编译原理

第三章 词法分析

第三章 词法分析

- 对于词法分析器的要求
- 词法分析器的设计
- 正规表达式与有限自动机
- 词法分析器的自动产生 --LEX

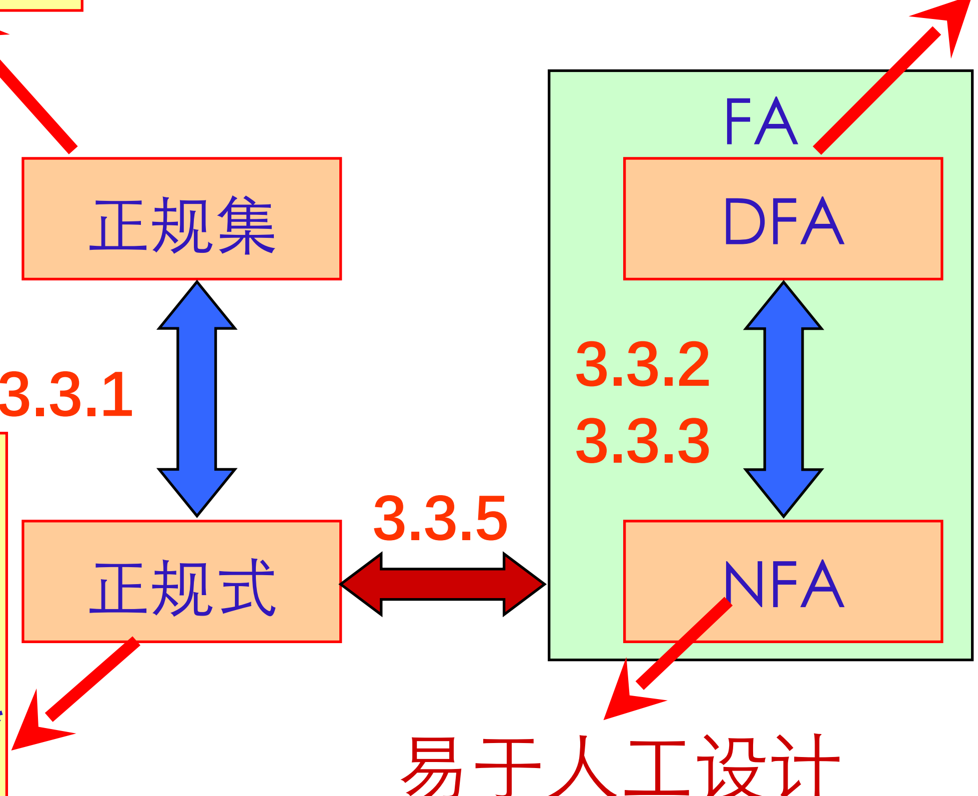
第三章 词法分析

- 对于词法分析器的要求
- 词法分析器的设计
- 正规表达式与有限自动机
- 词法分析器的自动产生 --LEX

关系图

```
curState = 初态
GetChar();
while( stateTrans[curState][ch] 有定义 ){
    // 存在后继状态，读入、拼接
    Concat();
    // 转换入下一状态，读入下一字符
   curState= stateTrans[curState][ch];
    if cur_state 是终态 then  返回 strToken 中的单
    GetChar( );
}
```

DIM,IF, DO,STOP,END
number, name, age
125, 2169,
…



DIM
IF
DO
STOP
END
letter(letter|digit)*
digit(digit)*

3.3.5 正规式与有限自动机的等价性

- 定理:

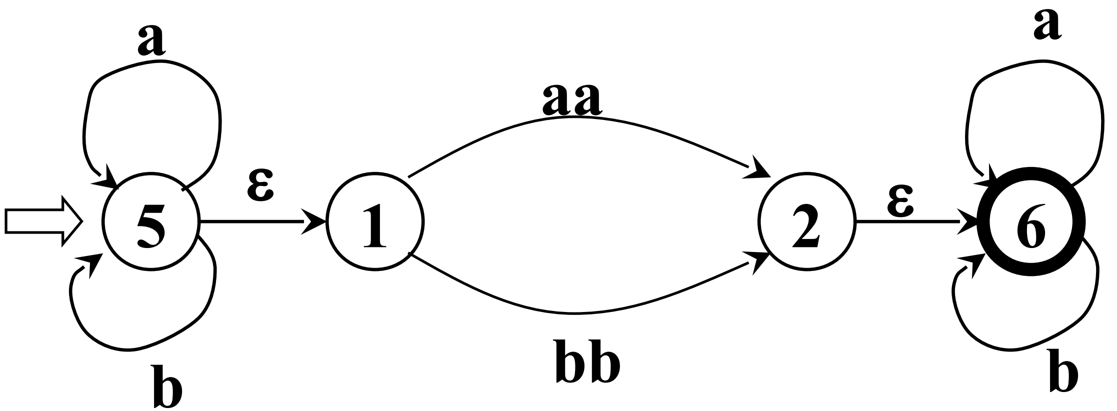
1. 对任何 FA M，都存在一个正规式 r，使得 L(r)=L(M)。
2. 对任何正规式 r，都存在一个 FA M，使得 L(M)=L(r)。

