

华中科技大学

编译原理

第3章 词法分析

2019年11月10日星期日

编译原理课程组

内容摘要

本章介绍词法分析阶段的基本原理和技术，主要内容是词法的3种形式描述工具及其相互转换、构造词法分析程序的技术线路。

重点讲解

3.1 词法分析程序设计

3.2 单词的描述工具

3.3 有穷自动机

3.4 正规式和有穷自动机的等价性

3.5 正规文法和有穷自动机间的转换

3.6 词法分析程序的自动构造工具

3.1 词法分析程序设计

3.1.1 词法分析任务

词法分析阶段是编译的第一阶段，它的主要任务是从左至右扫描文本格式的源程序，从基于字符理解的源程序中分离出符合源语言词法的单词，最终转换成基于单词理解的源程序。

输出形式为：（单词种类，单词）

单词种类类似于自然语言的词性，由构词规则等因素确定的。

计算机高级语言一般都有关键字、标识符、常数、运算符和定界符这5类单词。

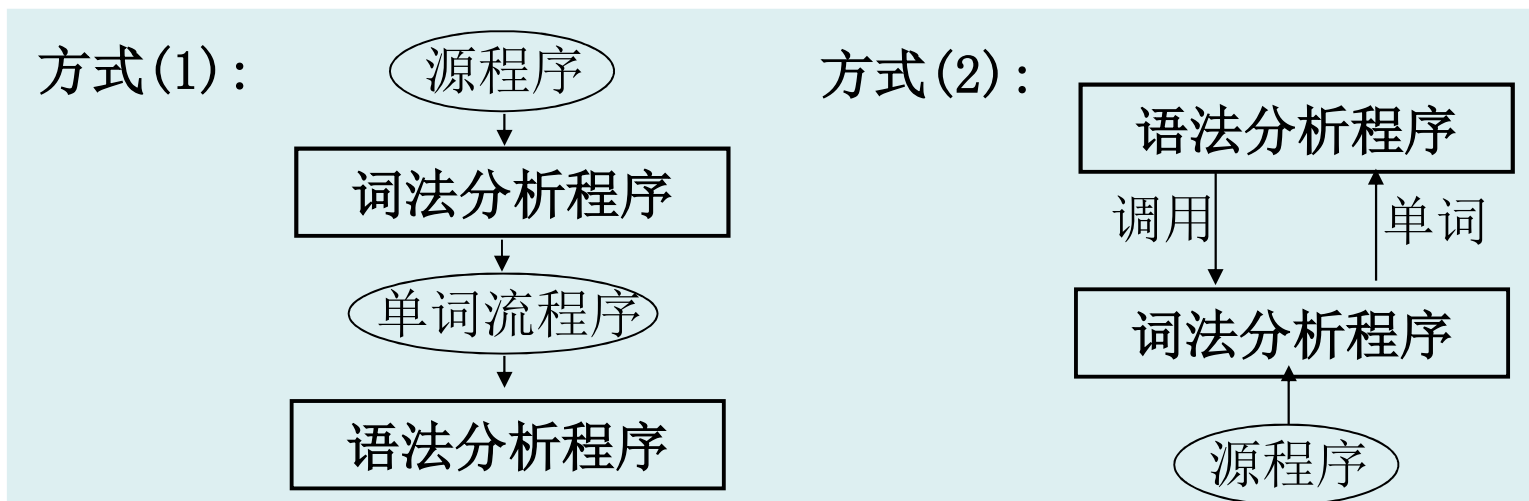


3.1.2 词法分析程序和语法分析程序的接口方式

词法分析程序通常与后阶段语法分析程序接口有下列两种方式。

(1)词法分析程序和语法分析程序各自独立一趟方式。即词法分析程序把字符流的源程序转换成单词流的内部程序形式，供语法分析程序之用。

(2)词法分析程序和语法分析程序合并为一趟方式。即词法分析程序由反复语法分析程序调用，每调用一次从源程序中一个新单词返回给语法分析程序。



3.2 单词的形式化描述工具

基于生成观点、计算观点和识别观点，分别形成了正规文法、正规式和有穷自动机 3种用于描述计算机高级语言词法的工具。

本节仅介绍正规文法和正规式以及两者之间的转换方法。

3.2.1 正规文法

下面是“标识符”单词的右线性正规文法描述实例。

$$\begin{array}{l} G_1[T]: \\ T \rightarrow c \mid cS \\ S \rightarrow c \mid d \mid cS \mid dS \end{array}$$

