编译原理

第3章词法分析与有穷自动机

第3章　词法分析与有穷自动机

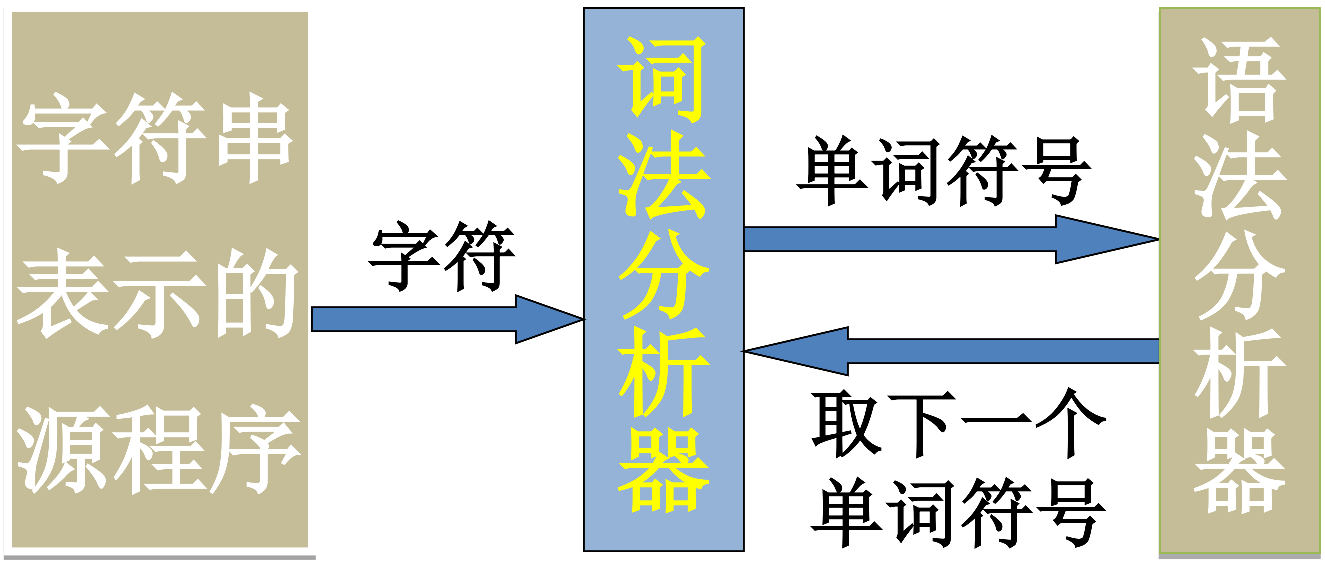
- 1、词法分析程序功能
- 2、单词符号及输出单词的形式
- 3、语言单词符号的两种定义方式
- 4、正规式与有穷自动机
- 5、正规文法与有穷自动机
- 6、词法分析程序的编写方法

3.1 词法分析程序功能

词法分析 Lexical Analysis → 语法分析 SyntaxAnalysis → ……

3.1 词法分析程序的功能

3.1 词法分析程序的功能



输入： if (a>1) b =100；

3.2 单词符号及输出单词的形式

语言的单词符号：语言中具有独立意义的最小语法单位。

```
int fib(int n) {
    if (n <= 1)
        return n;
    return fib(n - 1) + fib(n - 2);
}

char a[100] = "Enter an integer:
";

int main() {
    print_string(a);
    int max = read_int();
    int n = 1;
    do {
        print_int(fib(n++));
        print_string("\n");
    } while (n <= max);
}   return 0;
```

关键字 if、while、do

标识符 各种名字：变量名、常量名、数组名和函数名

常数　整型常数：125、实型常数：0.718、布尔型常数：TRUE

运算符 ＋、－、*、/、＜

分界符 ，、；、（、）、：

3.2 单词符号及输出单词的形式

```
%{
  enum yytokentype{
        NUMBER=258,
        ADD=259,
        SUB=260,
        MUL=261,
        DIV=262,
        ABS=263,
        EOL=264
  };
  int yylval;
%}
%%
"+" {return ADD;}
"-" {return SUB;}
"*" {return MUL;}
"/" {return DIV;}
"|" {return ABS;}
[0-9]+ {yylval=atoi(yytext);return NUMBER;}
\n {return EOL;}
[ \t] { }
.  {printf("Mystery character %c\n",*yytext);}
%%
main(int argc,char **argv)
{
    int tok;
    while(tok=yylex()){
        printf("%d",tok);
        if(tok==NUMBER)printf("=%d\n",yylval);
        else printf("\n");
    }
}
```

```
%token NUMBER
%token ADD SUB MUL DIV ABS
%token EOL

%%
calclist:
| calclist exp EOL {printf("=%d\n",$2);}
;
exp:factor default $$=$1
|exp ADD factor {$$=$1+$3;}
|exp SUB factor {$$=$1-$3;}
;
factor:term default $$=$1
|factor MUL term {$$=$1*$3;}
|factor DIV term {$$=$1/$3;}
;
term:NUMBER default $$=$1
|ABS term ABS {$$=$2>=0?$2:-$2; }
;
%%
main(int argc,char **argv)
{
     yyparse();
}
```

3.2 单词符号及输出单词的形式

单词符号可用二元式表示：

（单词种别，单词自身的值）

例：if (a>1) b =100;

标识符的种别编码	整数10；
常数的种别编码	整数11；
基本字if种别编码	2；
赋值号的种别编码	17；
大于号的种别编码	23；
分号的种别编码	26；
左括号的种别编码	29；
右括号的种别编码	30；

3.2 单词符号及输出单词的形式

例子: if (a>1) b =100;

标识符的种别编码	整数10；
常数的种别编码	整数11；
基本字if种别编码	2；
赋值号的种别编码	17；
大于号的种别编码	23；
分号的种别编码	26；
左括号的种别编码	29；
右括号的种别编码	30；

①	(2,)	if
②	(29,)	(
③	(10,'a')	a
④	(23,)	>
⑤	(11,1)	1
⑥	(30,)	)
⑦	(10,'b')	b
⑧	(17,)	=
⑨	(11,100)	100
⑩	(26,)	;

3.3 单词符号的两种定义方式

正规式

```
l(l|d)*
```

正规文法

```
I→l|Il|Id
或:
I→l|lT
T→l|d|lT|dT
```

课外阅读任务:
https://www.regular-expressions.info

3.3 单词符号的两种定义方式

```
\A(?:[a-z0-9!#$%&'*+/=?^_‘{|}~-]+(?:\.[a-z0-9!#$%&'*+/=?^_‘{|}~-]+)*
 |  "(?:[\x01-\x08\x0b\x0c\x0e-\x1f\x21\x23-\x5b\x5d-\x7f]
      |  \\[\x01-\x09\x0b\x0c\x0e-\x7f])*")
@ (?:(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?
  |  \[(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}
       (?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?|[a-z0-9-]*[a-z0-9]:
          (?:[\x01-\x08\x0b\x0c\x0e-\x1f\x21-\x5a\x53-\x7f]
          |  \\[\x01-\x09\x0b\x0c\x0e-\x7f])+)
     \])\z
```

```
\A[a-z0-9!#$%&'*+/=?^_‘{|}~-]+(?:\.[a-z0-9!#$%&'*+/=?^_‘{|}~-]+)*@
(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\z
```

```
\A(?=[a-z0-9@.!#$%&'*+/=?^_‘{|}~-]{6,254}\z)
 (?=[a-z0-9.!#$%&'*+/=?^_‘{|}~-]{1,64}@)
 [a-z0-9!#$%&'*+/=?^_‘{|}~-]+(?:\.[a-z0-9!#$%&'*+/=?^_‘{|}~-]+)*
@ (?:(?=[a-z0-9-]{1,63}\.)[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+
  (?=[a-z0-9-]{1,63}\z)[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\z
```

3.3.1 正规式和正规集

设字母表$\Sigma=\{a_1, a_2,\ldots, a_n\}$，在$\Sigma$上的正规式和它所表示的正规集用规则1-4定义：

1. Φ是Σ上的正规式，它所表示的正规集是Φ，即空集{ }。

2. ε是Σ上的正规式，正规集：空符号串集合，$\{\varepsilon\}$。

3. a_i是$\sum$上的一个正规式，它所表示的正规集是由单个符号a_i所组成，即$\{a_i\}$。

3.3.1 正规式和正规集

4. 如果e_1和e_2是Σ上的正规式，它们所表示的正规集为 $L(e_1)$和$L(e_2)$ ，则有：

(1) $e_1|e_2$是Σ上的一个正规式，
它所表示的正规集为：$L(e_1|e_2)=L(e_1) \cup L(e_2)$

(2) e_1e_2 是Σ上的一个正规式，
它所表示的正规集为：$L(e_1e_2)=L(e_1)L(e_2)$

(3) $(e_1)^*$是Σ上的一个正规式，
它所表示的正规集为：$L((e_1)^*)=(L(e_1))^*$

3.3.1 正规式和正规集

例1　设有字母表Σ={a,b}，根据正规式与正规集的定义，则有：

1. a 和 b是正规式，相应正规集为

$$L(a)=\{a\}, \quad L(b)=\{b\}$$

2. a|b 是正规式，相应正规集为

$$L(a|b)=L(a)\cup L(b)=\{a,b\}$$

3. ab 是正规式，相应正规集为

$$L(ab)=L(a)L(b)=\{a\}\{b\}=\{ab\}$$

3.3.1 正规式和正规集

4. $(a \mid b)^*$ 是正规式，相应正规集为

$$L((a|b)^*) = (L(a|b))^*$$
$$= \{a, b\}^* = \{\varepsilon, a, b, aa, ab, ba, bb, \cdots\}$$

提问：$\{a, b\}^*$ 的子集 $\{a^n b^n \mid n \geq 1\}$ 是正规集吗？

3.3.1 正规式和正规集

5. ba*是正规式，相应的正规集为

$L(ba^*)=L(b)L(a^*)=\{b, ba, baa, baaa, \cdots\}$

6. $(a|b)^*(aa|bb)(a|b)^*$ 是正规式，相应正规集为

$$\begin{aligned} &L((a|b)^*(aa|bb)(a|b)^*) \\ &\quad =L((a|b)^*)L(aa|bb)L((a|b)^*) \\ &\quad =\{a,b\}^*\{aa,bb\}\{a,b\}^* \end{aligned}$$

3.3.1 正规式和正规集

华中科技大学 HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

例2　设Σ={a,b,c},则 $aa^*bb^*cc^*$ 是∑上的一个正规式，它所表示的正规集：

L={ abc， aabc， abbc， abcc， aaabc， …}

={ }

3.3.1 正规式和正规集

例3 设程序语言字母表是键盘字符集合，部分单词符号可用如下正规式定义：

关键字	`if|else|while|do`
标识符	`l(l|d)*`
整常数	`dd*`
关系运算符	`<|<=|>|>=|<>`

3.3.1 正规式和正规集

标识符

ID=l(l|d)*

I→l|Il|Id

l代表a~z中任一字母
d代表0~9中任一数字

3.3.1 正规式和正规集

例 $(a|b)^*$ $(a^*b^*)^*$ ； $b(ab)^*$ $(ba)^*b$

如果正规式R_1和R_2描述的正规集相同，则称正规式R_1与R_2等价。记为 $R_1 = R_2$。

3.3.1 正规式和正规集

正规式具有如下性质：

令A , B 和 C 均为正规式，则

1. $A \mid B = B \mid A$　　　　(　　)　交换律
2. $A \mid (B \mid C) = (A \mid B) \mid C$　　　　(　　)　结合律
3. $A(BC) = (AB)C$　　　　(　　)
4. $A(B \mid C) = AB \mid AC$　　　　(　　)　分配律
5. $(A \mid B)C = AC \mid BC$　　　　(　　)
6. $A\varepsilon \mid \varepsilon A = A$
7. $A^* = AA^* \mid \varepsilon = A \mid A^* = (A \mid \varepsilon)^*$
8. $(A^*)^* = A^*$

3.3.2 正规文法与正规式

1. 正规文法到正规式的转换

(1) 将正规文法中的每个非终结符表示成关于它的一个正规式方程，获得一个联立方程组。

(2) 依照求解规则：

若 $x = \alpha x \mid \beta$ (或 $x = \alpha x + \beta$) 则解为 $x = \alpha^{*}\beta$

若 $x = x\alpha \mid \beta$ (或 $x = x\alpha + \beta$) 则解为 $x = \beta\alpha^{*}$

3.3.2 正规文法与正规式

例1 设有正规文法G：

$Z \rightarrow 0A$

$A \rightarrow 0A \mid 0B$

$B \rightarrow 1A \mid \varepsilon$

试给出该文法生成语言的正规式。

分析 首先给出相应的正规式方程组(方程组中用“＋”代替正规式中的“｜”)如下：

$Z = 0A \qquad (1)$

$A = 0A + 0B \qquad (2)$

$B = 1A + \varepsilon \qquad (3)$

3.3.2 正规文法与正规式

$Z = 0A$ (1)

$A = 0A + 0B$ (2)

$B = 1A + \varepsilon$ (3)

将(3)代入(2)中的B得

$A = 0A + 01A + 0$ (4)

对(4)利用分配律得

$A = (0 + 01)A + 0$ (5)

对(5)使用求解规则得

$A = (0 + 01)^* 0$ (6)

将(6)代入(1)式中的A，得

$Z = 0(0 + 01)^* 0$

即正规文法G[Z]所生成语言的正规式是 $R = 0(0 \mid 01)^* 0$

3.3.2 正规文法与正规式

例2 设有正规文法G：

$$A \to aB \mid bB$$

$$B \to aC \mid a \mid b$$

$$C \to aB$$

试给出该文法生成语言的正规式。

分析 首先给出相应的正规式方程组(方程组中用“＋”代替正规式中的“|”)如下：

$$A = aB + bB \quad (1)$$

$$B = aC + a + b \quad (2)$$

$$C = aB \quad (3)$$

3.3.2 正规文法与正规式

$$A = aB + bB \qquad (1)$$

$$B = aC + a + b \qquad (2)$$

$$C = aB \qquad (3)$$

将(3)代入(2)中的C得

$$B = aaB + a + b \qquad (4)$$

对(4)使用求解规则得

$$B = (aa)^*(a + b) \qquad (5)$$

(5)代入(1)中的B得

$$A = (a + b)(aa)^*(a + b)$$

即正规文法G[A]所生成语言的正规式是

$$R = (a \mid b)(aa)^*(a \mid b)$$

3.3.2 正规文法与正规式

例3 设有正规文法G，请给出其语言的正规式

$$Z \rightarrow U0 \mid V1$$
$$U \rightarrow Z1 \mid 1$$
$$V \rightarrow Z0 \mid 0$$

相应的正规式方程组为

$$Z = U0 + V1 \qquad (1)$$
$$U = Z1 + 1 \qquad (2)$$
$$V = Z0 + 0 \qquad (3)$$

3.3.2 正规文法与正规式

华中科技大学 HUAZHONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

$Z = U0 + V1$ (1)

$U = Z1 + 1$ (2)

$V = Z0 + 0$ (3)

(2)和(3)代入(1)得

$Z = Z10 + 10 + Z01 + 01$

$\quad = Z(10+01)+(10+01)$ (4)

