第四章

正则表达式

- 第二章和第三章分别讨论的正则文法和有穷状态自动机都是正则语言的形式化描述模型。
- 正则文法擅长语言的产生，有穷状态自动机擅长语言的识别。

- 本章讨论正则语言的另外一种描述模型—正则表达式，它对正则语言的表达具有特殊的优势：正则表达式比正则文法和有穷状态自动机更简单，更容易处理。而且，这种表达式还更接近语言的集合表示和语言的计算机表示。

- 语言的集合表示使得人们能更容易地理解和使用它，而适应计算机的表示形式又使得我们能更容易地使用计算机系统处理语言。所以，就这两方面而言，正则表达式使用起来更方便。

■ 在前面我们已经用到了一些类似正则表达式来表示语言(蓝色部分)：

$L(G)=\{0^n|n\geq 1\}$

$L(G)=\{a^nb^n|n\geq 1\}$

$L(M)=\{0^n1^m2^k|n,m,k\geq 1\}$

■优点：

- 比文法和有穷状态自动机简单；
- 更接近语言的集合表示和计算机表示。

章节目录

4.1 启示

4.2 正则表达式的形式定义

4.3 正则表达式与FA等价

4.4 正则语言等价模型的总结

4.5 本章小结

■ 产生下面语言L的正则文法为？

$L=\{a^nb^mc^k|n,m,k\geq 1\}\cup\{a^ic^nbxa^m|i\geq 0,n\geq 1,m\geq 2,$ x为d和e组成的串$\}$

$\{a^nb^mc^k|n,m,k\geq 1\}$

$A\rightarrow aA|aB$

$B\rightarrow bB|bC$

$C\rightarrow cC|c$

$\{a^ic^nbxa^m|i\geq 0,n\geq 1,$ $m\geq 2,$ x为d和e组成的串$\}$

$A\rightarrow aA|cE$

$E\rightarrow cE|bF$

$F\rightarrow dF|eF|aH$

$H\rightarrow aH|a$

$A\rightarrow aA|aB|cE$

$B\rightarrow bB|bC$

$C\rightarrow cC|c$

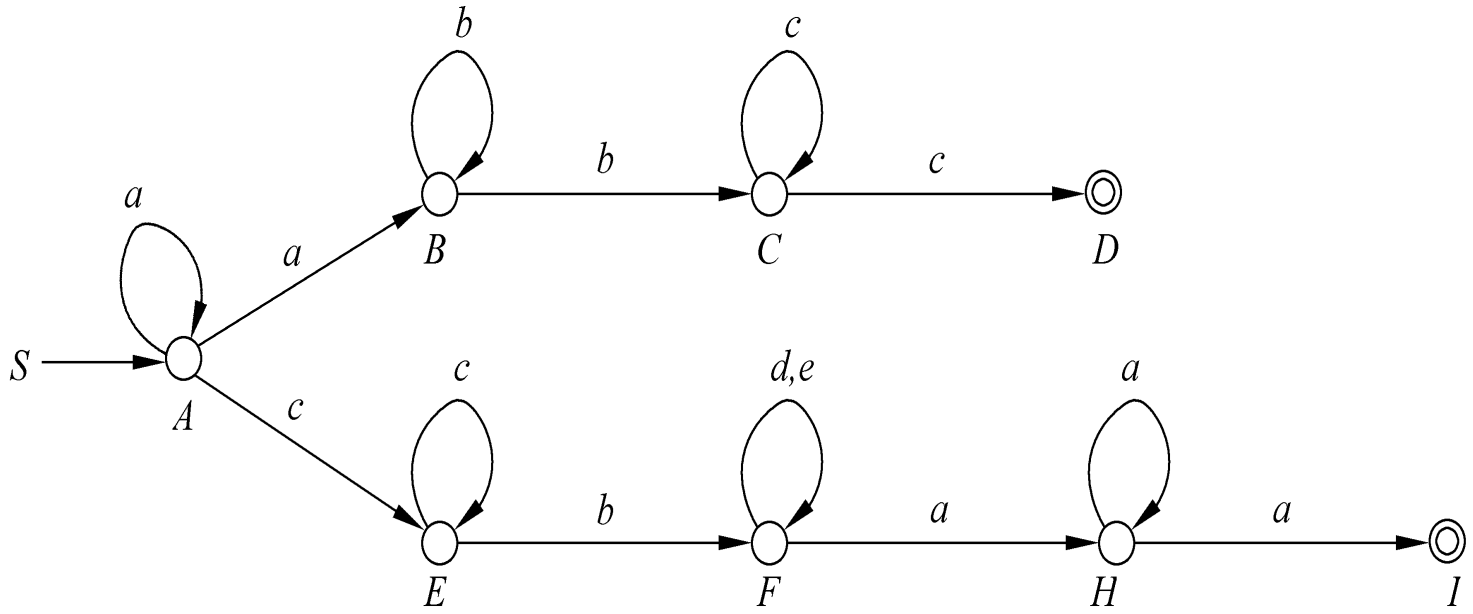
$E\rightarrow cE|bF$

$F\rightarrow dF|eF|aH$

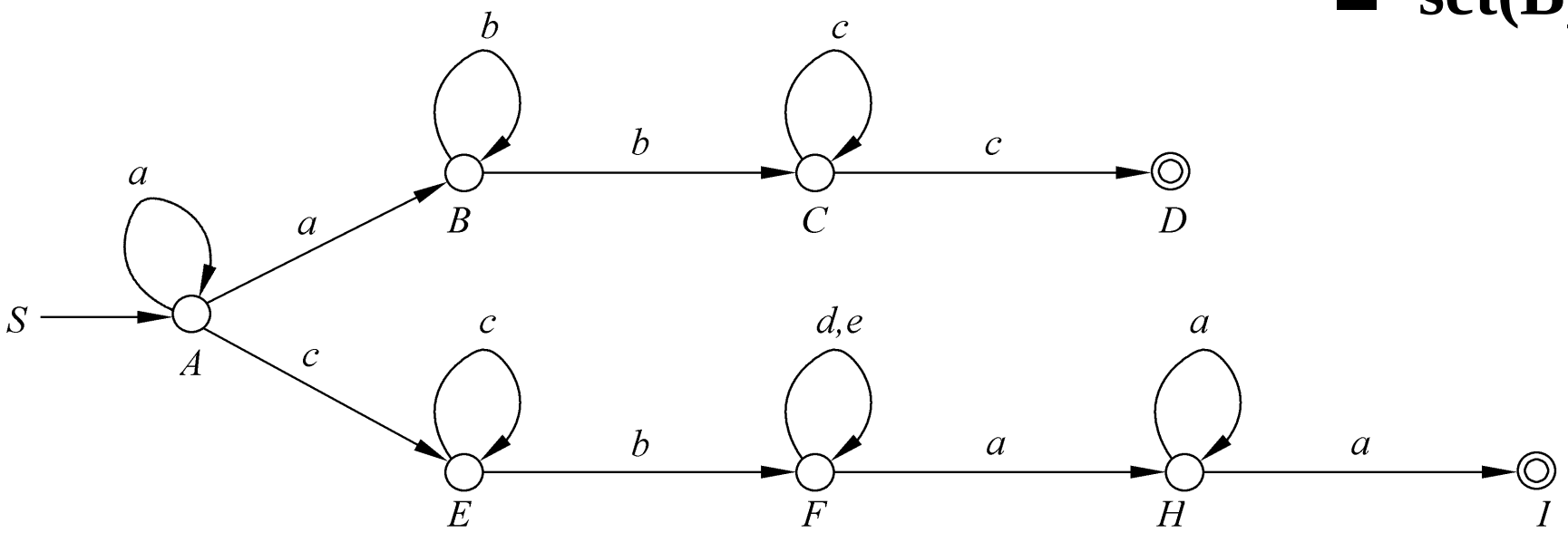
$H\rightarrow aH|a$

■ 产生下面语言的NFA M为

$\{a^n b^m c^k | n,m,k \geq 1\} \cup \{a^i c^n b x a^m | i \geq 0, n \geq 1, m \geq 2, x$为d和e组成的串$\}$

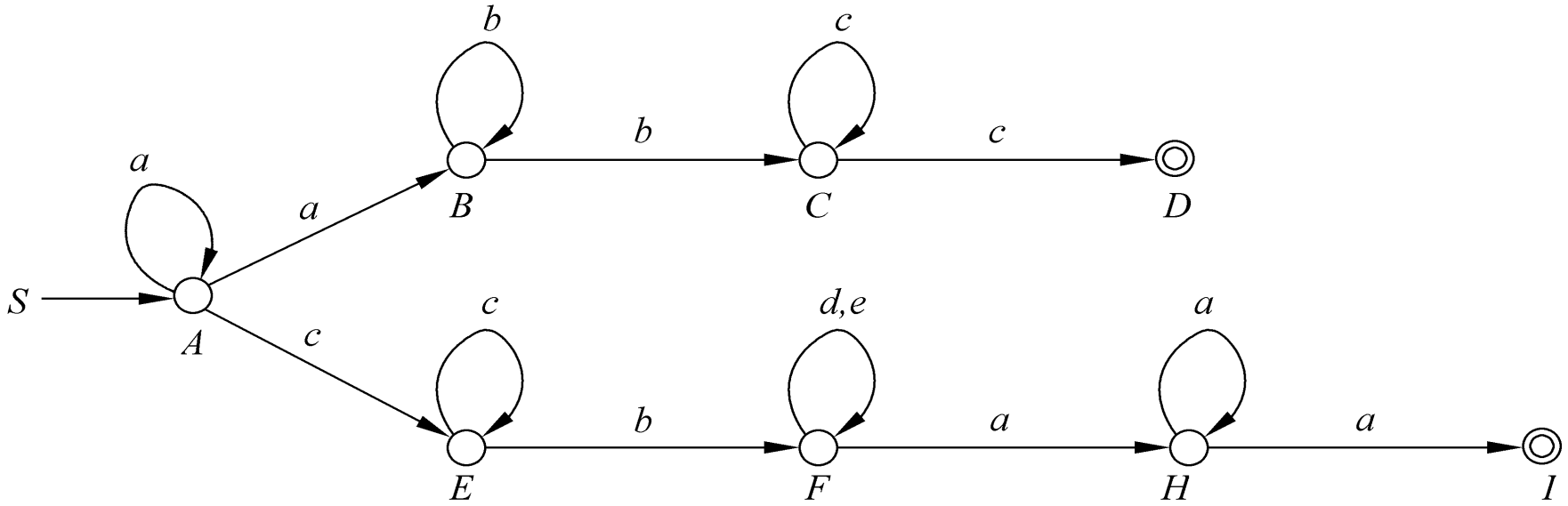


计算集合$\mathbf{set}(q), \forall q$：



- set(A)={a^n|n≥0}={a}*
- set(B)= set(A){a}{b^n|n≥0}
  ={a}*{a}{b}*={a}+{b}*
- set(C)= set(B){b}{c}*={a} +{b}*{b}{c}*={a}+{b}+{c}*
- set(D)= set(C) {c}={a}+{b}+{c}*{c}={a}+{b}+{c}+
- set(E)= set(A){c}{c}*={a}*{c}{c}*={a}*{c}+
- set(F)= set(E){b}{d, e}*={a}*{c}+{b}{d, e}*

4.1 启示



■set(H)= set(F){a}{a}*={a}*{c}+{d, e}*{a}{a}*

={a}*{c}+{d, e}*{a}+

■set(I)= set(H){a}={a}*{c}+{d, e}*{a}+{a}

L(M)= set(D)∪set(I)

= {a}+{b}+{c}+∪{a}*{c}+{d, e}*{a}+{a}

根据集合运算的定义，可得$\{d, e\}=\{d\}\cup\{e\}$。

从而，$\{d, e\}^*=(\{d\}\cup\{e\})^*$。

这样可以将L(M)写成如下形式：

$$L(M) = \{a\}^+\{b\}^+\{c\}^+\cup\{a\}^*\{c\}^+\{d, e\}^*\{a\}^+\{a\}$$

$$=\{a\}^+\{b\}^+\{c\}^+\cup\{a\}^*\{c\}^+(\{d\}\cup\{e\})^*\{a\}^+\{a\}$$

记作：

$$a^+b^+c^+ + a^*c^+(d+e)^* a^+a$$

$$=aa^*bb^*cc^* + a^*cc^*(d+e)^* aaa^*$$

章节目录

4.1 启示

4.2 正则表达式的形式定义

4.3 正则表达式与FA等价

4.4 正则语言等价模型的总结

4.5 本章小结

启示：四则运算表达式的递归定义

① 任何数都是四则运算表达式

② 如果a和b是四则运算表达式，以下均为四则运算表达式：

$$a+b, a-b, a\times b, a\div b\text{和}(a)$$

正则表达式的递归定义

定义4-1 设Σ是一个字母表，

① $\emptyset$是Σ上的正则表达式，它表示语言$\emptyset$；

② $\boldsymbol{\varepsilon}$是Σ上的正则表达式，它表示语言$\{\varepsilon\}$；

③ 对于$\forall a \in \Sigma$, $\boldsymbol{a}$是Σ上的正则表达式，它表示语言{a}；

④ 如果$\boldsymbol{r}$和$\boldsymbol{s}$分别是Σ上的表示语言R和S的正则表达式，则

$\boldsymbol{r}$与$\boldsymbol{s}$的“和” $(\boldsymbol{r}+\boldsymbol{s})$ 是Σ上的正则表达式，表示语言$R \cup S$；

$\boldsymbol{r}$与$\boldsymbol{s}$的“乘积”$(\boldsymbol{rs})$ 是Σ上的正则表达式，表示语言RS；

$\boldsymbol{r}$的克林闭包$\boldsymbol{r}^*$是Σ上的正则表达式，表示语言R^*；

⑤ 只有满足①②③④的表达式才是Σ上的正则表达式。

4.2 正则表达式的形式定义

例4-1 设$\Sigma=\{0,1\}$，下面是Σ上的一些正则表达式及其对应的语言。

正则表达式	表示的语言
$\mathbf{0}$	$\{0\}$
$\mathbf{1}$	$\{1\}$
$(\mathbf{01})$	$\{01\}$
$\mathbf{0+1}$	$\{0,1\}$
$((\mathbf{0+1})^*)$	$\{0,1\}^*$
$((\mathbf{00}))(((\mathbf{00})^*))$	$\{00\}\{00\}^*$
$((((\mathbf{0+1})^*)(\mathbf{0+1}))((\mathbf{0+1})^*))$	$\{0,1\}^+$
$((((\mathbf{0+1})^*)\mathbf{000})((\mathbf{0+1})^*))$	至少有3个连续0的串组成的语言
$((((\mathbf{0+1})^*)\mathbf{0})\mathbf{1})$	以01结尾的串组成的语言
$(\mathbf{1}(((\mathbf{0+1})^*)\mathbf{0}))$	以1开头0结尾的串组成的语言

4.2 正则表达式的形式定义

上述例子的正则表达式中括号太多，为了解决这一个问题，有如下约定：

（1）$\boldsymbol{r}$的正闭包 $\boldsymbol{r}^{+}$表示$\boldsymbol{r}$与$(\boldsymbol{r}^{*})$的乘积或$(\boldsymbol{r}^{*})$ 与$\boldsymbol{r}$的乘积，即 $\boldsymbol{r}^{+}=\boldsymbol{r}\boldsymbol{r}^{*}=\boldsymbol{r}^{*}\boldsymbol{r}$;

（2）闭包运算的优先级最高，乘运算的优先级次之，加运算的优先级最低。所以，当意义明确时，可以省略某些括号，如 $(((\boldsymbol{0}+\boldsymbol{1})^{*})\boldsymbol{000})((\boldsymbol{0}+\boldsymbol{1})^{*}))$可以写成$(\boldsymbol{0}+\boldsymbol{1})^{*}\boldsymbol{000}(\boldsymbol{0}+\boldsymbol{1})^{*}$，$(((\boldsymbol{0}+\boldsymbol{1})^{*})(\boldsymbol{0}+\boldsymbol{1}))((\boldsymbol{0}+\boldsymbol{1})^{*}))$可以写成$(\boldsymbol{0}+\boldsymbol{1})^{*}(\boldsymbol{0}+\boldsymbol{1})(\boldsymbol{0}+\boldsymbol{1})^{*}$;

（3）$\boldsymbol{r}$表示的语言$\mathbf{L}(\boldsymbol{r})$ 在意义明确时也可以直接记为$\boldsymbol{r}$;

（4）加、乘、闭包运算均执行左结合规则。

例4-2 正则表达式运算优先级。

正则表达式E	表示的语言L(E)
$\mathbf{1}+\mathbf{01}^{*}=\mathbf{1}+(\mathbf{0}(\mathbf{1}^{*}))$	• $\{1,0,01,011,0111,\cdots\}$
$\neq \mathbf{1}+(\mathbf{01})^{*}$	• $\{1,\varepsilon,01,0101,010101,\cdots\}$
$\neq (\mathbf{1}+\mathbf{01})^{*}$	• $\{1,01\}^{*}$ • $\{\varepsilon,1,01,11,101,0101,011\cdots\}$
$\neq (\mathbf{1}+\mathbf{0})\mathbf{1}^{*}$	• $\{1,0\}\{\varepsilon,1,11,111,\cdots\}$ • $\{1,0,11,01,111,011,1111,0111\cdots\}$

相等

定义4-2 设r, s是字母表Σ上的正则表达式，如果$L(r) = L(s)$，则称r和s相等，也称为等价，记作$r = s$。

可以证明，对字母表上的正则表达式r, s, t，下列各式成立。

① 结合律：(rs)t=r(st)；(r+s)+t=r+(s+t)
② 分配律：r(s+t)=rs+rt；(s+t)r=sr+tr
③ 交换律：r+s=s+r
④ 幂等律：r+r=r
⑤ 加法运算零元素：r+Φ=r
⑥ 乘法运算单位元：rε=εr=r
⑦ 乘法运算零元素：rΦ=Φr=Φ

4.2 正则表达式的形式定义

⑧ $L(\Phi)=\Phi$

⑨ $L(\varepsilon)=\{\varepsilon\}$

⑩ $L(a)=\{a\}$ （a是字母表Σ上的一个字符）

⑪ $L(rs)=L(r)L(s)$

⑫ $L(r+s)=L(r)\cup L(s)$

⑬ $L(r^*)=(L(r)^*)$

⑭ $L(\Phi^*)=\{\varepsilon\}$

⑮ $L((r+\varepsilon)^*)=L(r^*)$

⑯ $L((r^*)^*)=L(r^*)$

⑰ $L((r^*s^*)^*)=L((r+s)^*)$

⑱ 如果$L(r)\subseteq L(s)$,则$r+s=s$。

n次幂

定义4-3 设r是字母表Σ上的正则表达式， r的n次幂定义为：

- $r^0 = \varepsilon$
- $r^n = r^{n-1}\mathrm{r}\ (\mathrm{n} \geq 1)$

推论：$\mathrm{L}(r^n) = (\mathrm{L}(r))^n$

$r^n\ r^m = r^{n+m}$

例4-3 设$\boldsymbol{\Sigma}=\{0,1\}$，则

- **00**组成表示**语言{00}**
- $(0+1)^*00(0+1)^*$表示所有至少含有两个连续0的串组成的语言
- $(0+1)^*1(0+1)^9$表示所有倒数第10个字符为1的串组成的语言
- $L\big((0+1)^*011\big)=\{x|x$是以011结尾串$\}$
- $L(0^+1^+2^+)=\{0^n1^m2^k|n,m,k\geq 1\}$
- $L(0^*1^*2^*)=\{0^n1^m2^k|n,m,k\geq 0\}$
- $L(1(0+1)^*1+0(0+1)^*0)=\{x|x$的开头与结尾字符相同$\}$

例4-4 为语言L设计正则表达式：$L\{\omega|\omega \in \{0,1\}^*$，且$\omega$的右数第三个字符为1。$\}$

前面的字符 右数第三个 右数第2个 右数第1个

$$(0+1)^*1(0+1)(0+1)$$

例4-5 为语言L设计正则表达式：$L\{\omega|\omega \in \{0,1\}^*$ 且 ω不含有连续的0.}

方法一：将ω看作由多个“1”和“01”组成

1. 设计$(1+01)^*$

   检验空串：可以表示

   检验单独的0：不可以，无法表示以0结尾的串

2. 最终结果：$(1+01)^*(0+\varepsilon)$

例4-5 为语言L设计正则表达式：$L\{\omega|\omega \in \{0,1\}^*$ 且 ω不含有连续的0.}

方法二：将ω看作由多个“0后面连至少一个1”的串组成

1. 设计$(011^*)^*$

   无法表示以0结尾的串

   无法表示以1开头的串

2. 最终结果：$1^*(011^*)^*(0+\varepsilon)$

每个正则语言对应至少一个正则表达式，但一个正则表达式只对应一个正则语言

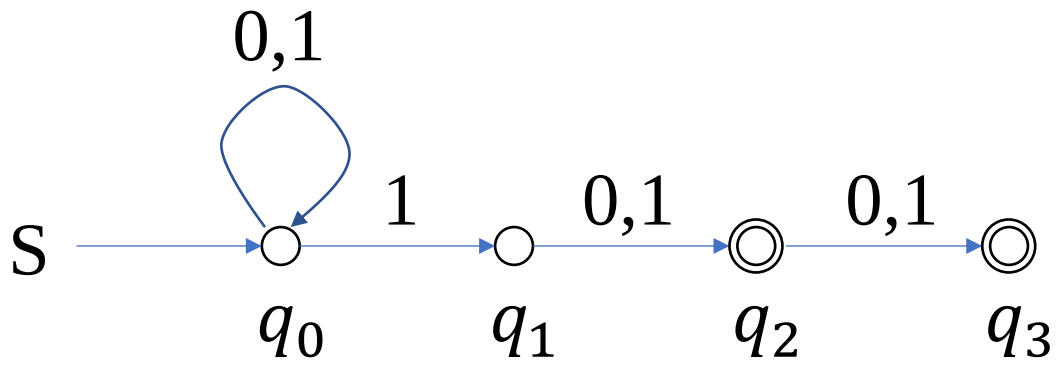
例4-6 为语言L设计正则表达式：$L\{\omega|\omega \in \{0,1\}^*$ 且 ω 不含有连续的0和连续的1.$\}$