T : 标识符 S: 字母数字串 c: 字母 d: 数字

3.2.2 正规式

基于字母表 Σ 上的正规式（也称为正则表达式）定义如下，正规式 e 的计算值称为正规集，记为 $L(e)$ 。

1. ε 是 Σ 上的正规式， $L(\varepsilon) = \{\varepsilon\}$
2. Φ 是 Σ 上的正规式， $L(\Phi) = \Phi$
3. 任何 $a \in \Sigma$ ， a 是 Σ 上的正规式， $L(a) = \{a\}$
4. 如果 e_1 和 e_2 是 Σ 上的正规式，则
 - 4.1 (e_1) 是 Σ 上的正规式， $L((e_1)) = L(e_1)$
 - 4.2 $e_1 \mid e_2$ 是 Σ 上的正规式， $L(e_1 \mid e_2) = L(e_1) \cup L(e_2)$
 - 4.3 $e_1 \cdot e_2$ 是 Σ 上的正规式， $L(e_1 \cdot e_2) = L(e_1) \cdot L(e_2)$
 - 4.4 e_1^* 是 Σ 上的正规式， $L(e_1^*) = L(e_1)^*$

3.2.2 正规式

例 3.1 令 $\Sigma = \{a,b\}$ 则 Σ 上正规式的例子如下 ,

a、 $a \mid b$ 、 ab 、 $(a \mid b)^*$ 、 $(a \mid b)^*a$,

且 $L(a) = \{a\}$

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

$L((a \mid b)^*) = L(L(a \mid b))^* = (\{a,b\})^* = \{a,b\}^*$

$L((a \mid b)^*a) = L((a \mid b)^*) \cdot L(a) = \{a,b\}^* \{a\}$

两个正规式 e_1 和 e_2 相等 , 是指正规式 e_1 和 e_2 计算值相等 (即 $L(e_1) = L(e_2)$) , 记为 $e_1 = e_2$ 。

设 r,s,t 为正规式 , 则正规式有如下定律 :

1. 交换律 : $r \mid s = s \mid r$

2. 结合律 : $(r \mid s) \mid t = r \mid (s \mid t)$

$(r \cdot s) \cdot t = r \cdot (s \cdot t)$

3. 分配律 : $r \cdot (s \mid t) = r \cdot s \mid r \cdot t$

$(s \mid t) \cdot r = s \cdot r \mid t \cdot r$

正规式应用举例

(1) 描述“标识符”单词的正规式

$$a(a \mid b)^*$$

其中, $\Sigma = \{a, b\}$, a —— 字母, b —— 数字

$$L(a(a \mid b)^*) = \{a\} \{a, b\}^*$$

(2) 描述“整数”单词的正规式

$$dd^* \mid +dd^* \mid -dd^*$$

其中, $\Sigma = \{+, -, d\}$, d —— 数字

$$L(dd^* \mid +dd^* \mid -dd^*) = \{+, -, \varepsilon\} \{d\} \{d\}^*$$

3.2.3 正规式和正规文法之间转换

定义 3.2 如果正规式 r 和文法 G , 有 $L(r) = L(G)$
则称正规式 r 和文法 G 是等价的。

正规式 $r \rightarrow$ 文法 G 转换方法

设 Σ 上正规式 r , 则等价文法 $G = (V_N, V_T, P, S)$ 。其中, $V_T = \Sigma$;
从形如产生式 $S \rightarrow r$ 开始, 按表4.1规则进行转换, 直到全部
形如产生式, 符合正规文法之规则形式为止, 可得到 P 和 V_N 。

规则1	$A \rightarrow xy$	$A \rightarrow xB, B \rightarrow y$
规则2	$A \rightarrow x*y$	$A \rightarrow xB, A \rightarrow y$ $B \rightarrow xB, B \rightarrow y$
规则3	$A \rightarrow x \mid y$	$A \rightarrow x, A \rightarrow y$
注: $A, B \in V_N$, B 为新增非终结符		

