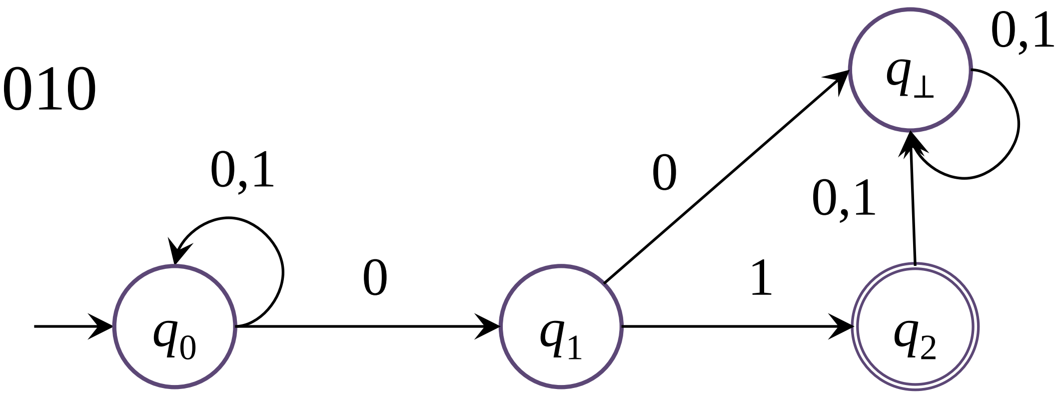
- 输入串10101

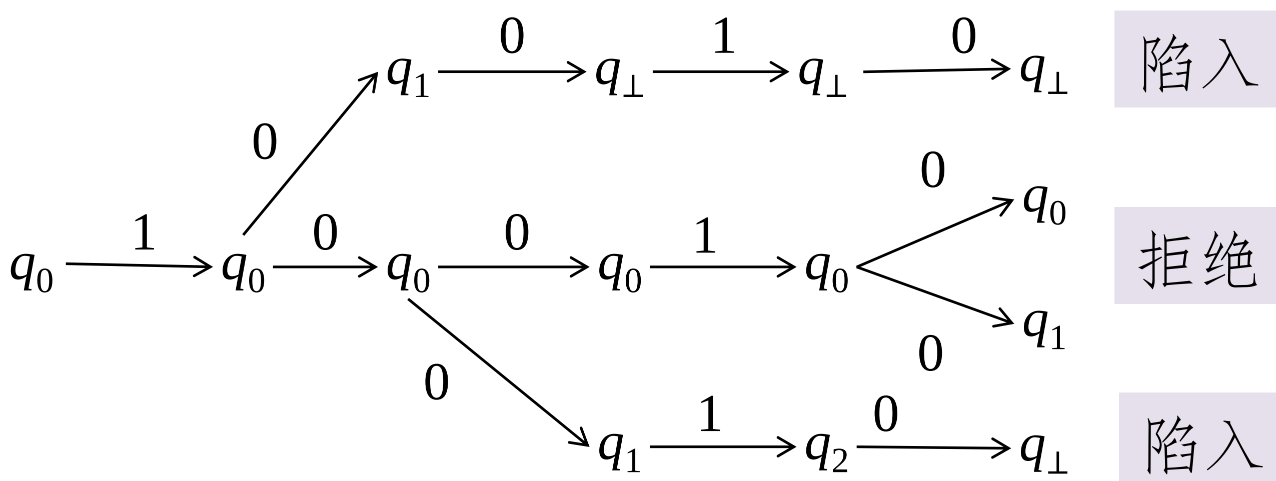




穷举与猜测

- 输入串10010





线索穷举

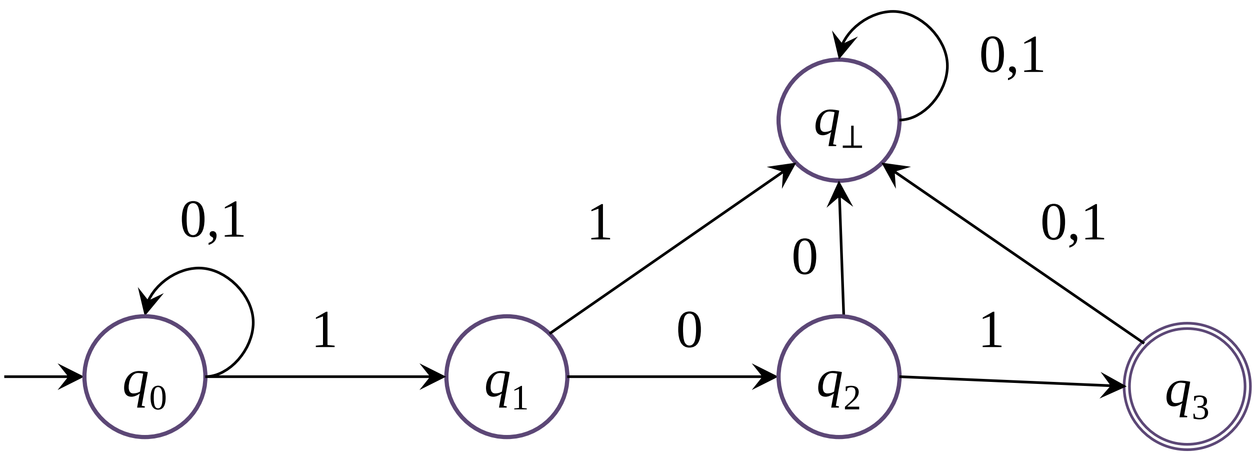
- 对于输入串，线索是自动机的一条以初始状态为起点的状态转移路径，该路径正好消耗掉输入串的某个前缀。线索穷举是有穷自动机的运行方式
  - DFA运行时只有一条线索（路径）。
  - 非确定有穷自动机（NFA）运行时有有穷个线索。
- 对于给定的输入串w，只要有一个线索成功，则w被自动机接受，否则不被接受
  - 成功的线索：始端为初始状态，标记为w，末端为接受状态。
- 在前例中，
  - 以101结尾的简化型NFA对于输入串110101有5个线索，其中有一个成功。
  - 以101结尾的完全型NFA对于输入串110010有4个线索，均不成功。
