F4 正则语言 2024

正则语言

正则语言

- 语言L是正则语言，当且仅当存在Σ上的DFA、NFA、ε-NFA或者RE（统称为语言识别器），接受L。

① ~~对于正则语言上的连接、并和闭包运算，其结果都是正则语言。语言上的运算还有很多，运算结果会不会超出正则语言范畴呢？这就是有关正则语言的"封闭性"问题。~~

② 断言某语言是正则的，只需要找到一个能接受它的语言识别器即可，但是要断言某语言是非正则的，就难以证明不存在接受它的语言识别器，需要依赖"泵引理"进行证明。

③ ~~通过自动机的判定性质知道给定串是否为它所接受。那么正则语言的判定性质呢？（空性、成员性、复杂度）~~

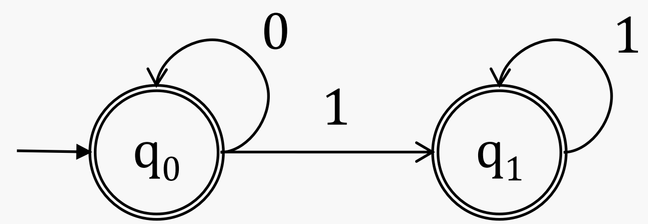
④ 我们知道，接受语言L的DFA不是唯一的，势必让我们对最简（状态最少）的那个DFA感兴趣，这就是"DFA最小化" 问题。

3.4.1 正则语言的泵引理

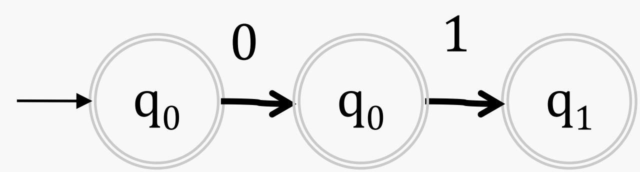
- 观察有n个状态的DFA A=(Q, Σ, υ, q_0, F)，
  - 若L(A)所有成员的长度都小于n，那么L(A)是正则语言。
  - 若w∈L(A)，且|w|>=n，那么A中有PATH(q_0,q,w)，其中q∈F。（始端为q_0末端属于F标记为w的路径）
  - PATH(q_0,q,w)有|w|+1个顶点，那么其中至少两个顶点必然相同。（鸽巢原理）
  - 如果p就是这样的顶点，那么PATH(q_0,q,w)可分段为PATH(q_0,p,x)，PATH(p,p,y)，PATH(p,q,z)。
  - 那么必然有xy^kz∈L(A), k≥0。
- 语言L不是正则语言，当且仅当没有能识别L的DFA、NFA和RE存在。

讨论PATH(q_0,q,w)分段情形

- 讨论PATH(q_0,q,w)分段情形，也就是正则语言上的情形：
  - PATH(q_0,p,x)， $|x|\geq 0, |xy|\leq n$
  - PATH(p,p,y) ， $|y|\geq 1, |xy| \leq n$
  - PATH(p,q,z) ， $|xyz|\geq n$。
  - 满足$w=xyz$， $xy^kz\in L(A), k\geq 0$。

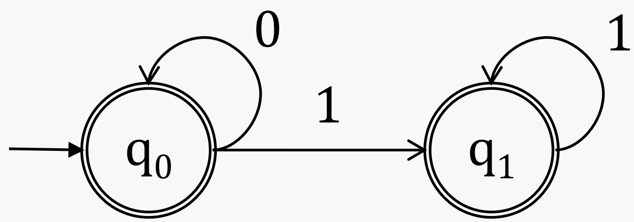


- 例： $L=\{0^m1^l \mid m,l\geq 0\}$
  - 对于n=2， $w=01\in L$， $w=xyz$， 其中，
  - $x=\varepsilon$， $y=0$， $z=1$
  - $|xy|=|0|\leq n$， $|w|=|xyz|\geq n$
  - $xy^kz\in L$， $k\geq 0$， 即$\varepsilon 0^k 1\in L$， $k\geq 0$



- $w= 0^m1^l$， 其中$m+l\geq 2$， $m,l \geq 0$
  - w有： 000*1*、011*、111*， 故x有： ？

例: $L=\{0^m1^l \mid m,l\geq 0\}$



- $w=0^m1^l$，其中$m+l\geq 2$，$m,l\geq 0$
- n=2，所有w，|w|≥2，为：000*1*、011*、111*

w	|w|≥n?	x	y	z	|xy|≤n?	$xy^kz\in L$?
000*1*	是	ε	0	00*1*	是	是
000*1*	是	ε	00	0*1*	是	是
000*1*	是	0	0	0*1*	是	是
011*	是	ε	0	11*	是	是
011*	是	ε	01	1*	是	不是
011*	是	0	1	1*	是	是
111*	是	ε	1	11*	是	是
111*	是	ε	11	1*	是	是
111*	是	1	1	11*	是	是

正则语言的泵引理

- 定理3.3（泵引理）设L是正则语言，则存在正整数n，对任意$w \in L$，如果$|w| \geq n$，那么有分段$w=xyz$，使得：
  ① $y \neq \varepsilon$
  ② $|xy| \leq n$
  ③ $xy^kz \in L$，$k \geq 0$

- 正则语言满足泵引理；满足泵引理的不一定是正则语言。
- 不满足泵引理必定不是正则语言。
- 泵引理常用于证明语言L是非正则语言（反证法）：
  - 对于任意正整数n，存在$w \in L$，$|w| \geq n$，存在切分$w=xyz$，其中，条件①和②满足但条件③不满足。

例：非正则语言$\{0^m1^m|m>1\}$

- $L=\{0^m1^m|m>1\}$
- 对于任意正整数n，选$w=0^n1^n$， $w\in L$， $|w|\geq n$，
- 设$w=xyz$，其中$x=0^{m-k}$，$y=0^k$，$z=0^{n-m}1^n$，$m\leq n$，$k>0$，则有：
- ①$y\neq\varepsilon$
- ②$|xy|\leq n$
- ③对于$i=0$，$xy^iz=0^{n-k}1^n\notin L$，矛盾