方法一： $(01)^* + (10)^* + 0(10)^* + 1(01)^*$

方法二： $(\varepsilon + 1)(01)^*(\varepsilon + 0)$

方法三： $(\varepsilon + 0)(10)^*(\varepsilon + 1)$

例4-7 已知FA如下，求其所表示语言的正则表达式。



- $(0+1)^*1(0+1)+(0+1)^*1(0+1)(0+1)$

利用分配律进行化简：

- $(0+1)^*1(\varepsilon+0+1)(0+1)$
- $(0+1)^*1\,(0+1)(\varepsilon+0+1)$

章节目录

4.1 启示

4.2 正则表达式的形式定义

4.3 正则表达式与FA等价

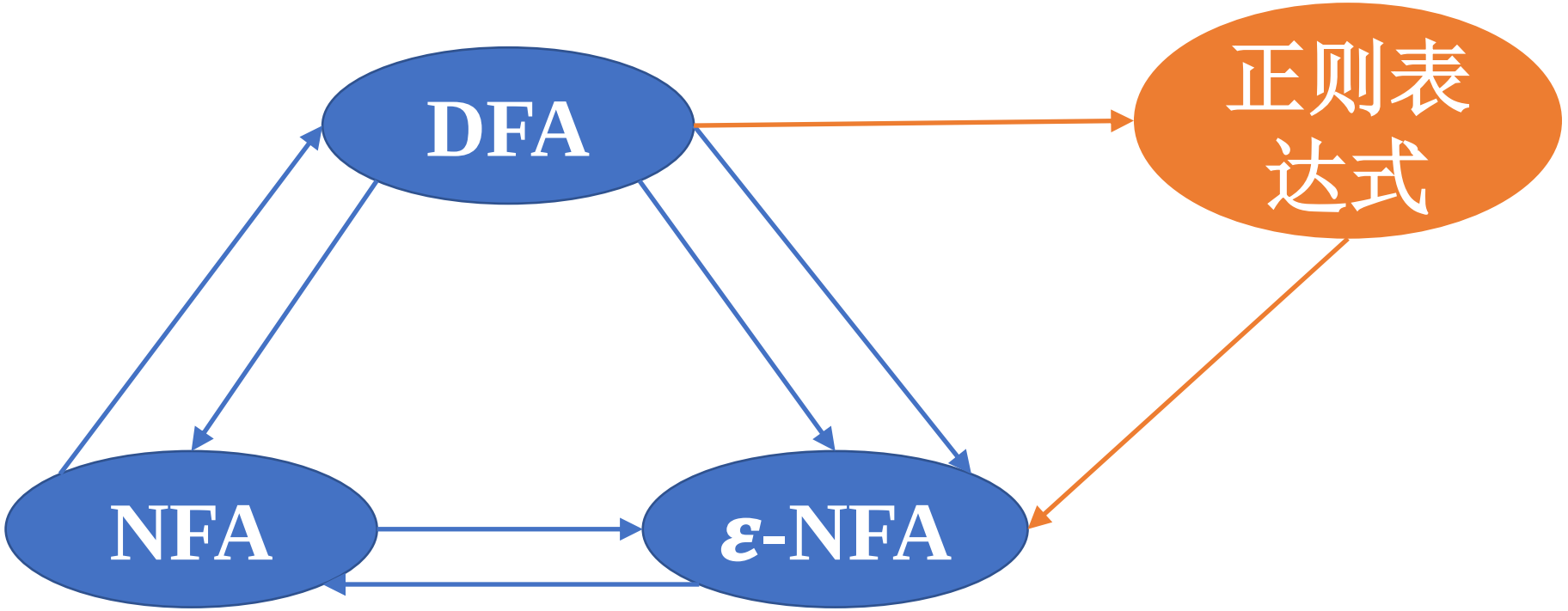
4.4 正则语言等价模型的总结

4.5 本章小结

- 之前讲过，给定一个FA，可以从其开始状态出发，依次计算出所给FA的各个状态q对应的set(q)，并且最终得到相应的FA接受的语言的RE表示。

- 这个计算含有较多的智力因素而难以自动化。本节讨论如何寻找一种比较“机械”的方法，使得计算机能够自动完成FA与RE之间的转换。

- **DFA, NFA, ε-NFA和正则表达式在表示语言的能力上是等价的**



4.3 正则表达式与FA等价

4.3.1 正则表达式到ε-NFA的等价变换

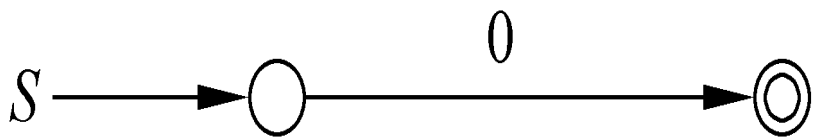
4.3.2 DFA到正则表达式的等价变换

4.3.1 正则表达式到ε-NFA的等价变换

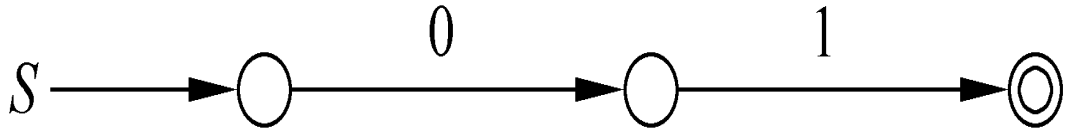
■显然，正则表达式0和01对应的FA如下所示，也不难得0+1和0^*对应的FA，那么如何得到更复杂的正则表达式对应的FA呢？

■按照构造一个给定模型的等价模型的经验，仍然需要从模型的基本定义入手，给出基本的构造方法。

- 0对应的FA

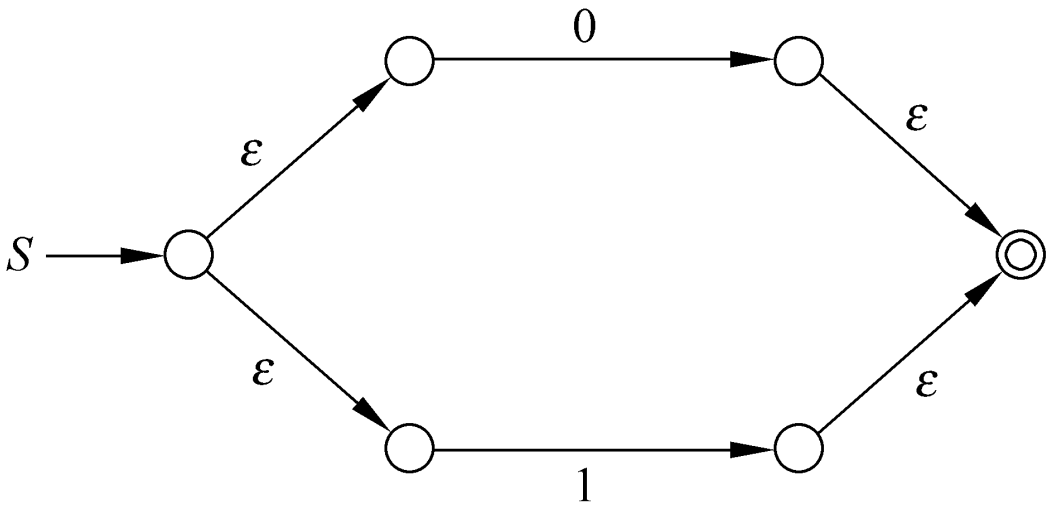


- 01对应的FA

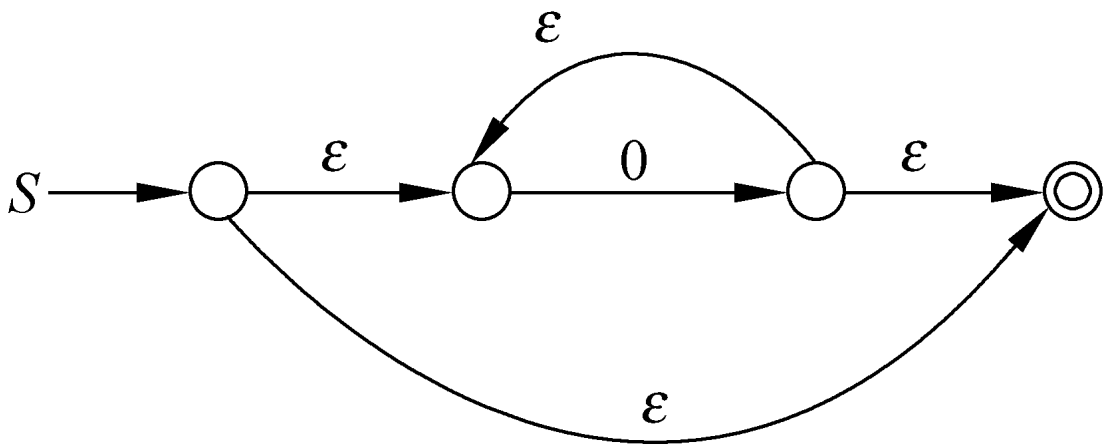


4.3.1 正则表达式到ε-NFA的等价变换

- **0+1对应的FA**



- **0^*对应的FA**



- **ε对应的FA**

- **∅对应的FA**

<u>定义4-4</u> 正则表达式r称为与FA M等价，如果L(r)=L(M)。

定理4-1 正则表达式表示的语言是正则语言。

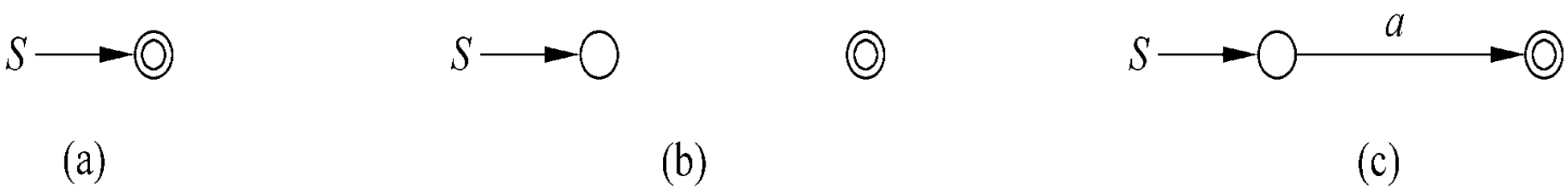
证明：只需证明对任意的正则表达式，可以构造出一个等价的FA。

为了叙述方便，假设正则表达式都是字母表Σ上的正则表达式，并且施归纳于正则表达式中所含的运算符的个数n，证明对于Σ上的任意正则表达式，存在FA M，使得L(M)=L(r)，且M恰有一个终止状态，而且M在终止状态下不做任何移动。

定理4-1 正则表达式表示的语言是正则语言。

证明：当n=0时，有如下三种情况。

r=ε　　r=Φ　　r=a



(a)　　(b)　　(c)

定理4-1 正则表达式表示的语言是正则语言。

证明：假设结论对$n \leq k$成立，则当n=k+1时，有如下三种情况。

（1）$r = r_1 + r_2$；　（2）$r = r_1 r_2$；　（3）$r = r_1^*$

由归纳假设，r_1、r_2中运算符个数不会大于k，则存在满足定理要求的ε-NFA：

- $M_1 = (Q_1, \Sigma, \delta_1, q_{01}, \{f_1\})$
- $M_2 = (Q_2, \Sigma, \delta_2, q_{02}, \{f_2\})$

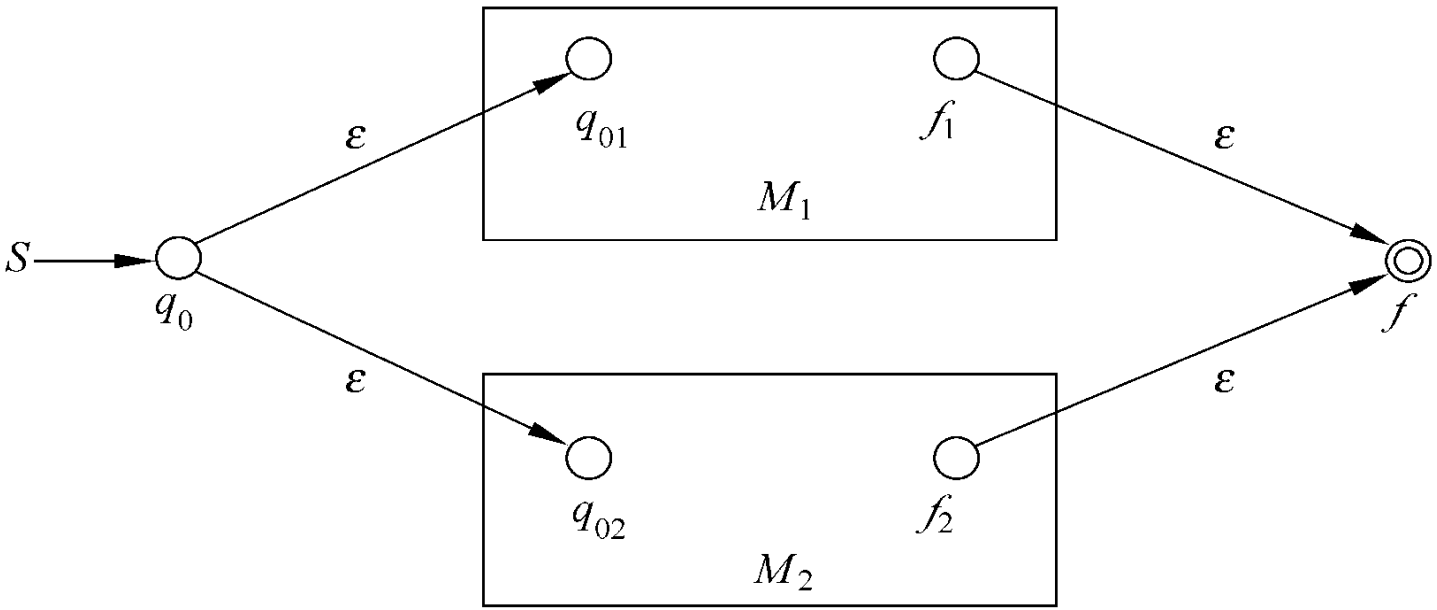
$L(M_1)=L(r_1)$，$L(M_2)=L(r_2)$。

$Q_1 \cap Q_2 = \Phi$。（可对状态重新命名）

定理4-1 正则表达式表示的语言是正则语言。

证明：当n=k+1时，取q_0,f$\notin Q_1 \cup Q_2$，令$M = (Q_1 \cup Q_2 \cup \{q_0, f\}, \Sigma, \delta, q_0, \{f\})$，其中，$\delta$的定义为

- $\delta(q_0, \varepsilon) = \{q_{01}, q_{02}\}$
- $\forall q \in Q_1, a \in \Sigma \cup \{\varepsilon\}, \delta(q,$
- $\forall q \in Q_2, a \in \Sigma \cup \{\varepsilon\}, \delta(q,$
- $\delta(f_1, \varepsilon) = \{f\}$
- $\delta(f_1, \varepsilon) = \{f\}$



定理4-1 正则表达式表示的语言是正则语言。

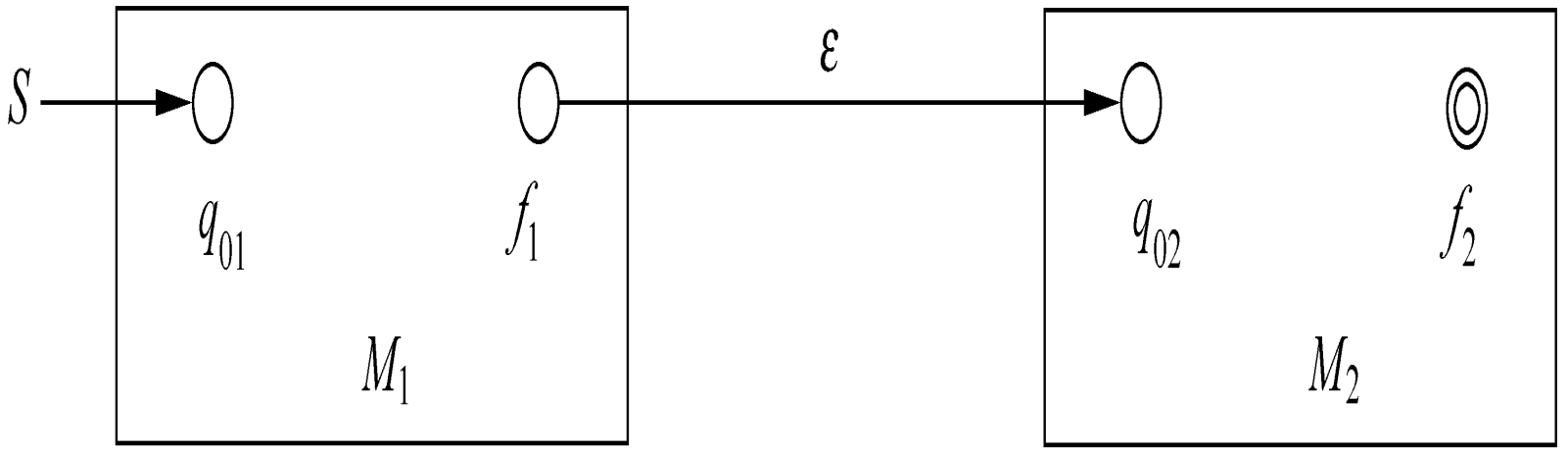
证明：（2）$r = r_1 r_2$；

同样对于上述的M_1和M_2，当n=k+1时，构造$M = (Q_1 \cup Q_2, \Sigma, \delta, q_{01}, \{f_2\})$，其中，$\delta$的定义为

- $\forall q \in Q_1 - \{f_1\}, a \in \Sigma$

  $\delta(q, a) = \delta_1(q, a)$
- $\forall q \in Q_2 - \{f_2\}, a \in \Sigma$

  $\delta(q, a) = \delta_2(q, a)$
- $\delta(f_1, \varepsilon) = \{q_{02}\}$

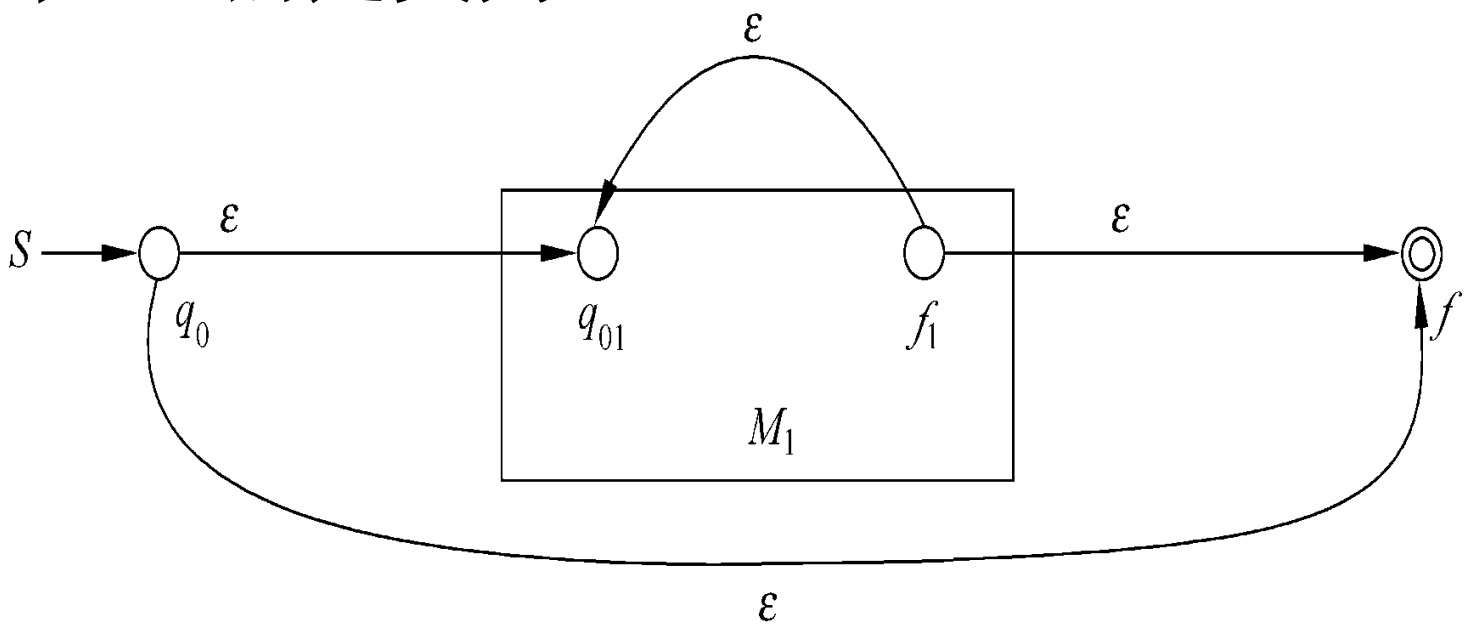


定理4-1 正则表达式表示的语言是正则语言。

证明：（3）$r = r_1^*$；

同样对于上述的M_1和M_2，当n=k+1时，构造$M = (Q_1 \cup \{q_0, f\}, \Sigma, \delta, q_0, \{f\})$，其中，$\delta$的定义为