- 对转换图概念拓广，令每条弧可用一个正规式作标记。（对一类输入符号）

- 证明:

1 对Σ上任一 NFA M，构造一个Σ上的正规式 r，使得 L(r)=L(M)。

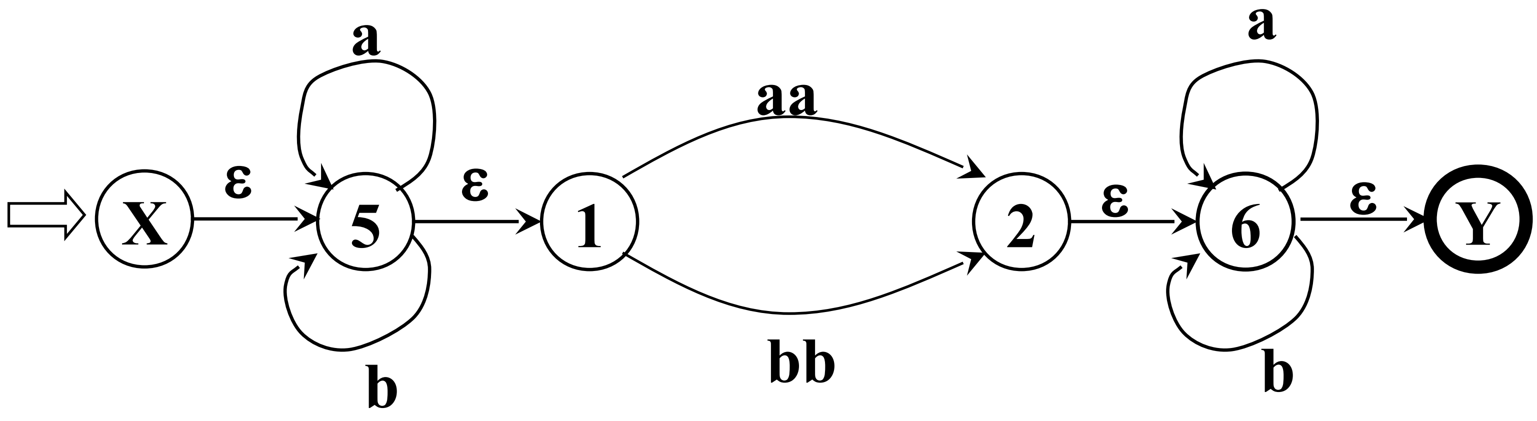
  - 首先，在 M 的转换图上加进两个状态 X 和 Y，从 X 用ε弧连接到 M 的所有初态结点，从 M 的所有终态结点用ε弧连接到 Y，从而形成一个新的 NFA，记为 M'，它只有一个初态 X 和一个终态 Y，显然 L(M)=L(M')。



- 证明：

1 对Σ上任一 NFA M ，构造一个Σ上的正规式 r，使得 L(r)=L(M) 。

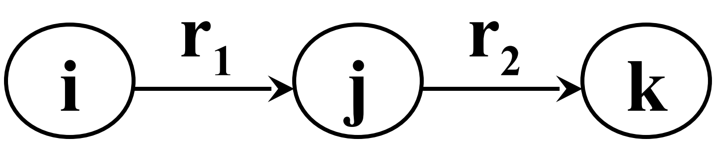
  - 首先，在 M 的转换图上加进两个状态 X 和 Y ，从 X 用ε弧连接到 M 的所有初态结点，从 M 的所有终态结点用ε弧连接到 Y ，从而形成一个新的 NFA，记为 M’ ，它只有一个初态 X 和一个终态 Y，显然 L(M)=L(M’) 。



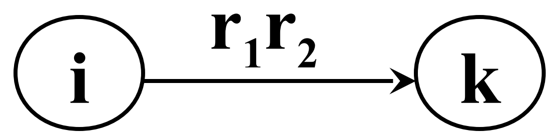
- 证明:

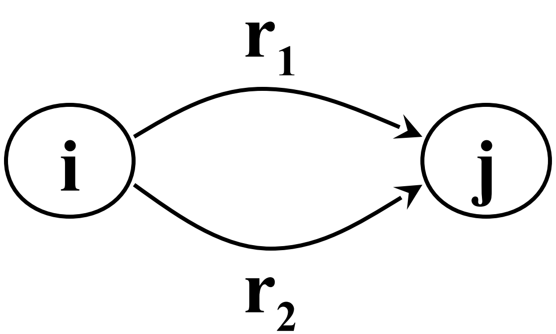
1 对Σ上任一 NFA M，构造一个Σ上的正规式 r，使得 L(r)=L(M)。