对(4) 用求解规则: $Z = (10 + 01)\ (10 + 01\ )^*$

即正规文法G[Z]所生成语言的正规式是

$R = (10 \mid 01)(10 \mid 01)^*$

3.3.2 正规文法与正规式

例4　已知描述“标识符”　单词符号的正规文法为

<标识符>→l|<标识符>l|<标识符>d

根据前述求解规则，可知该文法所描述语言的正规式是 l(l|d)*

3.3.2 正规文法与正规式

2. 正规式到正规文法的转换算法

(1) 令 $V_T=\Sigma$ 。

(2) 对任何正规式R,选择一个非终结符Z,生成规则Z→R,并令S＝Z。

(3) 若a和b都是正规式，对形如 A→ab 的规则转换成 A→aB 和 B→b ，B是新增的非终结符。

(4) 对形如A → a*b 的规则，转换成 A → aA | b 。

(5) 不断利用 (3)和(4)进行变换，直到每条规则最多含有一个终结符为止。

3.3.2 正规文法与正规式

例1　将 R=(a|b)(aa)*(a|b) 转换成相应的正规文法

令A是文法开始符号，根据规则(2)

$$A \rightarrow (a|b)(aa)^*(a|b)$$

根据规则(3)变换为

$$A \rightarrow (a|b)B$$

$$B \rightarrow (aa)^*(a|b)$$

3.3.2 正规文法与正规式

对B根据规则(4)变换为

A→ aB | bB

B→ aaB| a | b

根据规则(3)变换为

A→ aB | bB

B→ aC | a | b

C→ aB

A→(a|b)B

B→(aa)*(a|b)

规则(4)

A→a*b ⇔A→aA|b

规则（3）

A→ab ⇔A→aB B→b

3.3.2 正规文法与正规式

例2 将描述标识符的正规式 R=l(l|d)* 转换成正规文法

令I为文法的开始符号，根据规则(2)有

$$I \rightarrow l\,(l|d)^*$$

根据规则(3)变换为

$$I \rightarrow lT$$

$$T \rightarrow (l|d)^*$$

根据规则(4)变换为

$$I \rightarrow lT$$

$$T \rightarrow (l|d)T|\varepsilon$$

3.3.2 正规文法与正规式

进一步变换为

$I \rightarrow lT$

$T \rightarrow lT|dT|\varepsilon$

去掉 ε 规则

$I \rightarrow l|lT$

$T \rightarrow l|d|lT|dT$

即前面描述标识符的右线性文法。

3.4 正规式与有穷自动机

有穷自动机是具有离散输入与输出系统的一种抽象数学模型。

有穷自动机有“确定的”和“非确定的”两类；
都能准确地识别正规集。

3.4.1 确定有穷自动机

确定有穷自动机（DFA）

$M=(Q, \Sigma, f, S, Z)$

Q: 是有穷状态集合，每一个元素称为一个状态；

Σ:是有穷输入字母表，每个元素称为一个输入字符；

f: 是一个从$Q\times\Sigma$到Q的单值映射；

S: $S\in Q$，是唯一的一个初态；

Z: $Z\subseteq Q$，是一个终态集。

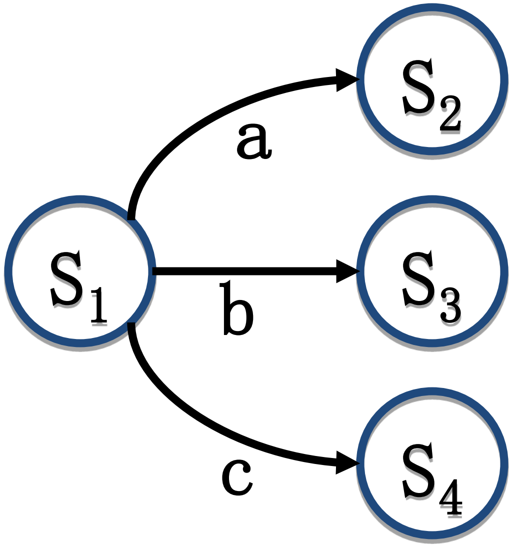
3.4.1 确定有穷自动机

$M=(Q, \Sigma, f, S, Z)$

f 是一个从 $Q\times\Sigma$ 到 Q 的单值映射：

$f(q_i,a) = q_j \qquad (q_i,q_j \in Q, a\in\Sigma)$

当前状态为 q_i，输入字符为a时，自动机将转换到下一个状态 q_j。q_j 称为 q_i 的一个后继状态。



3.4.1 确定有穷自动机

例 设DFA $M=(\{q_0, q_1, q_2\},\{a, b\}, f, q_0,\{q_2\})$

其中 $f(q_0, a) = q_1 \quad f(q_0, b) = q_2$

$f(q_1, a) = q_1 \quad f(q_1, b) = q_1$

$f(q_2, a) = q_2 \quad f(q_2, b) = q_1$

状态转换矩阵

状态 \ 字符	a	b
q_0	q_1	q_2
q_1	q_1	q_1
q_2	q_2	q_1

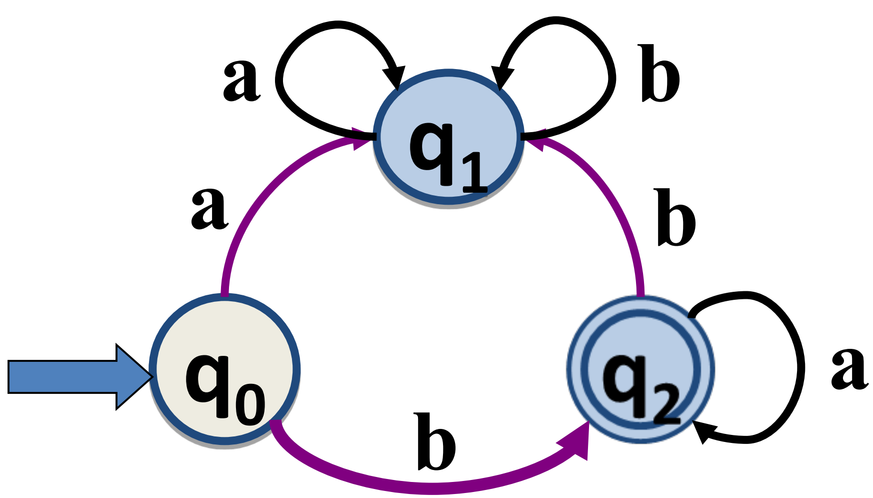
3.4.1 确定有穷自动机

一个 DFA也可以表示成一张状态转换图。
例DFA $M=(\{q_0, q_1, q_2\},\{a, b\}, f, q_0, \{q_2\})$ 的状态转换图如下图所示。

$f(q_0, a) = q_1$ $f(q_0, b) = q_2$

$f(q_1, a) = q_1$ $f(q_1, b) = q_1$

$f(q_2, a) = q_2$ $f(q_2, b) = q_1$

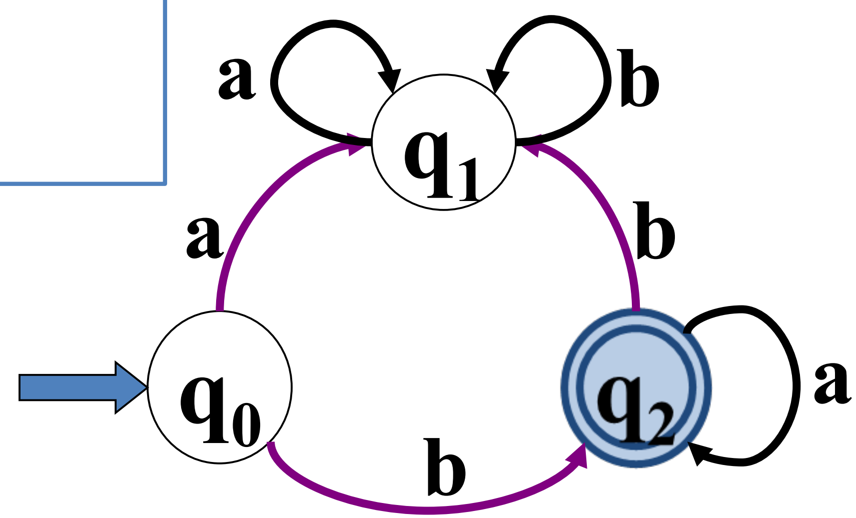


3.4.1 确定有穷自动机

对Σ*中的任何符号串β，若存在一条从初态结到终态结的道路，在这条路上所有弧的标记连结成的符号串等于β，则称β为DFA M所识别(或接受)。

M所识别的符号串的全体记为L(M)，称为DFA M所识别的语言。

DFA M所识别的语言为
$L(M) = ba^*$。



DFA M

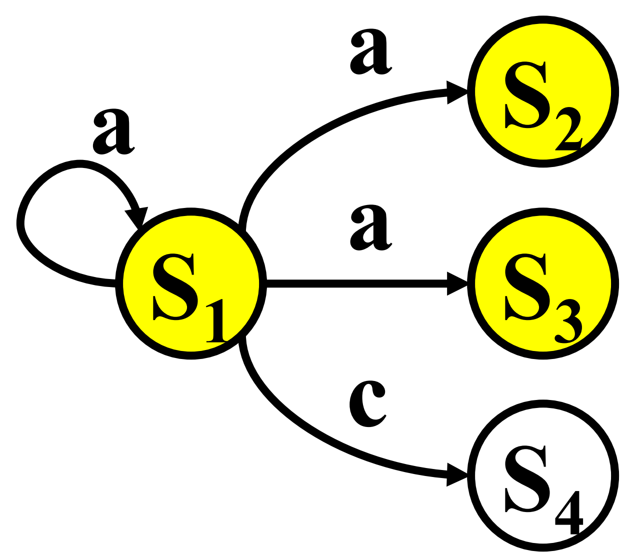
3.4.2 非确定有穷自动机

非确定有穷自动机（NFA）

一个非确定有穷自动机M是一个五元组

M=(Q, Σ, f, S, Z)

Q, Σ, Z 意义同DFA , f 和 S不同于DFA 。



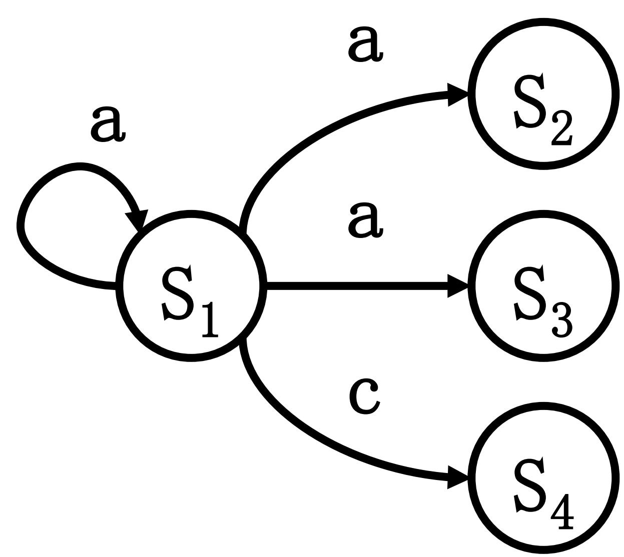
3.4.2 非确定有穷自动机

(1) f：是一个多值函数，$f(q_i, a)$ ={某些状态的集合}

由图可知 $f(S_1, a)=\{S_1, S_2, S_3\}$

允许 $f(q_i, \varepsilon)$={某些状态的集合}。

(2) $S \subseteq Q$，是非空初态集。



3.4.2 非确定有穷自动机

例　设有NFA M=({1,2,3},{a,b},f,{1,3},{2})

其中 $f(1, a) = \{3\}$　　$f(1, b) = \{1,2\}$

$f(2, a) = \Phi$　　$f(2, b) = \{3\}$

$f(3, a) = \Phi$　　$f(3, b) = \{2\}$

例中 NFA M′ 对应的状态转换矩阵如下表所示。

状态＼字符	a	b
1	{3}	{1,2}
2	Φ	{3}
3	Φ	{2}

3.4.2 非确定有穷自动机

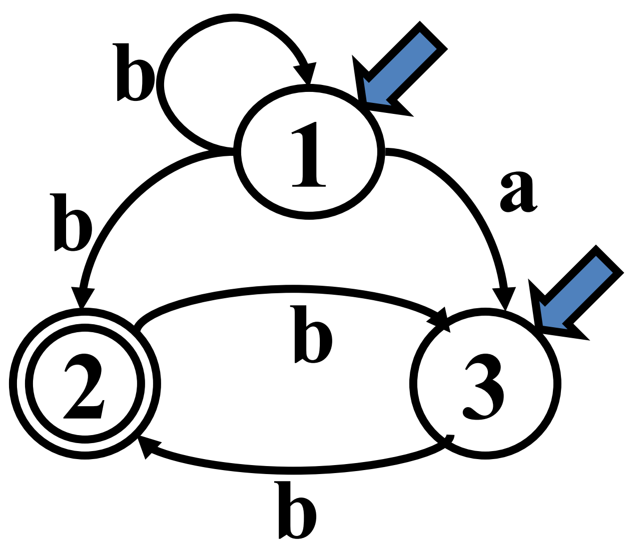
例　设有NFA M=({1,2,3},{a,b},f,{1,3},{2})

其中 f (1, a) = {3}　　f (1 , b) = {1,2}

f (2, a) = Φ　　f (2 , b) = {3}

f (3, a) = Φ　　f (3 , b) = {2}

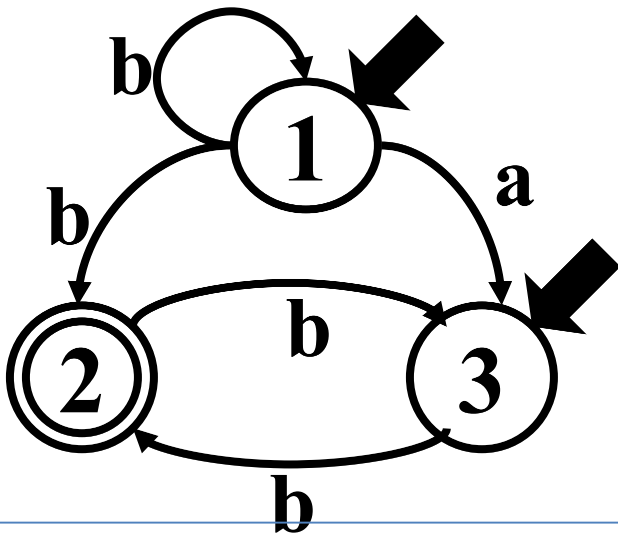
例中 NFA M′ 对应的状态转换图如下图所示。



3.4.2 非确定有穷自动机

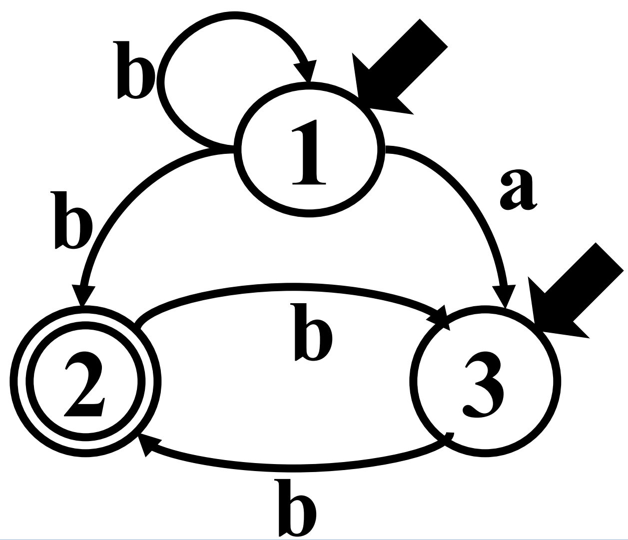
例中NFA M' 所识别的语言为

$L(M') = b^*(b|ab)(bb)^*$



3.4.2 非确定有穷自动机

NFA中，同一个符号串 β可由多条路来识别。
例中 NFA M' , 符号串β =bbb可由3条路来识别。



①：状态1→状态2→状态3→状态2;

②：状态1→状态1→状态1→状态2;

③：状态3→状态2→状态3→状态2;

3.4.2 非确定有穷自动机

提问：DFA是NFA 的特例？
提问：NFA是否比DFA描述能力更强呢？

对于每个NFA M 是否存在 DFA M' ,使 L(M) = L(M')？

是特例
不，两者是相当的

3.4.2 非确定有穷自动机

对于每个NFA M 存在 DFA M’,使 L(M) = L(M’)。

利用有穷自动机构造词法分析程序:

1. 从语言单词的描述中构造出非确定的有穷自动机,
2. 再将非确定的有穷自动机转化为确定的有穷自动机,
3. 并将其化简为状态最少化的**DFA**,
4. 然后对**DFA**的每一个状态构造一小段程序将其转化为识别语言单词的词法分析程序。

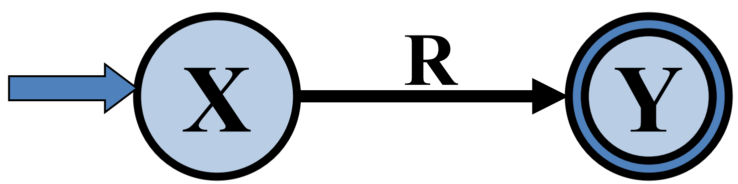
3.4.3 由正规式R构造NFA

输入：字母表Σ上的正规式R

输出：识别(接受)语言L(R)的NFA N

方法：

1. 引进初始结点X和终止结点Y

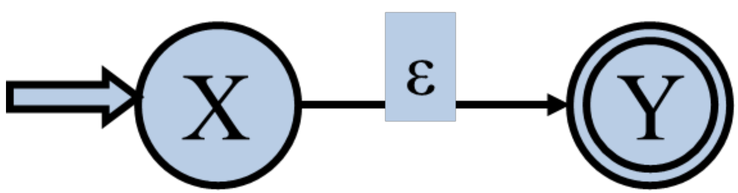


2. 分析R的语法结构，用规则（1）-（4）对R中的每个基本符号构造NFA。

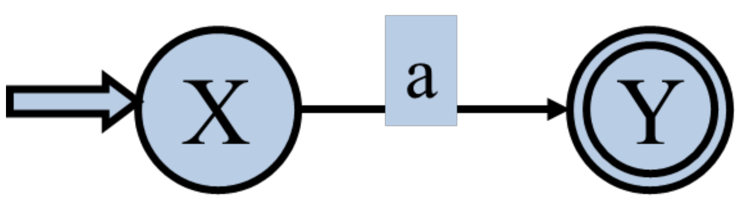
3.4.3 由正规式R构造NFA

(1)　　R=Φ，构造NFA如图所示。

(2)　　R= ε　，构造NFA如图所示。

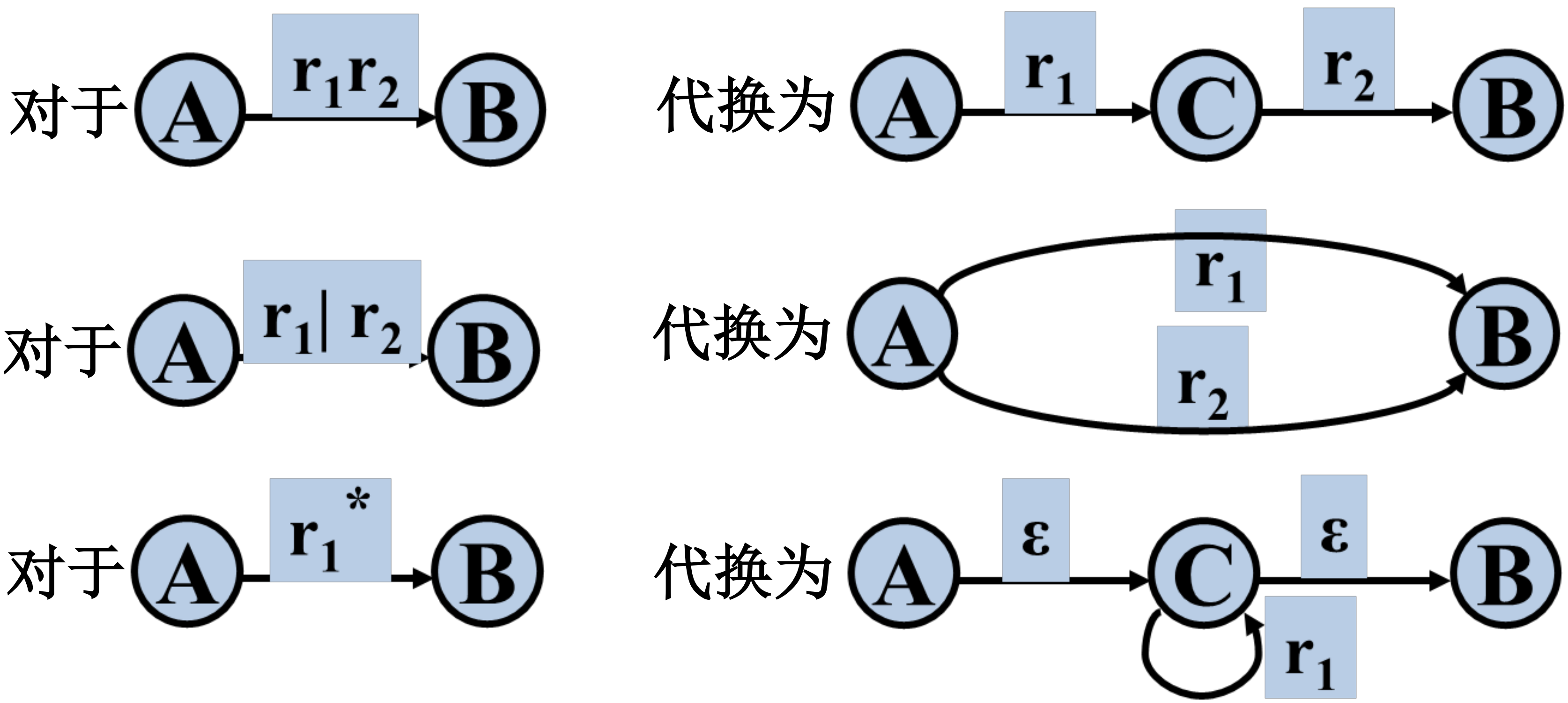


(3)　　R= a（a∈Σ），构造NFA如图所示。



3.4.3 由正规式R构造NFA

(4) 若R是复合正规式，则按下图的转换规则对R进行分裂和加进新结，直至每个边上只留下一个符号或 ε 为止。



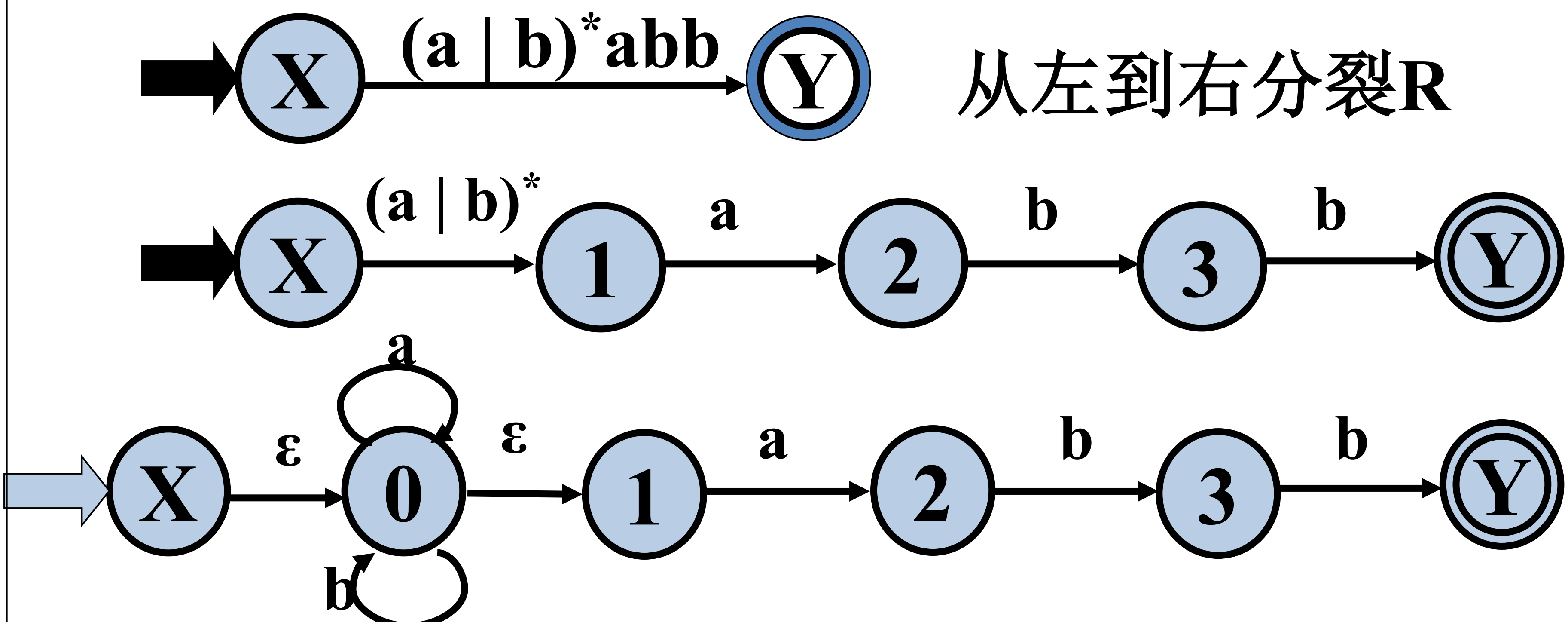
3.4.3 由正规式R构造NFA

3. 整个分裂过程中，所有新节点均采用不同的名字，保留X，Y为全图唯一初态结和终态结。

3.4.3 由正规式R构造NFA

例1 试构造识别语言$R=(a \mid b)^*abb$ 的NFA N, 使$L(N)=L(R)$。

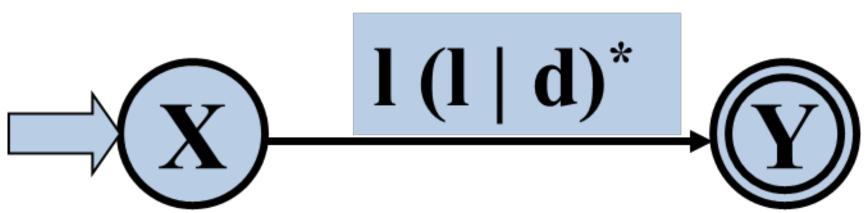
首先将R表示成如下拓广转换图



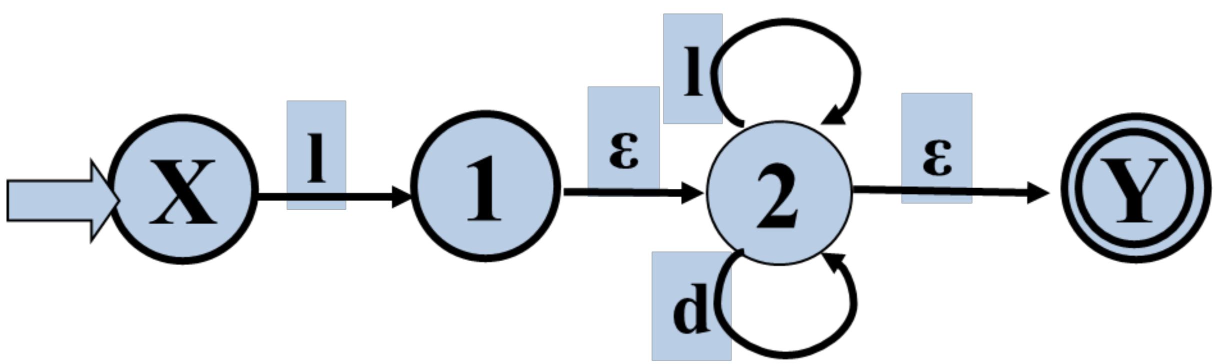
3.4.3 由正规式R构造NFA

例2 试构造识别标识符的NFA，描述标识符的正规式R= l (l | d)*

首先将R表示成如下拓广转换图



从左到右分裂R



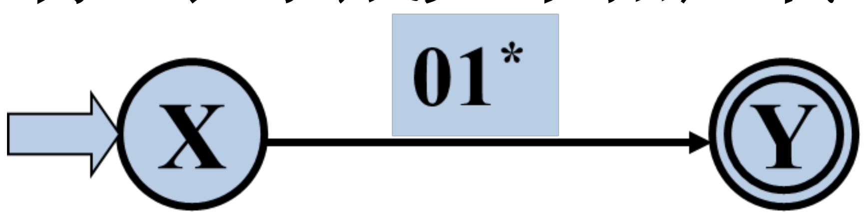
3.4.3 由正规式R构造NFA

例3 试构造正规式 $R= 0 (1^*)^* \mid 01$ 的NFA。

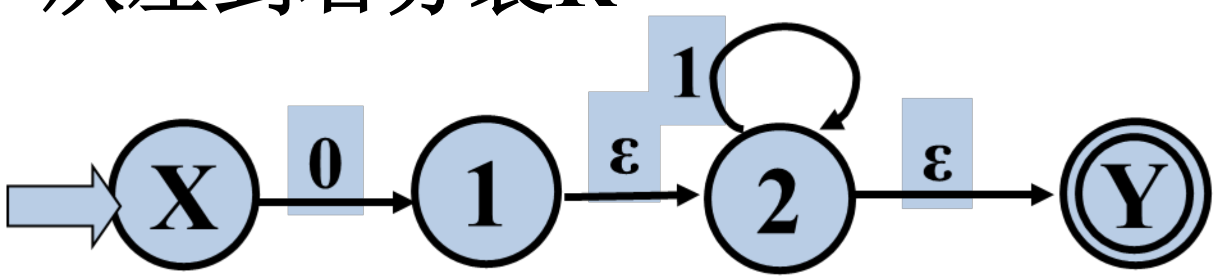
首先利用正规式的等价性化简正规式

$\because (1^*)^*=1^* \quad \therefore R=01^* \mid 01=0 (1^* \mid 1) =01^*$

将R表示成如下拓广转换图



从左到右分裂R



3.4.4 NFA确定化为DFA的方法

对于NFA，状态转换函数 f 是一个多值函数

$$f(q, a)=\{q_1, q_2, q_3 \cdots, q_n\}$$

为将NFA转换为DFA，将状态集合 $\{q_1, q_2, q_3 \cdots, q_n\}$ 看作一个状态A。

3.4.4 NFA确定化为DFA的方法

输入：一个**NFA N**

输出：一个接受（识别）相同语言的**DFA M**

1. 状态集合 I 的 ε-闭包:

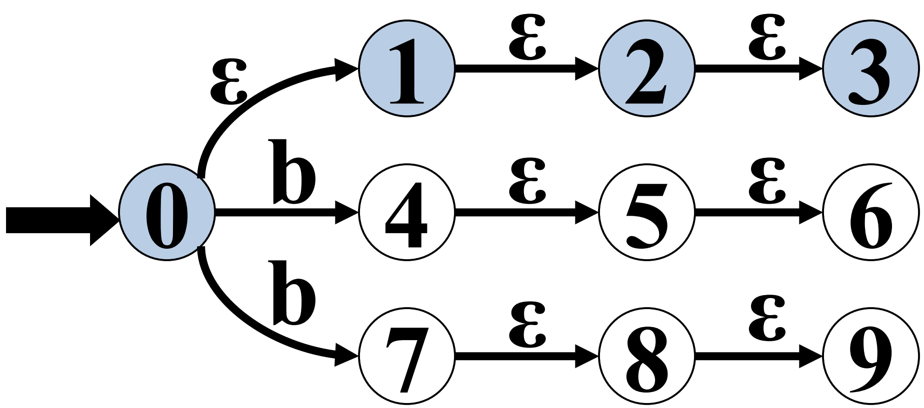
设I是NFA N的一个状态子集,
ε-closure(I)(或ε-闭包）定义如下：