- $\forall q \in Q_1 - \{f_1\}, a \in \Sigma,$
  $$\delta(q, a) = \delta_1(q, a)$$
- $\delta(f_1, \varepsilon) = \{q_{01}, f\}$
- $\delta(q_0, \varepsilon) = \{q_{01}, f\}$



4.3.1 正则表达式到ε-NFA的等价变换

- 按照上述方法构造与给定RE的等价FA时，该FA有可能含有许多的空移动。
- 可以按照自己对给定RE的“理解”以及对FA的“理解”，“直接地”构造出一个比较“简单”的FA。
- 由于“直接地”构造出的FA的正确性依赖于构造者的“理解”，所以它的正确性缺乏有力的保证。

4.3.1 正则表达式到ε-NFA的等价变换

例4-10 构造与 (0+1)*0+(00)*等价的ε-NFA。

按照定理4-1证明中的方法，该FA的构造可以按如下步骤进行：

① 构造与0等价的ε-NFA M_1。

② 构造与1等价的ε-NFA M_2。

③ 构造与0+1等价的ε-NFA M_3。

④ 构造与$(0+1)^*$等价的ε-NFA M_4。

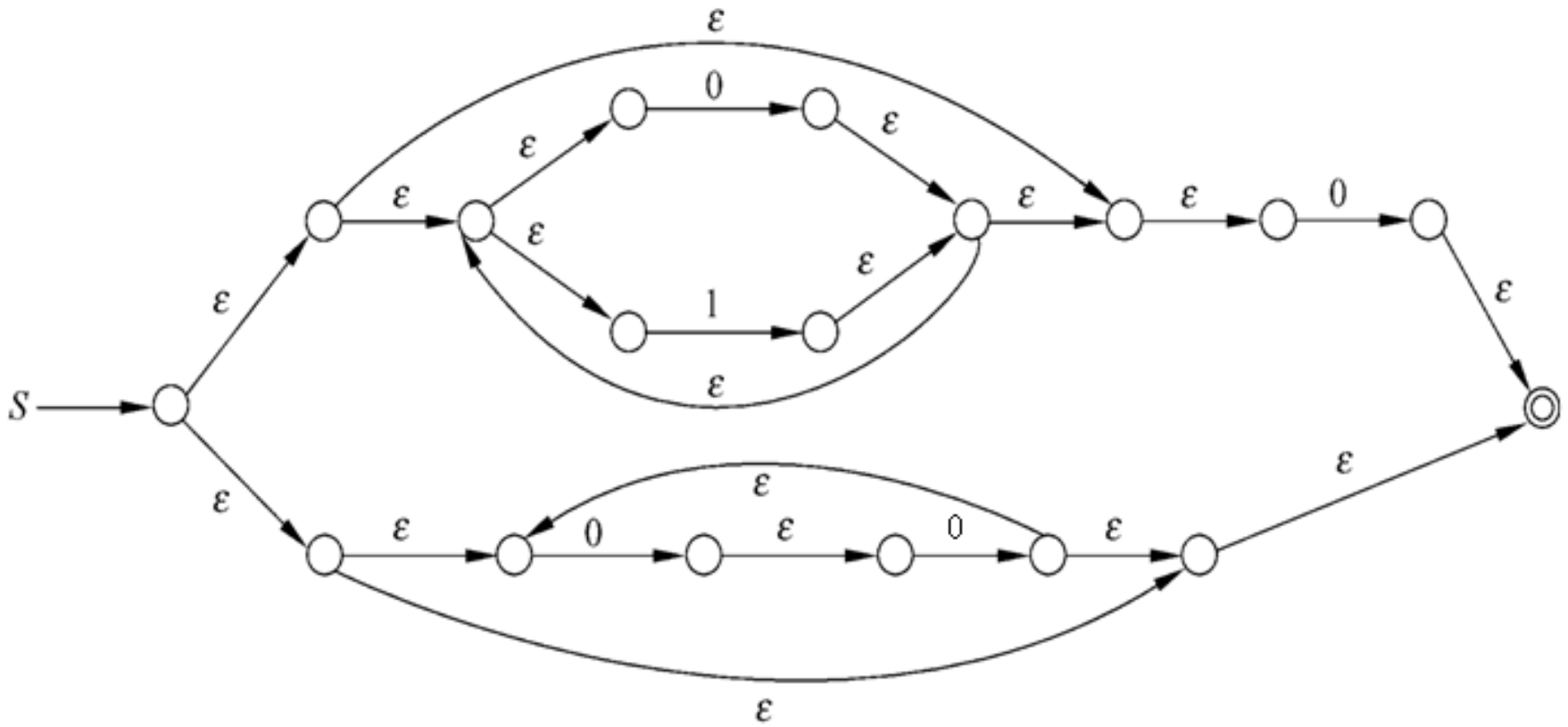
⑤ 构造与$(0+1)^*0$等价的ε-NFA M_5。

⑥ 构造与00等价的ε-NFA M_6。

⑦ 构造与$(00)^*$等价的ε-NFA M_7。

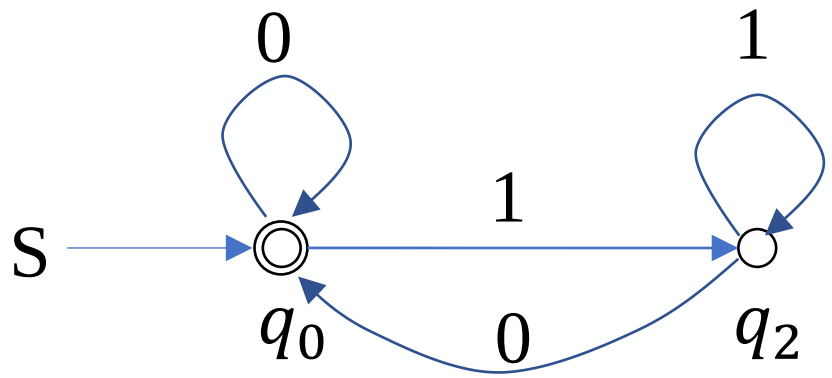
⑧ 构造与$(0+1)^*0+ (00)^*$等价的ε-NFA M_8。

例4-10 构造与 (0+1)*0+(00)*等价的ε-NFA。



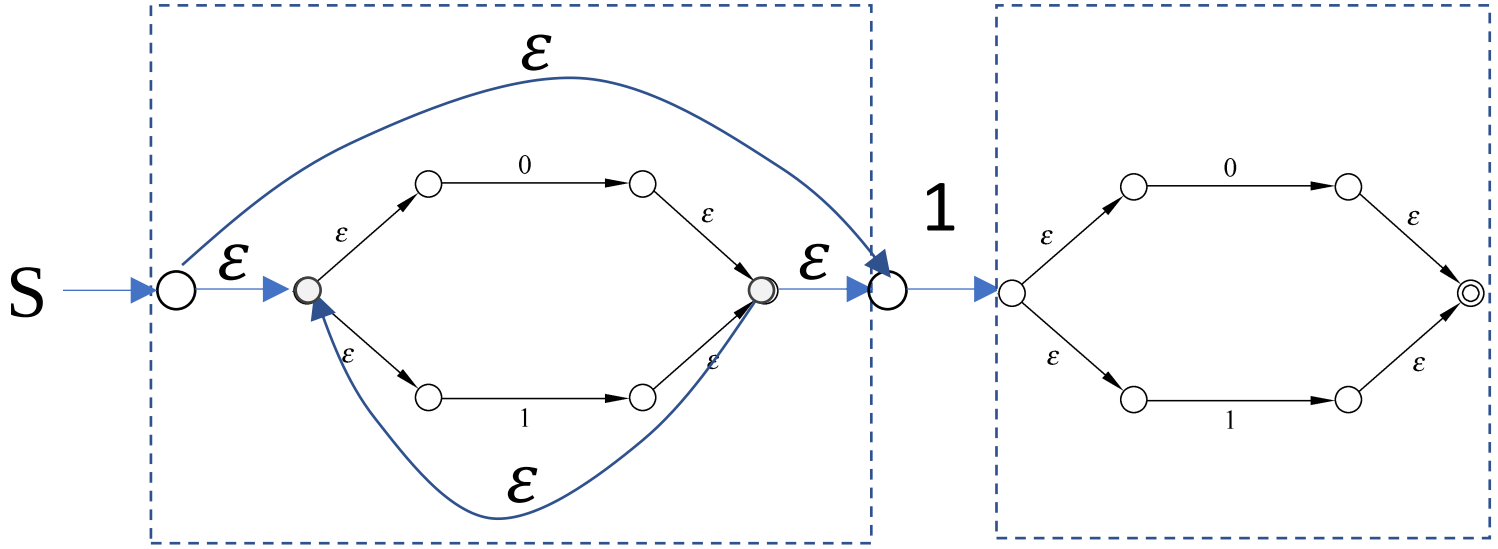
例4-10 构造与 (0+1)*0+(00)*等价的ε-NFA。

按照对 (0+1)*0+(00)*的“理解”，可以“直接地”构造FA，如下。



问题：验证FA的正确性较困难，因此对于比较复杂的正则表达式，多采用比较“机械”的方法来构造。

例4-11 构造与 (0+1)*1(0+1)等价的ε-NFA。



4.3 正则表达式与FA等价

4.3.1 正则表达式到ε-NFA的等价变换

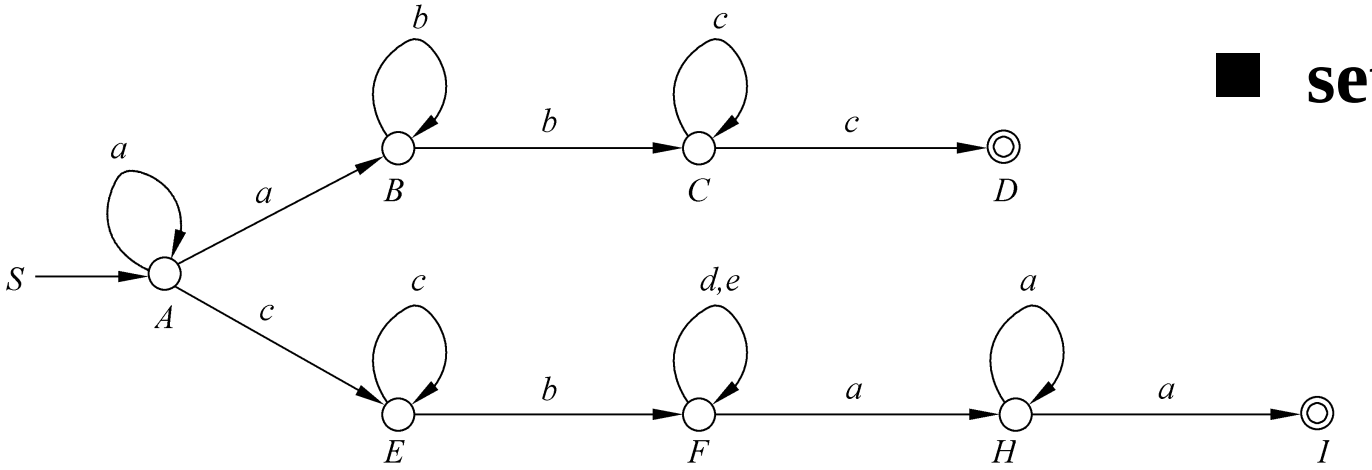
4.3.2 DFA到正则表达式的等价变换

4.3.2 DFA到正则表达式的等价变换

- 正则表达式表示的是正则语言，然而，是不是所有的正则语言都可以用正则表达式表示呢？

- **DFA是正则语言的等价描述模型，而且有了构造与给定正则表达式等价的FA的经验，所以，现在探讨如何从给定的DFA构造等价的正则表达式。**

Callback:



- set(A)={a^n|n≥0}={a}*

  set(A) =a^*

- set(B)= set(A){a}{b^n|n≥0}

  ={a}*{a}{b}*={a}+{b}*

  set(B) =set(A)ab^*=a^*ab^*

- set(C)= set(B){b}{c}*={a}*{a}{b}*{b}{c}*={a}+{b}+{c}*

  set(C)=set(B)bc^* = $a^*ab^*bc^*$

- set(D)= set(C) {c}={a}+{b}+{c}*{c}={a}+{b}+{c}+

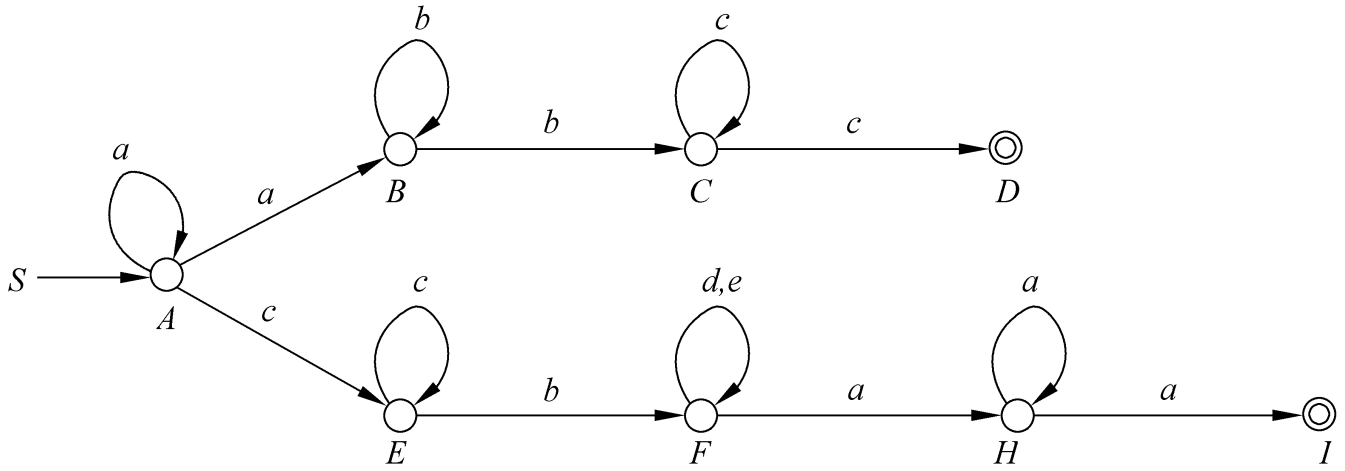
  set(D)=set(C)c= $a^*ab^*bc^*$ c=$aa^*bb^*c^*c$

- set(E)= set(A){c}{c}*={a}*{c}{c}*={a}*{c}+

  set(E)=set(A)cc^* =a^*cc^* set(F)=set(E)b(d+e) *=$a^*cc^*b(d+e)^*$

- set(F)= set(E){b}{d,e}*={a}*{c}+{b}{d,e}*

Callback:



- 问题：这个计算过程难以“机械”地进行，尤其是对较复杂的DFA更是如此。
- 从FA的状态转移图入手，并在转换过程中充分考虑各个状态对应集合之间的关系，对完成这一“等价变换”工作可能是比较有利的。

■set(H)= set(F){a}{a}*={a}*{c}+{d，e}*{a}+

set(H)=set(F)aa*= a*cc*b(d+e)* aa*

■set(I)= set(H){a}={a}*{c}+{d，e}*{a}+{a}

set(I)=set(H)a= a*cc*b(d+e)* aa* a

L(M)= set(D)∪set(H)=aa*bb*cc*+a*cc*(d+e)* aaa*

set(D)+set(I)= aa*bb*cc* +a*cc*b(d+e)* aa* a

4.3.2 DFA到正则表达式的等价变换

设DFA M如下，对状态进行编号

$$M = (\{q_1, q_2, \cdots, q_n\}, \Sigma, \delta, q_1, F)$$

令$R_{ij}^k = \{x | \delta(q_i, x) = q_j$，且对于$x$的任意前缀$y(y \neq x, y \neq \varepsilon)$，如果$\delta(q_i, x) = q_l$，则$l \leq k\}$

也就是说R_{ij}^k是所有将DFA从给定状态q_i引导到状态q_j，并且“途中”不经过下标大于k的状态的所有字符串。值得提醒的是，i和j的值不受小于或等于k的限制。对于$\forall q_i, q_j \in \{q_1, q_2, \cdots, q_n\}$，$R_{ij}^n$是所有可以将DFA从状态$q_i$引导到状态$q_j$的字符串的集合。

4.3.2 DFA到正则表达式的等价变换

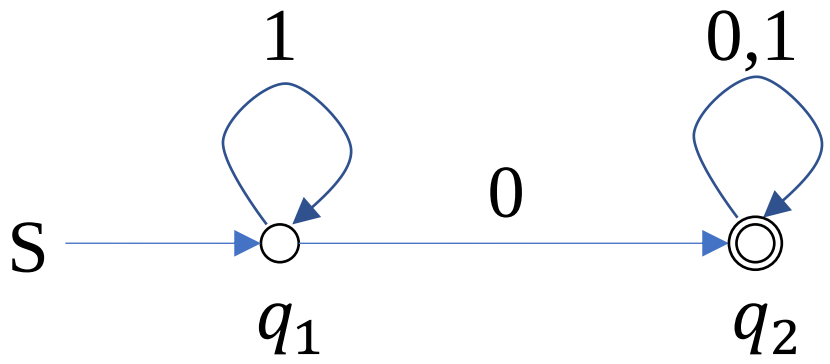
为了方便计算，可以将R_{ij}^{k}递归定义为：

$$R_{ij}^{0}=\begin{cases}\{a|\delta(q_i,a)=q_j\}, & i\neq j\\ \{a|\delta(q_i,a)=q_j\}\cup\{\varepsilon\}, & i=j\end{cases}$$

$$R_{ij}^{k}=R_{ik}^{k-1}\,(R_{kk}^{k-1})^{*}R_{kj}^{k-1}\cup R_{ij}^{k-1}$$

显然，$L(M)=\cup_{q_f\in F}R_{1f}^{n}$

例4-12 将下面DFA转化为正则表达式。



$$R_{ij}^{k} = R_{ij}^{k-1} + R_{ik}^{k-1}\,(R_{kk}^{k-1})^{*} R_{kj}^{k-1}$$

$$R_{ij}^{1} = R_{ij}^{0} + R_{i1}^{0}\,(R_{11}^{0})^{*} R_{1j}^{0}$$

R_{ij}^{k}	k=0
R_{11}^{0}	$\varepsilon+1$
R_{12}^{0}	0
R_{21}^{0}	$\emptyset$
R_{22}^{0}	$\varepsilon+0+1$

R_{ij}^{k}	k=1
R_{11}^{1}	$(\varepsilon+1)+(\varepsilon+1)(\varepsilon+1)^{*}(\varepsilon+1)$
R_{12}^{1}	$0+(\varepsilon+1)(\varepsilon+1)^{*}0$
R_{21}^{1}	$\emptyset+\emptyset(\varepsilon+1)^{*}(\varepsilon+1)$
R_{22}^{1}	$\varepsilon+0+1+\emptyset(\varepsilon+1)^{*}0$

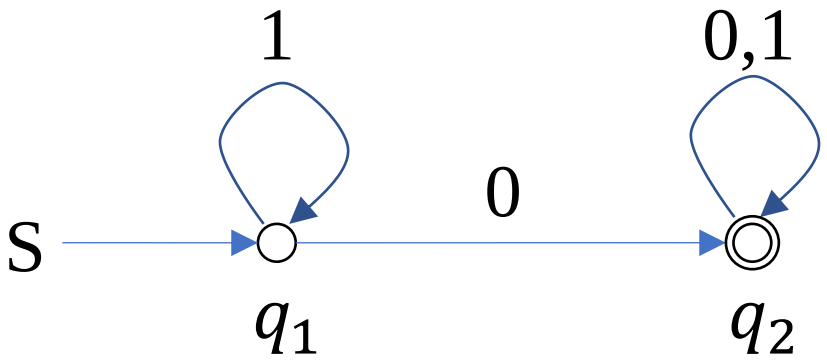
$R_{11}^{0}+R_{11}^{0}(R_{11}^{0})^{*}R_{11}^{0}$ 1^{*}