正规式转换成正规文法

例 3.2 将 $\{a,d\}$ 上的正规式 $a(a \mid d)^*$ ，转换成等价的正规文法 $G[S]$ 。

$S \rightarrow a(a \mid d)^*$	($S \rightarrow r$)
$S \rightarrow aA, A \rightarrow (a \mid d)^*$	(规则1)
$S \rightarrow aA, A \rightarrow (a \mid d)B, A \rightarrow \varepsilon, B \rightarrow (a \mid d)B, B \rightarrow \varepsilon$	(规则2)
$S \rightarrow aA, A \rightarrow aB \mid dB, A \rightarrow \varepsilon, B \rightarrow aB \mid dB, B \rightarrow \varepsilon$	(分配律)
$S \rightarrow aA, A \rightarrow aB, A \rightarrow dB, A \rightarrow \varepsilon, B \rightarrow aB, B \rightarrow dB, B \rightarrow \varepsilon$	

最后得到正规文法 $G[S]$ 如下：

$$\begin{aligned}
 G &= (V_N, V_T, P, S), \text{ 其中,} \\
 V_N &= \{S, A, B\} \\
 V_T &= \{a, d\} \\
 P &= \{S \rightarrow aA, A \rightarrow aB, A \rightarrow dB, A \rightarrow \varepsilon, \\
 &\quad B \rightarrow aB, B \rightarrow dB, B \rightarrow \varepsilon\}
 \end{aligned}$$

正规文法转换成正规式

基本上是正规式到正规文法的逆过程，按下列规则将文法处理成只剩下一个开始符号定义的正规式。

规则1	$A \rightarrow xB, B \rightarrow y$	$A \rightarrow xy$
规则2	$A \rightarrow xA \mid y$	$A \rightarrow x^*y$
规则3	$A \rightarrow x, A \rightarrow y$	$A \rightarrow x \mid y$
注意此处规则2与前面的区别，具有单向性。		

正规文法转换成正规式

正规文法G[S]

$S \rightarrow aA, \quad S \rightarrow a$

$A \rightarrow aA, \quad A \rightarrow dA$

$A \rightarrow a, \quad A \rightarrow d$

规则1

$A \rightarrow xB, B \rightarrow y$

$A \rightarrow xy$

规则2

$A \rightarrow xA \mid y$

$A \rightarrow x^*y$

规则3

$A \rightarrow x, \quad A \rightarrow y$

$A \rightarrow x \mid y$

整理成:

$S \rightarrow aA \mid a$

$A \rightarrow a(A \mid d)^*(a \mid d) \mid a \mid d$

代数变换:

$S = a(a \mid d)^*(a \mid d) \mid a$

$= a((a \mid d)^*(a \mid d) \mid \epsilon)$

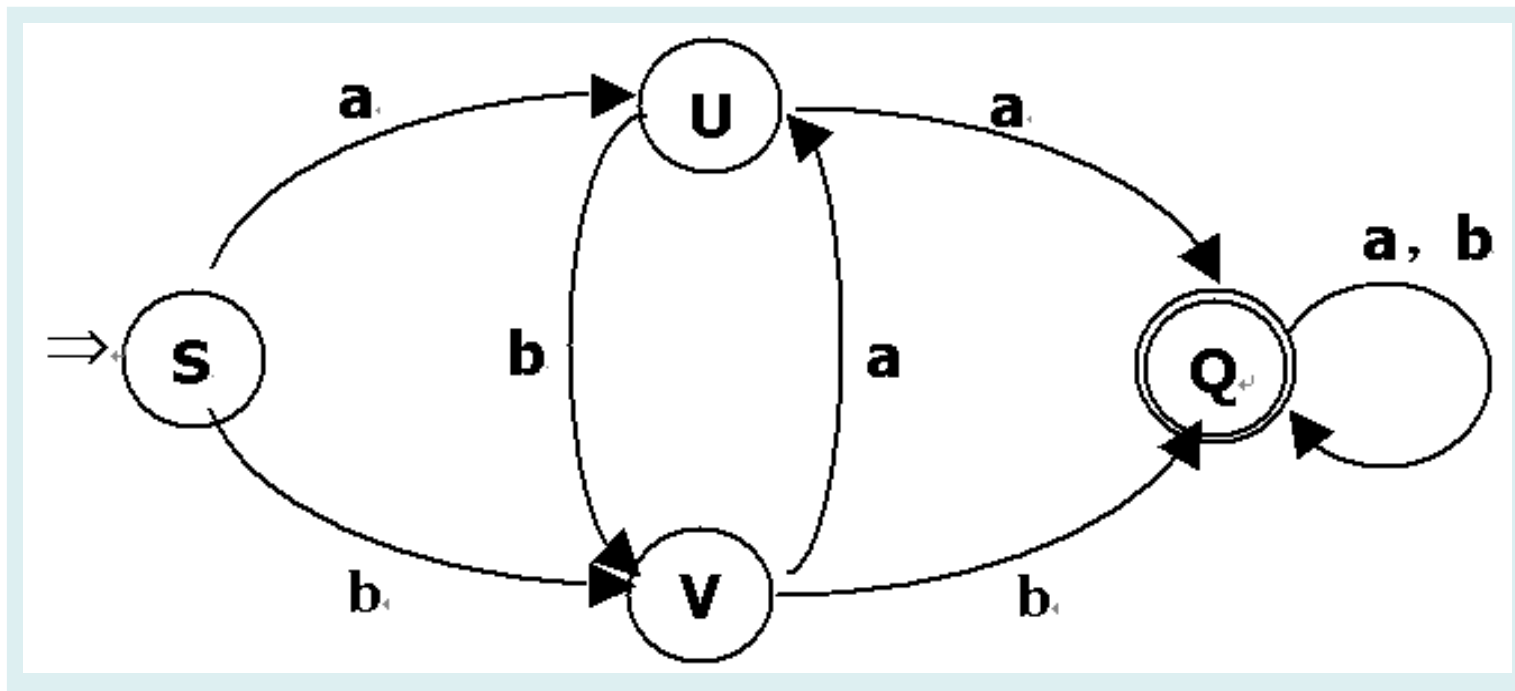
$= a(a \mid d)^*$

如何使用规则处理正规文法G[A]?