(1) 若s∈I , 则 s∈ε-closure(I)

(2) 若s∈I，从s出发经过任意条ε弧能到达的状态 s’,
则s’∈ε-closure(I)

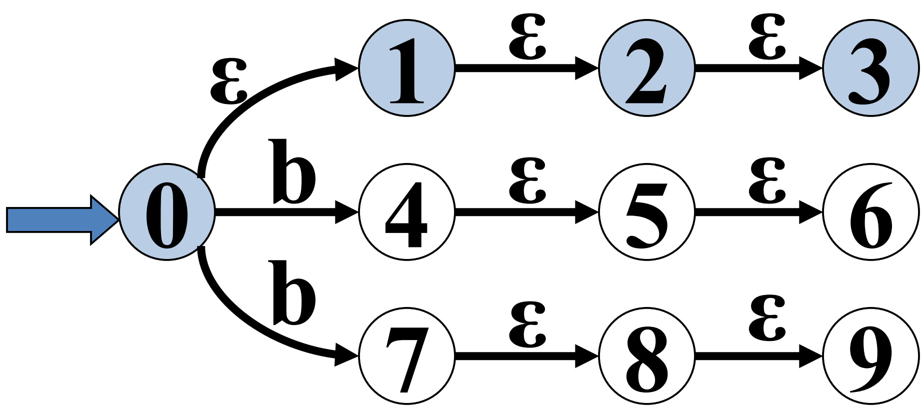
3.4.4 NFA确定化为DFA的方法

例子：状态集合 I={0}，I的 ε-闭包如下：



ε–closure({0})={0,1,2,3}

3.4.4 NFA确定化为DFA的方法



若令A={0,1,2,3},则

$J=f(A,b)=f(0,b)\cup f(1,b)\cup f(2,b)\cup f(3,b)=\{4,7\}$

令B= ε–closure({4,7})={4,5,6,7,8,9}

则定义B是DFA在状态A下遇到符号b后转移到的状态。

2. 从 NFA N 构造 DFA M 的算法

已知 NFA N=(Q, ∑, f, x, {y})

计算 DFA M=(Q', ∑, f ', 初态, 终态集)

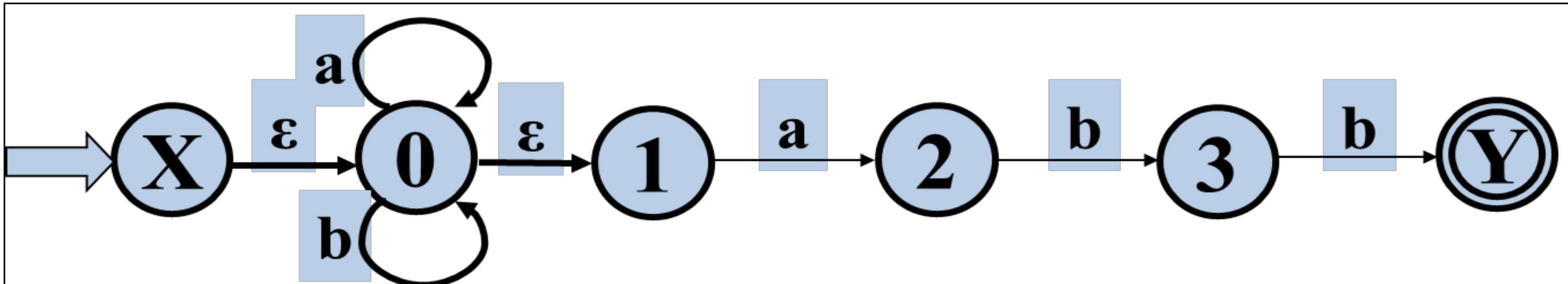
开始令Q'={ }

3.4.4 NFA确定化为DFA的方法

```
计算DFA M的初态ε–closure({x}),并置无标记送入Q';
while ( Q'中存在一个无标记的状态 T={ s1, s2, s3, …, sn} )
{  标记T;
   for  ( 每个输入符号a )                /* 求解 f '( T, a )=U */
    { J = f ({s1, s2, s3, …, sn},a )
          = f ( s1, a )∪f ( s2 , a )∪… ∪f ( sn, a );
      U = ε–closure( J );
      if (U不在Q'中 && U不为空) 置U为无标记并送入Q';
      if (U不为空) 置M[T, a ]=U;
      if (U中至少有一个是N的终态) U为M的终态;
    }//end of for
 }//end of while
```

3.4.4 NFA确定化为DFA的方法

例1 将下图所示的NFA N确定化。



其等价DFA的开始状态:

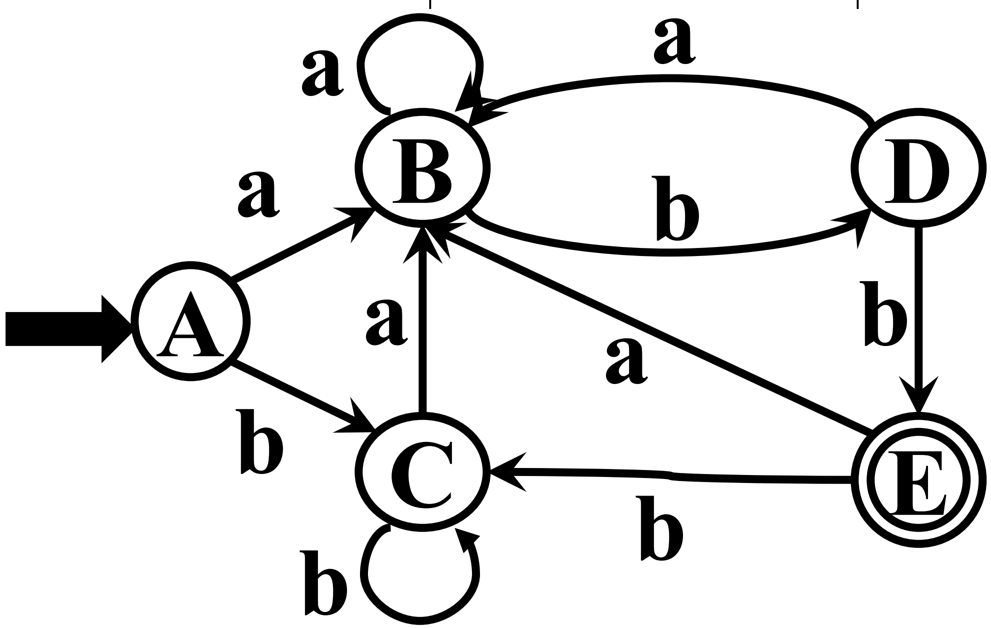
A=ε–closure({X})
 ={X, 0, 1}

A作为无标记,加入Q'

	Q' \ 符号	a	b
A	{ X, 0, 1 }	{ 0, 1, 2 }	{ 0, 1 }
B	{ 0, 1, 2 }	{ 0 ,1, 2 }	{ 0, 1, 3 }
C	{ 0, 1 }	{ 0, 1, 2 }	{ 0, 1 }
D	{ 0, 1, 3 }	{ 0, 1, 2 }	{ 0, 1, Y }
E	{ 0, 1, Y }	{ 0, 1, 2 }	{ 0, 1 }

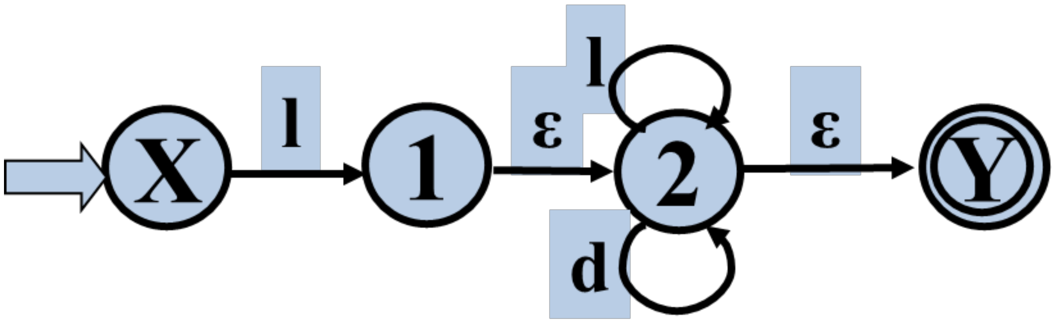
3.4.4 NFA确定化为DFA的方法

	Q′	a	b
A	{ X, 0, 1 }	{ 0, 1, 2 }	{ 0, 1 }
B	{ 0, 1, 2 }	{ 0 ,1, 2 }	{ 0, 1, 3 }
C	{ 0, 1 }	{ 0, 1, 2 }	{ 0, 1 }
D	{ 0, 1, 3 }	{ 0, 1, 2 }	{ 0, 1, Y }
E	{ 0, 1, Y }	{ 0, 1, 2 }	{ 0, 1 }



3.4.4 NFA确定化为DFA的方法

例2 将下面的NFA N确定化。



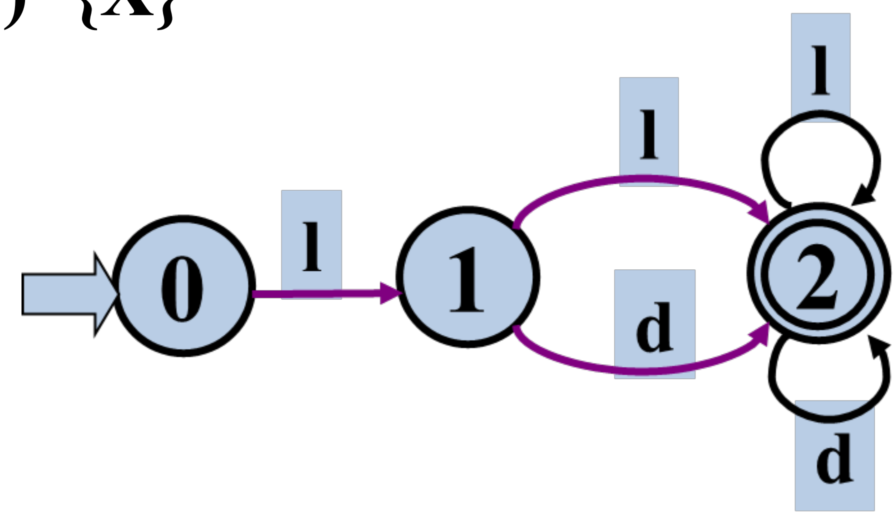
1. 确定其初态，命名为0状态

$$0 = \varepsilon\text{–CLOSURE}(\{X\}) = \{X\}$$

2. 计算状态转换矩阵

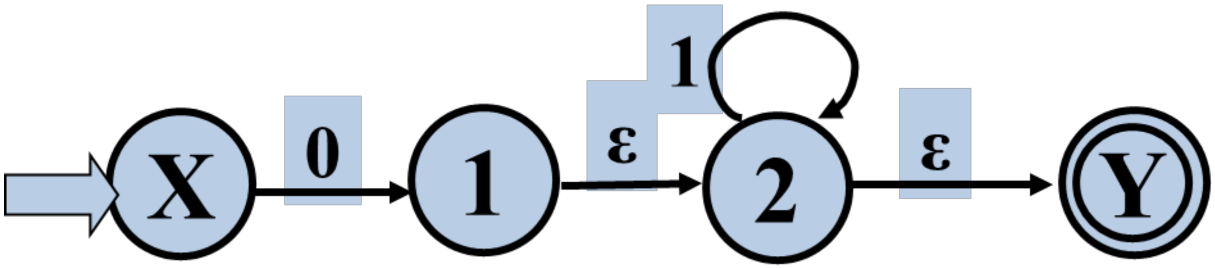
	Q′	l	d
0	{X}	{1,2,Y}	Φ
1	{1,2,Y}	{2,Y}	{2,Y}
2	{2,Y}	{2,Y}	{2,Y}

3. 画出DFA



3.4.4 NFA确定化为DFA的方法

例3 将下面的NFA N确定化。



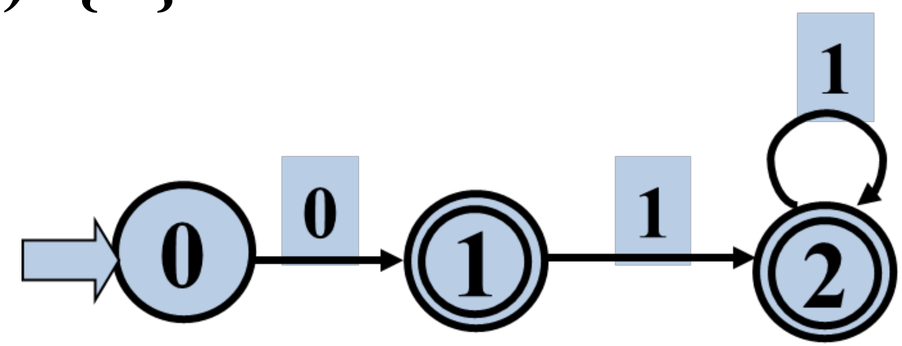
首先确定其初态，命名为0状态

$0 = \varepsilon\text{–CLOSURE}(\{X\}) = \{X\}$

计算状态转换矩阵

	Q′	0	1
0	{X}	{1,2,Y}	Φ
1	{1,2,Y}	Φ	{2,Y}
2	{2,Y}	Φ	{2,Y}

画出DFA



3.4.5 DFA的化简

1. DFA的化简

问题：能否寻找一个状态数比 M 少的 DFA M'，使 L(M)=L(M')？

进一步问题：
还有没有更少的？如何证明是最少的？

化简的DFA满足两个条件:
(1) 无多余状态;
(2) 无等价状态;

3.4.5 DFA的化简

2. 多余状态

有穷自动机的多余状态：从该自动机的开始状态出发，经任何输入串也不能到达的状态。

3.4.5 DFA的化简

3.等价状态

设 DFA $M=(Q, \Sigma, f, S, F)$, $s, t\in Q$，若对任何 $\alpha\in\Sigma^*$, $f(s,\alpha)\in F$ 当且仅当 $f(t,\alpha)\in F$，则 s 和 t 是等价状态。

如果 s 和 t 不等价, 则称 s 和 t 可区别。

注意：终态与非终态是可区别的。

原因：终态有一条到达自身的ε道路，而非终态没有到达终态的ε道路。

3.4.5 DFA的化简

4. 两个状态等价的条件:

(1) 一致性条件：状态s和t必须同时为终态或非终态。

(2) 蔓延性条件：对于所有输入符号a，状态 s 和 t 必须转到等价的状态里。

5. 化简（最小化）方法

输入：一个DFA M 。

输出：接受与M相同语言的DFA M'，且其状态数最少。

3.4.5 DFA的化简

- 无多余状态下，将M的状态集 Q 分划成不相交的子集，使每个子集中任何两个状态是等价的；而任何两个属于不同子集的状态，都是可区别的。

- 在每个子集中，任取一个状态作“代表”，删去子集中其余状态，并把指向其余状态的箭弧都改作指向 “代表”的状态。

3.4.5 DFA的化简

化简算法（子集法）：

1. 将DFA M的状态集Q分成终态集F和非终态集¬F，形成分划∏=(F, ¬F)。

2. 对∏使用如下方法构造新分划$\prod_{NEW}$：

对∏中的每个状态子集G:

(1) 把G分划成新的子集，使得G的两个状态s和t属于同一子集，当且仅当对任何输入符号a，状态s和t转换到的状态都属于∏的同一子集。

(2)用G分划出的所有新子集替换G，形成新的分划∏NEW；

3.4.5 DFA的化简

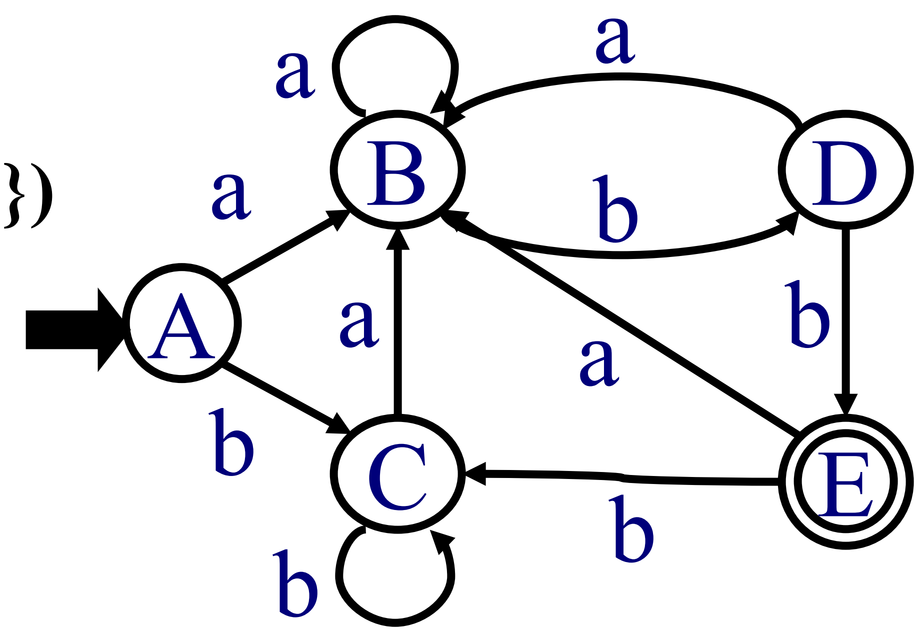
3. 如果Π_{NEW}==Π，则执行第4步；否则令$\Pi = \Pi_{NEW}$，重复第2步。

4. 分划结束，对分划中的每个状态子集G，选出一个状态作代表，删去其它等价的状态，并把指向其它状态的箭弧改为指向作为代表的状态。

例1. 将右面的DFA最小化

分析　由图可知，给定的DFA中无多余状态。

初始分划∏=({E} {A,B,C,D})



$\{A,B,C,D\}_a=\{B\}$

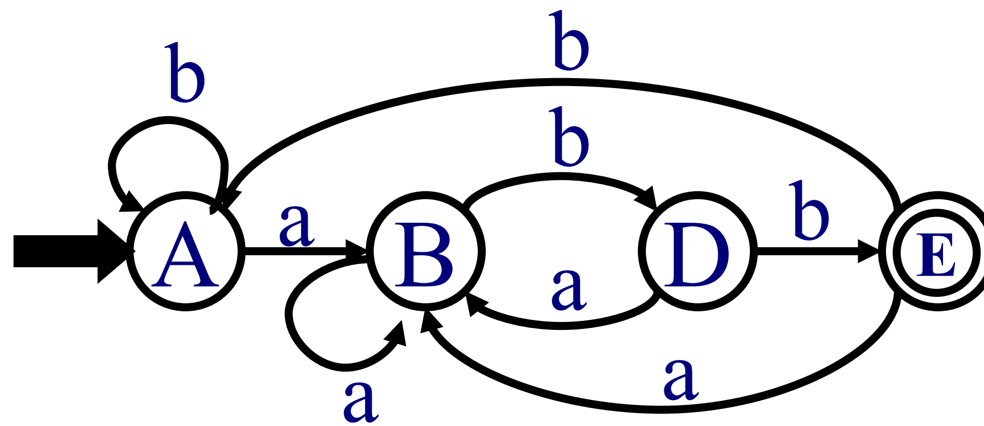
$\{A,B,C,D\}_b=\{C,D,E\}$

∏=({E}{A,B,C}{D})

$\{A,B,C\}_a=\{B\}$

$\{A,B,C\}_b=\{C,D\}$

∏=({E}{A,C}{B}{D})



$\{A,C\}_a=\{B\}$

$\{A,C\}_b=\{C\}$

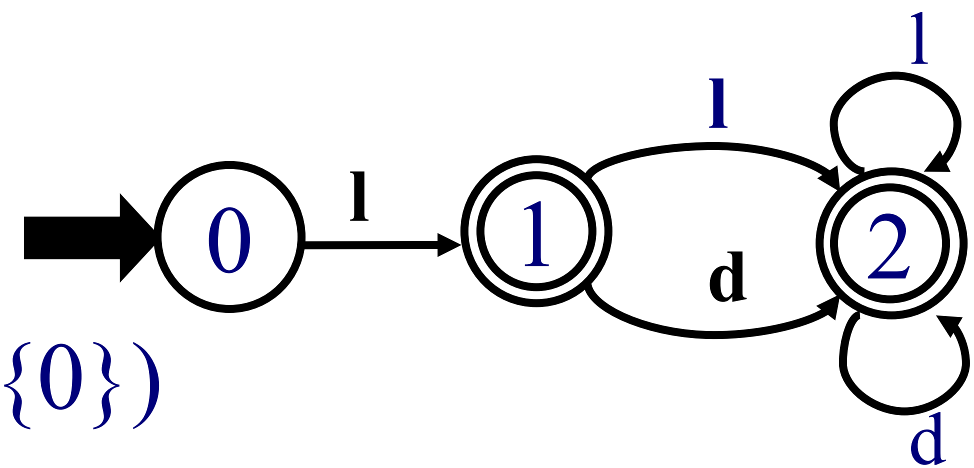
$\prod_{new}$=({E}{A,C}{B}{D})

$\prod_{new}$ == ∏，A,C等价，保留……

3.4.5 DFA的化简

例2. 将DFA M最小化

分析 由图可知，给定的DFA无多余状态。



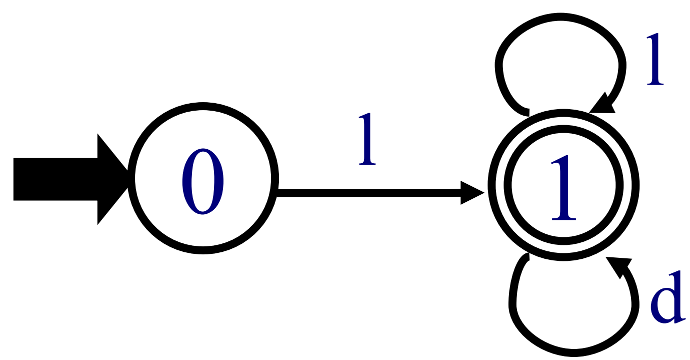
初始分划$\Pi=(\{1,2\}\{0\})$

$\{1,2\}_1=\{2\}$

$\{1,2\}_d=\{2\}$

$\Pi_{new}=(\{1,2\}\{0\})$

$\Pi_{new}==\Pi$，状态1,2等价，保留1得：



3.4.6 有穷自动机到正规式

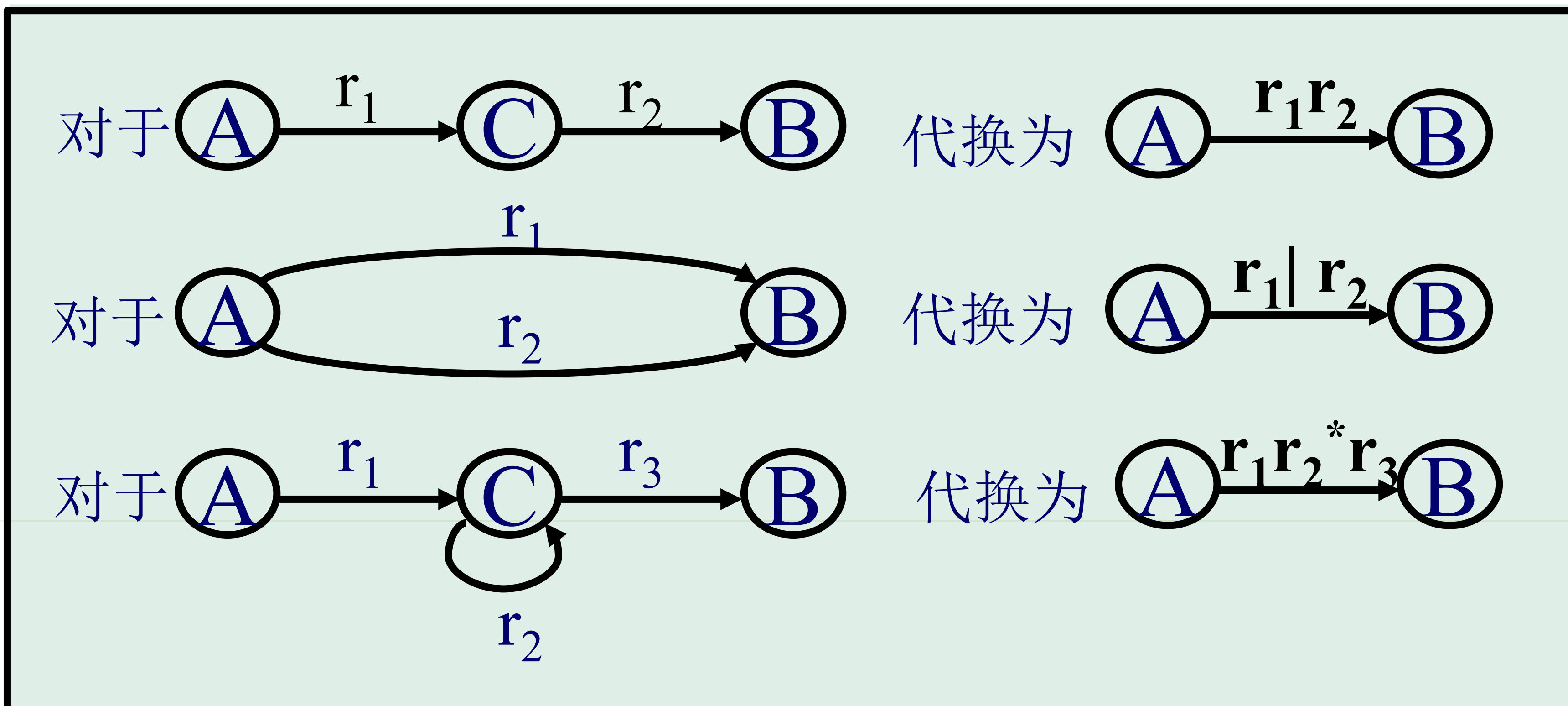
算法：有穷自动机M转正规式R

1. 在M的转换图上添加两个结点X和Y；从X用ε连接到M的所有初态结点，从 M的所有终态结点用ε连线连结到Y；构成新的非确定有穷自动机M’，具有唯一的初态X和终态Y。显然，L(M)=L(M’)。（两个NFA是等价的）

3.4.6 有穷自动机到正规式

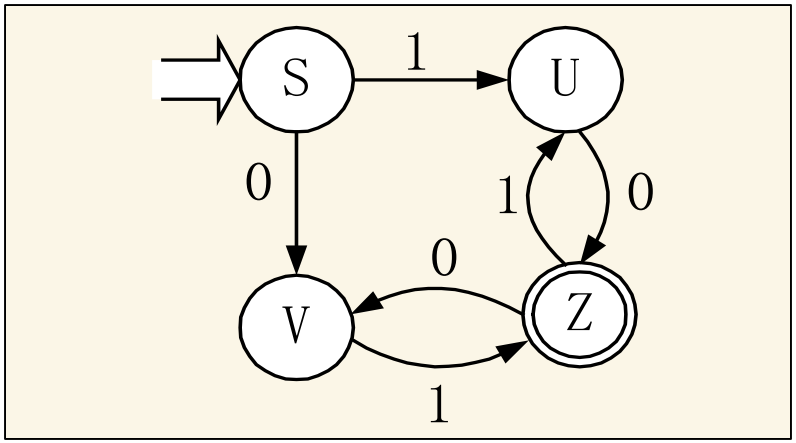
2.逐步消去M’中的其它结点，直至只剩下X，Y结点。在消除结点过程中，逐步用正规式来标记相应的箭弧。

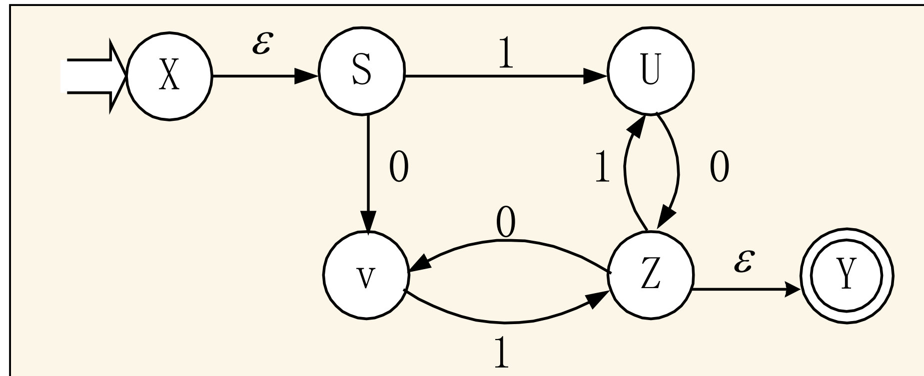
消去规则：



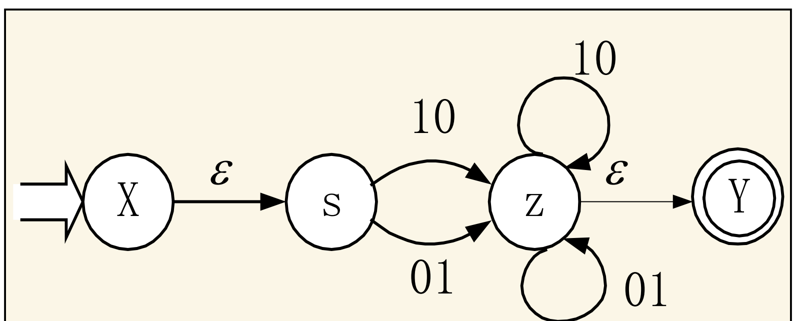
3.4.6 有穷自动机到正规式

例1. 设有穷自动机的状态图如图所示。试求该自动机识别语言的正规式。

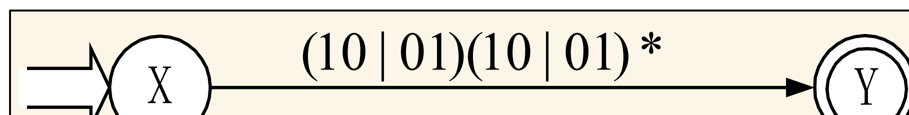




(a)



(b)



(c)

$$R=(10|01)(10|01)^*$$

算法：右线性正规文法G转有穷自动机M

给定右线性正规文法$G=(V_N, V_T, P, S)$

$A \to aB$
$A \to a$

构造相应有穷自动穷自动机
$M=(Q, \Sigma, f, q_0, Z)$

1. 令$Q=V_N \cup \{D\}$ $(D \notin V_N)$
$Z=\{D\}$ $\Sigma=V_T$ $q_0=S$

$a=\varepsilon$ $A \to B$
令$f(A, \varepsilon)=B$

2. 对G中每一形如
$A \to aB$ $(A, B \in V_N, a \in V_T \cup \{\varepsilon\})$
的产生式，令 $f(A, a)=B$

3.5.1 右线性正规文法到有穷自动机

3. 对G中每一形如$A \rightarrow a(A \in V_N, a \in V_T)$的产生式，令 $f(A, a)=D$

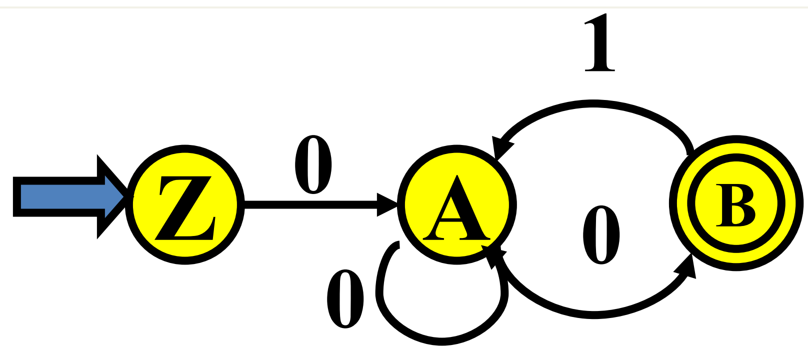
4. 对G中每一形如$A \rightarrow \varepsilon$ $(A \in V_N)$的产生式，令A为接受状态或令 $f(A, \varepsilon)=D$