$R_{12}^{0}+R_{11}^{0}(R_{11}^{0})^{*}R_{12}^{0}$ $1^{*}0$

$R_{21}^{0}+R_{21}^{0}(R_{11}^{0})^{*}R_{11}^{0}$ $\emptyset$

$R_{22}^{0}+R_{21}^{0}(R_{11}^{0})^{*}R_{12}^{0}$ $\varepsilon+0+1$

例4-12 将下面DFA转化为正则表达式。



$$R_{ij}^{k} = R_{ij}^{k-1} + R_{ik}^{k-1}\,(R_{kk}^{k-1})^{*} R_{kj}^{k-1}$$

$$R_{ij}^{2} = R_{ij}^{1} + R_{i2}^{1}\,(R_{22}^{1})^{*} R_{2j}^{1}$$

$R_{ij}^{(k)}$	k=2		
$R_{11}^{(2)}$	$1^{*} + 1^{*}0(\varepsilon + 0 + 1)^{*}\emptyset$	$R_{11}^{1} + R_{12}^{1}(R_{22}^{1})^{*}R_{21}^{1}$	1^{*}
$R_{12}^{(2)}$	$1^{*}0 + 1^{*}0(\varepsilon + 0 + 1)^{*}(\varepsilon + 0 + 1)$	$R_{12}^{1} + R_{12}^{1}(R_{22}^{1})^{*}R_{22}^{1}$	$1^{*}0(0 + 1)^{*}$
$R_{21}^{(2)}$	$\emptyset + (\varepsilon + 0 + 1)(\varepsilon + 0 + 1)^{*}\emptyset$	$R_{21}^{1} + R_{22}^{1}(R_{22}^{1})^{*}R_{21}^{1}$	$\emptyset$
$R_{22}^{(2)}$	$\varepsilon + 0 + 1 + (\varepsilon + 0 + 1)(\varepsilon + 0 + 1)^{*}(\varepsilon + 0 + 1)$	$R_{22}^{1} + R_{22}^{1}(R_{22}^{1})^{*}R_{22}^{1}$	$(0 + 1)^{*}$

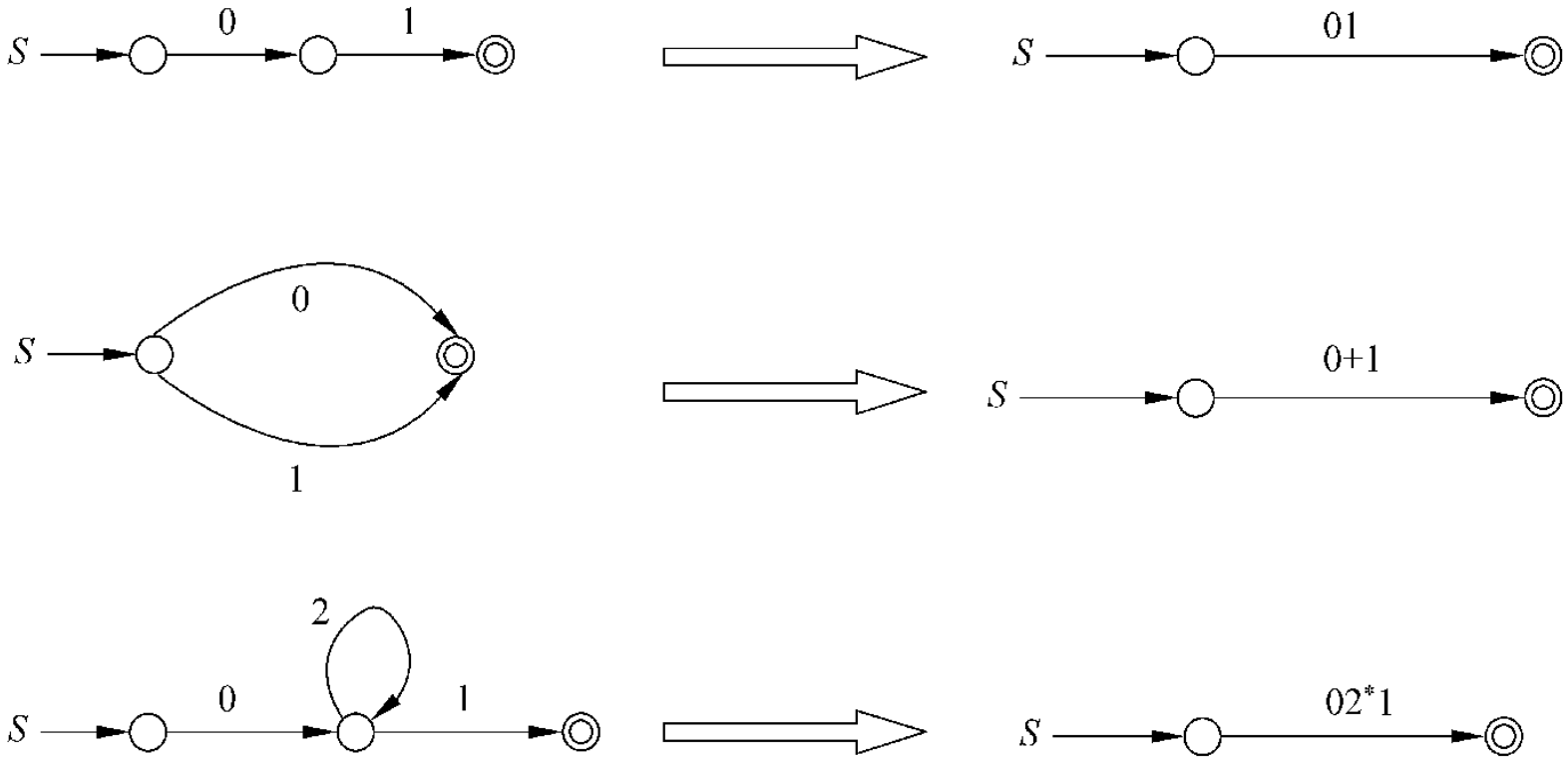
q_1开始状态， q_2接收状态，该DFA的正则表达式为R_{12}^{2}：$1^{*}0(0 + 1)^{*}$

4.3.2 DFA到正则表达式的等价变换

定理4-2 正则语言可以用正则表达式表示。

- 上面所给的方法对计算机系统来说比较方便，但是“手工”来计算比较繁琐。
- 下面介绍一种称为“图上作业法”的方法，通过对DFA的状态转移图进行处理，来获取它相应的正则表达式。

图上作业法



图上作业法操作步骤

（1）预处理：

① 在状态转移图中增加标记为X和Y的状态，从标记为X的状态到标记为q_0的状态引一条标记为ε的弧；从标记为q(q∈F)的状态到标记为Y的状态分别引一条标记为ε的弧。

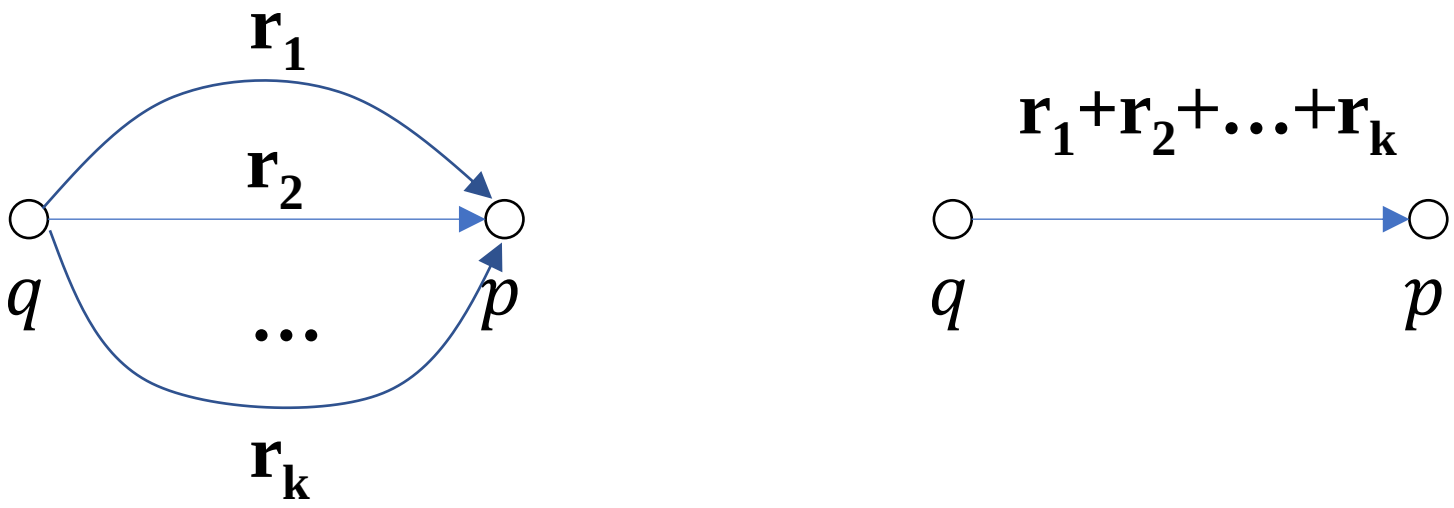
② 去掉所有的不可达状态。

（2）对通过步骤(1)处理所得到的状态转移图重复进行并弧和去状态等操作，直到该图中不再包含除了标记为X和Y外的其他状态，并且这两个状态之间最多只有一条弧。

图上作业法操作步骤

① 并弧

- 将从q到p的标记为r_1，r_2，…，r_k并行弧用从q到p的、标记为$r_1+r_2+\ldots+r_k$的弧取代这k个并行弧。



图上作业法操作步骤

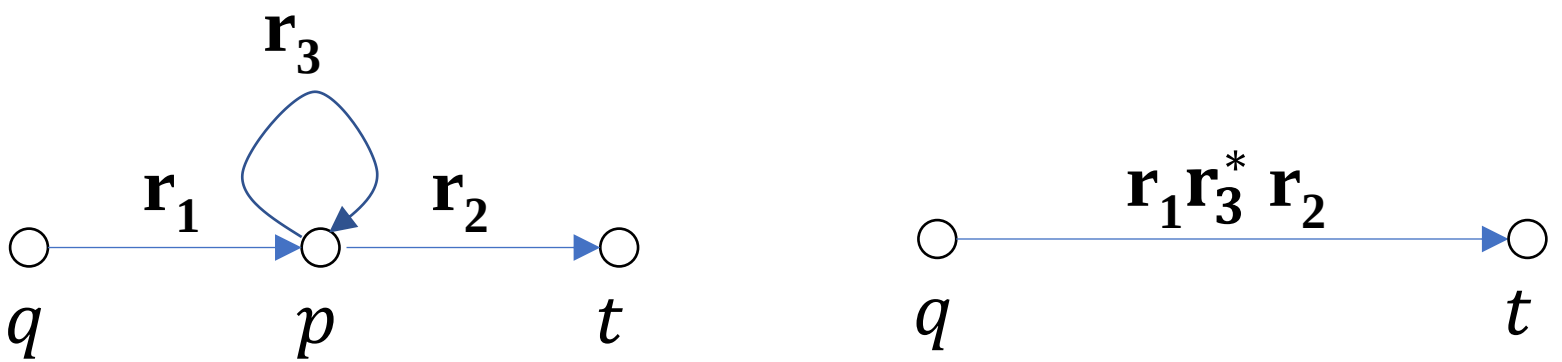
② 去状态1

- 如果从q到p有一条标记为r_1的弧，从p到t有一条标记为r_2的弧，不存在从状态p到状态p的弧，将状态p和与之关联的这两条弧去掉，用一条从q到t的标记为r_1r_2的弧代替。

图上作业法操作步骤

③ 去状态2

- 如果从q到p有一条标记为r_1的弧，从p到t有一条标记为r_2的弧，从状态p到状态p标记为r_3的弧，将状态p和与之关联的这三条弧去掉，用一条从q到t的标记为$r_1 r_3^* r_2$的弧代替。