证明泵引理

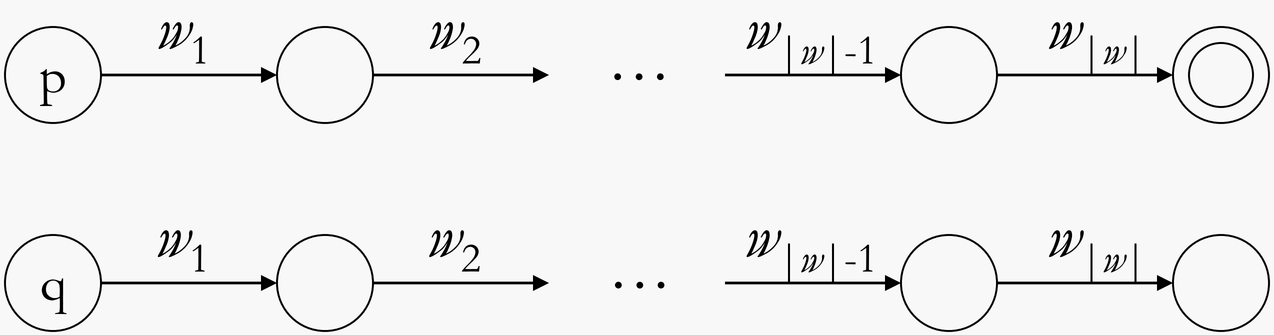
- 证明：假设L是正则语言，那么存在DFA $A=(Q, \Sigma, \delta, q_0, F)$有$L=L(A)$。
- 不妨假设A有n个状态，$|Q|=n$并考虑长度不小于n的串$w=a_1 \ldots a_m$，其中$m \geq n$并且$a_1, \ldots, a_m \in \Sigma$。
- 对于$i=0,1,\ldots,n$定义状态$p_i=\hat{\delta}(q_0, a_1 \ldots a_i)$，注意$p_0=q_0$。
- 由鸽巢原理，p_0到p_n这$n+1$个状态不可能都不同，因为Q只有n个元素。因此，我们能够找到两个不同的非负整数i和j，满足$0 \leq i < j \leq n$，使得$p_i=p_j$。
- 至此，取$w=xyz$，其中$x= a_1 \ldots a_i$，$y= a_{i+1} \ldots a_j$，$z= a_{j+1} \ldots a_m$，得到$xy^kz \in L, k \geq 0$。

例4.3

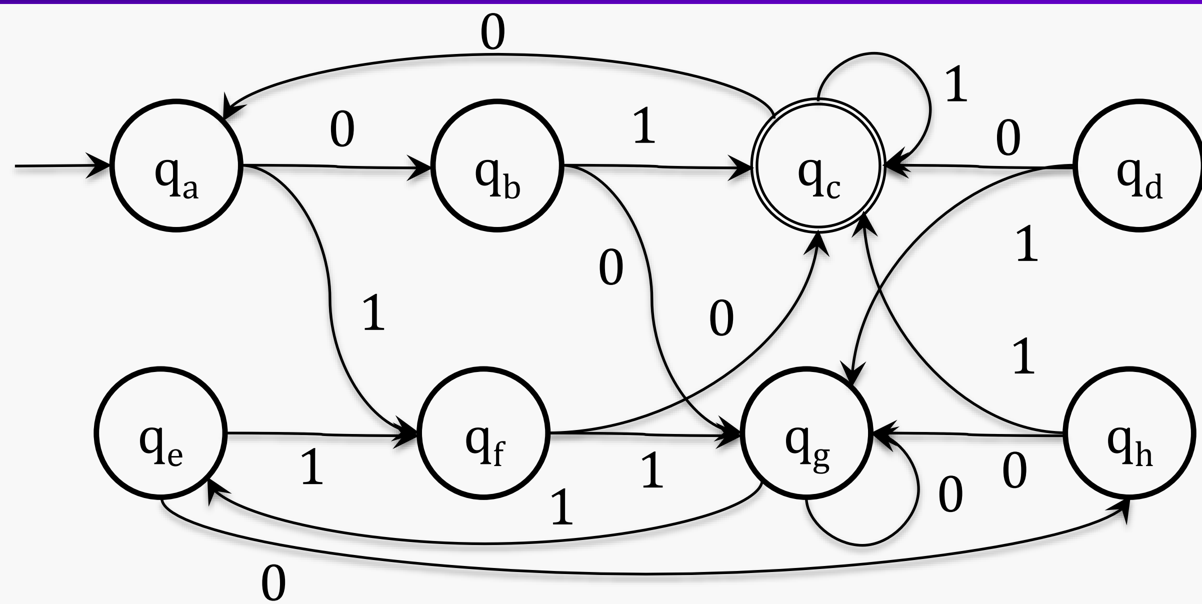
- 证明$L = \{1^p | p$是素数$\}$不是正则语言。
- 反证法，假定L是正则的，
- 考虑某个素数$q \geq n+2$, 其中n是泵长度
- 选择串$w = 1^q$，我们可以写成$w=xyz$使得$y \neq \varepsilon$且$|xy| \leq n$
- 令$|y|=m$, 那么$|xz|=q-m$, 考虑串$s=xy^{q-m}z$, 根据泵引理属于L
- $|xy^{q-m}z|=|xz|+(q-m)|y|=q-m+(q-m)m=(m+1)(q-m)$
- 注意到$m+1>1$, 当$y \neq \varepsilon$
- 另注$q \geq n+2$ 则$q-m>1$，（$m \leq n$）
- 矛盾。

3.7 DFA最小化

- DFA $(Q, \Sigma, \upsilon, q_0, F)$的任意两个状态之间会有什么关系？
- 状态p和q是等价的，如果，
  - $\forall w \in \Sigma^* \cdot (\tilde{\upsilon}(p, w) \in F \leftrightarrow \tilde{\upsilon}(q, w) \in F)$
- 状态p和q是可区分的，如果，
  - $\exists w \in \Sigma^* \cdot (\tilde{\upsilon}(p, w) \in F \wedge \tilde{\upsilon}(q, w) \notin F)$，或者，
  - $\exists w \in \Sigma^* \cdot (\tilde{\upsilon}(p, w) \notin F \wedge \tilde{\upsilon}(q, w) \in F)$



例: 状态p和q等价或可区分



$(q_c, q_a)\varepsilon$ $(q_c, q_b)\varepsilon$ $(q_c, q_d)\varepsilon$ $(q_c, q_e)\varepsilon$ *可区分的*
$(q_c, q_f)\varepsilon$ $(q_c, q_g)\varepsilon$ $(q_c, q_h)\varepsilon$
$(q_a, q_b)1$ $(q_a, q_d)0$ $(q_a, q_f)0$ $(q_a, q_g)01$ $(q_a, q_h)1$
$(q_b, q_d)1$ $(q_b, q_e)1$ $(q_b, q_f)1$ $(q_b, q_g)1$
$(q_d, q_e)0$ $(q_d, q_g)0$ $(q_d, q_h)0$ ……

(q_b, q_h): $1q_c$ 或 $0q_g$ (q_d, q_f) : $0q_c$或$1q_g$
(q_a, q_e): $1q_f$ 或$01q_c$ 或 $00q_g$ *等价的*

最小化DFA的填表算法

输入： DFA $(Q,\Sigma,\upsilon,q_0,F)$

输出： DFA $(\mathcal{P},\Sigma,move[G,a],G_0,F')$, $q_0 \in G_0$, $\forall G \in F' \cdot G \cap F \neq \varphi$

对Q元素建立索引|Q[i], i=1,..|Q|

$\mathcal{C}=F\times(Q\backslash F)$;