  - 以101结尾的完全型NFA对于输入串110101有几个个线索？成功者为何？

NFA的线索穷举成为一个树

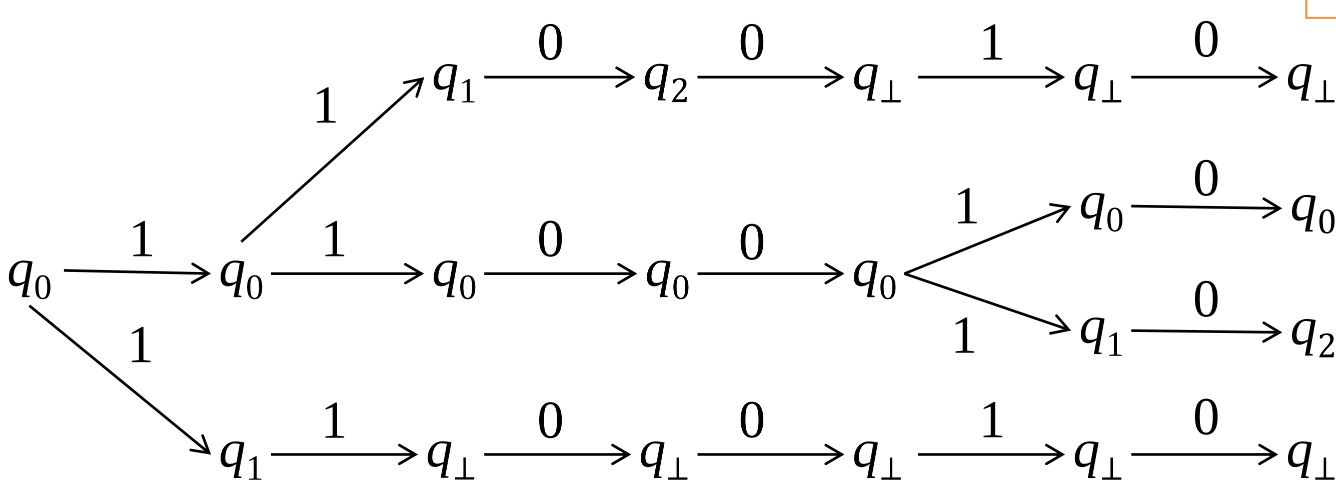
- w为输入串，w的第i（$i \geqslant 0$）个前缀就是长度为i的前缀，记为i前缀，等于$\pi(w, i)$
- w=hello-world
- w前缀：ε，h，he，hel，hell，hello，hello-，hello-w，…，hello-world
- w有$|w|+1$个前缀
- 给定输入串的线索穷举树▶
- 根为初始状态，层数为0
- 从根到每一个叶子的路径都是一条线索（每个线索都在树中恰有一条从根到叶子的路径对应）
- 第i层（i>0）的状态集合=T_i
- 以根为起点，无论沿着那条线索，都消耗掉输入串的i前缀并转移到T_i中的状态
- T_i就是活动状态集，也是w的i前缀的当前状态集。