图上作业法操作步骤

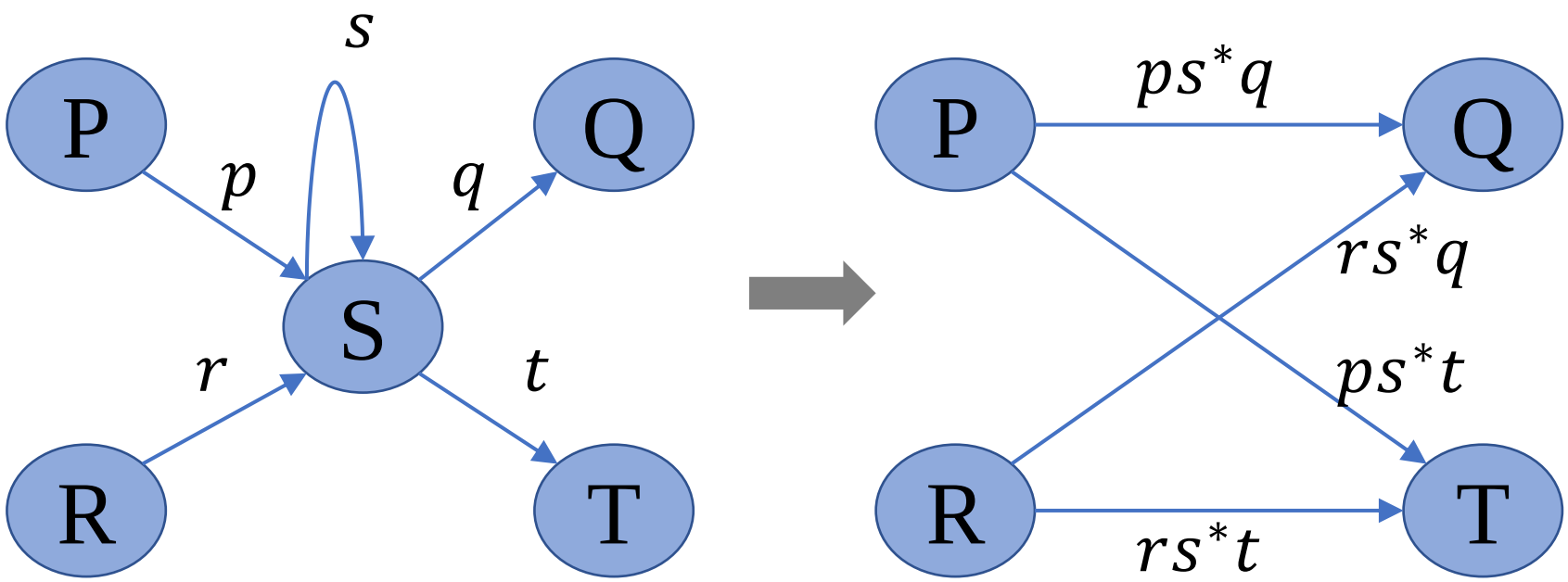
④ 去状态3

- 如果图中只有三个状态，而且不存在从标记为X的状态到达标记为Y的状态的路，则将除标记为X的状态和标记为Y的状态之外的第3个状态及其相关的弧全部删除

（3） 从标记为X的状态到标记为Y的状态的弧的标记为所求的正则表达式。如果此弧不存在，则所求的正则表达式为 Φ。

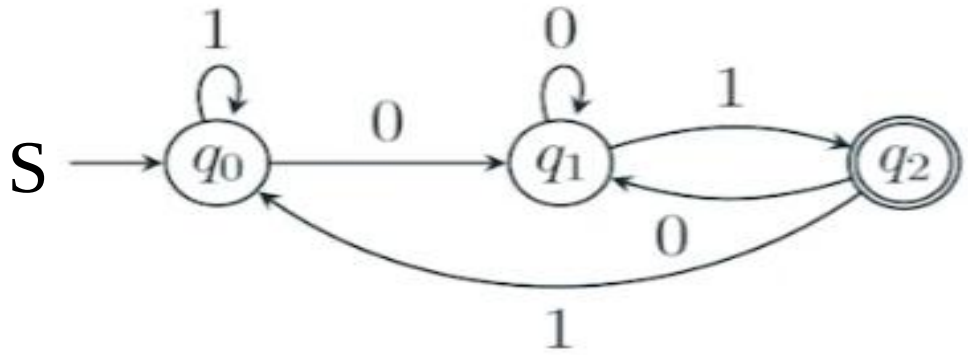
例4-13 利用图上作业法，将下图中状态S去掉。

若要删除状态S，需添加相应路径。

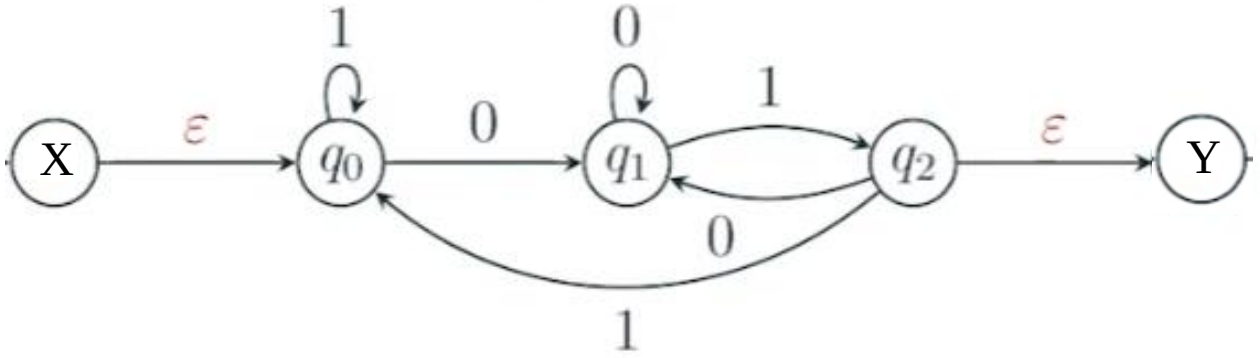


4.3.2 DFA到正则表达式的等价变换

例4-14 利用图上作业法，构造下图自动机的正则表达式。

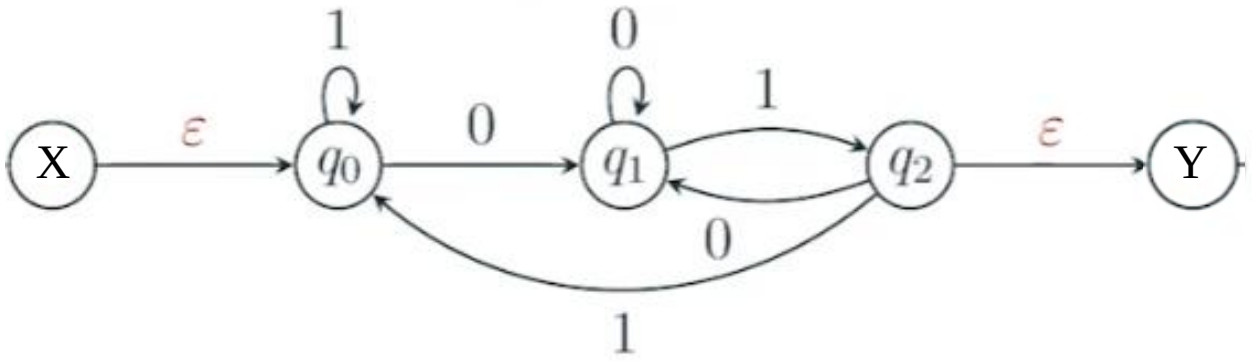


1. 利用空转移，添加新的开始状态X和结束状态Y。

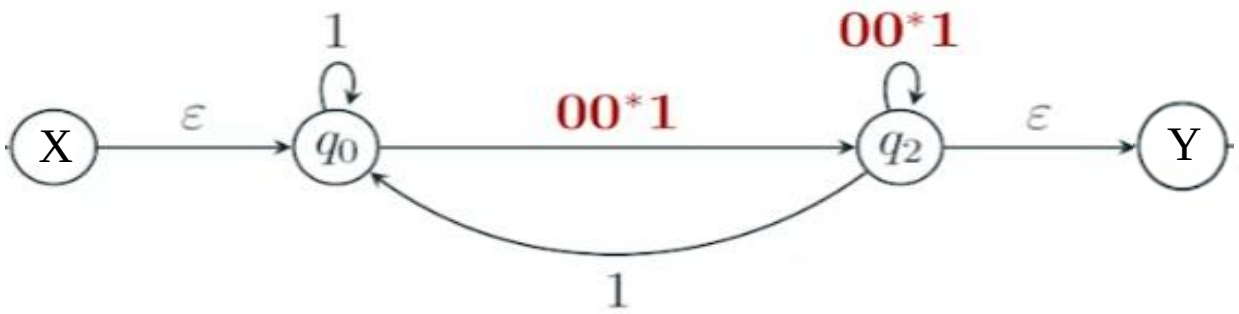


4.3.2 DFA到正则表达式的等价变换

1. 利用空转移，添加新的开始状态X和结束状态Y。

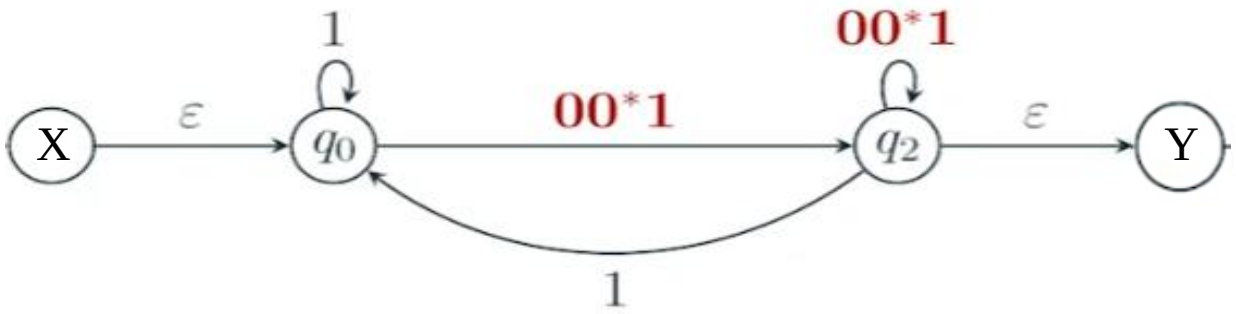


2. 消除状态q_1，添加路径$q_0 \to q_2$和$q_2 \to q_2$：

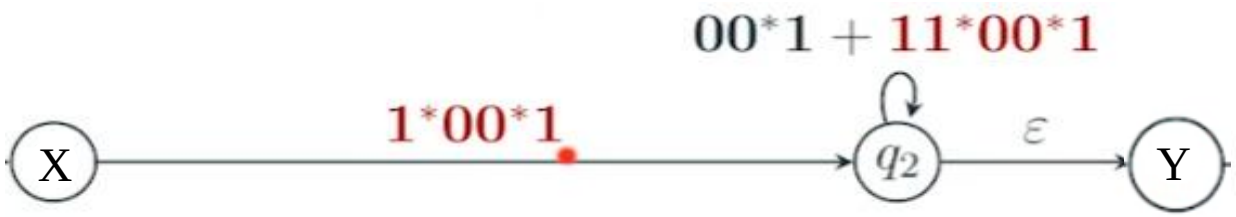


4.3.2 DFA到正则表达式的等价变换

2. 消除状态q_1，添加路径$q_0 \to q_2$和$q_2 \to q_2$：

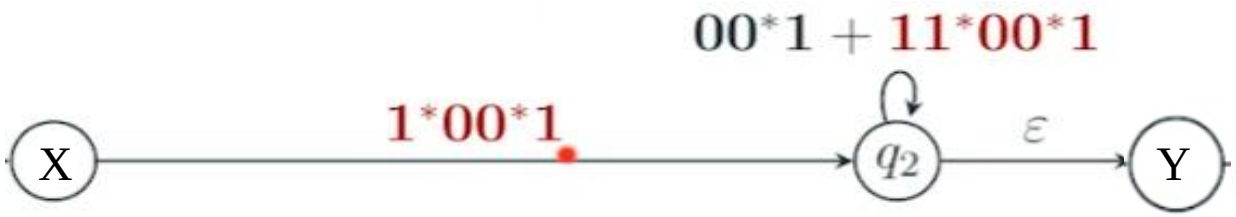


3. 消除状态q_0，添加路径$X \to q_2$和$q_2 \to q_2$：

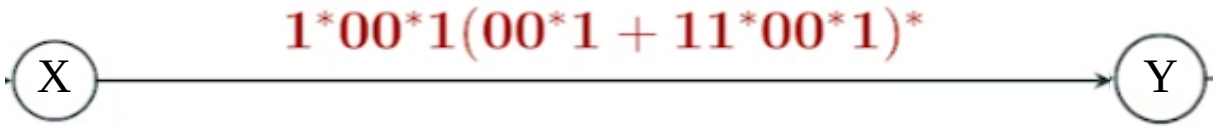


4.3.2 DFA到正则表达式的等价变换

3. 消除状态q_0，添加路径$X \to q_2$和$q_2 \to q_2$：



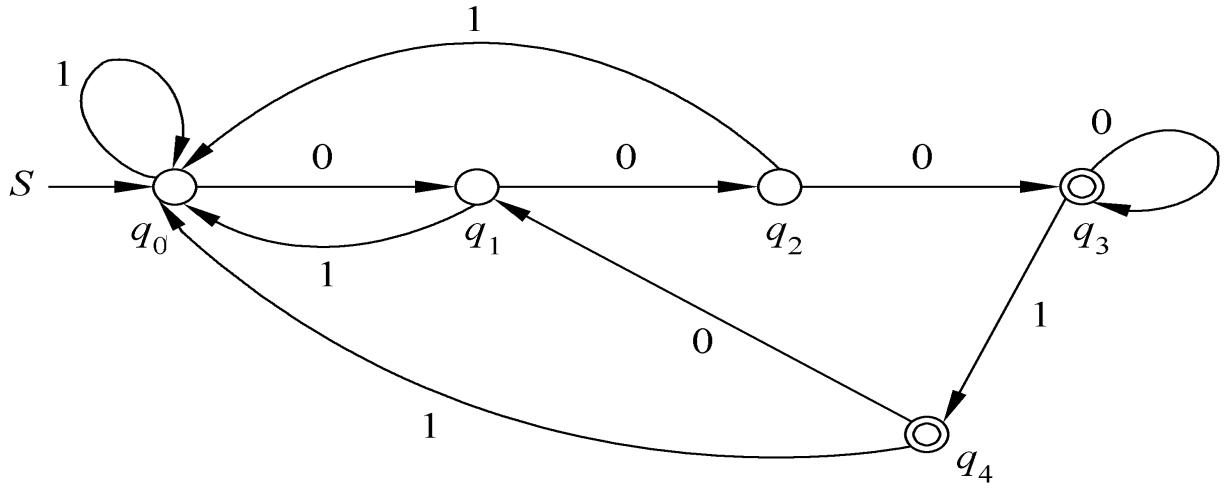
4. 消除状态q_2，添加路径$X \to Y$：



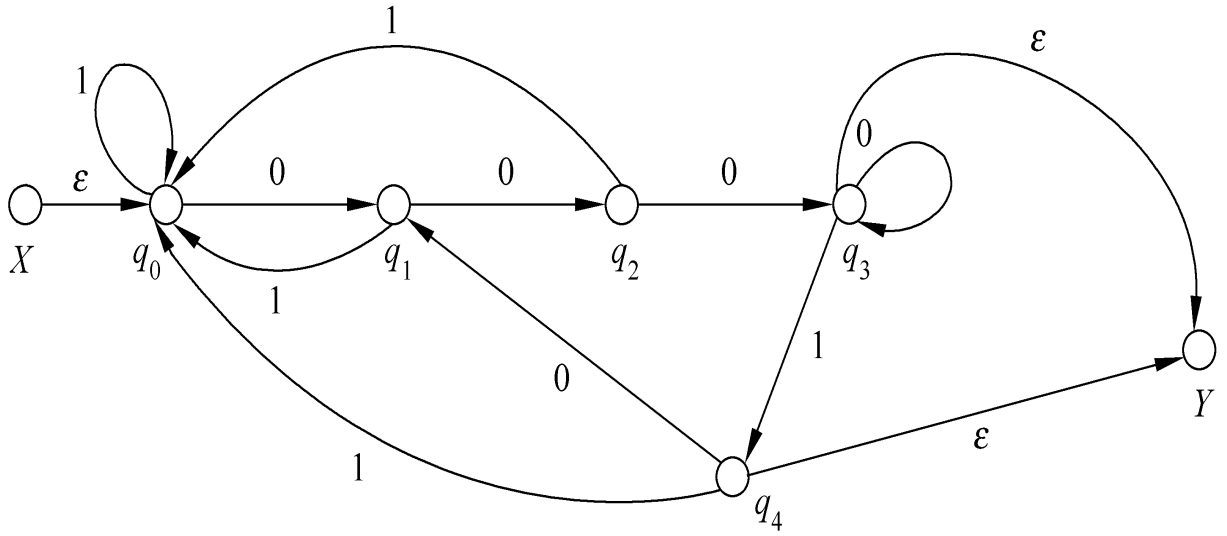
因此，该DFA的正则表达式为：

$$1^*00^*1(00^*1 + 11^*00^*1)^*$$

例4-15 求与下图所示的DFA等价的正则表达式。

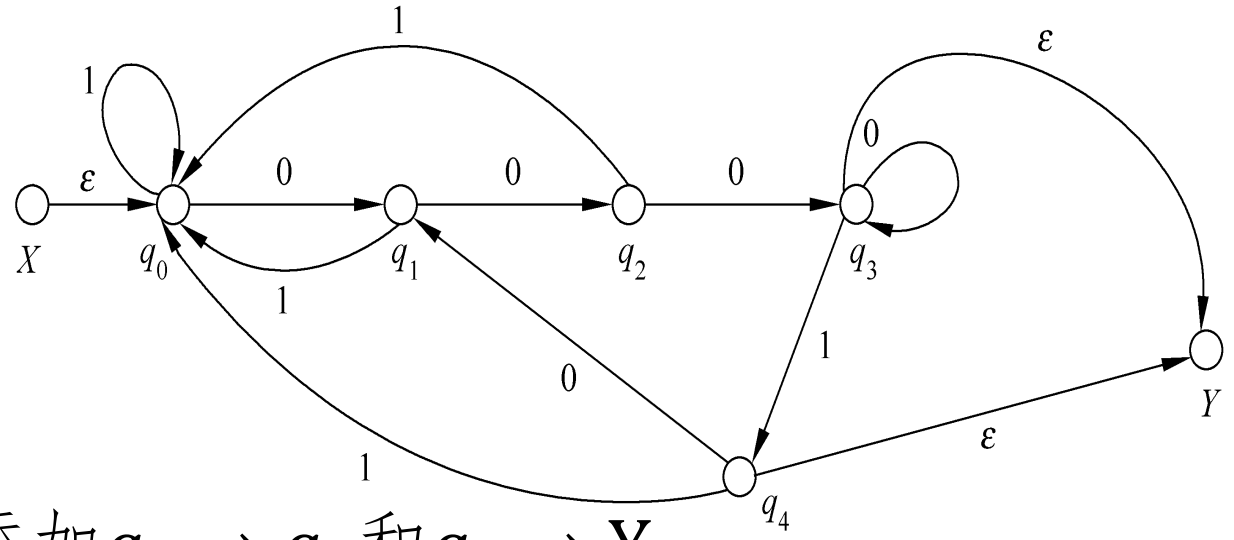


1. 预处理

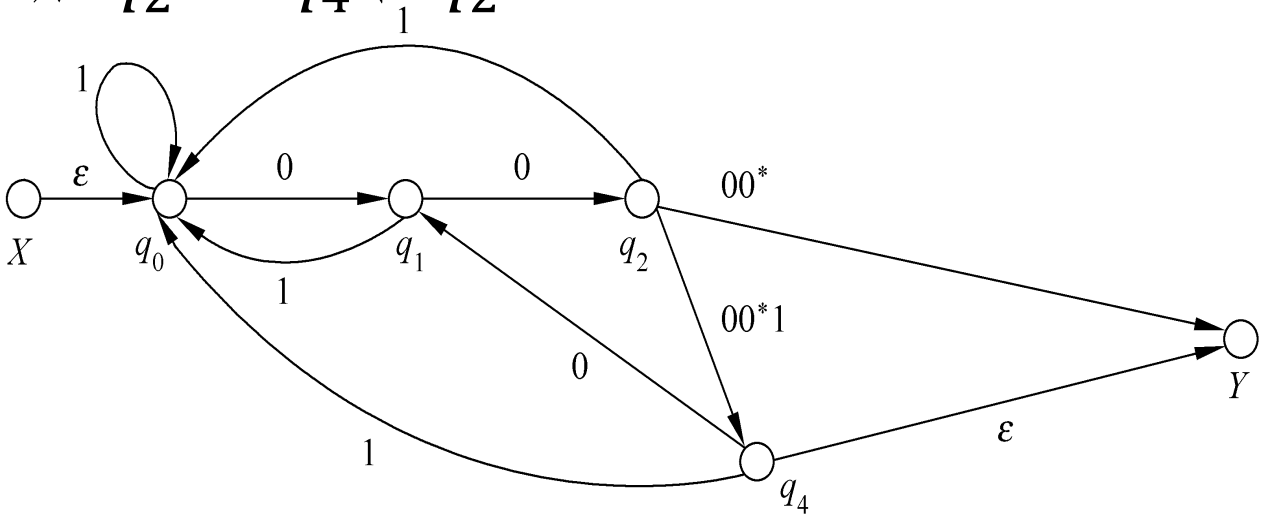