  - 首先，在 M 的转换图上加进两个状态 X 和 Y，从 X 用ε弧连接到 M 的所有初态结点，从 M 的所有终态结点用ε弧连接到 Y，从而形成一个新的 NFA，记为 M’，它只有一个初态 X 和一个终态 Y，显然 L(M)=L(M’)。
  - 然后，反复使用下面的一条规则，逐步消去的所有结点，直到只剩下 X 和 Y 为止;

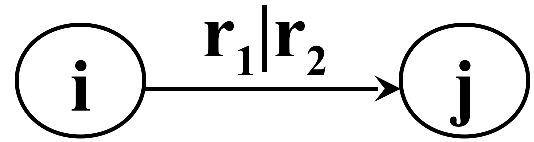


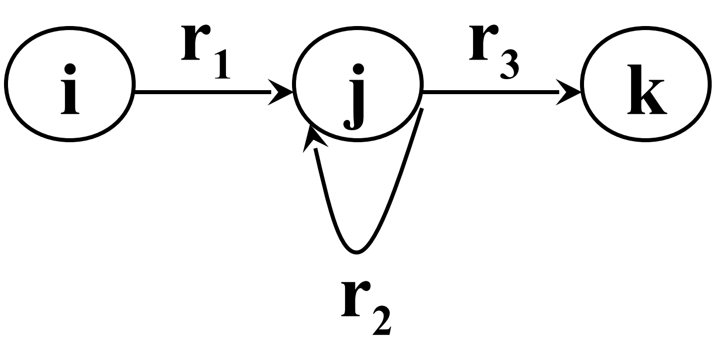
代之为



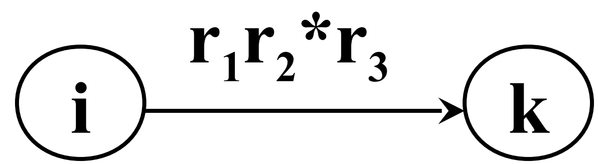


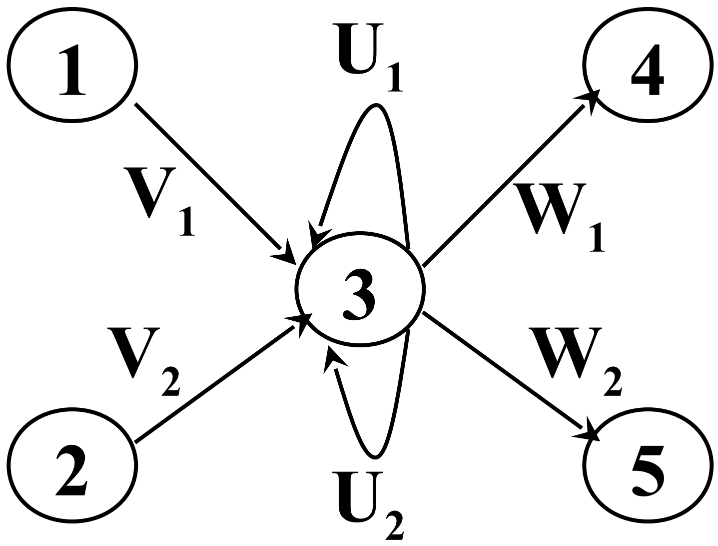
代之为

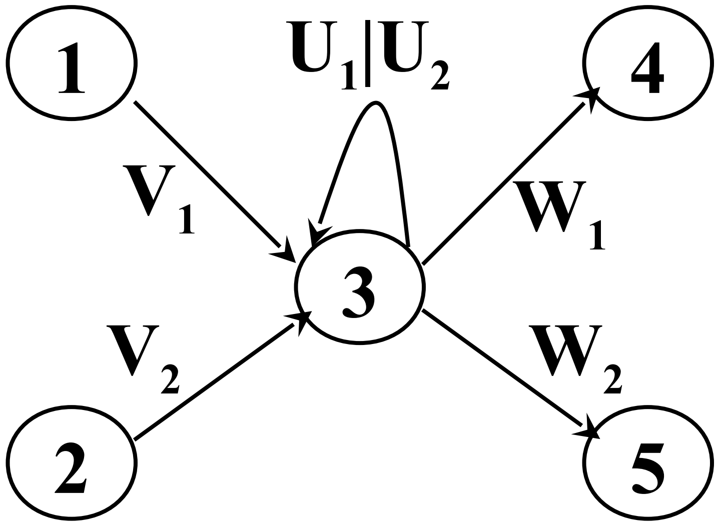


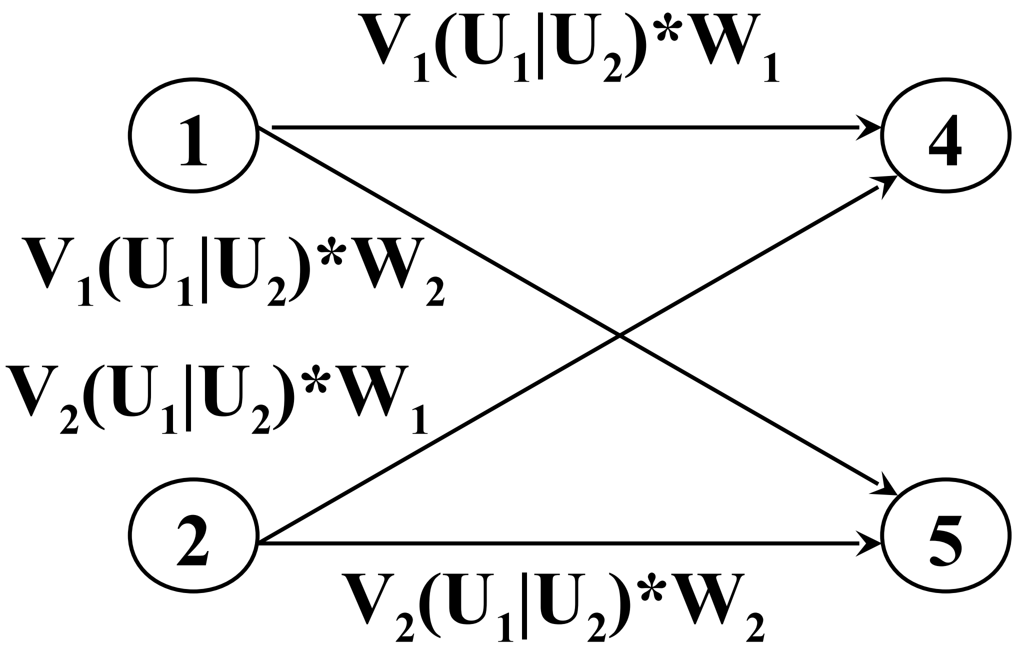


代之为




1
4
3
2
5
U_1
U_2
V_1
V_2
W_1
W_2


1
4
3
2
5
$U_1|U_2$
V_1
V_2
W_1
W_2


$V_1(U_1|U_2)*W_1$
1
4
$V_1(U_1|U_2)*W_2$
$V_2(U_1|U_2)*W_1$
2
5
$V_2(U_1|U_2)*W_2$

- 最后， X 到 Y 的弧上标记的正规式即为所构造的正规式 r