例1 构造下述文法G[Z]的有穷自动机。

Z→0A

A→0A | 0B

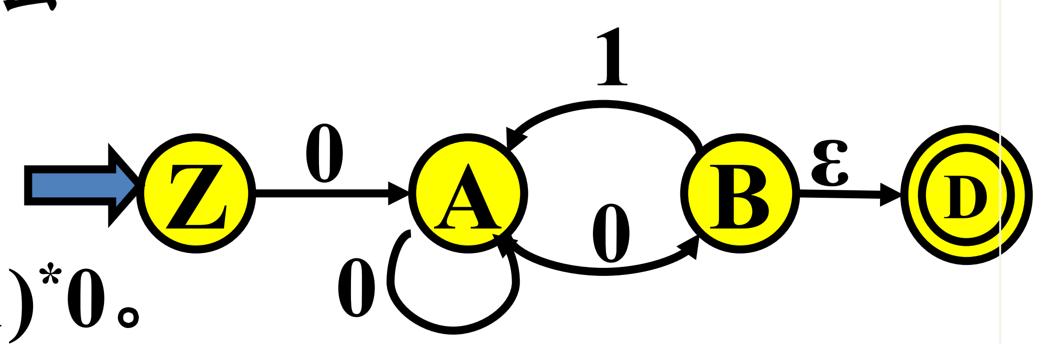
B→1A | ε



(a)

其状态图如图(a)或(b)所示。

显然,自动机M是非确定的。
它识别的语言就是文法
G[Z]所描述的语言即



(b)

$L(G[Z])=L(M)=0(0|01)^*0$。

3.5.2 左线性正规文法到有穷自动机

给定左线性正规文法

$G = (V_N, V_T, P, S)$

$A \to Ba$

$A \to a$

则相应的有穷自穷自动机

$$M = (Q, \Sigma, f, q_0, Z)$$

1. 令 $Q = V_N \cup \{q_0\}$ $(q_0 \notin V_N)$

$Z = \{S\}$ $\Sigma = V_T$

2. 对G中每一形如

$A \to Ba$ $(A, B \in V_N, a \in V_T \cup \{\varepsilon\})$

的产生式, 令 $f(B, a) = A$

a= ε $A \to B$

令 $f(B, \varepsilon) = A$

3. 对G中每一形如

$A \to a$ ($A \in V_N$, $a \in V_T \cup \{\varepsilon\}$)

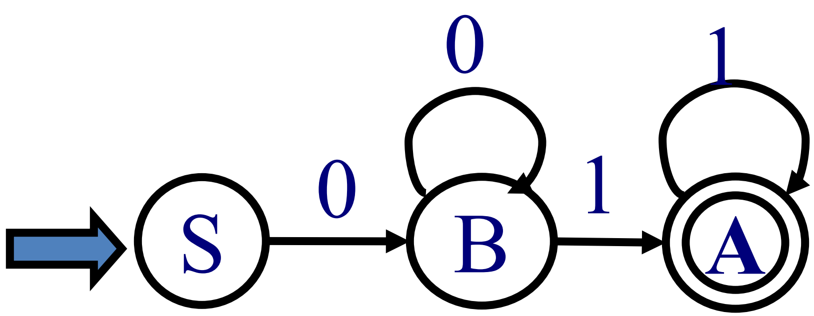
的产生式, 令 $f(q_0, a) = A$

例1. 构造下述文法G[A]的自动机。

A→A1 | B1

B→B0 | 0

构造自动机状态图如下图所示:



显然,该自动机是确定的。

识别的语言就是文法G[A]所描述的语言。

即 $L(G[A])=L(M)=00^{*}11^{*}$

3.5.3 有穷自动机到正规文法

设有穷自动机 $M = (Q, \Sigma, f, q_0, Z)$

则对应的正规文法 $G = (V_N, V_T, S, P)$

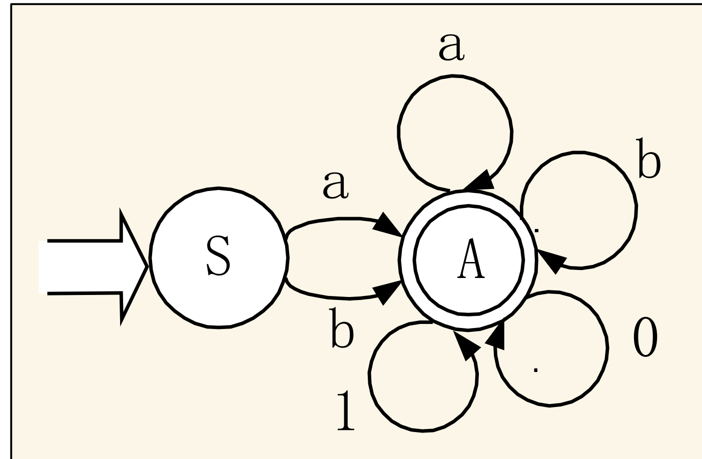
1. 令 $V_N = Q$, $V_T = \Sigma$, $S = q_0$;
2. 若 $f(A,a)=B$ 且 $B \notin Z$ 时，则将产生式 $A \rightarrow aB$ 加到P中;
3. 若 $f(A,a)=B$ 且 $B \in Z$ 时，则将产生式 $A \rightarrow aB \mid a$ 或将产生式 $A \rightarrow aB$、$B \rightarrow \varepsilon$ 加到P中;
4. 若文法的开始符号S是一个终态，则将产生式 $S \rightarrow \varepsilon$ 加到P中。

例1 设有穷自动机 M=({S,A},{a,b,0,1} ,f , S , {A})

其中 f(S,a)=A　　f(S,b)=A

f(A,a)=A　　f(A,b)=A

f(A,0)=A　　f(A,1)=A



M的状态转换图如图所示，构造与M等价的正规文法G。

G=({S,A},{a,b,0,1},P,S)

其中P: S→aA | bA　　A→aA | bA | 0A | 1A | ε

或P: S→aA | bA　　A→aA | bA| 0A | 1A

| a | b　　| a | b | 0 | 1

自动机M所识别的语言L(M)=L(G)=(a|b)(0|1|a|b)*。

例2 设DFA M=({A,B,C,D},{0,1}, δ, A,{B})

其中： δ(A,0)=B δ(A,1)=D

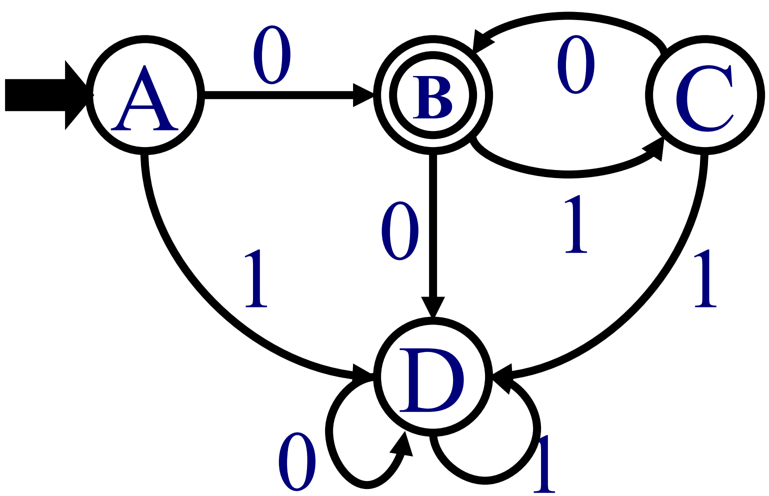
δ(B,0)=D δ(B,1)=C

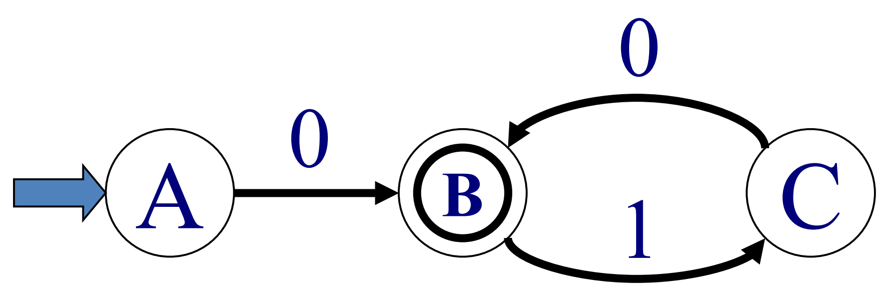
δ(C,0)=B δ(C,1)=D

δ(D,0)=D δ(D,1)=D

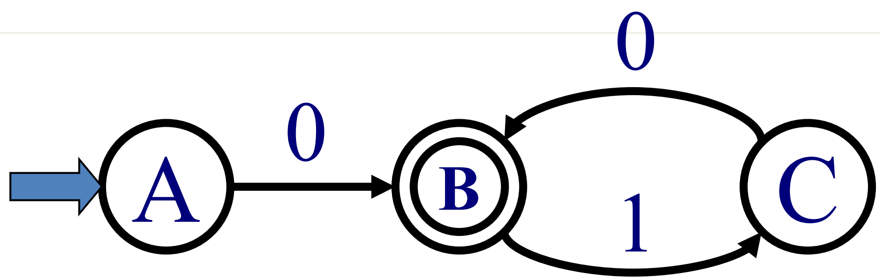
构造一个右线性文法G,使得L(G)=L(M)。

该自动机相应的状态转换图如下图所示。





3.5.3 有穷自动机到正规文法



根据转换规则所求右线性文法为

G=({A,B,C},{0,1}, P, A)其中P为

$A \to 0B \mid 0$		$A \to 0B$
$B \to 1C$	或	$B \to 1C \mid \varepsilon$
$C \to 0B \mid 0$		$C \to 0B$

该自动机所识别的语言为 0(10)*。

3.6 词法分析程序的编写

方法:

第一种：手工方式，根据识别语言单词的状态转换图，使用某种高级语言（如**C**语言），直接编写词法分析程序。

第二种：利用词法分析程序的自动生成工具**Lex/Flex**自动生成词法分析程序。

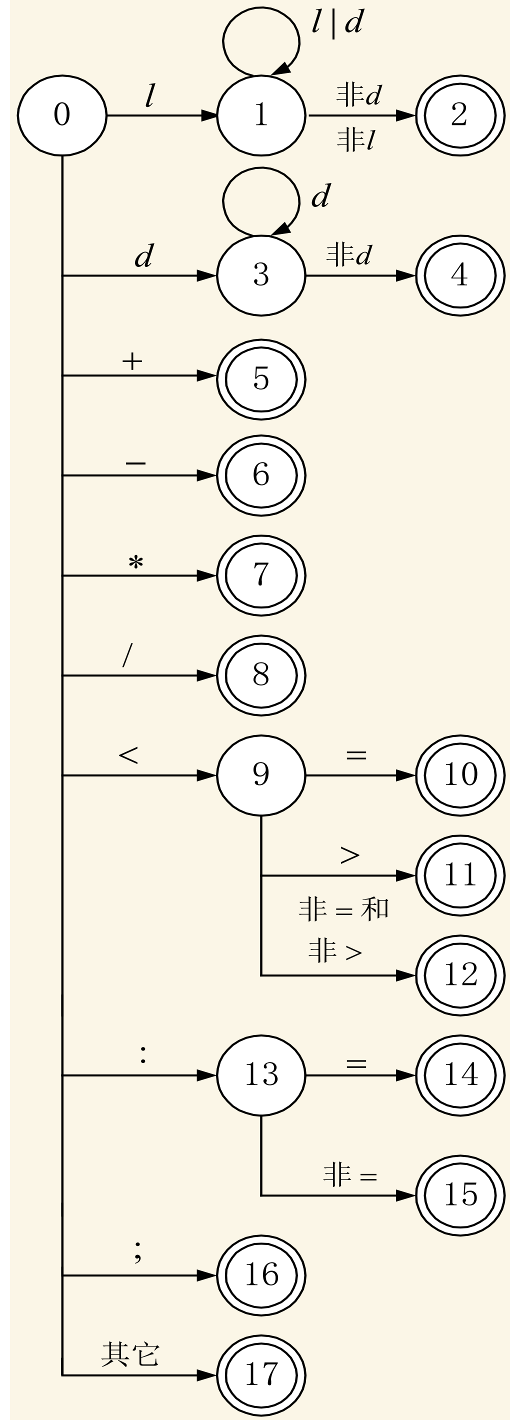
例子：下表列出了某个简单语言的所有单词符号，以及它们的种别编码和单词值。

单词符号	种别编码	单词值
begin	1	
end	2	
if	3	
then	4	
else	5	
while	6	
do	7	
标识符	10	内部字符串
整常数	11	二进制数值
+	13	
-	14	
*	15	
/	16	
<=	17	
<>	18	
<	19	
:	21	
:=	22	
;	23	

右图是识别前表单词符号的状态转换图：

图中，状态0为初态，带双圈均为终态；状态17是识别不出单词符号的出错情况。l代表任一字母，d代表任一数字。

根据转换图，用C语言直接编写出识别该语言所有单词的词法分析程序。



3.6 词法分析程序的编写方法

规定1：所有关键字，不得用作自定义标识符，（关键字作为特殊标识符处理，预先安排在关键字表中）。识别到标识符时，查关键字表，判断是否为关键字。

规定2：若关键字、标识符和常数间没有运算符、界符间隔，则必须至少用一个空白符作间隔（此时空白符有意义）。

根据状态转换图构造词法分析程序的方法:
让每个状态对应一小段程序。

全局变量、函数如下:

1.ch 字符变量，存放当前读进的源程序字符;

2. token 字符数组,存放构成单词符号的字符串;

3. getch( )读字符函数，每次从输入缓冲区中读进源程序的下一个字符放在ch中，并把读字符指针指向下一个字符;

4. getbc( )函数，调用时检查ch中的字符是否为空白字符，若是，则反复调用getbc( )，直至ch中读入一个非空白字符为止;

3.6 词法分析程序的编写方法

5. concat( )函数，每次调用把当前ch中的字符与token中的字符串联接。例如，token字符数组中原有“ab”, ch中存放着“c”,调用concat( )后，token数组中的值变为“abc”;

6. letter(ch) / degit(ch)函数，分别判定 ch 中的字符是否为字母/数字，给出true 或 false;