4.3.2 DFA到正则表达式的等价变换

1. 预处理

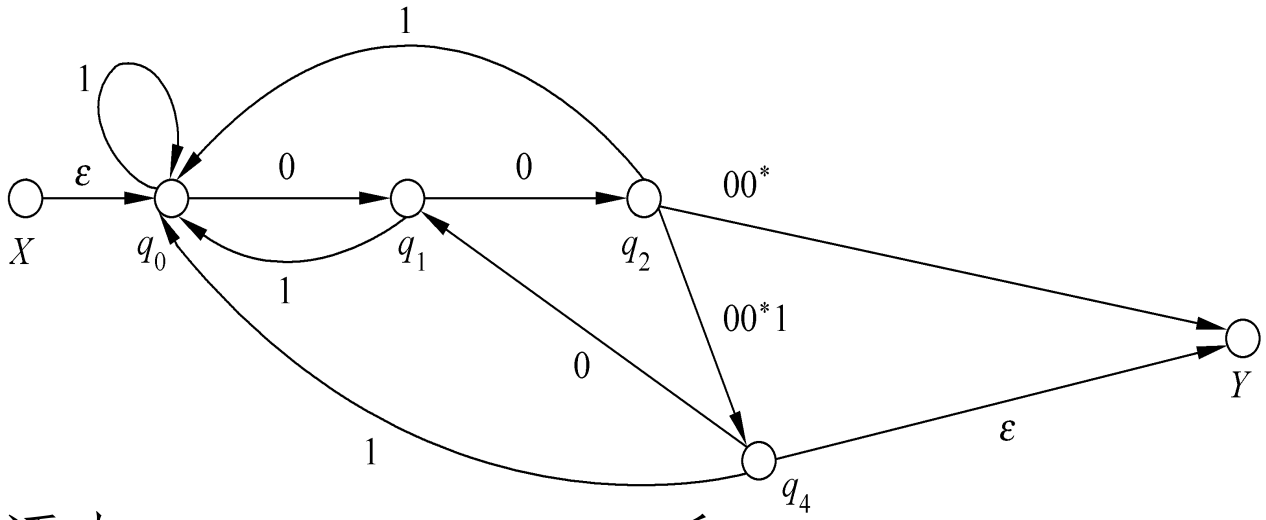


2. 去掉状态q_3，添加$q_2 \to q_4$和$q_2 \to$ Y

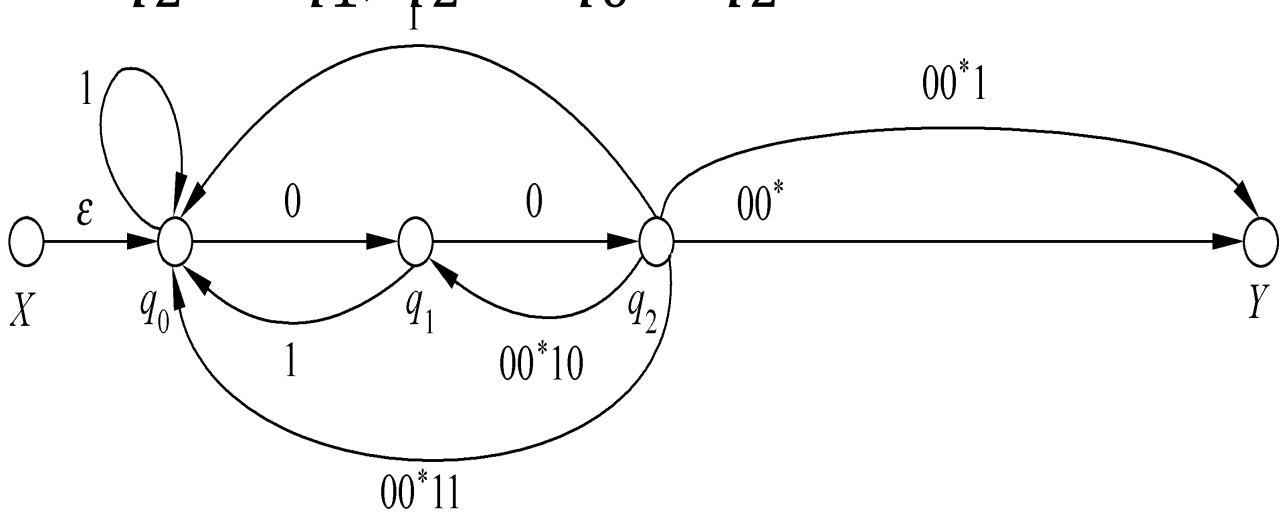


4.3.2 DFA到正则表达式的等价变换

2. 去掉状态q_3，添加$q_2 \to q_4$和$q_2 \to \mathrm{Y}$

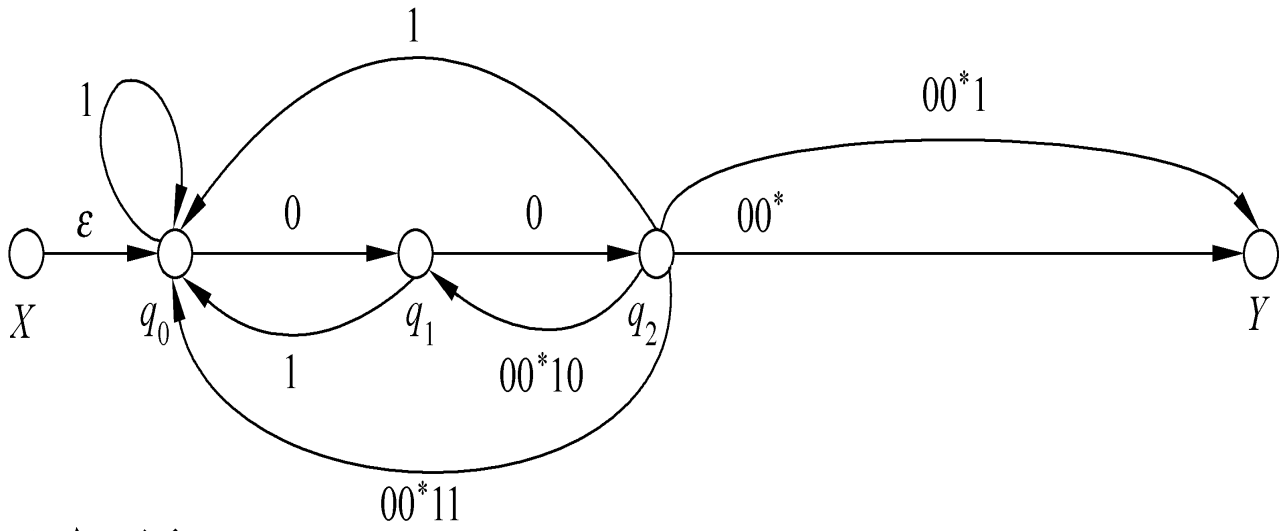


3. 去掉状态q_4，添加$q_2 \to q_1, q_2 \to q_0$和$q_2 \to Y$

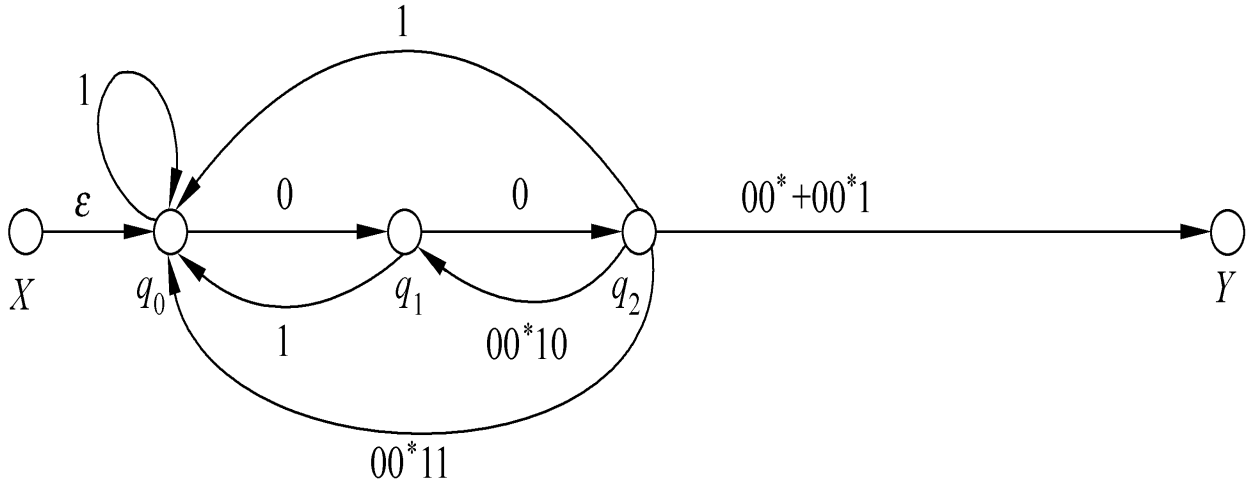


4.3.2 DFA到正则表达式的等价变换

3. 去掉状态q_4，添加$q_2 \to q_1, q_2 \to q_0$和$q_2 \to Y$

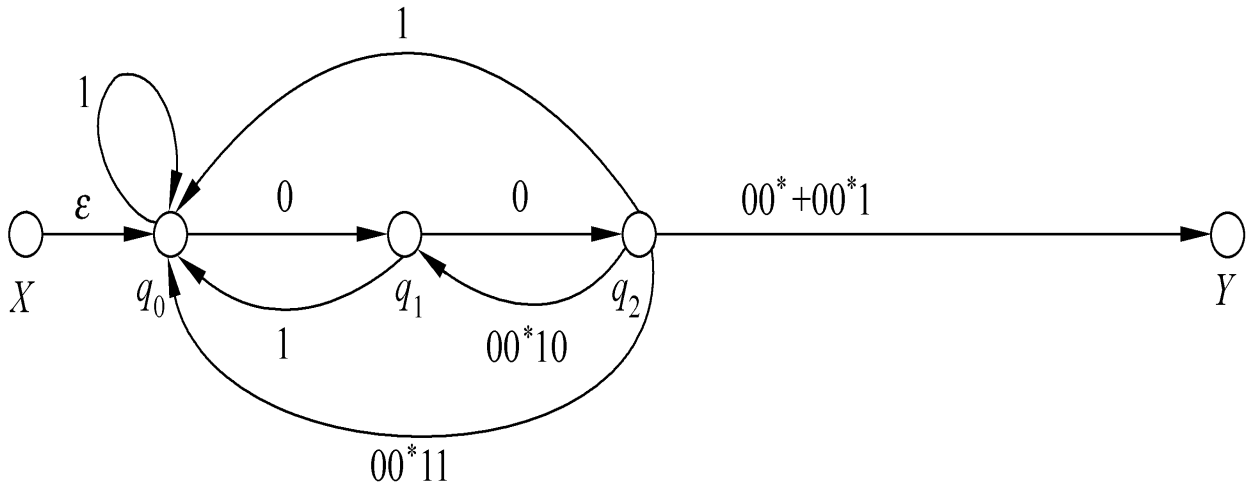


4. 合并q_2到Y的两条弧

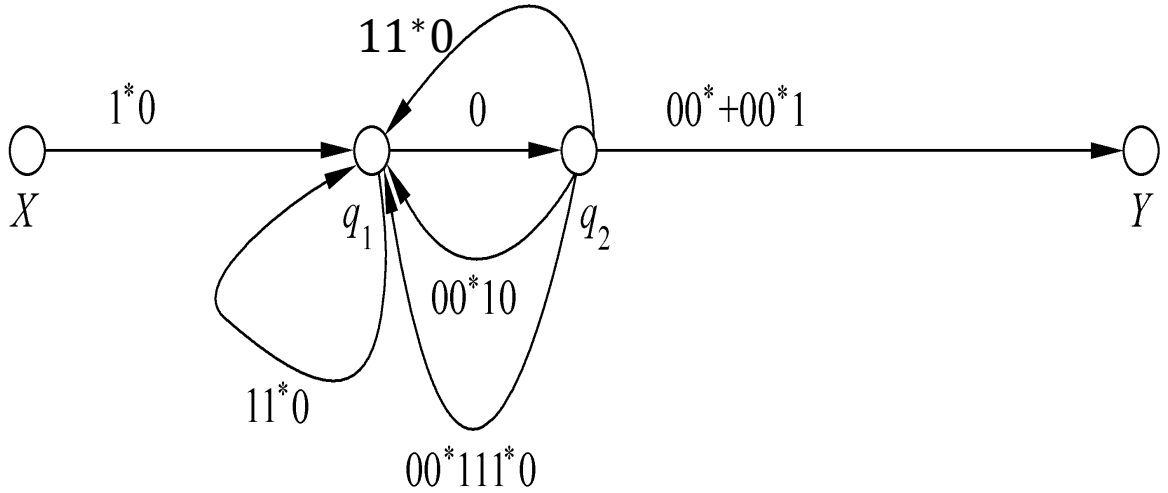


4.3.2 DFA到正则表达式的等价变换

4. 合并q_2到Y的两条弧

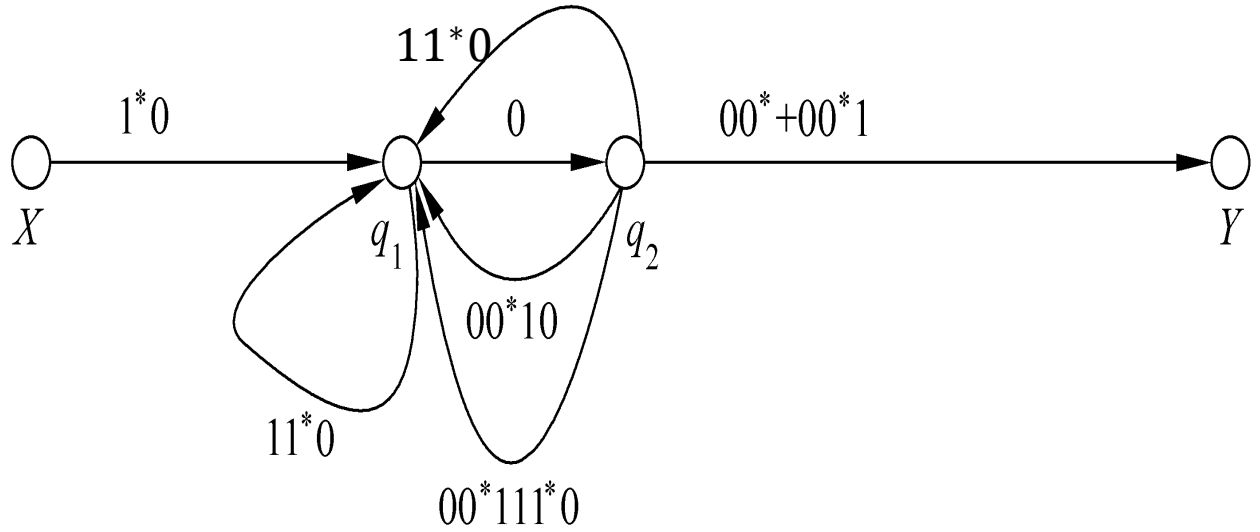


5. 去掉状态q_0，添加$X \to q_1, q_1 \to q_1, q_2 \to q_1$和$q_2 \to q_1$

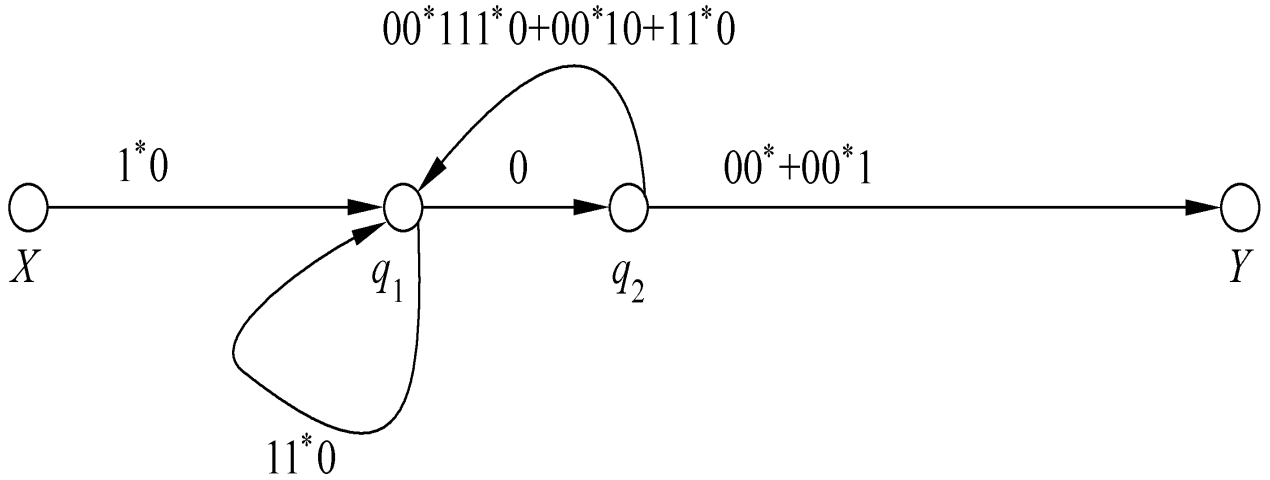


4.3.2 DFA到正则表达式的等价变换

5. 去掉状态q_0，添加$X \to q_1, q_1 \to q_1, q_2 \to q_1$和$q_2 \to q_1$

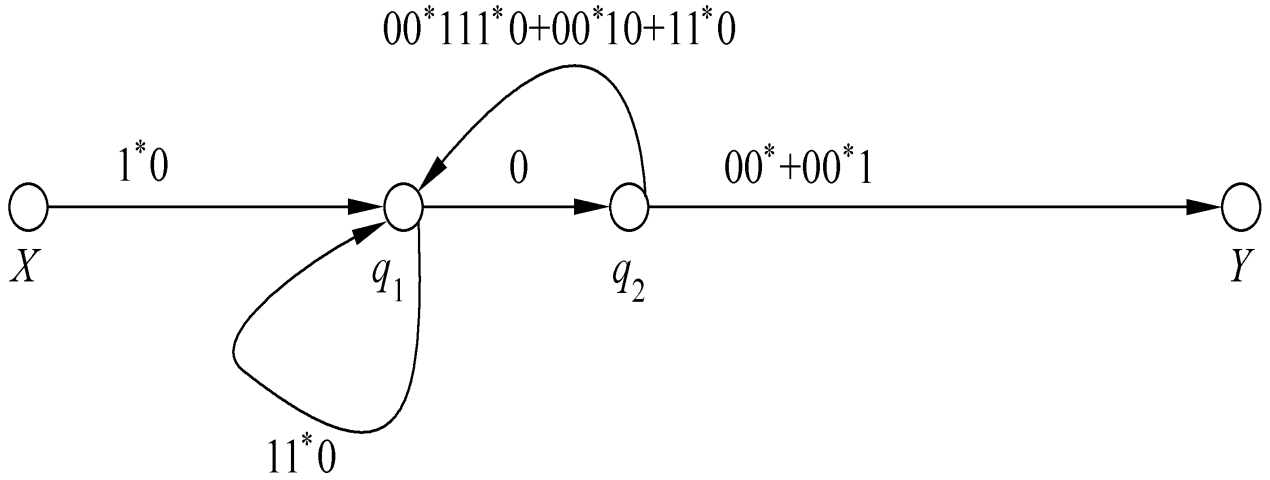


6. 并弧$q_2 \to q_1$

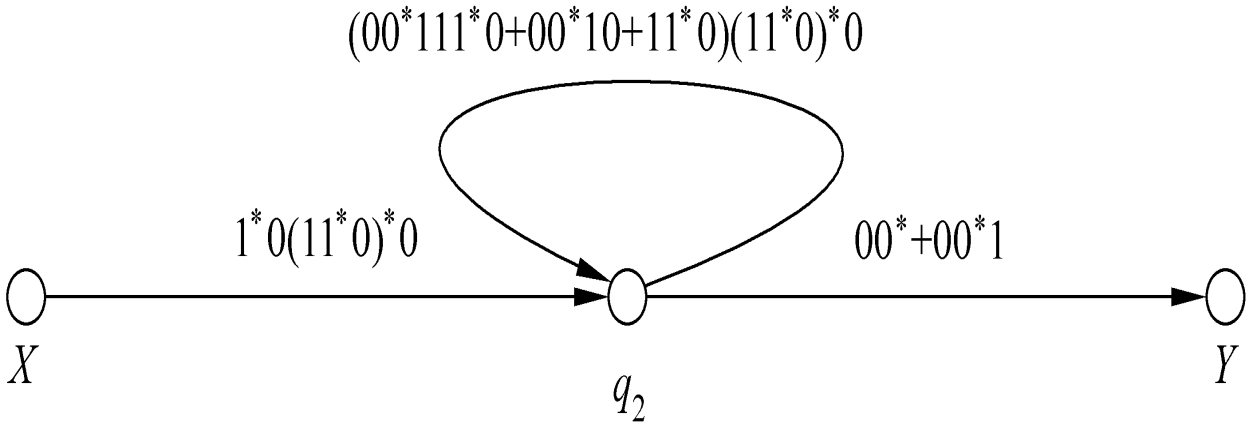


4.3.2 DFA到正则表达式的等价变换

6. 并弧$q_2 \to q_1$

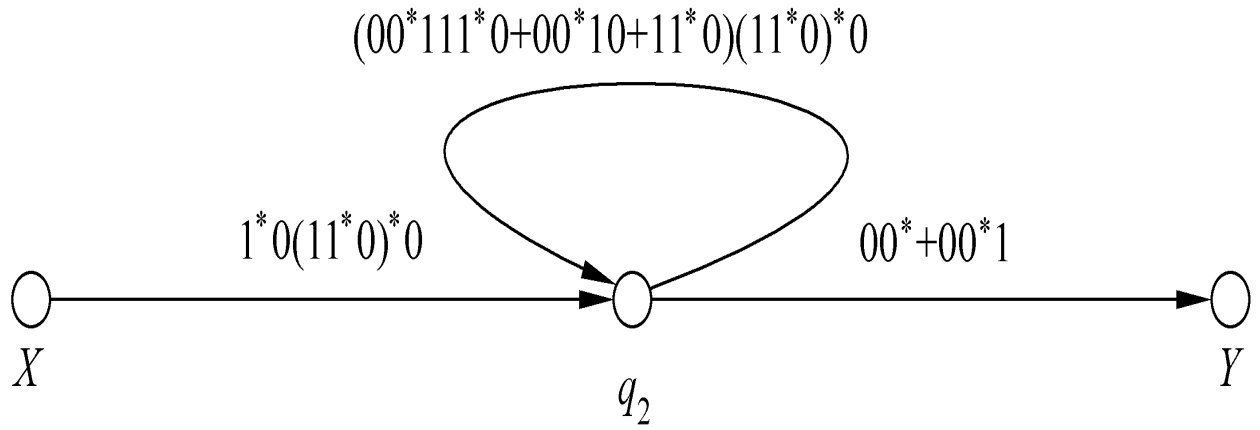


7. 去掉状态q_1，添加X $\to q_2$和$q_2 \to q_2$

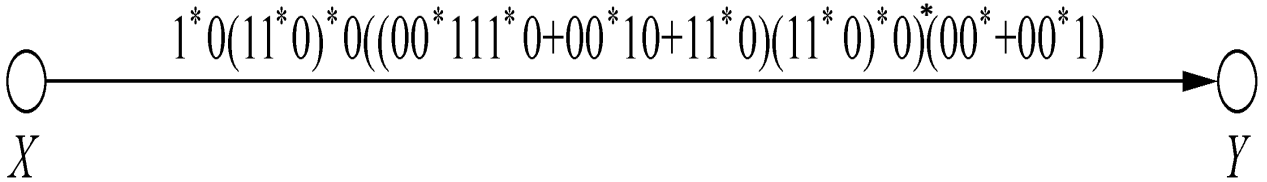


4.3.2 DFA到正则表达式的等价变换

7. 去掉状态q_1，添加$X \to q_2$和$q_2 \to q_2$



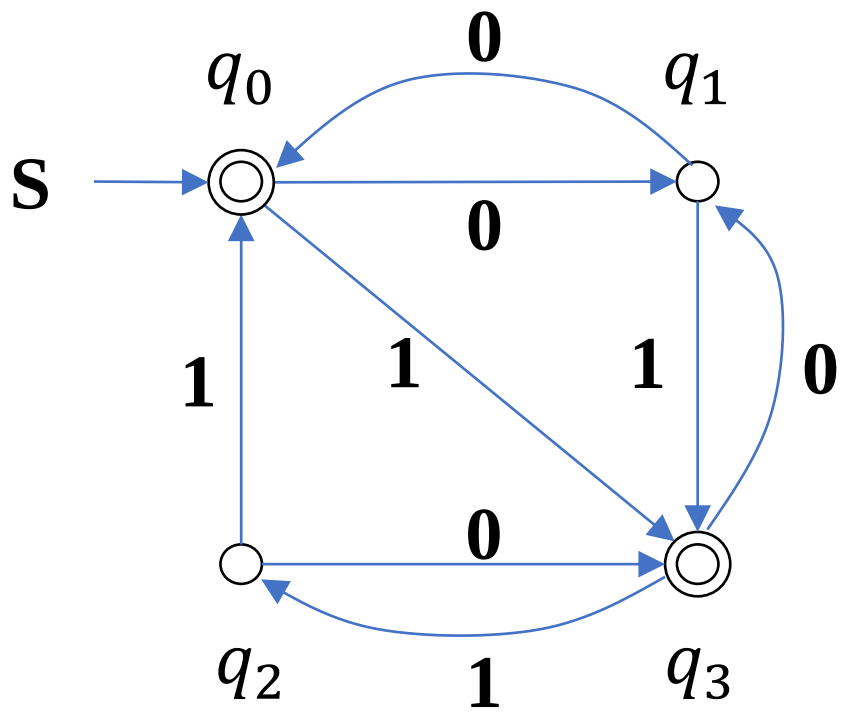
8. 去掉状态q_2，添加$X \to Y$