对Q\F表示为序列S，有(Q\F)#个元素；

for(q∈Q\F

发现等价对的填表算法

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

输出： $\{(q, p) \mid q, p\in Q$且q与p是等价的$\}$

$\mathbb{T}$初始化为$Q\times Q$的下三角；

foreach $q\in F$ do 标记$\mathbb{T}$的q行和q列诸元素；

置c为yes；

while c do {

    置c为no；

    foreach $\mathbb{T}$的未标记元素$\mathbb{T}[q, p]$do

        foreach $a\in\Sigma$ do

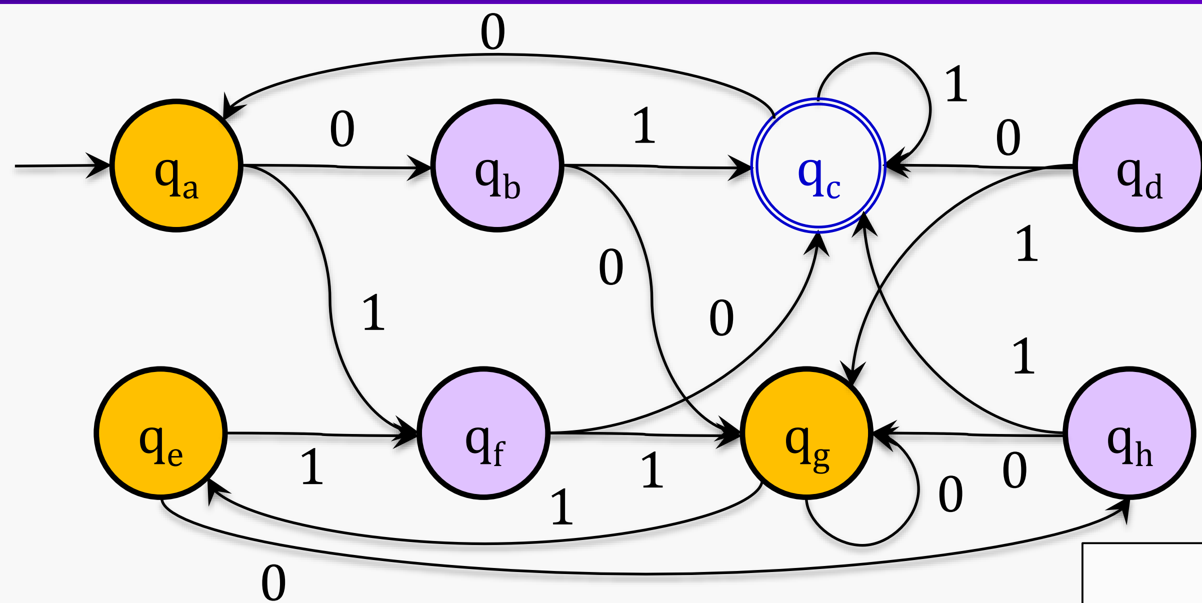
            if ($\mathbb{T}[\upsilon(q, a) , \upsilon(p, a)]$有标记) {

                标记$\mathbb{T}[q, p]$；

                置c为yes； }}

return $\mathbb{T}$的所有未带标记元素。

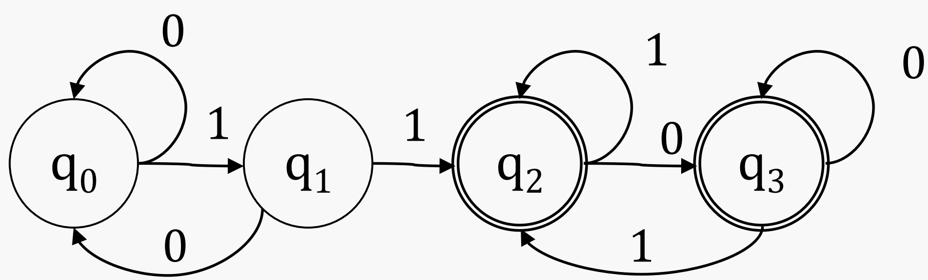
例: 填表算法



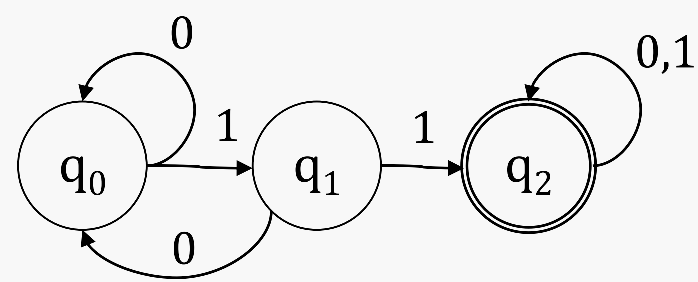
q_b	X1						
q_c	X	X					
q_d	X0	X1	X				
q_e		X1	X	X0			
q_f	X0	X0	X		X0		
q_g	X1	X1	X	X0	X1	X0	
q_h	X1		X	X0	X1	X1	X1
	q_a	q_b	q_c	q_d	q_e	q_f	q_g

	0	1
*{c}	{a,e}	{c}
→{a,e}	{b,h}	{d,f}
{d,f}	{c}	{g}
{b,h}	{g}	{c}
{g}	{g}	{a,e}

例: 填表算法



图(b)

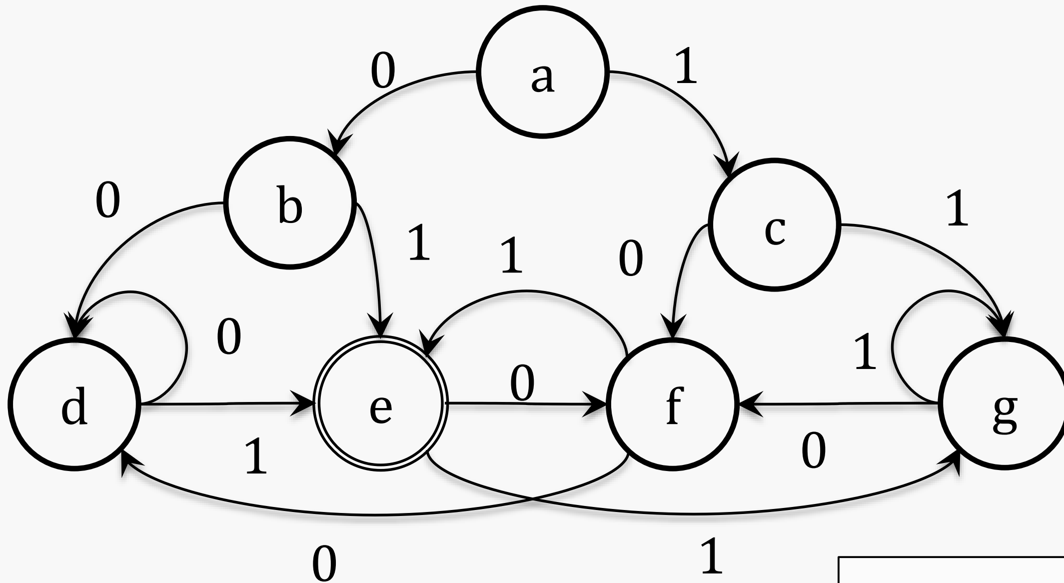


图(c)