7. reserve( )整型函数，对token中的字符串查关键字表，若是关键字, 则送回它的编码，否则送回标识符的种别码10。

3.6 词法分析程序的编写方法

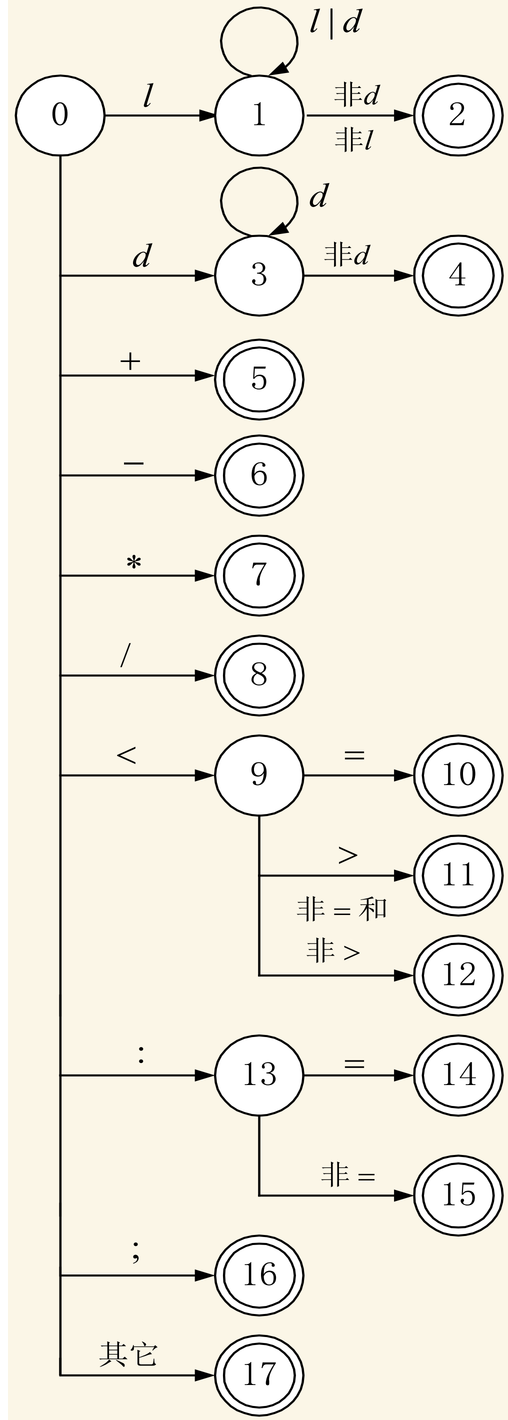
8. retract( )函数，读字符指针回退一个字符。

9. return( )函数，收集并携带必要的信息返回调用程序，即返回语法分析程序。

10. dtb( ) 十进制转换函数, 它将token中的数字串转换成二进制数值表示, 并以此作为函数值返回。

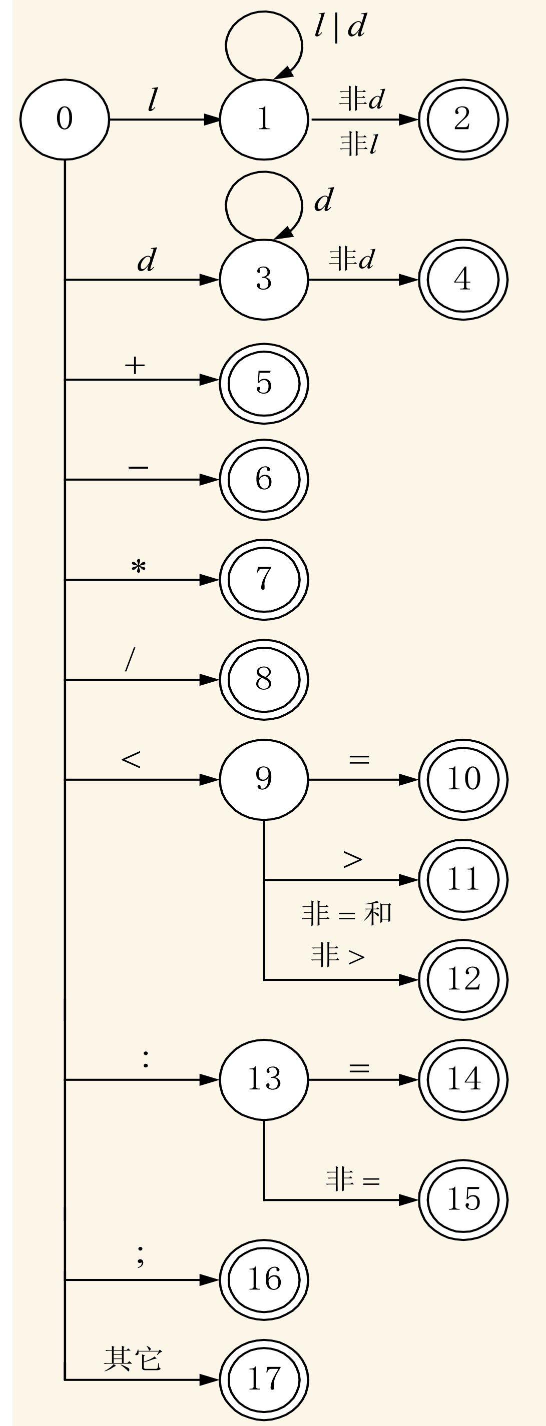
根据状态转换图用C语言编写出词法分析程序参考如下:

```
Scaner( )
{   token=NULL;
    getch( );
    getbc( );
    if  (letter(ch))
    {
      while(letter(ch) || digit(ch))
       {   concat( );
           getch( );
       }
        retract( );
        c=reserve( );
        if(c!=10)  return(c,token);
        else  return( 10,token);
     }
```

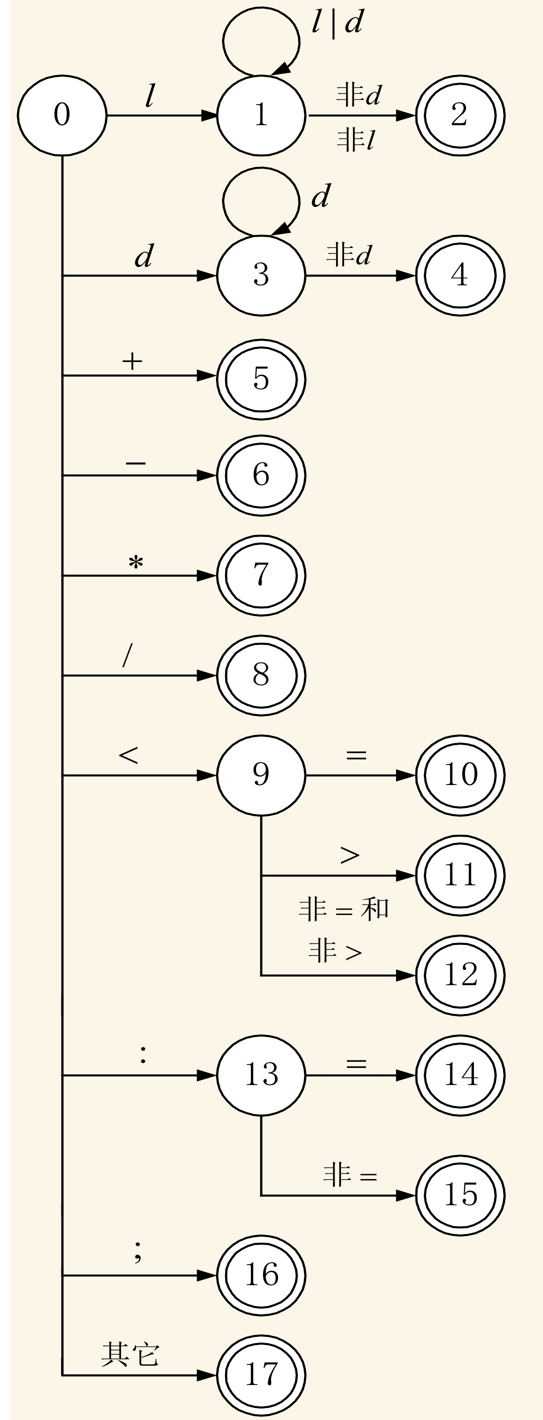


相对于状态转换图用C语言编写出词法分析程序如下:

```
else if(digit(ch))
  {
      while (digit(ch))
      {
          concat( );
          getch( );
      }
      retract( );
      return(11,dtb( ));
  }
```



```
else  switch(ch)
{  case'+': return(13, —);  break ;
   case'-': return(14, —);  break ;
   case'*': return(15, —);  break ;
   case'/': return(16, —);  break ;
   case'<':  getch( );
   if (ch= = `=´)  return(17, —);
   else if(ch= = `>´)   return(18, —);
   retract( );   return(19, —);   break;
   case': ':  getch( );
   if(ch= = '=')   return(22, —);
   retract( );   return(21, —);   break;
   case';': return(23, —);    break;
   default: error( );  break;
   }
}
```



3.6 词法分析程序的编写方法

构造出识别语言单词符号的有穷自动机，就很容易构造出识别语言单词符号的词法分析程序。

本章小结

本章重点介绍了词法分析程序的设计思想和构造方法。主要内容有：

1. 词法分析程序的功能：从左到右扫描源程序字符串，根据语言的词法规则识别出单词符号。

输出单词符号的形式是二元组：

（单词种别，单词自身值）

本章小结

2．单词符号的两种定义方式

- 正规文法
- 正规式

例：定义“标识符”单词的正规式：**l (l | d)***

正规文法： ＜标识符＞→ **l** |＜标识符＞**l**
|＜标识符＞**d**

本章小结

3．有穷自动机：确定、非确定两大类：

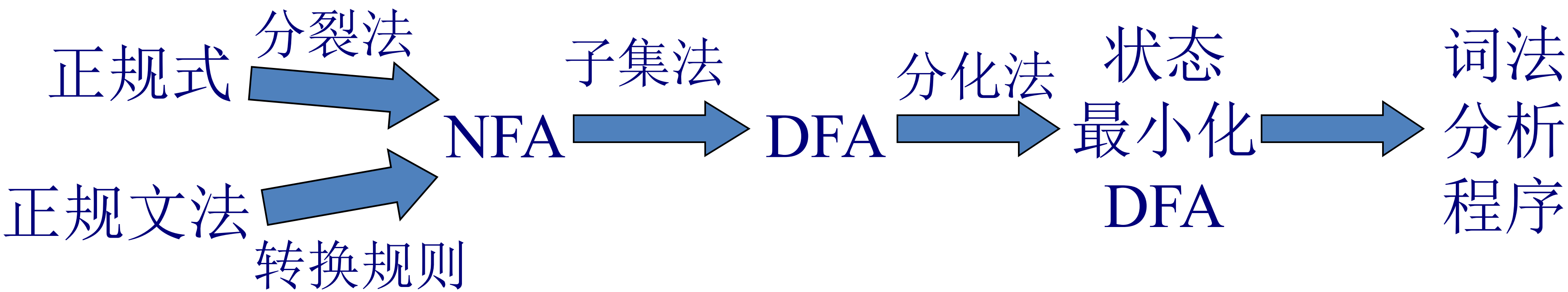
DFA M =（Q , Σ , f , S , Z ）其中是**f**单值映射函数，**S**是唯一初态

NFA N =（Q , Σ , f , S , Z ）其中**f**是多值映射函数，**S**为非空初态集。

有穷自动机通常表示为状态转换图，它是有穷自动机的非形式化描述。

本章小结

从单词两种定义方式中构造词法分析程序的过程是：



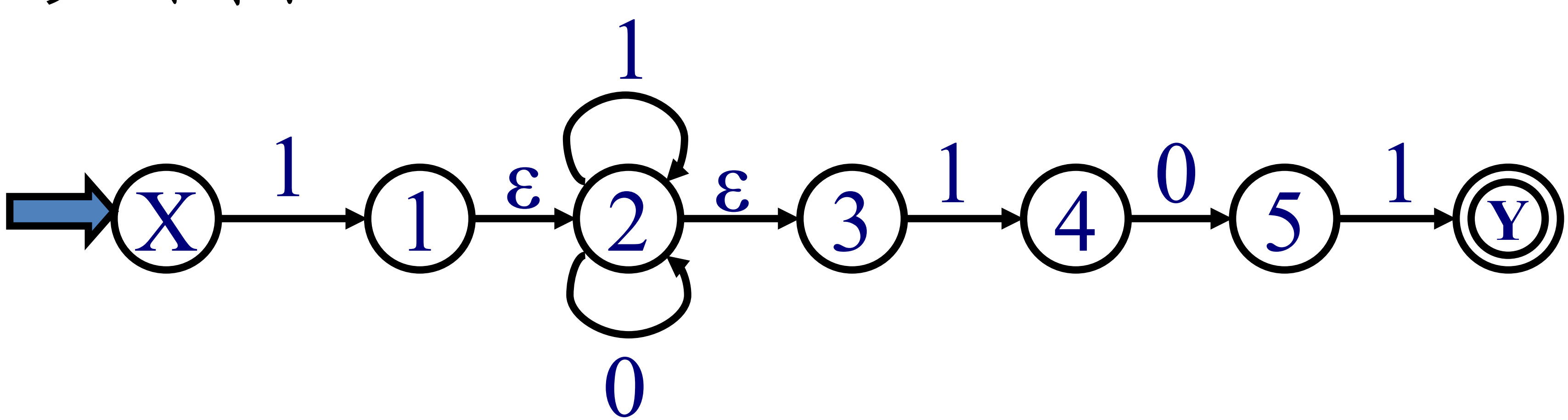
本章小结

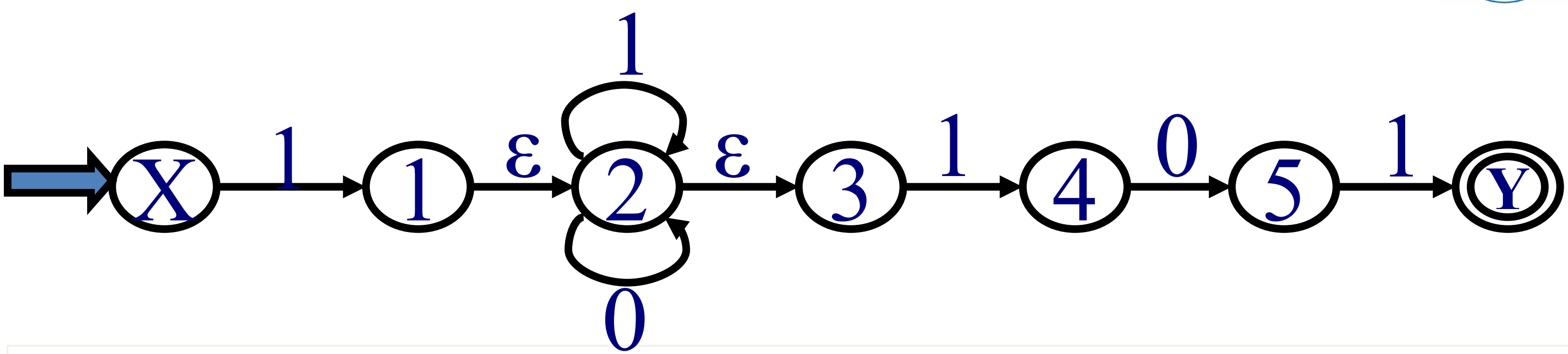
4．正规式、正规文法和有穷自动机三者都是描述正规集的工具，它们的描述能力是等价的，它们之间可相互转换。

5．证明两正规式是等价的,如果它们的最小状态DFA是相同的。也可以利用正规式的基本等价关系将一个正规式化简来证明两正规式之间的等价性或两正规式识别的语言一样。

例1 构造正规式R=1(0|1)*101的状态最小化的DFA

分析 首先对R采用分裂法构造NFA，见下图：





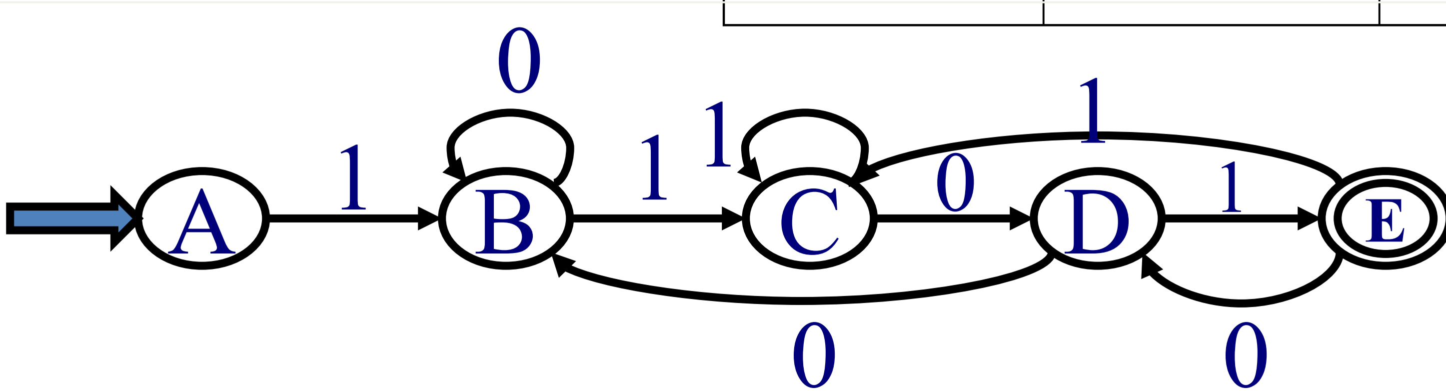
对NFA采用子集法构造其等价的DFA的状态转换矩阵，见右表

	状态 \ 字符	0	1
A	{X}	Φ	{1,2,3}
B	{1,2,3}	{2,3}	{2,3,4}
F	{2,3}	{2,3}	{2,3,4}
C	{2,3,4}	{2,3,5}	{2,3,4}
D	{2,3,5}	{2,3}	{2,3,4,Y}
E	{2,3,4,Y}	{2,3,5}	{2,3,4}