- 显然 L(r)=L(M)=L(M’)

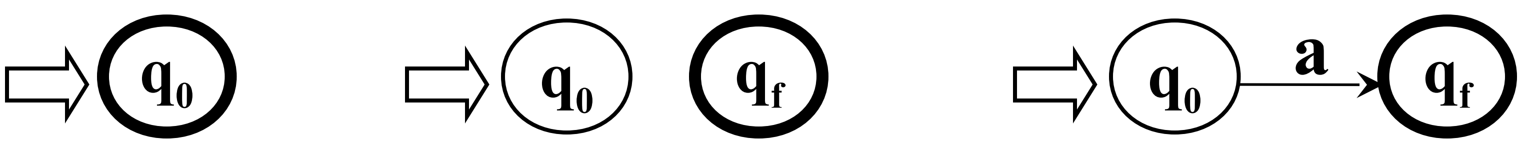
1. 对任何 FA M，都存在一个正规式 r，使得 L(r)=L(M)。

2. 对任何正规式 r，都存在一个 FA M，使得 L(M)=L(r)。

- 证明 2：对于Σ上的正规式 r，构造一个 NFA M，使 L(M)=L(r)，并且 M 只有一个终态，而且没有从该终态出发的箭弧。

下面使用关于 r 中运算符数目的归纳法证明上述结论。

(1) 若 r 具有零个运算符，则 $r=\varepsilon$ 或 $r=\phi$ 或 r=a，其中 $a \in \Sigma$。此时下图所示的三个有限自动机显然符合上述要求。



(2) 假设结论对于少于 k(k≥1) 个运算符的正规式成立。

当 r 中含有 k 个运算符时， r 有三种情形：

- 情形 1： $r=r_1|r_2$， r_1 和 r_2 中运算符个数少于 k 。从而，由归纳假设，对 r_i 存在 $M_i=\langle S_i, \Sigma_i, \delta_i, q_i, \{f_i\}\rangle$， 使得 $L(M_i)=L(r_i)$， 并且 M_i 没有从终态出发的箭弧（ i=1,2 ） 。不妨设 $S_1 \cap S_2=\phi$， 在 $S_1 \cup S_2$ 中加入两个新状态 q_0， f_0 。