q_1	X1		
q_2	X	X	
q_3	X	X	
	q_0	q_1	q_2

	0	1
→{0}	{0}	{1}
{1}	{0}	{2,3}
*{2,3}	{2,3}	{2,3}

例: 填表算法

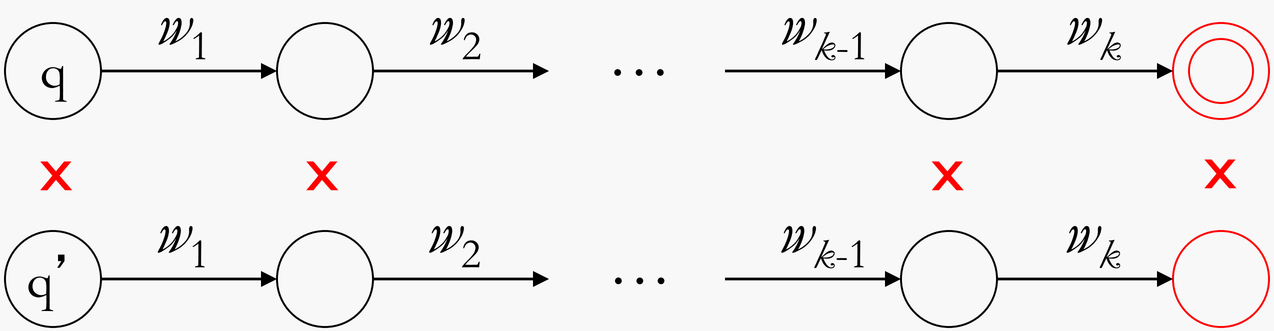


b	X1					
c	?	X1				
d	X1		X1			
e	X	X	X	X		
f	X1		X1	?	X	
g	?	X1	?	X1	X	X1
	a	b	c	d	e	f

	0	1
→{a,g,c}	{d,b,f}	{a,g,c}
{d,b,f}	{d,b,f}	{e}
*{e}	{d,b,f}	{a,g,c}

证明填表算法能行

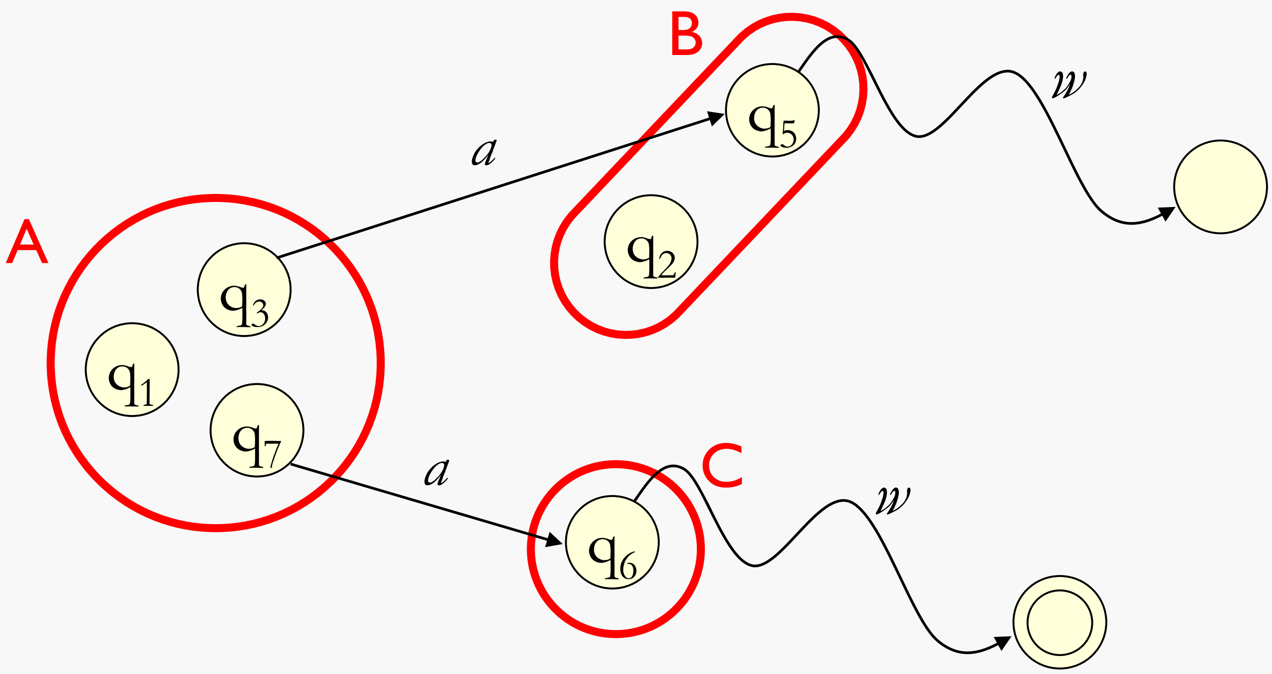
- 定理4.20 如果通过填表算法不能区分两个状态，那么它们是等价的。



反证法：假定不能发现q和q'

证明填表算法能行

- 为什么当我们合并等价状态时不会产生不一致性?



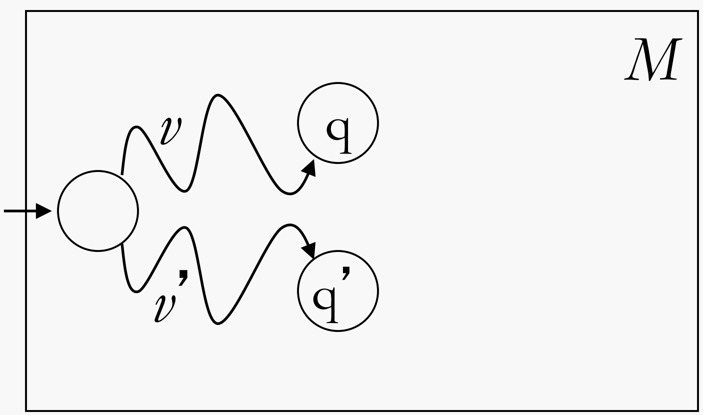
不一致：合并了q3和q7到A中
B和C可区分，所以存在w一个到达接受状态另一个不能
结果A中状态不全是等价的

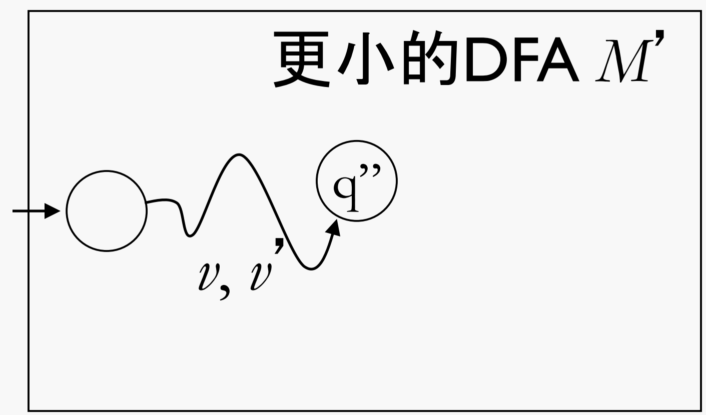
证明填表算法能行

- 为什么没有更小的DFA了?