线索穷举树举例

输入串: 110010



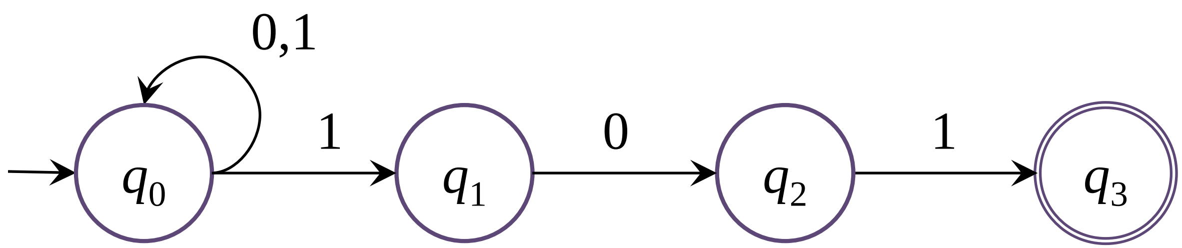
i	T_i	i前缀
0	$\{q_0\}$	ε
1	$\{q_0, q_1\}$	1
2	$\{q_0, q_1, q_\perp\}$	11
3	$\{q_0, q_2, q_\perp\}$	110
4	$\{q_0, q_\perp\}$	1100
5	$\{q_0, q_1, q_\perp\}$	11001
6	$\{q_0, q_2, q_\perp\}$	110010