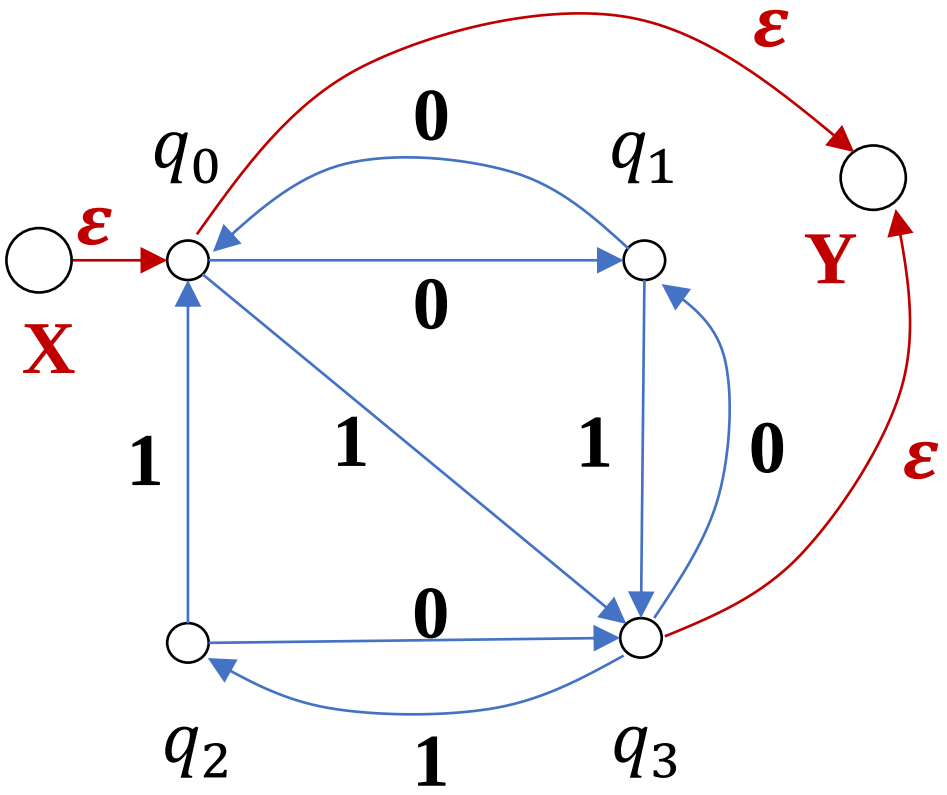
结果为：

$1^*0(11^*0)^*0((00^*111^*0+00^*10+11^*0)(11^*0)^*0)^*(00^*+00^*1)$

例4-16 求与下图所示的DFA等价的正则表达式，去状态顺序为q_2, q_1, q_3, q_0。

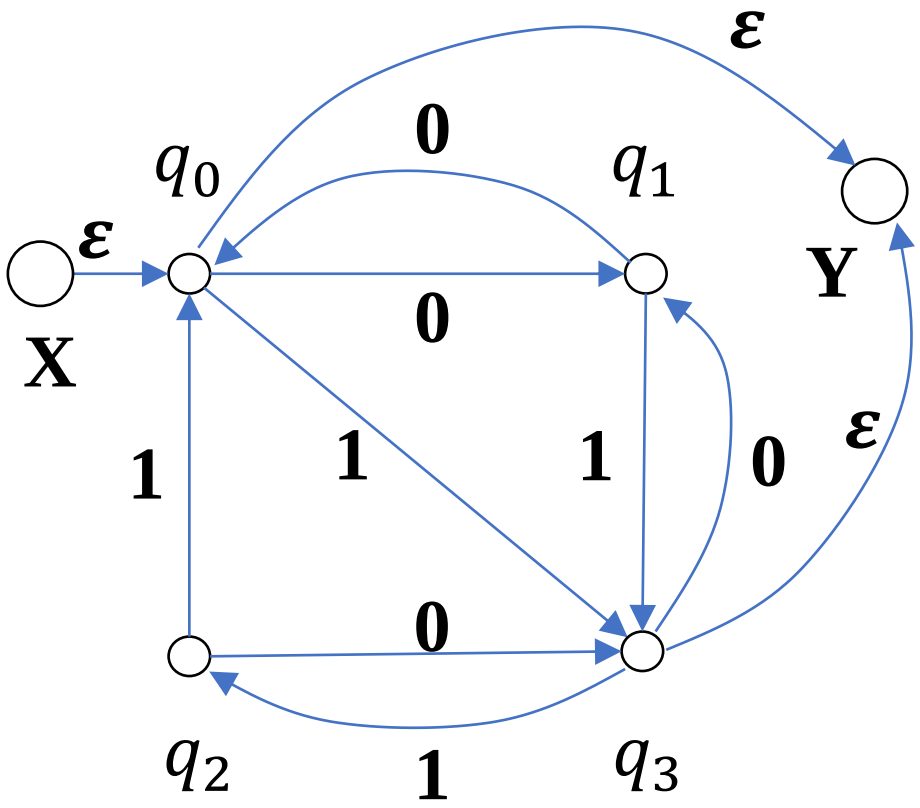


1. 预处理

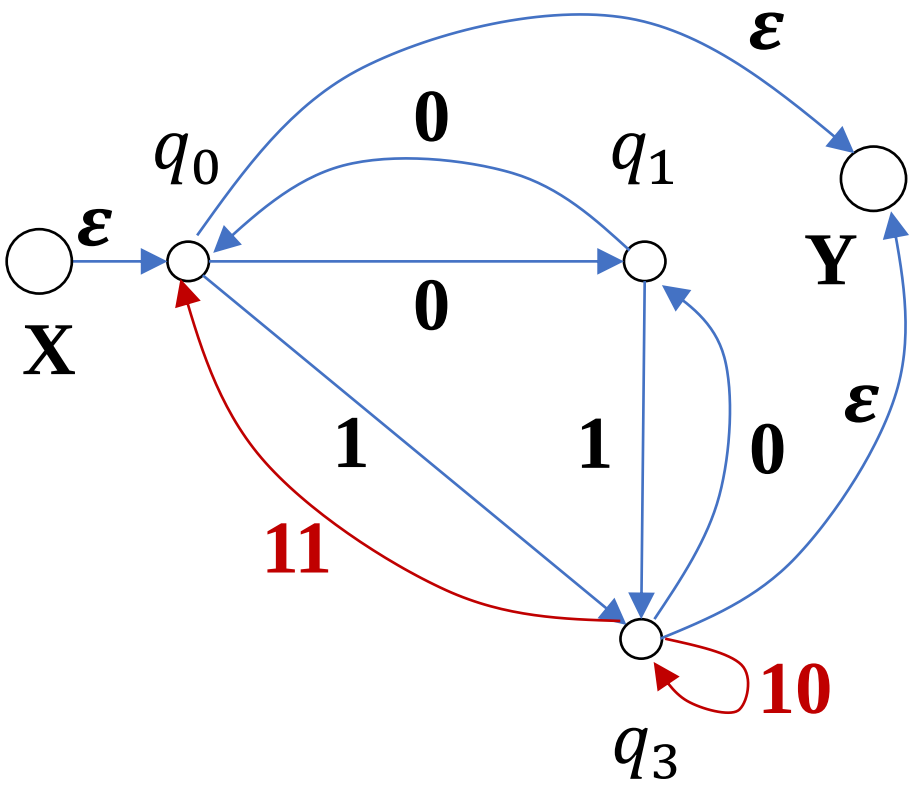


例4-16 求与下图所示的DFA等价的正则表达式，去状态顺序为q_2, q_1, q_3, q_0。

1. 预处理

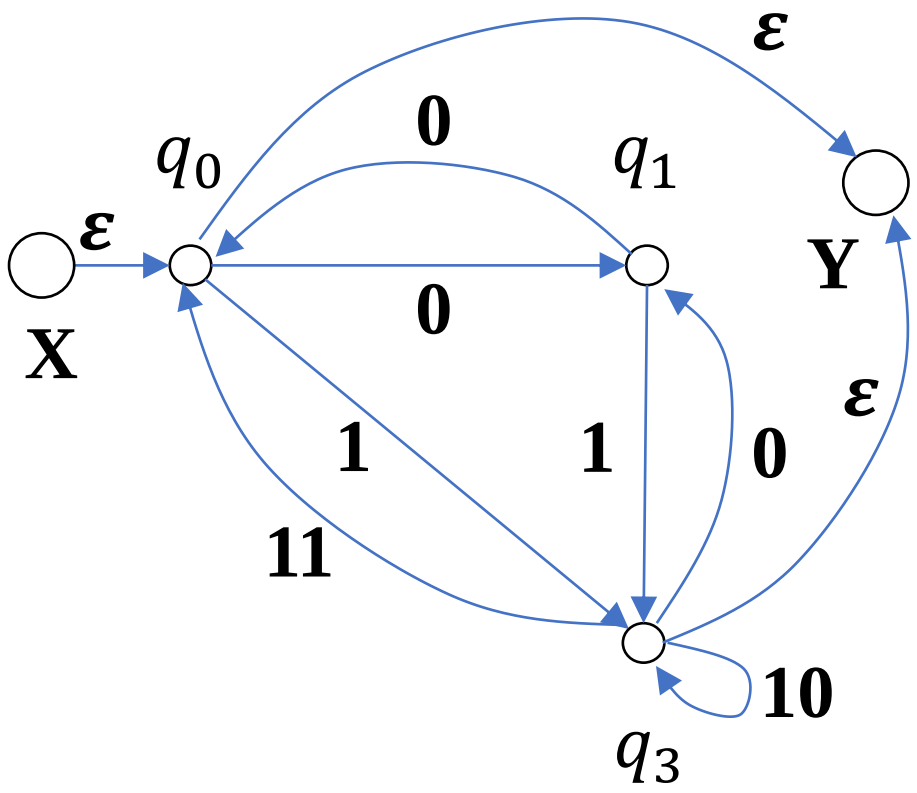


2.去状态q_2

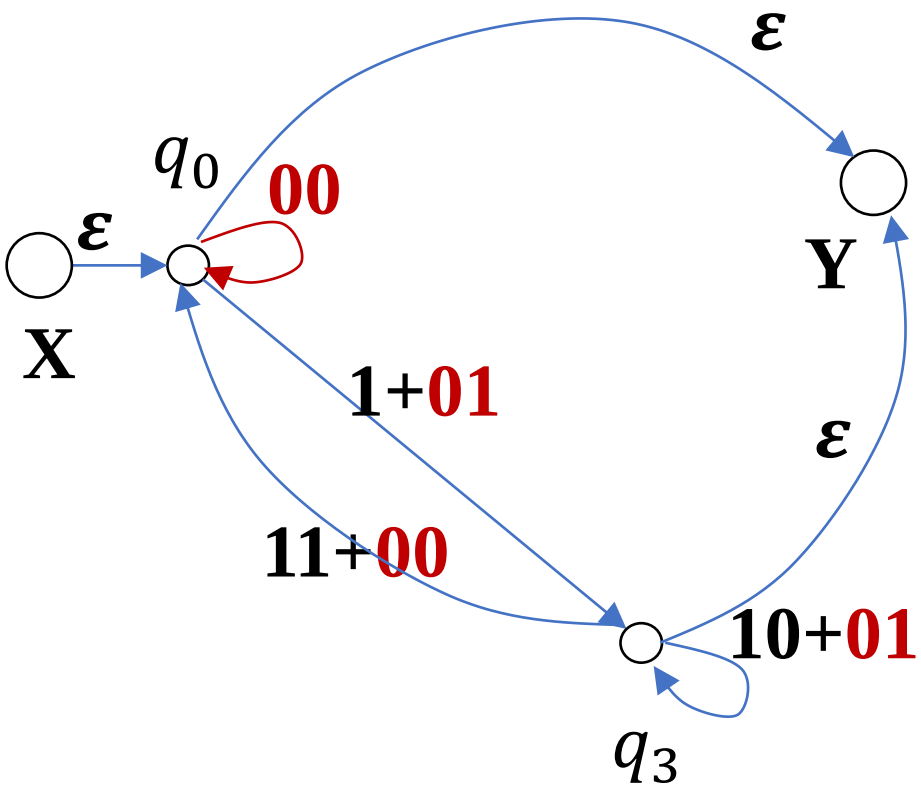


例4-16 求与下图所示的DFA等价的正则表达式，去状态顺序为q_2, q_1, q_3, q_0。

2.去状态q_2

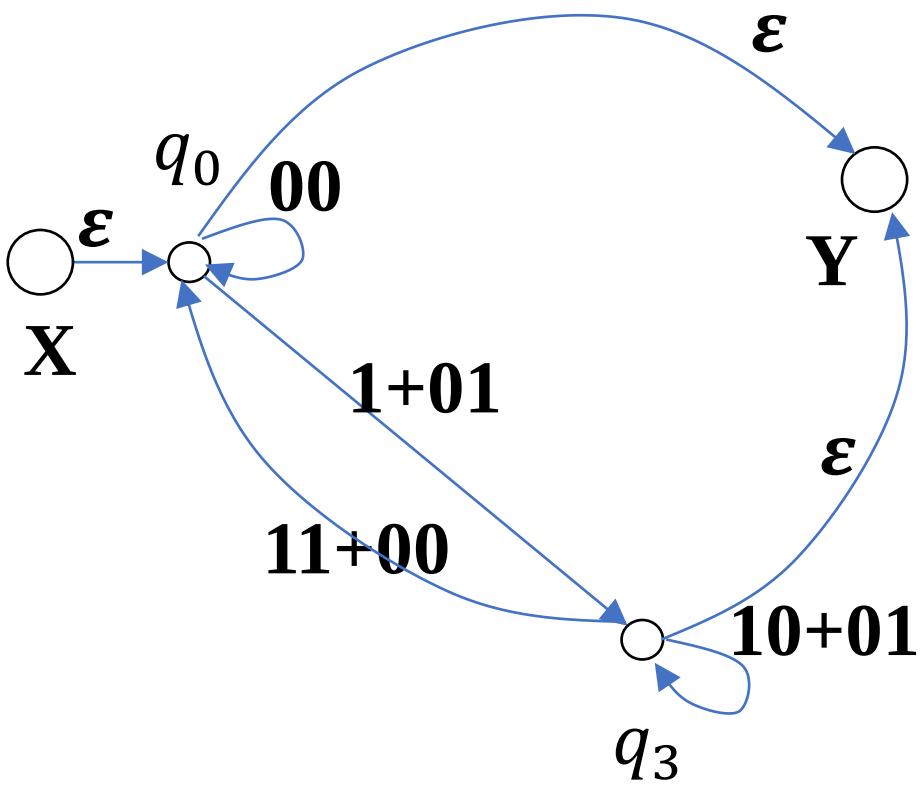


3.去状态q_1

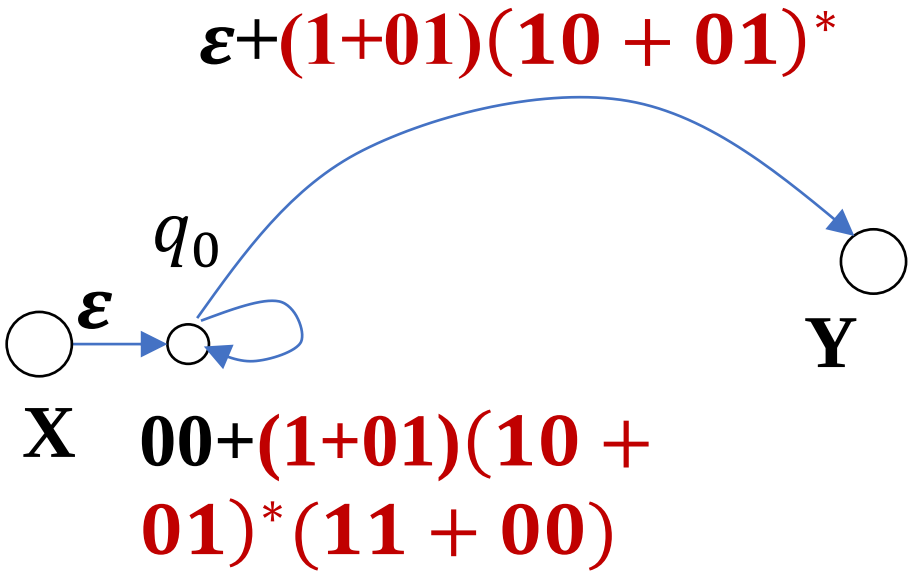


例4-16 求与下图所示的DFA等价的正则表达式，去状态顺序为q_2, q_1, q_3, q_0。

3.去状态q_1

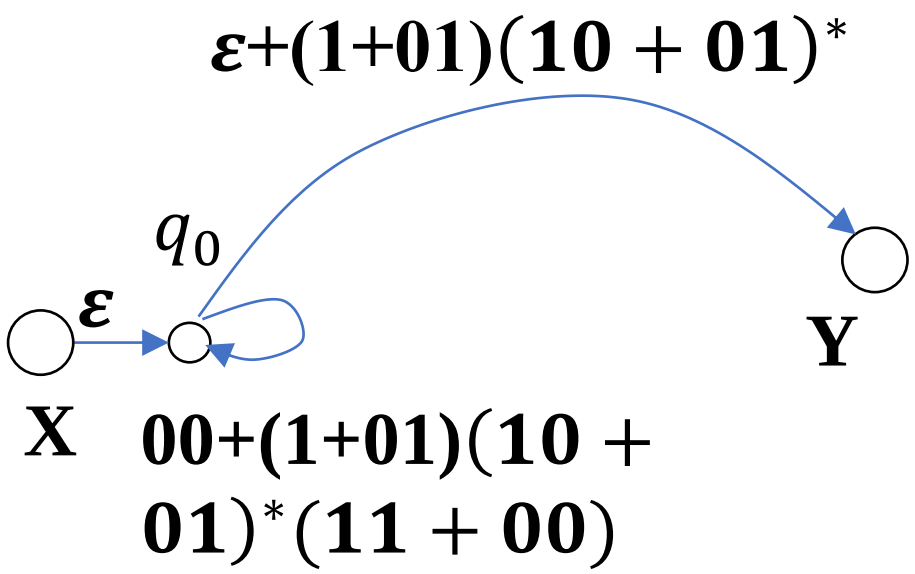


4.去状态q_3

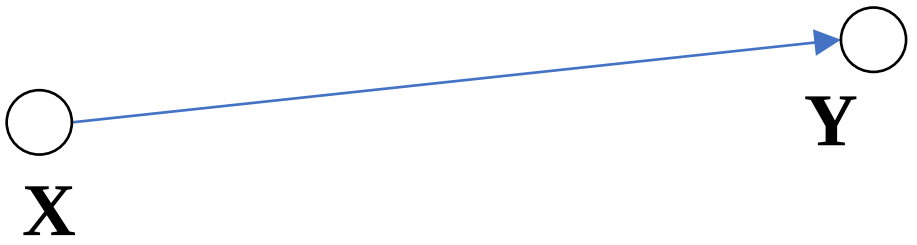


例4-16 求与下图所示的DFA等价的正则表达式，去状态顺序为q_2, q_1, q_3, q_0。

4.去状态q_3



5.去状态q_0



(00+(1+01)(10 + 01)*(11 + 00))*(ε+(1+01)(10 + 01)*)

以下几点值得注意：

- 如果去状态的顺序不一样，则得到的RE可能在形式是不一样，但它们都是等价的。
- 当DFA的终止状态都是不可达的时候，状态转移图中必不存在从开始状态到终止状态的路。按照上述过程，最终会去掉除了X和Y外的所有状态和弧。此时，相应的RE为Φ。
- 不计算自身到自身的弧，如果状态q的入度为n，出度为m，则将状态q及其相关的弧去掉之后，需要添加nm条新弧。

4.3.2 DFA到正则表达式的等价变换

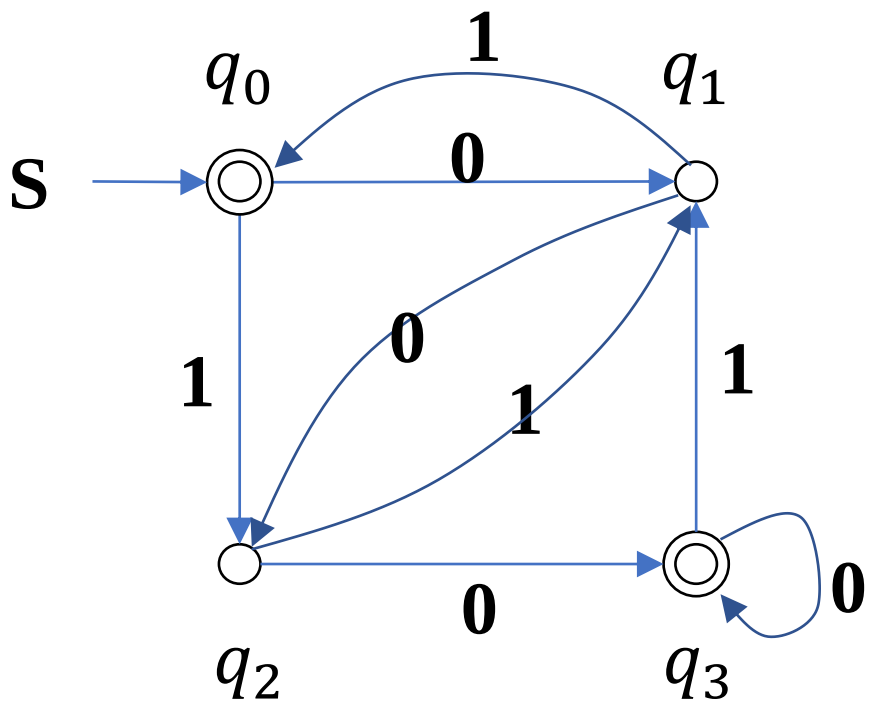
以下几点值得注意：

- 对操作的步骤实施归纳，可以证明图上作业法的正确性。
- 按照所给的方法，不会将状态X和Y去掉。实际上，这里所给的方法也可以说是个算法，而且在计算机系统中是不难实现的。