假定有

根据鸽巢原理必定出现下面情况:

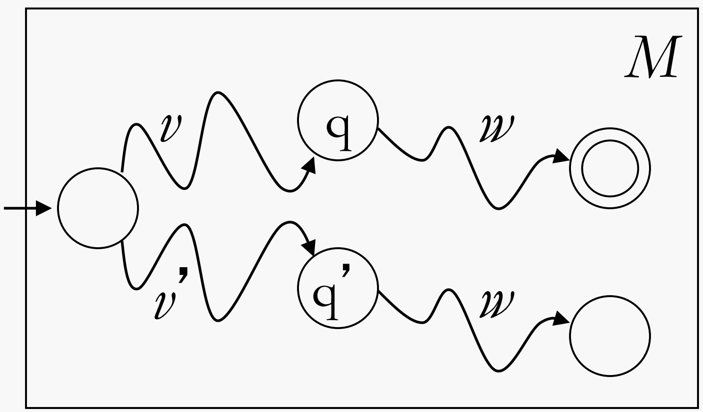


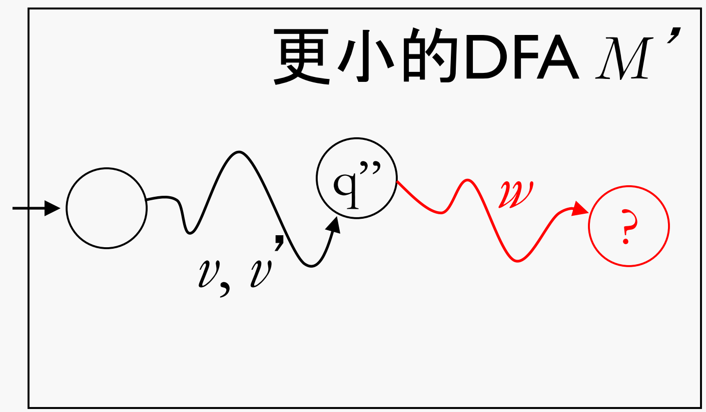


证明填表算法能行

- 为什么没有更小的DFA了?

但是之后





每个状态序对都是可区分的.

q'' 不能存在!

DFA最小化的划分算法

也称Hopcroft算法

- 算法的结果是对Q的一个划分$\mathcal{P}$，每个划分块G∈$\mathcal{P}$都是一致的极大划分块，
- G相对于$\mathcal{P}$是一致的，当且仅当，

$$\forall a\in\Sigma \cdot \exists G'\in\mathcal{P} \cdot \forall q\in G \cdot \upsilon(q, a)\in G'$$

- 如果G是相对于$\mathcal{P}$一致的，记为$\forall a\in\Sigma \cdot \exists G'\in\mathcal{P} \cdot move[G, a]=G'$
- G是极大的，当且仅当，

$$\text{如果}\exists G'\in\mathcal{P} \cdot \forall a\in\Sigma \cdot move[G, a]=move[G', a]\text{那么}G=G'$$

DFA最小化的划分算法

输入： DFA $(Q, \Sigma, \upsilon, q_0, F)$

输出： DFA $(\mathcal{P}, \Sigma, move[G, a], G_0, F')$, $q_0 \in G_0$, $\forall G \in F' \cdot G \cap F \neq \varphi$

$\mathcal{P} = \{F, Q \backslash F\}$

while ($G \in \mathcal{P}$ && G未标记)

if (G是一致的) 标记G //G是一致的，

//若$\forall p,q \in G \cdot \forall a \in \Sigma \cdot$ $MOVE[G,p,a] = MOVE[G,q,a]$

else{按最大化一致性原则得出G的划分$\mathcal{G}$;

$\mathcal{P}$中去除G; $\mathcal{P} = \mathcal{P} \cup \mathcal{G}$;

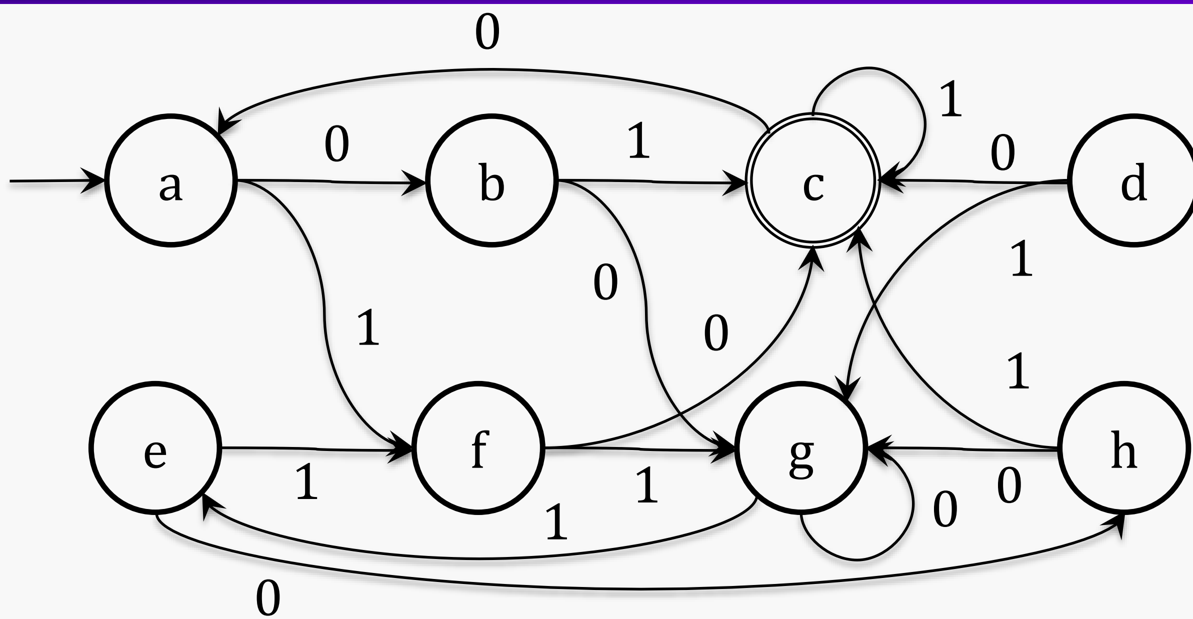
$\mathcal{P}$中去除所有元素的标记; }

令$MOVE[G,q,a] = G'$, 其中$G, G' \in \mathcal{P}, a \in \Sigma, \upsilon(q,a) \in G', q \in G$;

令$move[G,a] = MOVE[G,-,a]$。

（*DFA最小化的Hopcroft算法*）

例: 采用Hopcroft算法最小化



G1={c}
G2={a,b,d,e,f,g,h)