令 $M=<S_1 \cup S_2 \cup \{q_0,f_0\}, \Sigma_1 \cup \Sigma_2, \delta, q_0, \{f_0\}>$，其中δ定义如下：

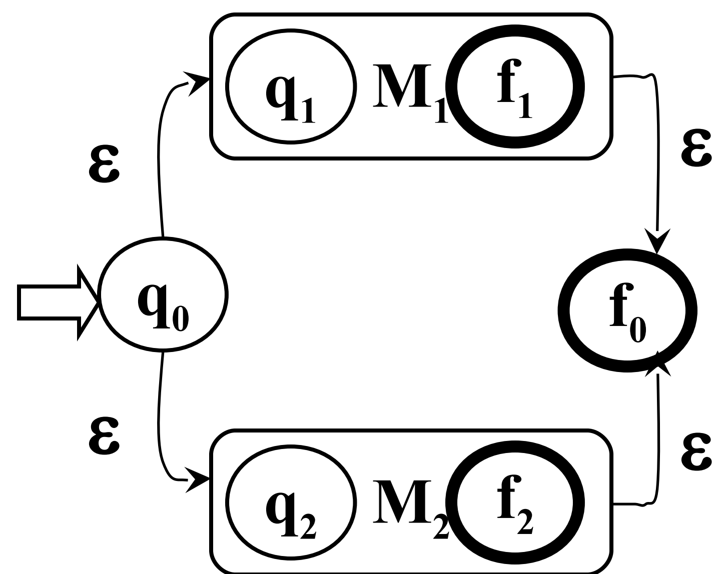
(a) $\delta(q_0, \varepsilon)=\{q_1,q_2\}$

(b) $\delta(q,a)= \delta_1(q,a)$，当 $q \in S_1-\{f_1\}$, $a \in \Sigma_1 \cup \{\varepsilon\}$

(c) $\delta(q,a)= \delta_2(q,a)$，当 $q \in S_2-\{f_2\}$, $a \in \Sigma_2 \cup \{\varepsilon\}$

(d) $\delta(f_1,\varepsilon)=\delta(f_2,\varepsilon)=\{f_0\}$ 。

$r=r_1|r_2$



M 的状态转换如右图所示。

$L(M)=L(M_1) \cup L(M_2)$

$=L(r_1) \cup L(r_2)=L(r)$

- 情形 2： $r=r_1r_2$，设 M_i 同情形 1(i=1,2)。

令 $M=<S_1\cup S_2, \Sigma_1\cup\Sigma_2, \delta, q_1, \{f_2\}>$ ，其中δ定义如下：

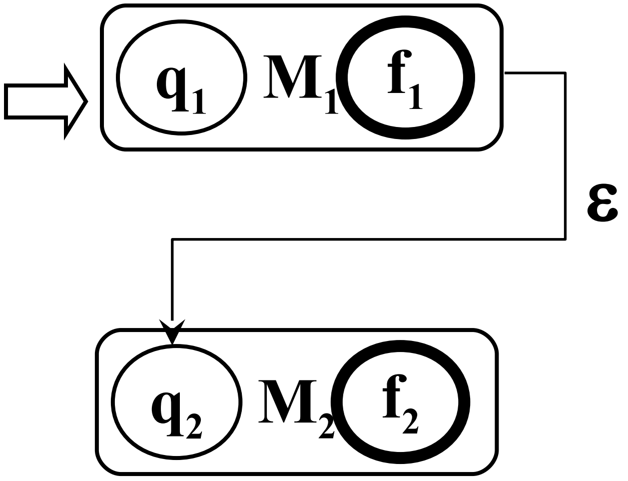
(a) $\delta(q,a)= \delta_1(q,a)$，当 $q\in S_1-\{f_1\}, a\in\Sigma_1\cup\{\varepsilon\}$

(b) $\delta(q,a)= \delta_2(q,a)$，当 $q\in S_2, a\in\Sigma_2\cup\{\varepsilon\}$

(c) $\delta(f_1,\varepsilon)=\{q_2\}$

$r=r_1r_2$

M 的状态转换如右图所示。



$L(M)=L(M_1)L(M_2)$

$=L(r_1)L(r_2)=L(r)$。

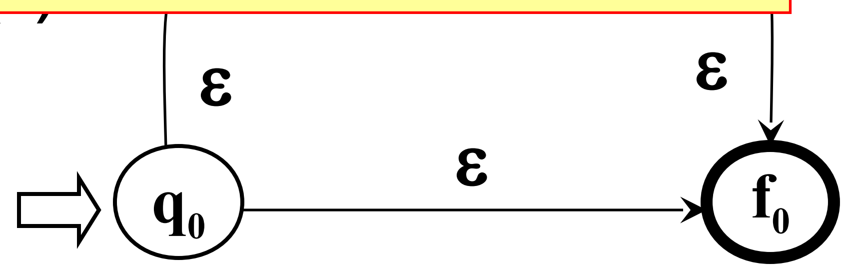
- 情形 3： $r=r_1^*$。设 M_1 同情形 1。

令 $M=<S_1\cup\{q_0, f_0\}, \Sigma_1, \delta, q_0, \{f_0\}>$，其中 q_0, $f_0\notin S_1$，δ定义如下:

(a) $\delta(q_0,\varepsilon)=\delta(f_1,\varepsilon)=\{q_1, f_0\}$

1. 对任何 FA M，都存在一个正规式 r，使得 L(r)=L(M)。

2. 对任何正规式 r，都存在一个 FA M，使得 L(M)=L(r)。

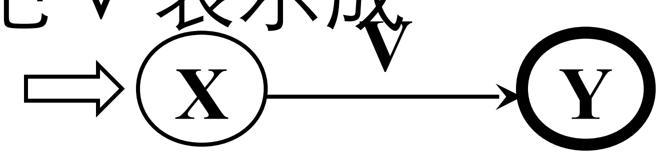


至此，结论 2 获证。

上述证明过程实质上是一个将正规表达式转换为有限自动机的算法

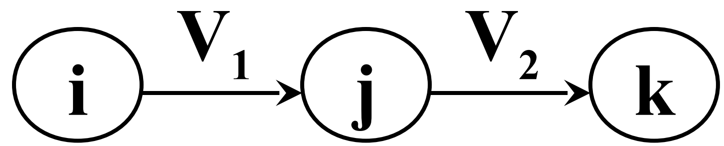
1) 构造Σ上的 NFA M' 使得 L(V)=L(M')

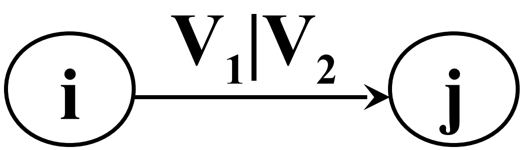