$A \rightarrow aB, \quad A \rightarrow b,$

$B \rightarrow cA, \quad B \rightarrow d$

3.3 有穷自动机



M接受的aaa和M不接受的aba的识别过程。

定义3.3 确定有穷自动机

一个确定的有穷自动机DFA M 是一个五元组： $M=(K, \Sigma, f, S, Z)$ 。
其中：

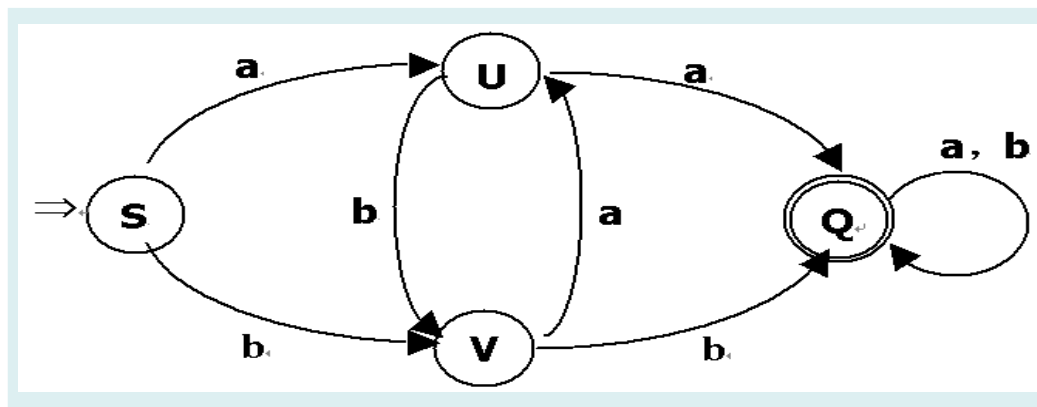
K 是非空有穷集，每个元素称为状态；

Σ 是有穷字母表；

f 是 $K \times \Sigma \rightarrow K$ 映射，称为状态转换函数；

$S \in K$ ，称为开始状态；

$Z \subset K$ ，称为结束状态集，或接受状态集。

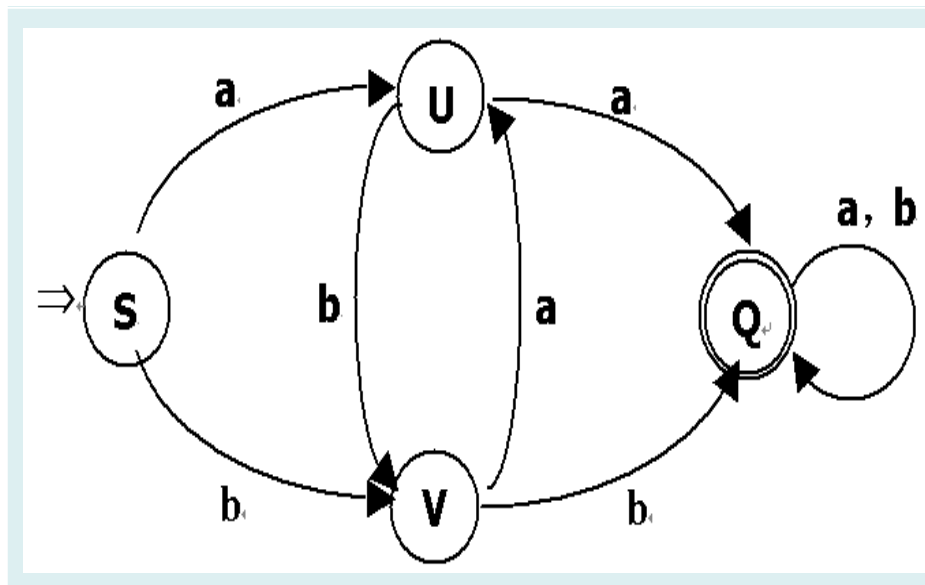


确定有穷自动机转换函数的扩充

转换函数 f 可以扩充为 $f': K \times \Sigma^* \rightarrow K$ 映射, 并以 f 替代 f' 使用。

设 $a \in \Sigma$, $\beta \in \Sigma^*$, $q \in K$, 即

$$f'(q, a\beta) = \begin{cases} f(q, a) & (\beta = \varepsilon) \\ f'(f(q, a), \beta) & (\beta \neq \varepsilon) \end{cases}$$



$$\begin{aligned} f'(S, aaa) &= f'(f(S, a), aa) \\ &= f'(U, aa) \\ &= f'(f(U, a), a) \\ &= f'(Q, a) \\ &= f(Q, a) = Q \in Z \end{aligned}$$

定义3.4 确定有穷自动机识别的语言

设DFA $M = (K, \Sigma, f, S, Z)$ ，如果 $\alpha \in \Sigma^*$ ， $f'(S, \alpha) \in Z$ ，则称符号串 α 是DFA M 所接受(或识别)的。DFA M 所接受的符号串的集合记为 $L(M)$ ，即

$$L(M) = \{ \alpha \mid \alpha \in \Sigma^*, f'(S, \alpha) \in Z \}。$$

一个DFA $M = (K, \Sigma, f, S, Z)$ ，以带权有向图 $G = (V, E)$ 观点，还可采用图形直观描述：

- 顶点表示状态(即 $V = K$)
- 加上粗箭头的顶点表示开始状态
- 双圈顶点表示接受状态
- 权为 a 的弧 $\langle A, B \rangle (\in E)$ 表示 $f(A, a) = B$ 。