本章小结

对DFA采用分化的方法化简，得到状态最小化的DFA，见下图：

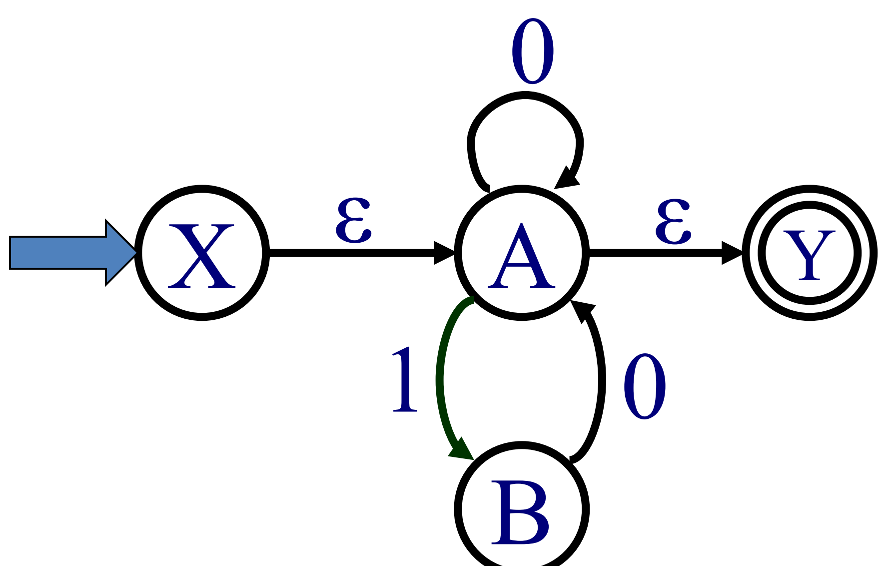
	状态 \ 字符	0	1
A	{X}	Φ	{1,2,3}
B	{1,2,3}	{2,3}	{2,3,4}
	{2,3}	{2,3}	{2,3,4}
C	{2,3,4}	{2,3,5}	{2,3,4}
D	{2,3,5}	{2,3}	{2,3,4,Y}
E	{2,3,4,Y}	{2,3,5}	{2,3,4}



例2. 构造一个DFA它接收∑={0,1}上所有满足如下条件的字符串，每个1都有0直接跟在右边。

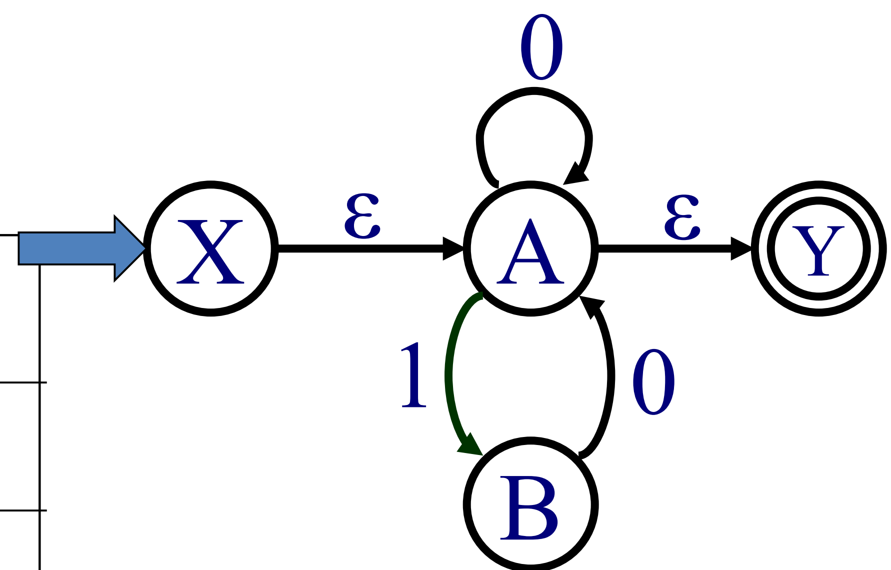
分析：给出言的正规式R=(0|10)*

分裂法从正规式构造NFA

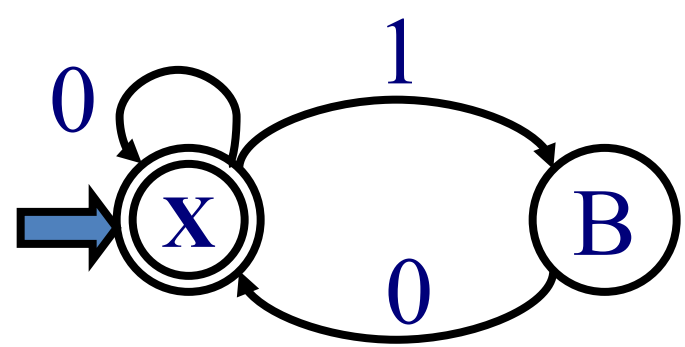


采用子集法将NFA确定化为DFA

状态 \ 字符	0	1
{X,A,Y}	{A,Y}	{B}
{A,Y}	{A,Y}	{B}
{B}	{A,Y}	Φ



采用分化方法将DFA化简

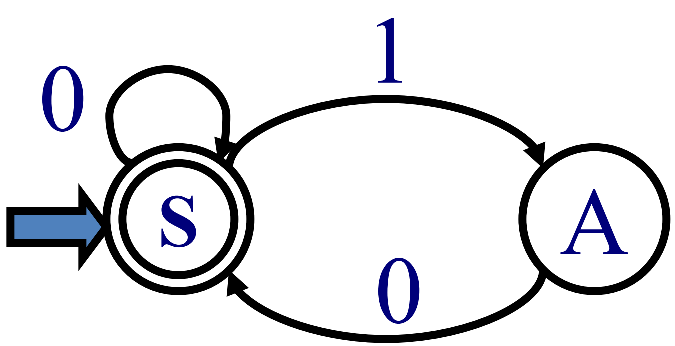


例2. 构造一个DFA它接收$\sum=\{0,1\}$上所有满足如下条件的字符串，每个1都有0直接跟在右边。

分析 给出描述语言的正规文法

$$S\rightarrow 0S \mid 1A \mid \varepsilon$$
$$A\rightarrow 0S$$

根据右线性文法构造有穷自动机的方法，构造出如下的状态转换图:



例3. 给出下述文法所对应的正规式：

$S \to 0A \mid 1B$

$A \to 1S \mid 1$

$B \to 0S \mid 0$

首先给出该正规文法对应的正规式方程组：

$S=0A+1B \quad (1)$

$A=1S+1 \quad (2)$

$B=0S+0 \quad (3)$

将(2)、(3)代入(1)得

$S=01S+01+10S+10 \quad (4)$

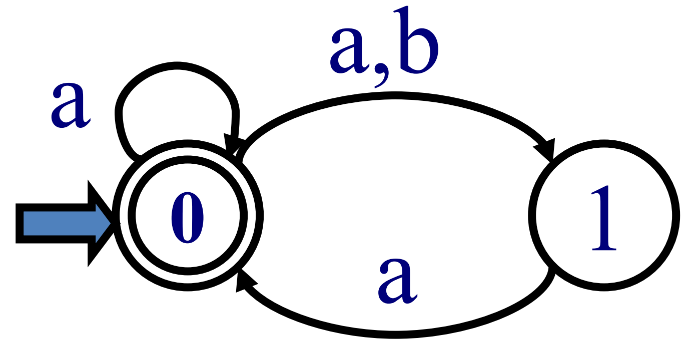
对(4)使用求解规则得

$S=(01+10)^*(01+10)$

即正规文法所生成语言的正规式是 $(01|10)^*(01|10)$。

本章小结

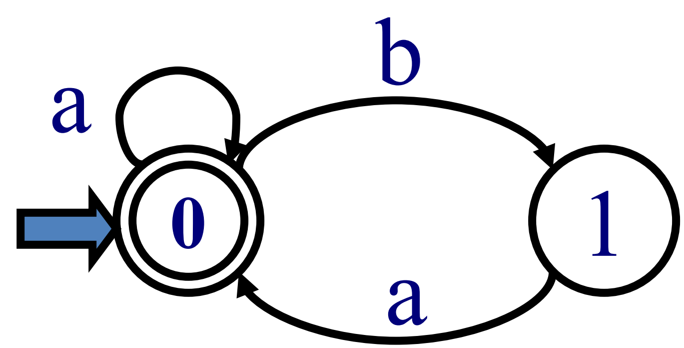
例4 将右图确定化和最小化。



图示是一个无ε 边转移的NFA,采用子集法将NFA确定化为DFA

状态＼字符	a	b
{0}	{0,1}	{1}
{0,1}	{0,1}	{1}
{1}	{0}	Φ

分化法DFA化简：



本章小结

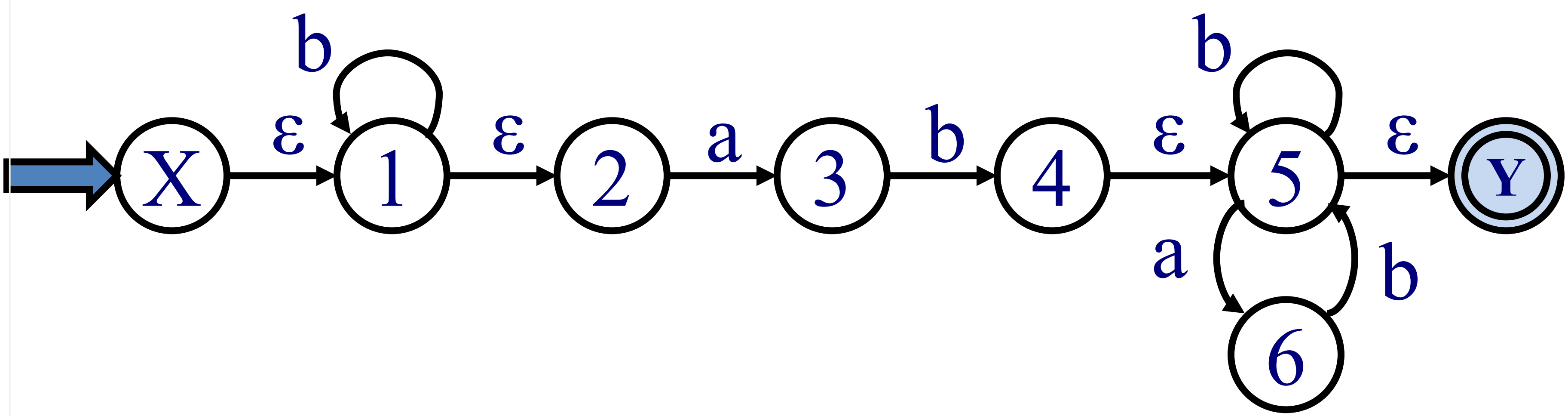
例4. 设字母表Σ={a,b}，给出Σ上的正规式 R=b*ab(b|ab)*

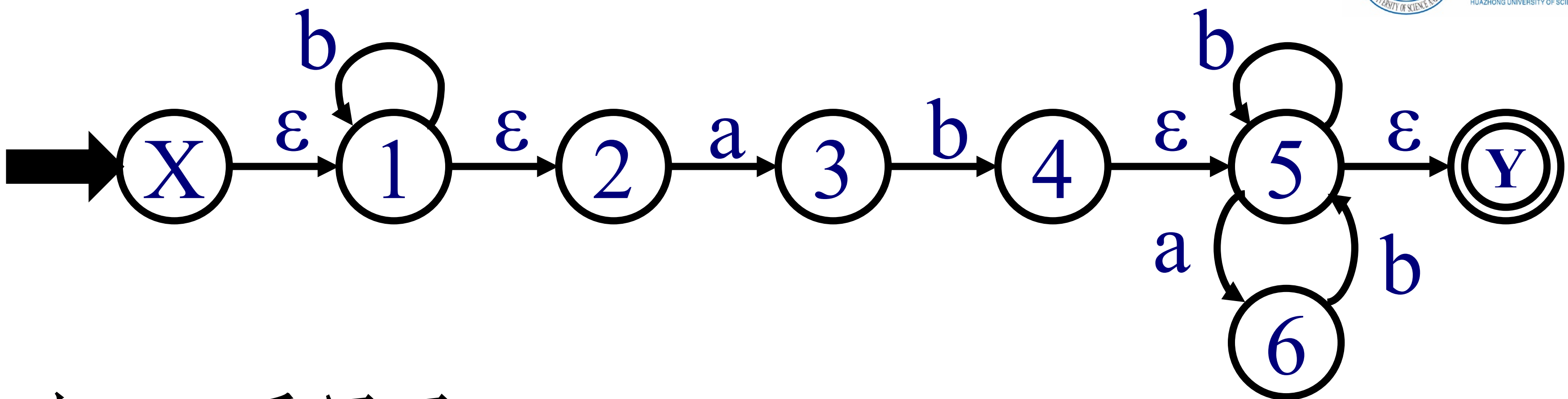
1. 试构造状态最小化的DFA M，使得L(M)=L(R)。

2. 求右线性文法G，使L(G)=L(M)。

本章小结

对正规式R=b*ab(b|ab)*采用分列法构造NFA，见下图。





对**NFA**采用子集法构造其等价的**DFA**的状态转换矩阵，见表。

	状态＼字符	a	b
X	{X,1,2}	{3}	{1,2}
B	{1,2}	{3}	{1,2}
A	{3}	Φ	{4,5,Y}
Y	{4,5,Y}	{6}	{5,Y}
E	{6}	Φ	{5,Y}
F	{5,Y}	{6}	{5,Y}

分化法得最小DFA：

初始分划∏=({Y,F}{X,B,A,E})

$\{Y,F\}_a=\{E\}$ $\{Y,F\}_b=\{F\}$

$\{X,B,A,E\}_a=\{A,A,\Phi,\Phi\}$

分划∏=({Y,F}{X,B}{A,E})

$\{X,B\}_a=\{A\},\{X,B\}_b=\{B\}$

$\{A,E\}_a=\{\Phi\},\{A,E\}_b=\{Y,F\}$

$\prod_{new}=(\{Y,F\}\{X,B\}\{A,E\})$

$\prod_{new}== \prod$

状态Y,F等价、X,B等价、A,E等价；保留Y,X,A化简得最小DFA如图：

	状态＼字符	a	b
X	{X,1,2}	{3}	{1,2}
B	{1,2}	{3}	{1,2}
A	{3}	Φ	{4,5,Y}
Y	{4,5,Y}	{6}	{5,Y}
E	{6}	Φ	{5,Y}
F	{5,Y}	{6}	{5,Y}