首先，把 V 表示成



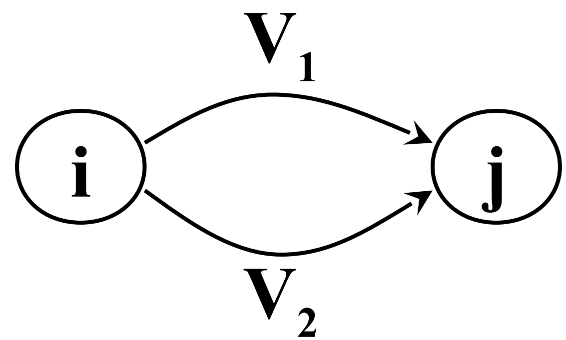
按下面的三条规则对 V 进行分裂

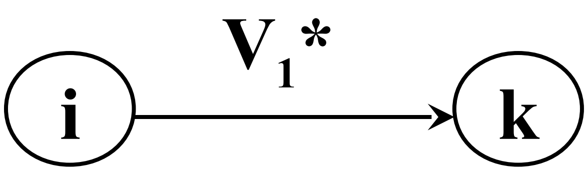
代之为



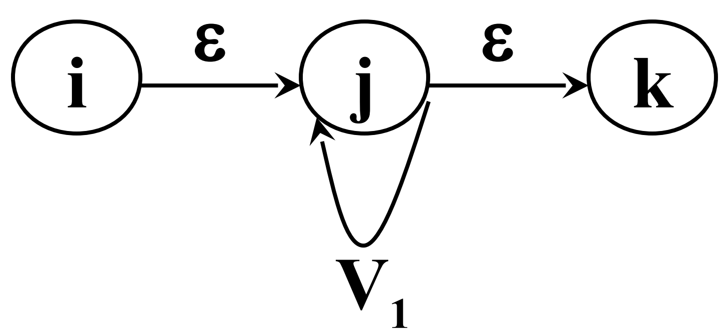


代之为



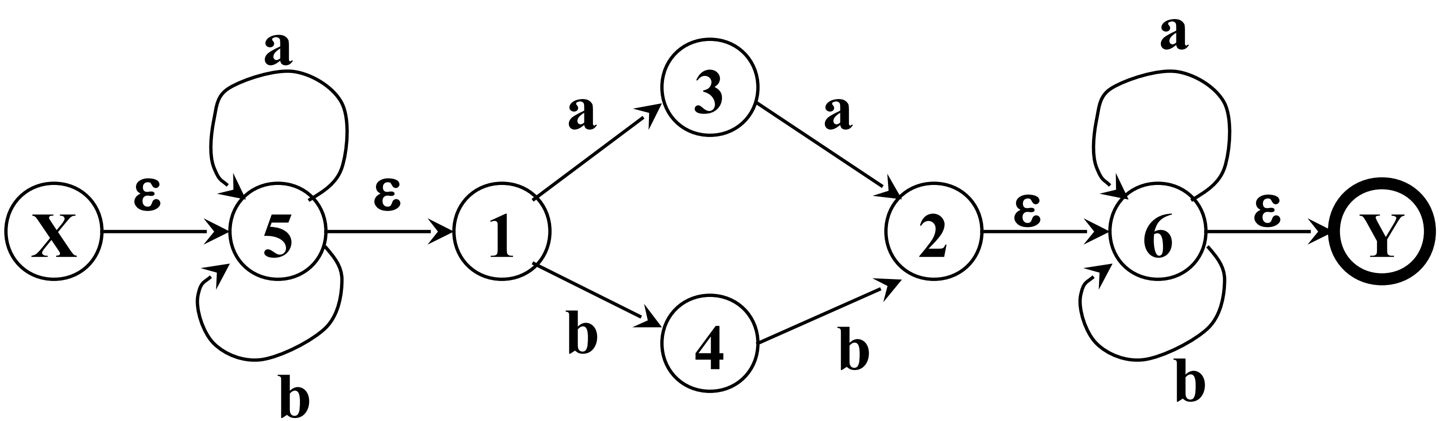


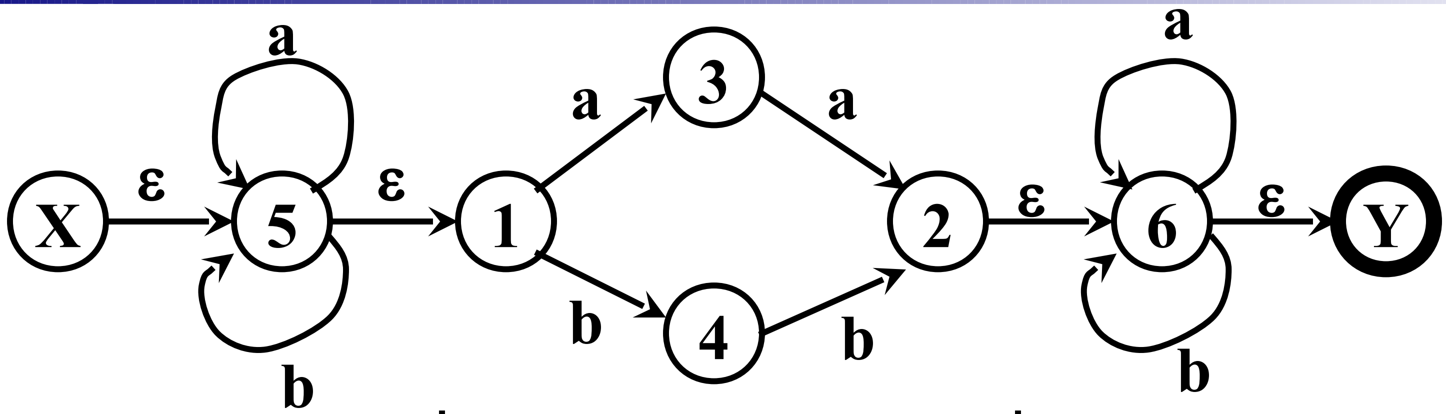
代之为



- 逐步把这个图转变为每条弧只标记为Σ上的一个字符或ε，最后得到一个 NFA M’，显然 L(M’)=L(V)