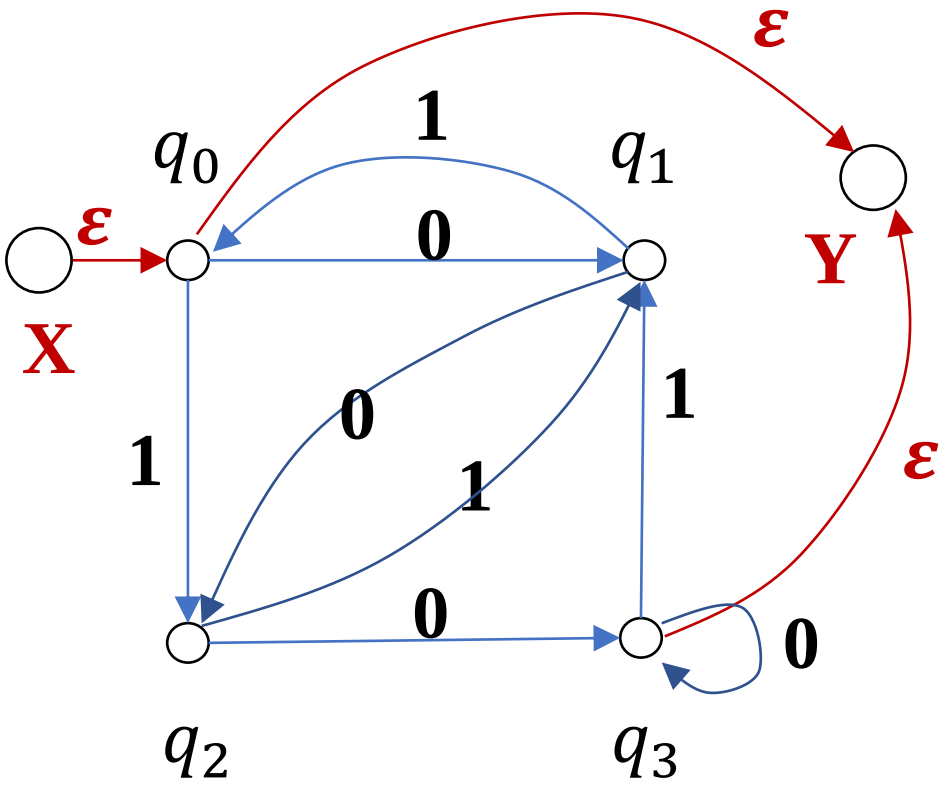
推论4-1 正则表达式是正则语言的表示模型。

4.3.2 DFA到正则表达式的等价变换

课堂练习：利用图上作业法求下图对应的正则表达式，去状态顺序为q_1, q_2, q_3, q_0。



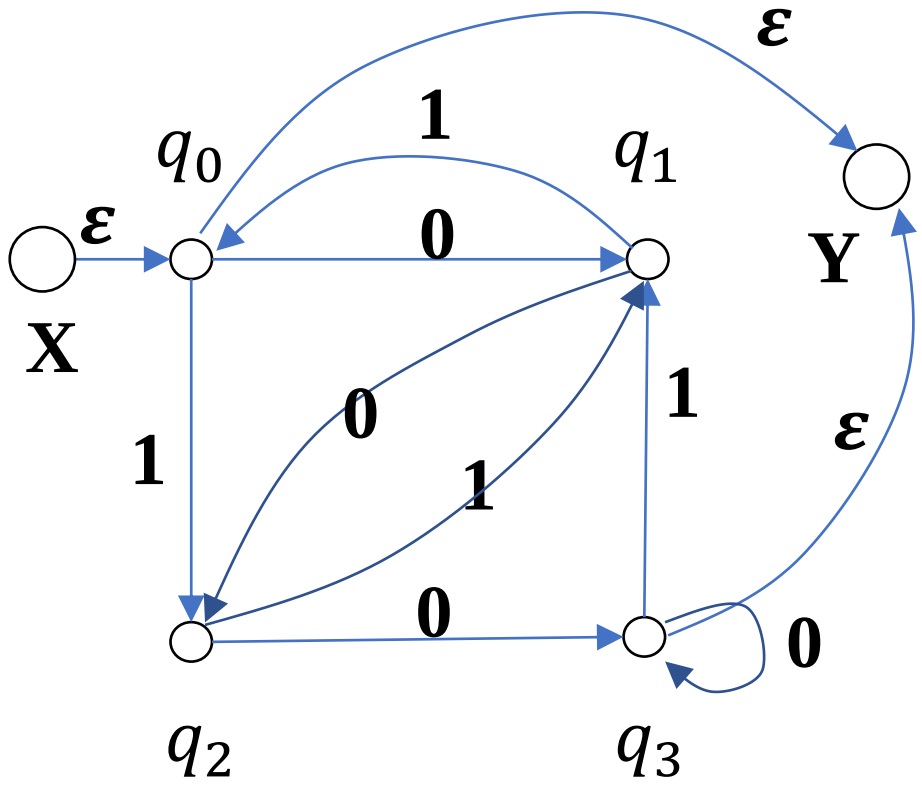
1. 预处理



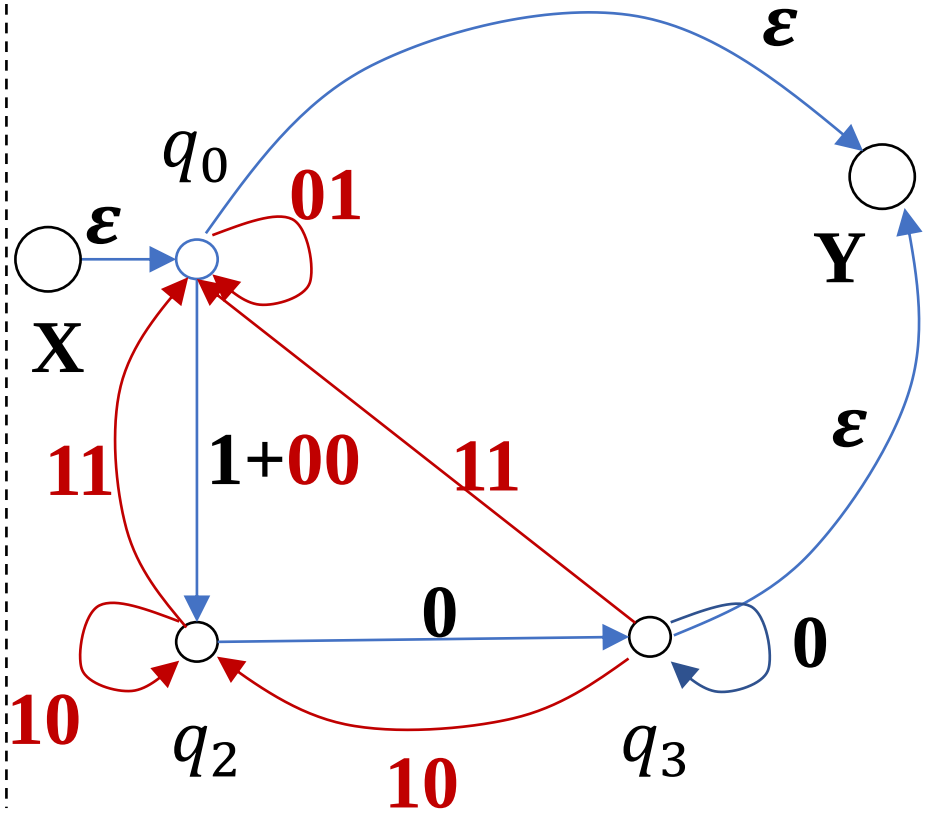
4.3.2 DFA到正则表达式的等价变换

课堂练习：利用图上作业法求下图对应的正则表达式，去状态顺序为q_1, q_2, q_3, q_0。

1. 预处理



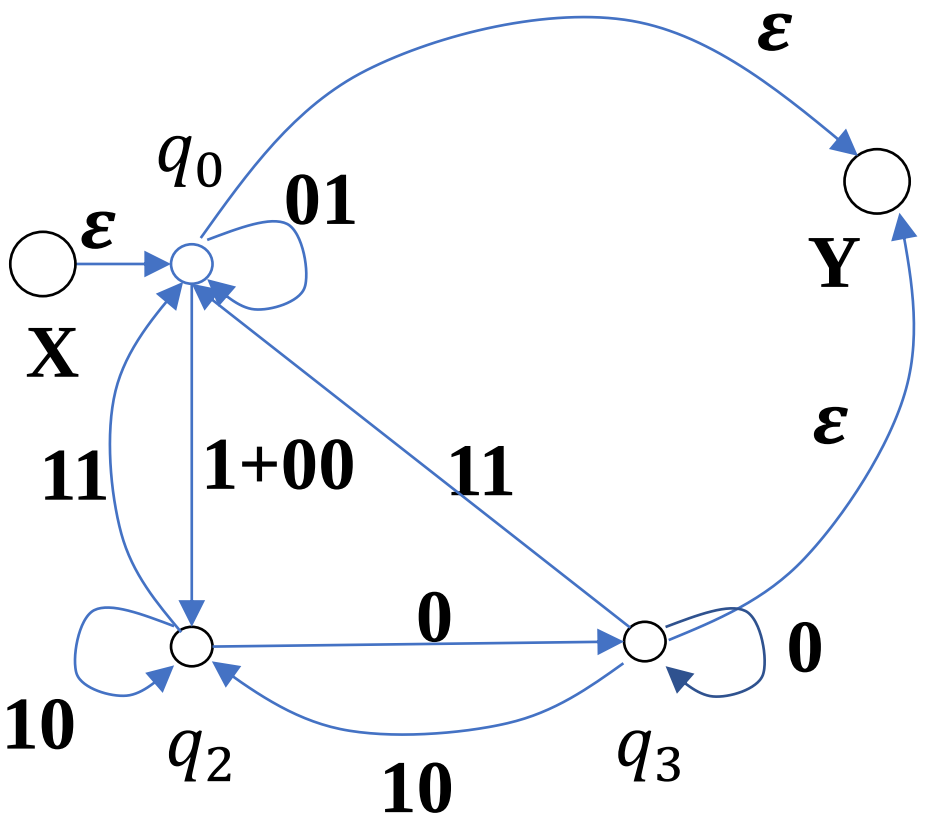
2.去状态q_1



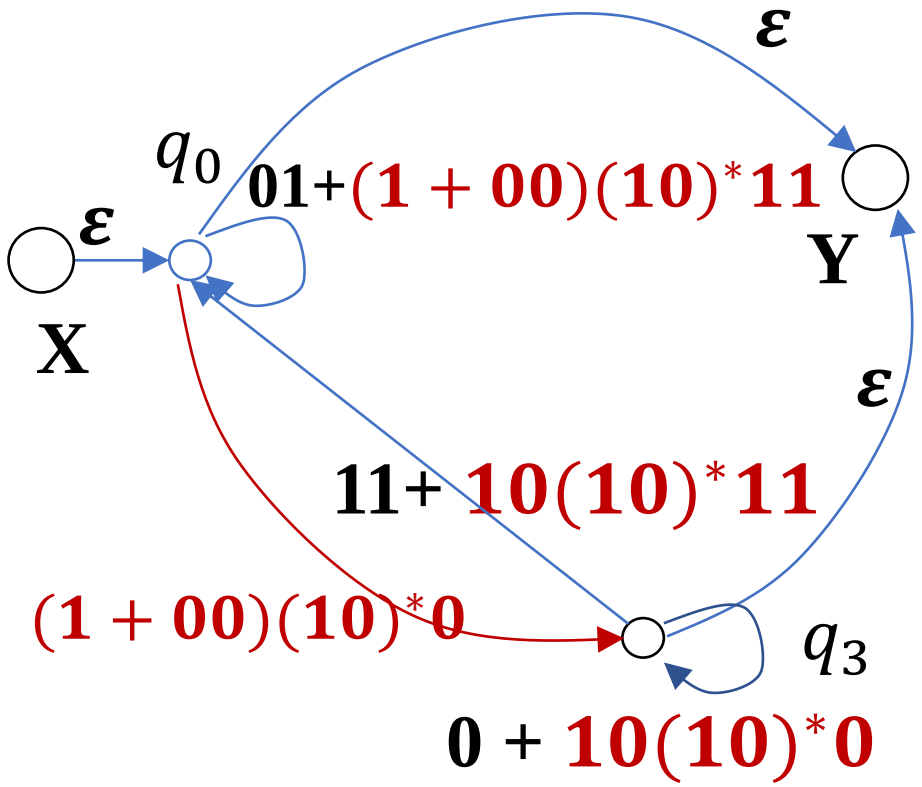
4.3.2 DFA到正则表达式的等价变换

课堂练习：利用图上作业法求下图对应的正则表达式，去状态顺序为q_1, q_2, q_3, q_0。

2.去状态q_1



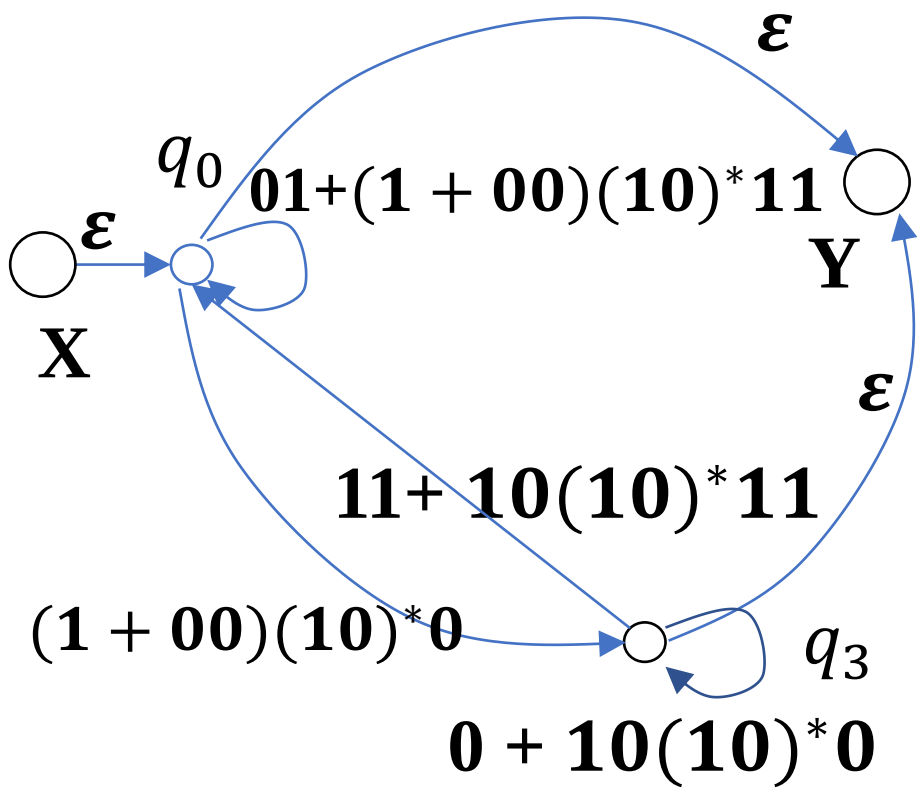
3.去状态q_2



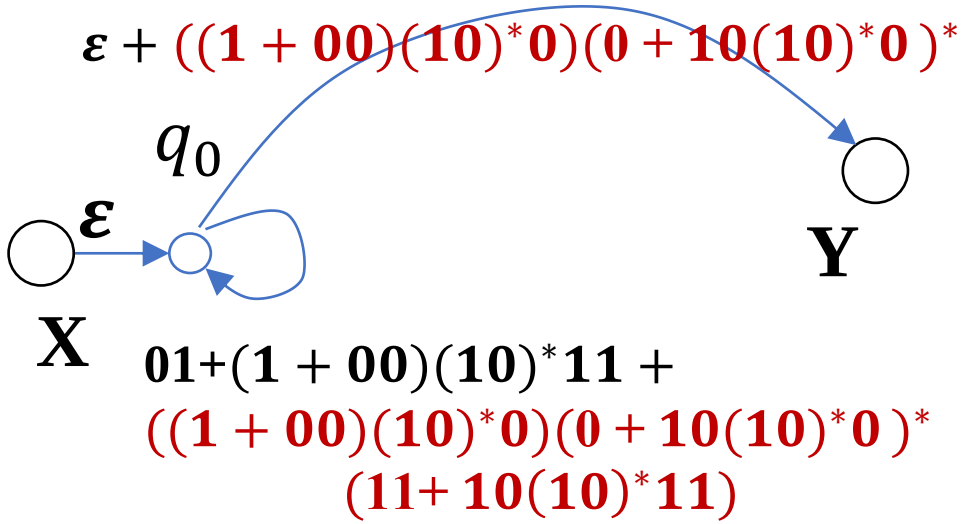
4.3.2 DFA到正则表达式的等价变换

课堂练习：利用图上作业法求下图对应的正则表达式，去状态顺序为q_1, q_2, q_3, q_0。

3.去状态q_2



4.去状态q_3

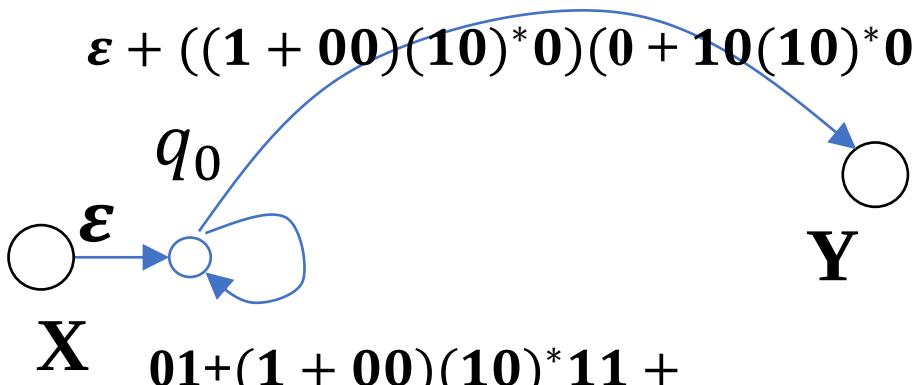


4.3.2 DFA到正则表达式的等价变换

课堂练习：利用图上作业法求下图对应的正则表达式，去状态顺序为q_1, q_2, q_3, q_0。

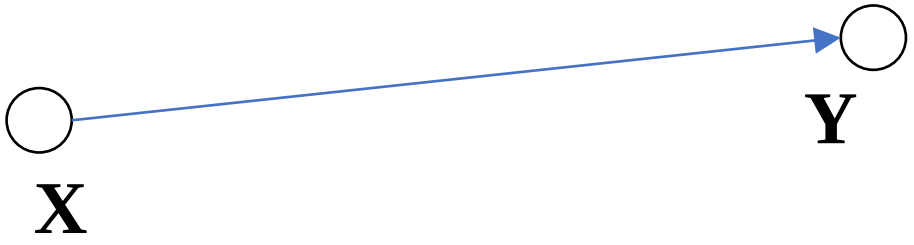
4.去状态q_3

$\varepsilon + ((1+00)(10)^*0)(0+10(10)^*0)^*$



$01+(1+00)(10)^*11 +$
$((1+00)(10)^*0)(0+10(10)^*0)^*$
$11+10(10)^*11$

5.去状态q_0



$(01+(1+00)(10)^*11 +$
$((1+00)(10)^*0)(0+10(10)^*0)^*11+$
$10(10)^*11)^* \ (\varepsilon + ((1+00)(10)^*0)(0+$
$10(10)^*0)^*)$

章节目录

4.1 启示

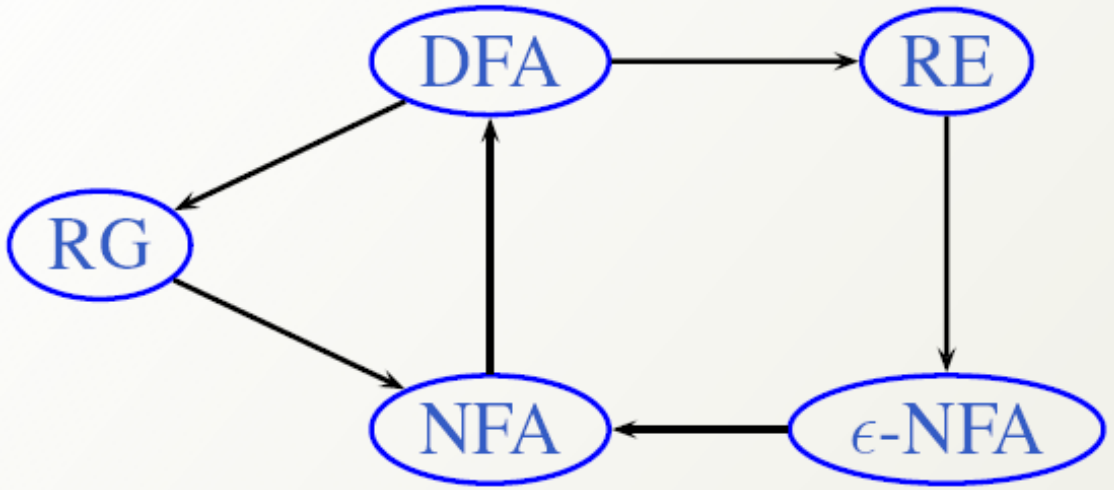
4.2 正则表达式的形式定义

4.3 正则表达式与FA等价

4.4 正则语言等价模型的总结

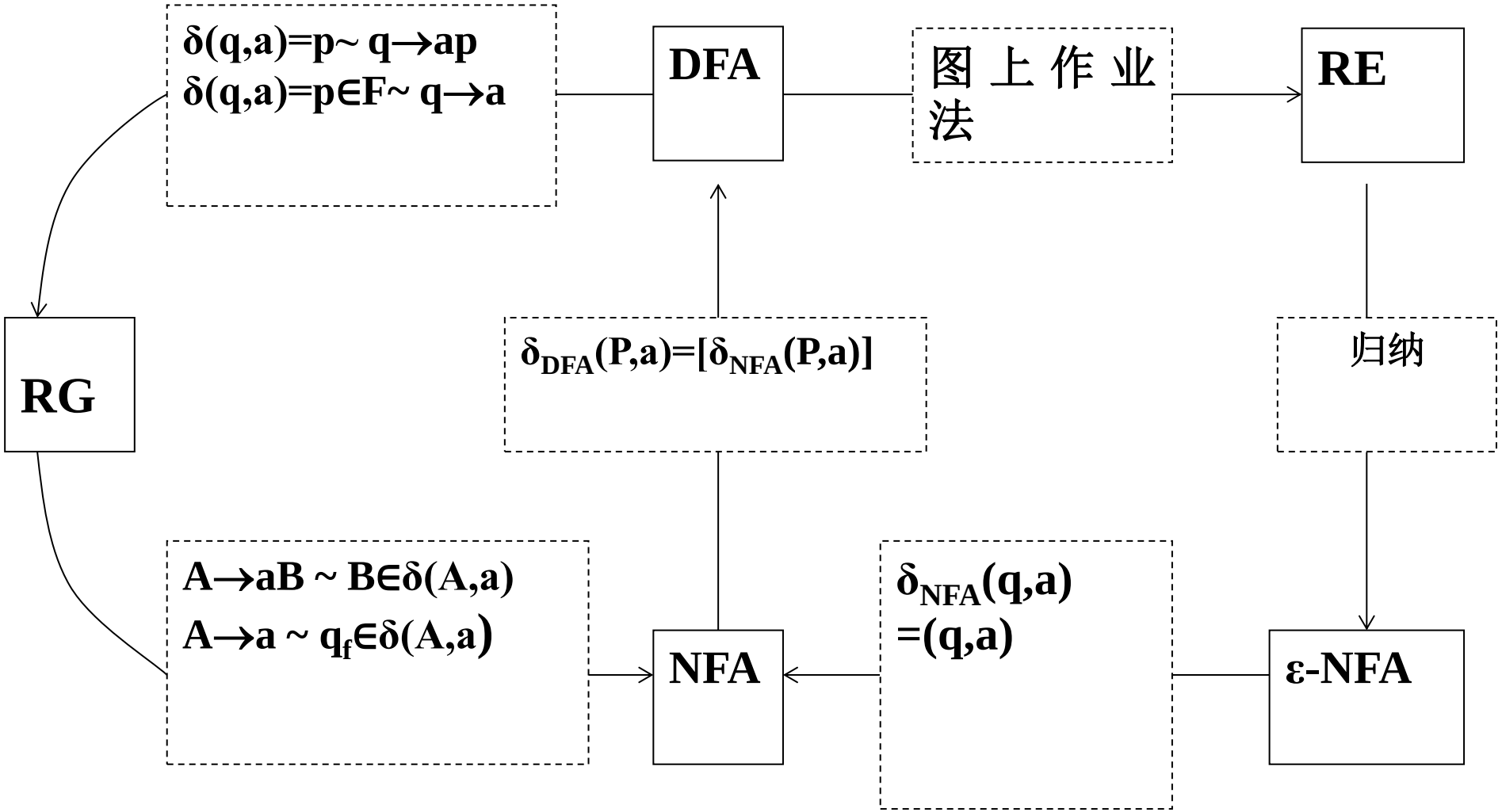
4.5 本章小结

4.4 正则语言等价模型总结



- DFA⇒RG: 右、左线性文法
- RG⇒NFA: 右、左线性文法
- DFA⇒RE: 图上作业法
- RE⇒ ϵ-NFA: 并、串联逐步构造
- ϵ-NFA⇒NFA: ϵ-CLOSURE
- NFA⇒DFA: 用一个状态对应一个状态集

4.4 正则语言等价模型总结



章节目录

4.1 启示

4.2 正则表达式的形式定义

4.3 正则表达式与FA等价

4.4 正则语言等价模型的总结

4.5 本章小结

4.5 本章小结

本章讨论了RE及其与FA的等价性。

① 正则表达式的形式定义及构造方法；

② 根据RE构造出与它等价的ε-NFA；

③ 反过来，可以用图上作业法构造出与给定的DFA等价的RE。

Thanks!