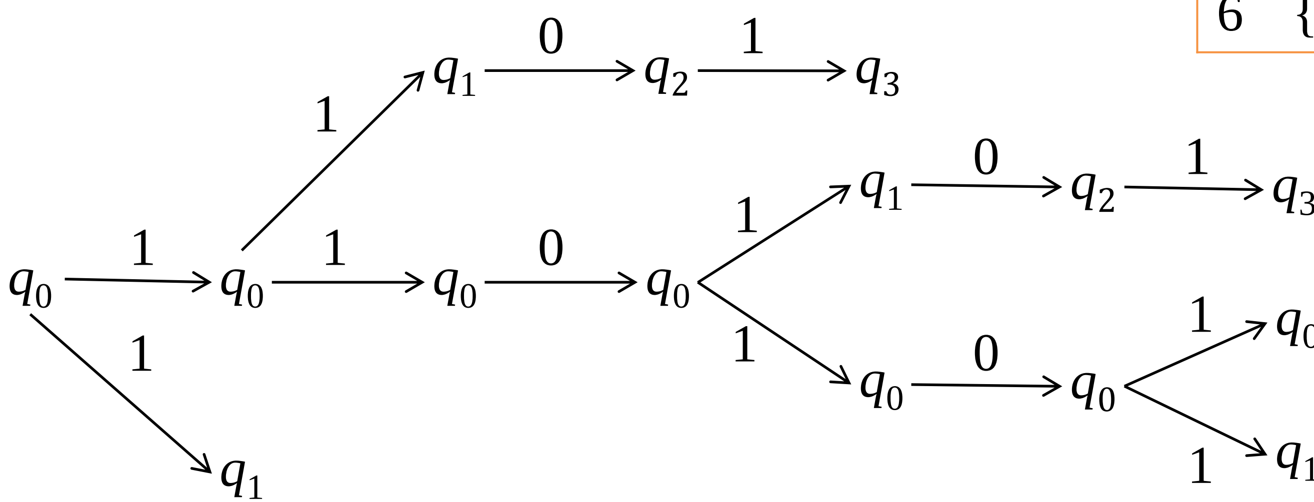


线索穷举树举例

输入串: 110101

i	T_i	i前缀
0	$\{q_0\}$	ε
1	$\{q_0,q_1\}$	1
2	$\{q_0,q_1\}$	11
3	$\{q_0,q_2\}$	110
4	$\{q_0,q_1,q_3\}$	1101
5	$\{q_0,q_2\}$	11010
6	$\{q_0,q_1,q_3\}$	110101