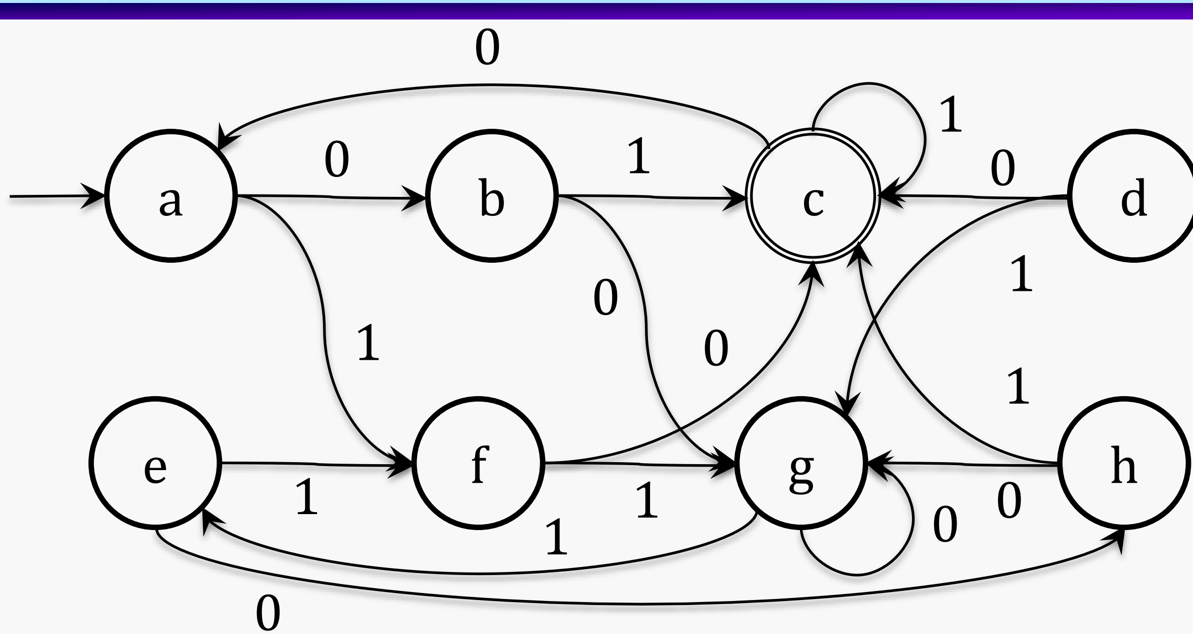
G2	0	1
a	G2	G2
b	G2	G1
d	G1	G2
e	G2	G2
f	G1	G2
g	G2	G2
h	G2	G1