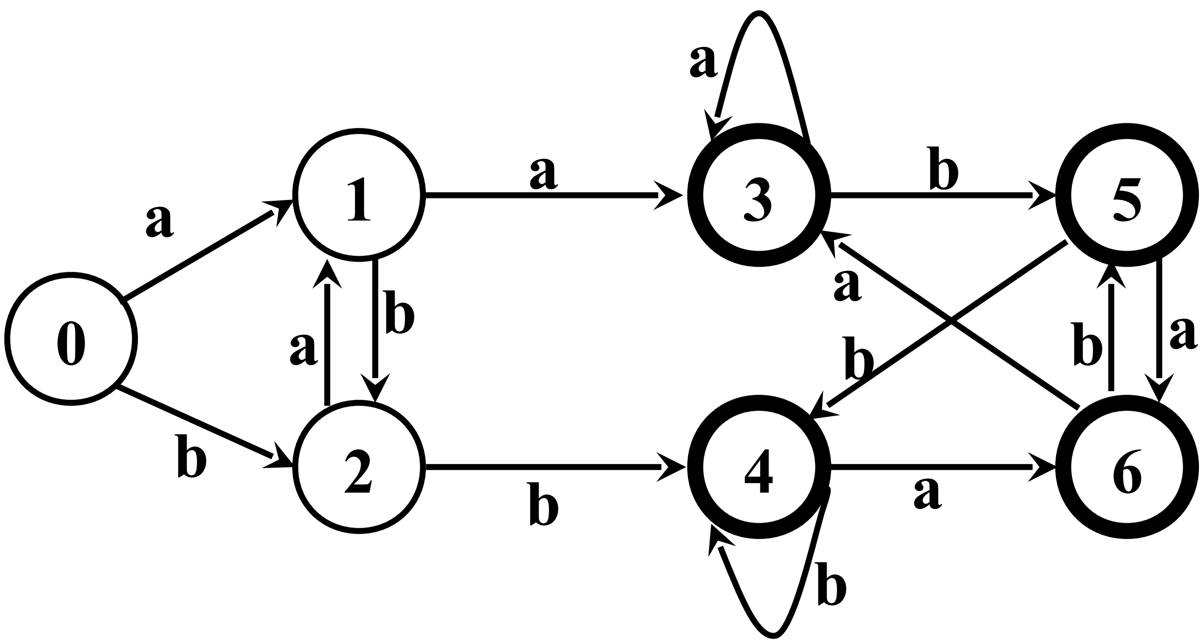
- (a|b)*(aa|bb)(a|b)*





I	I_a	I_b
{X,5,1}	{5,3,1}	{5,4,1}
{5,3,1}	{5,2,3,1,6,Y}	{5,4,1}
{5,4,1}	{5,3,1}	{5,2,4,1,6,Y}
{5,2,3,1,6,Y}	{5,2,3,1,6,Y}	{5,4,6,1,Y}
{5,2,4,1,6,Y}	{5,3,6,1,Y}	{5,2,4,1,6,Y}
{5,4,6,1,Y}	{5,3,6,1,Y}	{5,2,4,1,6,Y}
{5,3,6,1,Y}	{5,2,3,1,6,Y}	{5,4,6,1,Y}

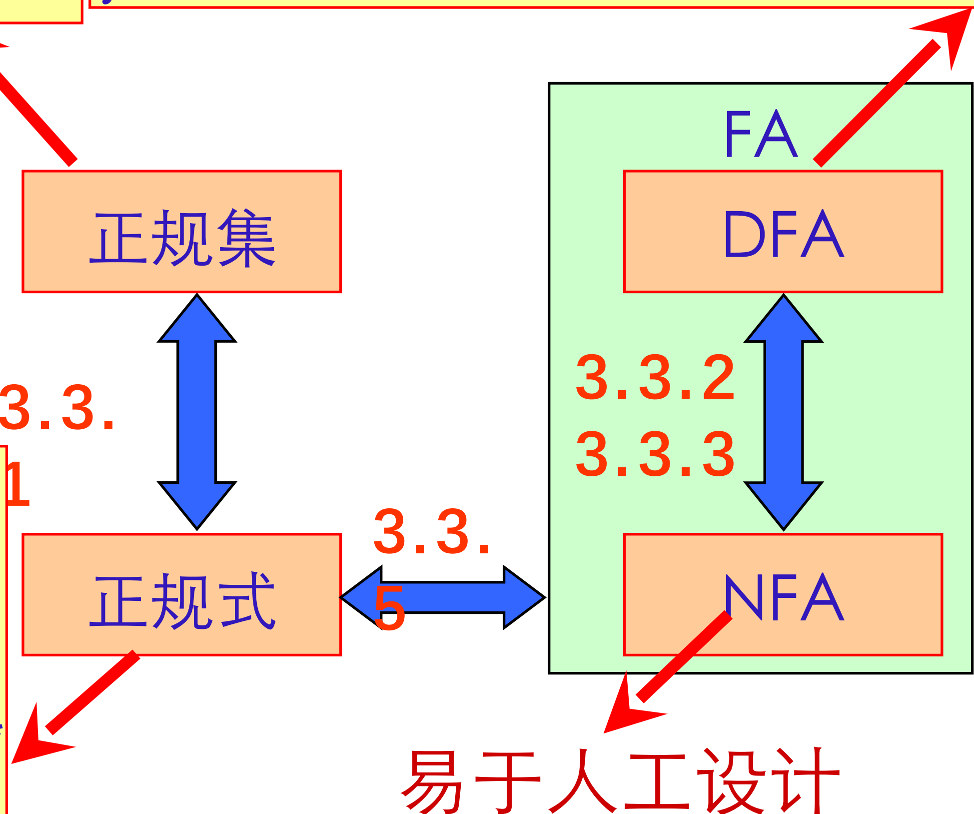
I	a	b
0	1	2
1	3	2
2	1	4
3	3	5
4	6	4
5	6	4
6	3	5



小结

DIM,IF, DO,STOP,END
number, name, age
125, 2169
…

```
curState = 初态
GetChar();
while( stateTrans[curState][ch] 有定义 ){
    // 存在后继状态，读入、拼接
    Concat();
    // 转换入下一状态，读入下一字符
   curState= stateTrans[curState][ch];
    if cur_state 是终态 then  返回 strToken 中的单
    GetChar( );
}
```



DIM
IF
DO
STOP
END
letter(letter|digit)*
digit(digit)*