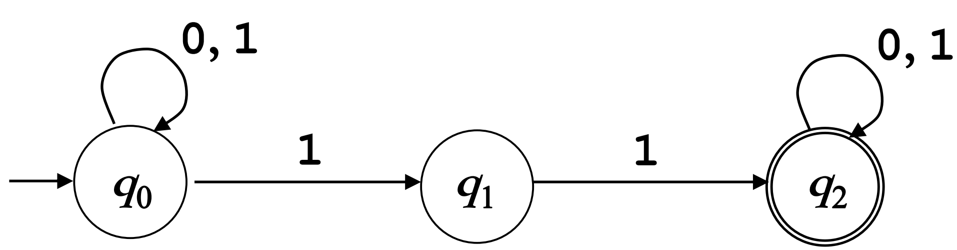


NFA例

- 构造字母表{0, 1}上的NFA 接受含有子串 11的串。

10011010, 11001, 11101 接受

ε, 000, 010101 拒绝

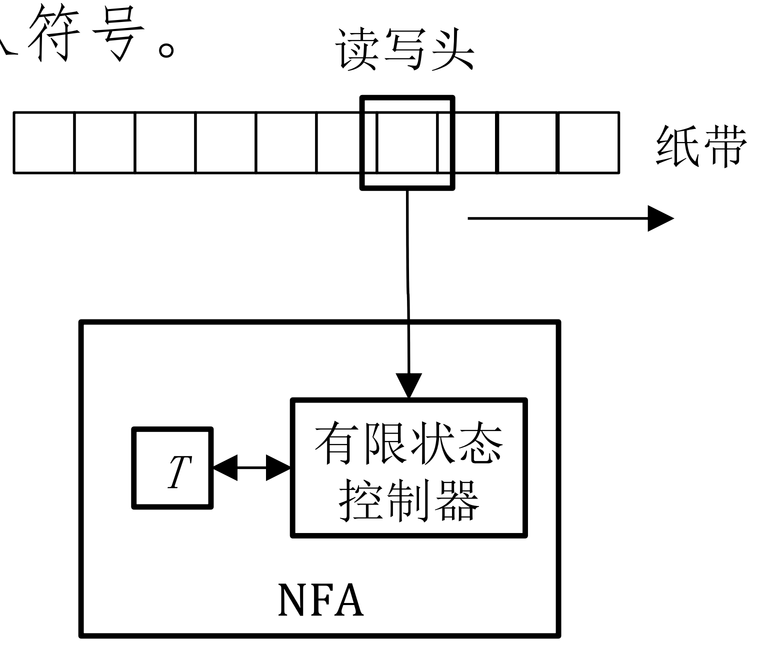


NFA与DFA有什么不同？

- DFA：从同一个状态射出的同标记箭弧最多有一条。
- NFA：从同一个状态射出的同标记箭弧可以有多条。

- ε-NFA：有ε转移的NFA。
- ε转移：状态间的ε转移不消耗输入符号。

- 在2.3节介绍NFA、
- 在2.4节介绍ε-NFA
- 之后，二者不再区分，统称NFA



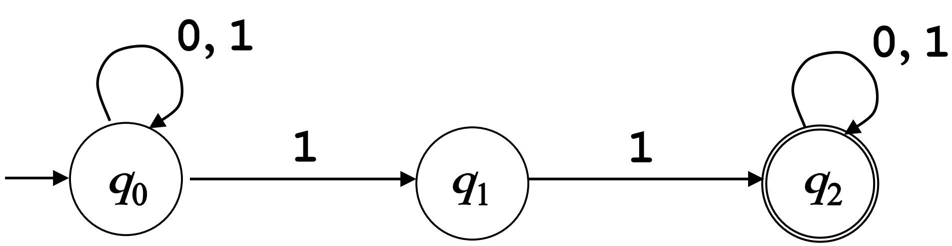
表示NFA：代数、转移图、转移表

- NFA $A=(Q, \Sigma, \upsilon, q_0, F)$
  - 状态集 $Q=\{q_0, q_1, q_2\}$
  - 字母表 $\Sigma=\{0, 1\}$
  - 转移函数υ: $Q\times\Sigma\rightarrow 2^Q$
  - 初始状态$q_0\in Q$
  - 接受状态集合 $F=\{q_2\}\subseteq Q$

$$\upsilon=\{((q_0,0),\{q_0\}), ((q_0,1),\{q_0,q_1\}), ((q_1,0),\{\}),((q_1,1),\{q_2\}), ((q_2,_1),\{q_2\}),((q_2,1),\{q_2\})\}$$

NFA $A=(\{q_0,q_1,q_2\}, \{0,1\}, \upsilon, q_0, \{q_2\})$

两种定义：
转移函数υ是映射；υ允许$\perp$值
用$\upsilon(q, a)=\perp$表示$\upsilon(q, a)$无定义。



	0	1
$\rightarrow q_0$	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\emptyset$	$\{q_2\}$
$*q_2$	$\{q_2\}$	$\{q_2\}$

NFA的判定性质

- 输入：NFA $A=(Q, \Sigma, \upsilon, q_0, F)$，$w\in\Sigma^*$
- 输出：接受、拒绝w

```
int nfa(w) {
T = {q0};                      //活动状态集初始化
x = w;                         //剩余串初始化
while (x != ε) {
        x = ay;                        //求出当前输入符号a
        T = ∪q∈T · υ(q, a);    //更新活动状态集
        x = y;}                        //更新剩余串
return T∩F != ∅;          //活动状态集中若有接受状态则接受
}
```

NFA的扩展转移函数

- 为了方便表示读入一个串所发生的连续转移，扩展转移函数υ为$\tilde{\upsilon}$，$\tilde{\upsilon}$被称为扩展转移函数。

q 是NFA的状态，
对于串$w=a_1a_2...a_n$，如果，
$T_0=\{q\}$，
$T_i=\bigcup p\in T_{i-1} \cdot \upsilon(p, a_i), 1\le i\le n$，
那么，$\tilde{\upsilon}(q, w)=T_n$

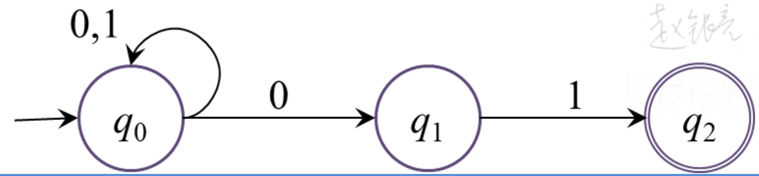
- 用归纳方式定义$\tilde{\upsilon}$为

基础：$\tilde{\upsilon}(q, \varepsilon) = \{q\}$

归纳：$\tilde{\upsilon}(q, wa) = \bigcup p\in\tilde{\upsilon}(q, w) \cdot \upsilon(p, a)$

- $\tilde{\upsilon}(q, w)$：对于始端为q，标记为w的所有路径，由它们的末端组成的集合。

例2.11的示例



$$\tilde{\upsilon}(q_0, 01011) = \bigcup_{p\in\tilde{\upsilon}(q_0, 0101)} \upsilon(p, 1)$$

$$= \bigcup_{p\in[\bigcup_{p\in\tilde{\upsilon}(q_0, 010)} \upsilon(p, 1)]} \upsilon(p, 1)$$

$$= \bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in\tilde{\upsilon}(q_0, 01)} \upsilon(p, 0)]} \upsilon(p, 1)]} \upsilon(p, 1)$$