G1={c}
G2={a,e,g}
G3={d,f}
G4={b,h}

G2	0	1
a	G4	G3
e	G4	G3
g	G2	G2

G1={c}
G2={a,e}
G3={d,f}
G4={b,h}
G5={g}

检查一致性并构建转移表



G1={c}
G2={a,e}
G3={d,f}
G4={b,h}
G5={g}

G1	0	1
c	G2	G1

G2	0	1
a	G4	G3
e	G4	G3

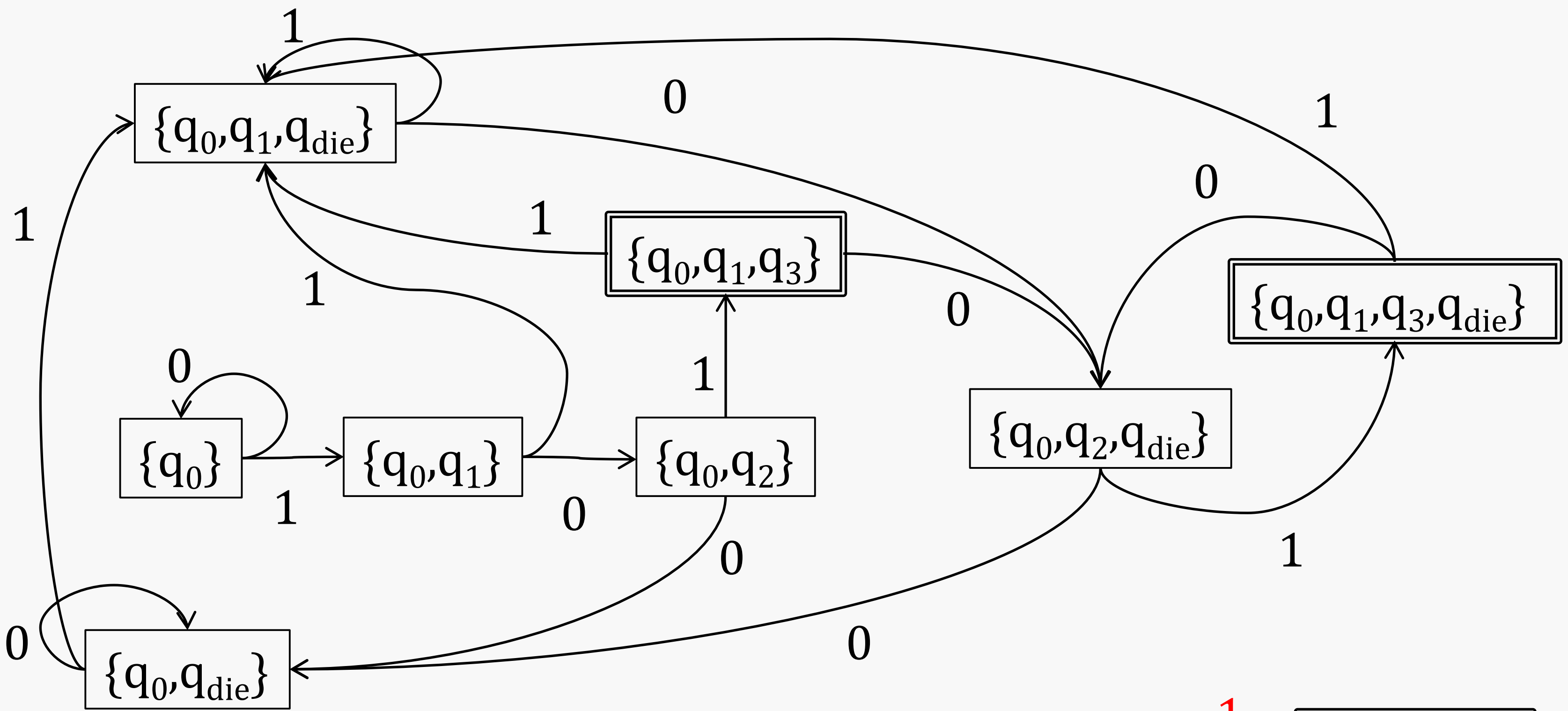
G5	0	1
g	G5	G2

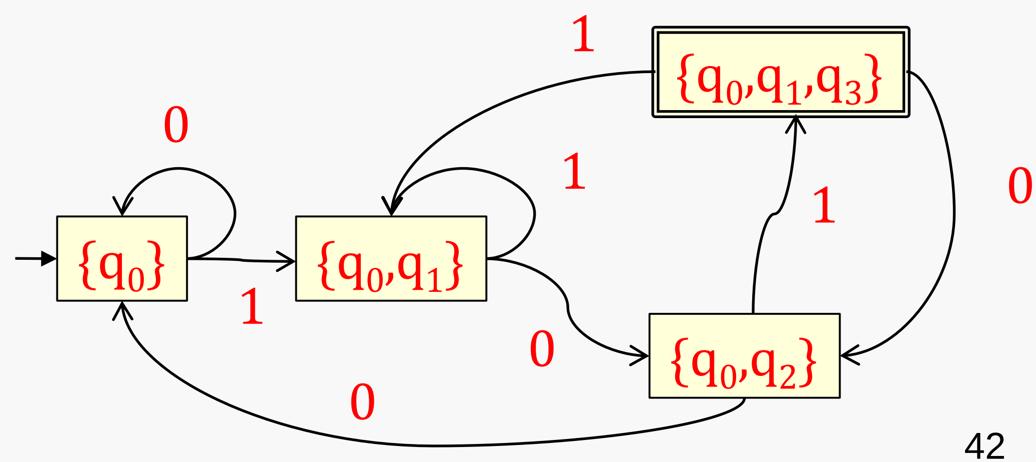
G3	0	1
d	G1	G5
f	G1	G5

G4	0	1
b	G5	G1
h	G5	G1

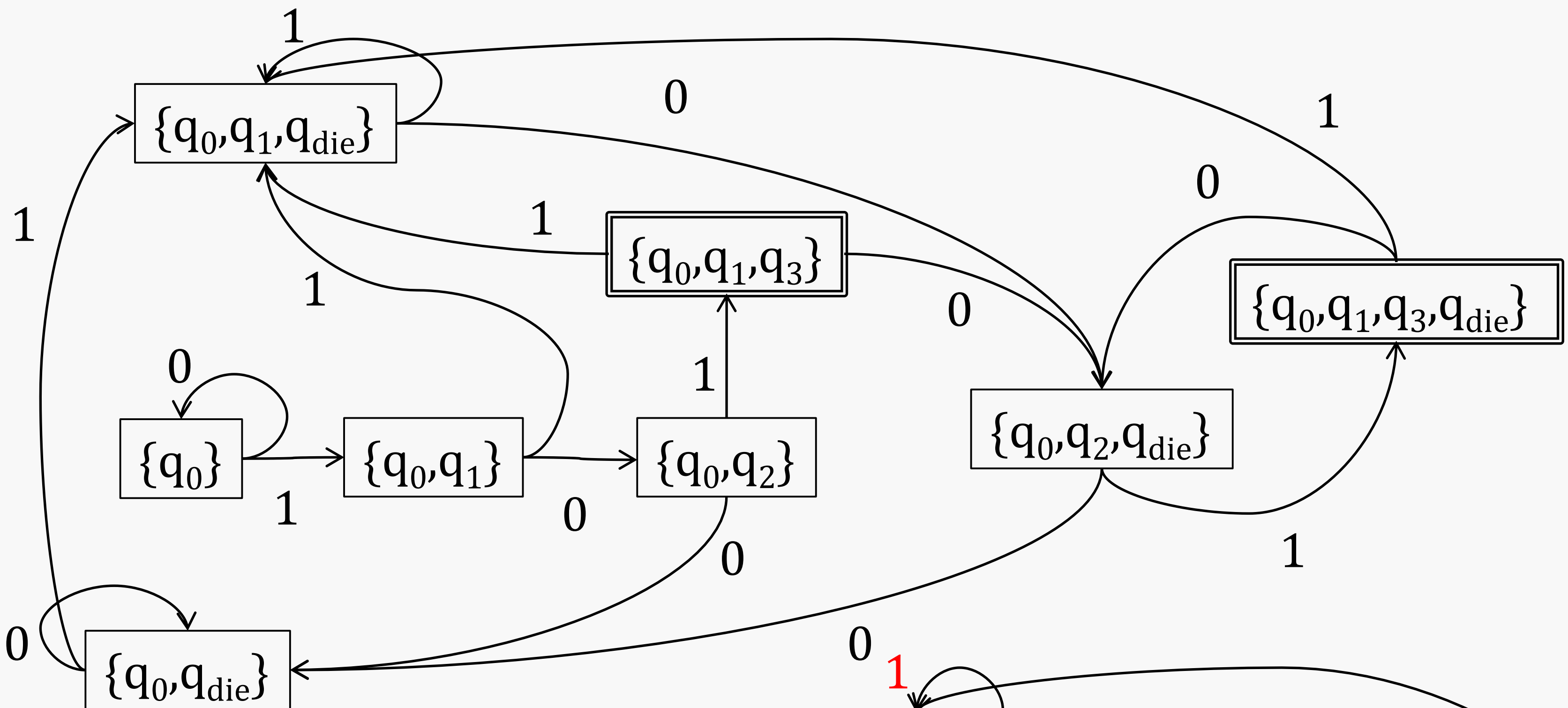
	0	1
*G1={c}	G2	G1
→G2={a,e}	G4	G3
G3={d,f}	G1	G5
G4={b,h}	G5	G1
G5={g}	G5	G2

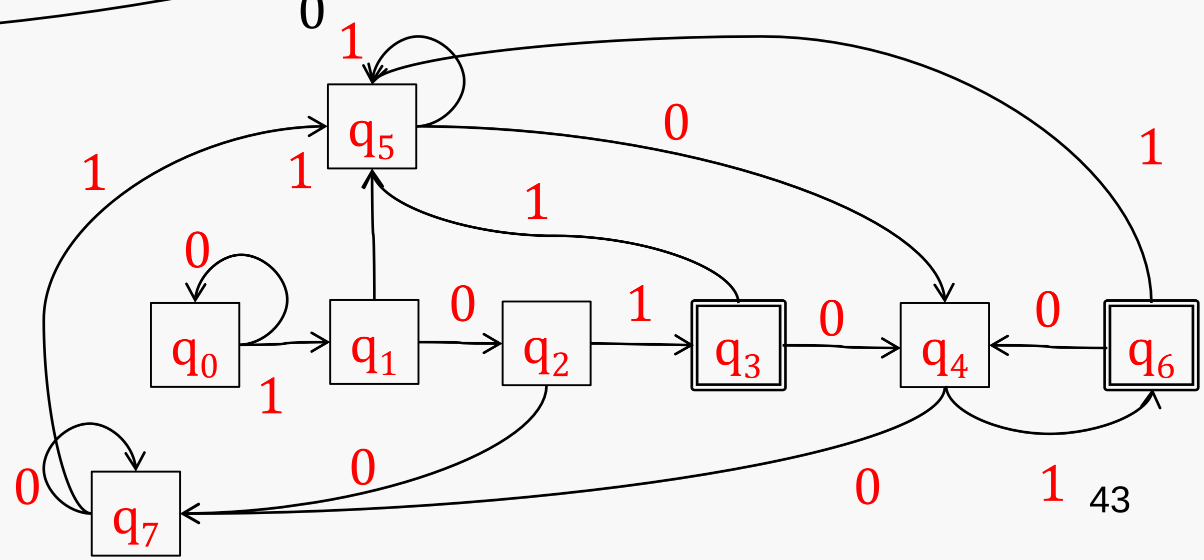
3-后缀为101的0-1串



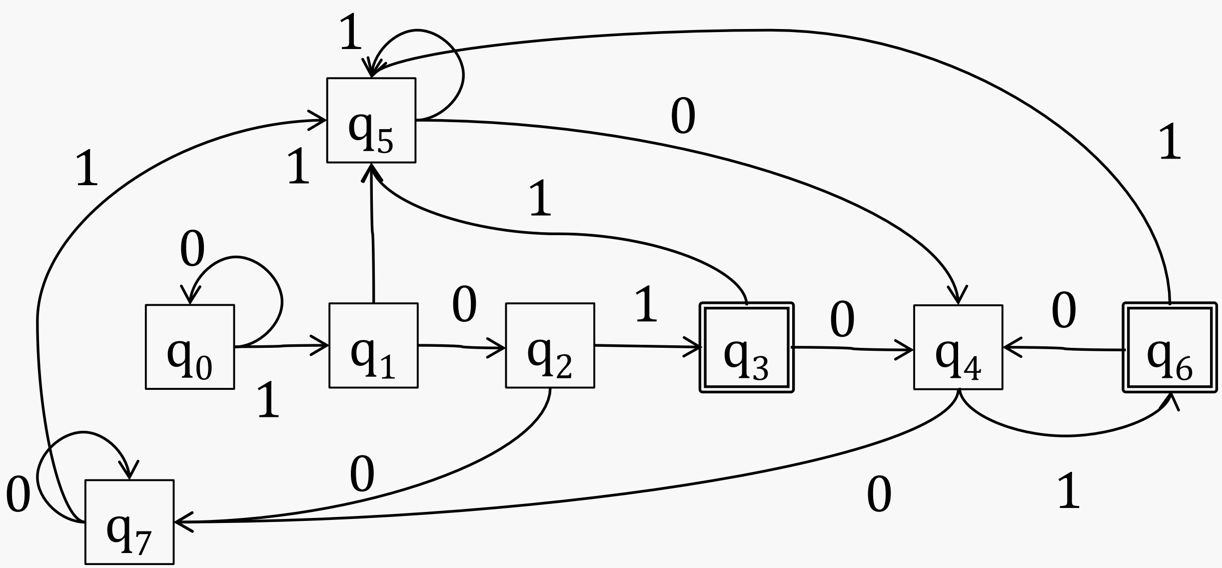


3-后缀为101的0-1串





用划分法进行最小化



G2	0	1
q0	G2	G4
q7	G2	G4

G3	0	1
q2	G2	G1
q4	G2	G1

G4	0	1
q1	G3	G4
q5	G3	G4

G1	0	1
q3	G3	G4
q6	G3	G4

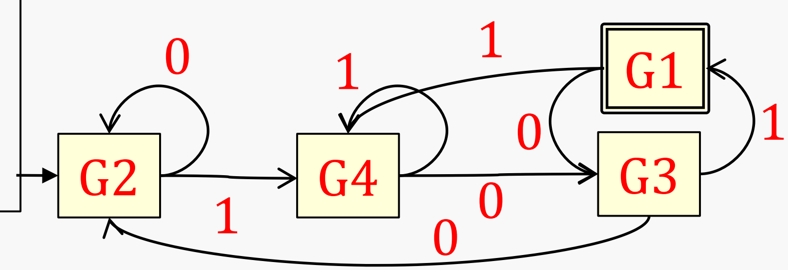
G1={q3, q6}
G2={q0,q1,q2,q4,q5,q7}

G2	0	1
q0	G2	G2
q1	G2	G2
q2	G2	G1
q4	G2	G1
q5	G2	G2
q7	G2	G2

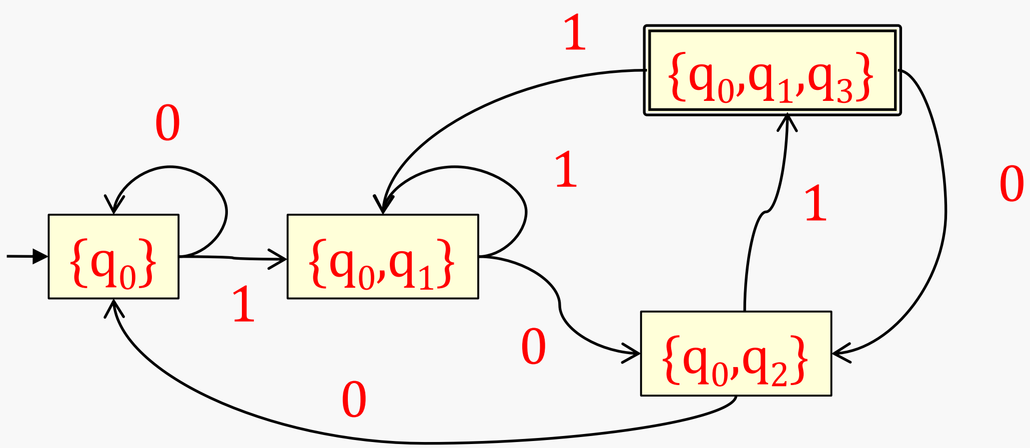
G1={q3, q6}
G2={q0,q1,q5,q7}
G3={q2,q4}

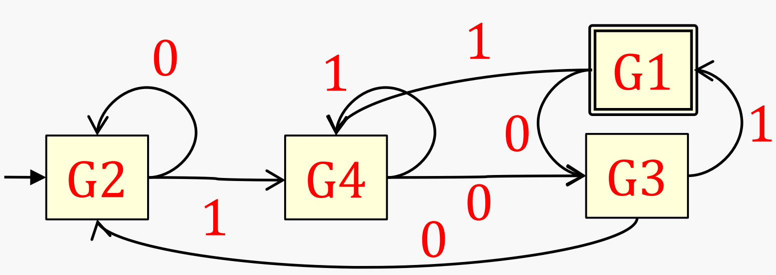
G2	0	1
q0	G2	G2
q1	G3	G2
q5	G3	G2
q7	G2	G2

G1={q3, q6}
G2={q0,q7}
G3={q2,q4}
G4={q1,q5}



3-后缀为101的0-1串





两种方法最小化 DFA方法的比较

填表算法

① 找出等价状态;

② 合并等价状态。

③ 构建等价块间转移关系。

Hopcropt算法

① 分裂划分块为最大化一致性划分块;

② 构建划分块间转移关系。

小结

- 知识点：泵引理、最小化
- 知识点：可区分的状态、等价的状态
- 最小化算法

- 作业：p73: 3.7(c)；3.10-12