$$= \bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in\tilde{\upsilon}(q_0, 0)} \upsilon(p, 1)]} \upsilon(p, 0)]} \upsilon(p, 1)]} \upsilon(p, 1)$$

$$= \bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in\tilde{\upsilon}(q_0, \varepsilon)} \upsilon(p, 0)]} \upsilon(p, 1)]} \upsilon(p, 0)]} \upsilon(p, 1)]} \upsilon(p, 1)$$

$$= \bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in\{q_0\}} \upsilon(p, 0)]} \upsilon(p, 1)]} \upsilon(p, 0)]} \upsilon(p, 1)]} \upsilon(p, 1)$$

$$= \bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in\{q_0, q_1\}} \upsilon(p, 1)]} \upsilon(p, 0)]} \upsilon(p, 1)]} \upsilon(p, 1)$$

$$= \bigcup_{p\in[\bigcup_{p\in[\bigcup_{p\in\{q_0, q_2\}} \upsilon(p, 0)]} \upsilon(p, 1)]} \upsilon(p, 1)$$

$$= \bigcup_{p\in[\bigcup_{p\in\{q_0, q_1\}} \upsilon(p, 1)]} \upsilon(p, 1)$$

$$= \bigcup_{p\in\{q_0, q_2\}} \upsilon(p, 1)$$

$$= \{q_0\}$$

$$\tilde{\upsilon}(q_0, \varepsilon) = \{q_0\}$$

$$\tilde{\upsilon}(q_0, 0) = \bigcup_{p\in\tilde{\upsilon}(q_0, \varepsilon)} \upsilon(p, 0) = \{q_0, q_1\}$$

$$\tilde{\upsilon}(q_0, 01) = \bigcup_{p\in\tilde{\upsilon}(q_0, 0)} \upsilon(p, 1) = \upsilon(q_0, 1) \cup \upsilon(q_1, 1) = \{q_0, q_2\}$$

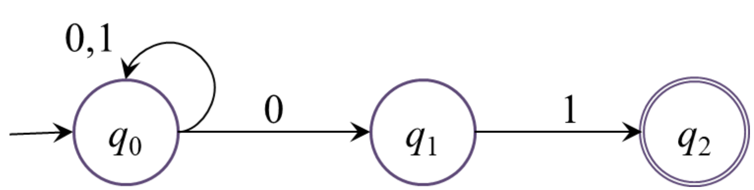
$$\tilde{\upsilon}(q_0, 010) = \bigcup_{p\in\tilde{\upsilon}(q_0, 01)} \upsilon(p, 0) = \upsilon(q_0, 0) \cup \upsilon(q_2, 0) = \{q_0, q_1\}$$

$$\tilde{\upsilon}(q_0, 0101) = \bigcup_{p\in\tilde{\upsilon}(q_0, 010)} \upsilon(p, 1) = \upsilon(q_0, 1) \cup \upsilon(q_1, 1) = \{q_0, q_2\}$$

$$\tilde{\upsilon}(q_0, 01011) = \bigcup_{p\in\tilde{\upsilon}(q_0, 0101)} \upsilon(p, 1) = \upsilon(q_0, 1) \cup \upsilon(q_2, 1) = \{q_0\}$$

NFA的语言

- NFA $A=(Q, \Sigma, \upsilon, q_0, F)$的语言，
- $L(A) = \{w \in \Sigma^* \mid \tilde{\upsilon}(q_0, w) \cap F \neq \emptyset\}$。
- 这个定义也显示，只要有一个成功线索，就表示接受该输入串。
- 证明例2.11接受{0,1}上2后缀等于01的串集合（语言）
  - 断言1：$w \in \Sigma^*, q_0 \in \tilde{\upsilon}(q_0, w)$；
  - 断言2：$q_1 \in \tilde{\upsilon}(q_0, w)$，当且仅当w以0结尾；
  - 断言3：$q_2 \in \tilde{\upsilon}(q_0, w)$，当且仅当w以01结尾。



小结

- DFA：定义及表示、判定性算法、扩展转移函数、语言与DFA构建。
- NFA：定义及表示，判定性算法、扩展转移函数、线索穷举，活动状态集

- 作业：
- 2.5节习题 2.1(2); 2.3~2.4