$f(A, a) = B$ 也读作“状态 A 经过 a 转换到状态 B ”。

例 3.3 DFA M定义如下，并转换直观状态图表示，讨论所接受的符号串情况。

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

其中 $K = \{S, U, V, Q\}$

$\Sigma = \{a, b\}$

f : $f(S, a) = U$ $f(S, b) = V$

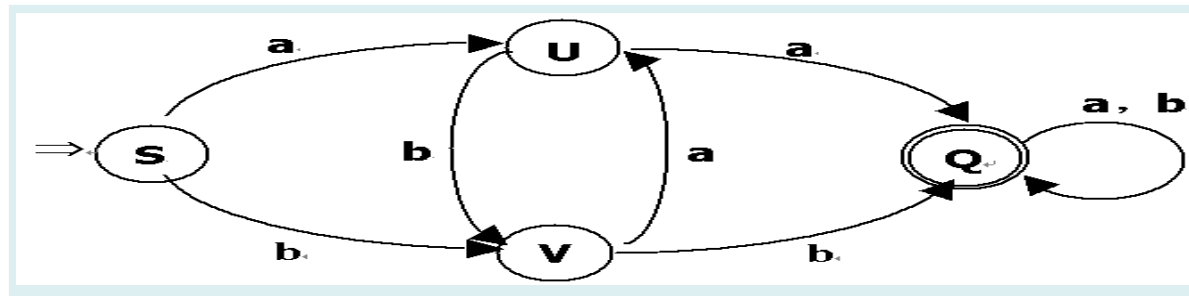
$f(U, a) = Q$ $f(U, b) = V$

$f(V, a) = U$ $f(V, b) = Q$

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

$Z = \{Q\}$

DFA M的状态图表示如下。



M接受的aaa和M不接受的aba的识别过程

定义3.5 不确定有穷自动机

一个不确定的有穷自动机NFA M 是一个五元组： $M=(K, \Sigma, f, S, Z)$ 。
其中：

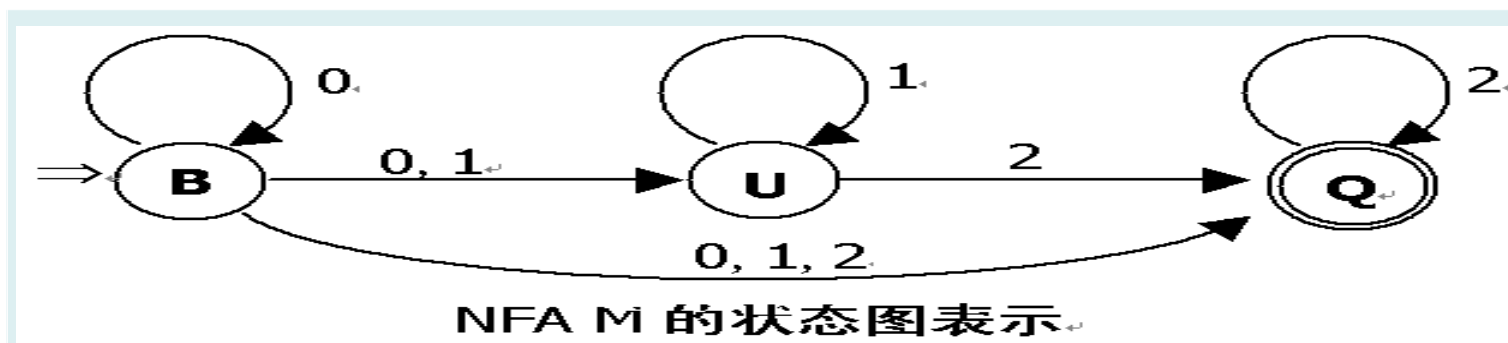
K 是非空有穷集，每个元素称为状态；

Σ 是有穷字母表；

f 是 $K \times \Sigma \cup \{\varepsilon\} \rightarrow \rho(K)$ 映射； f 称为状态转换函数， $\rho(K)$ 表示 K 之幂集。

$S \subseteq K$ ，称为开始状态集；

$Z \subseteq K$ ，称为结束状态集，或接受状态集。



例 3.4 NFA M定义如下, 并讨论所接受的符号串情况。

$M = (K, \Sigma, f, S, Z)$, 其中,

$K = \{B, U, Q\}$

$\Sigma = \{0, 1, 2\}$

f : $f(B, 0) = \{S, U, Q\}$ $f(B, 1) = \{U, Q\}$

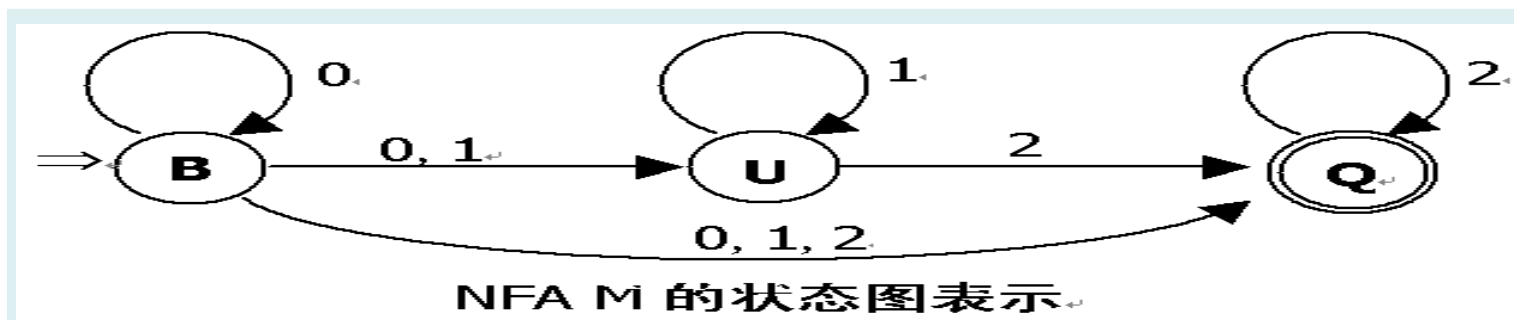
$f(B, 2) = \{Q\}$ $f(U, 0) = \Phi$ $f(U, 1) = \{U\}$

$f(U, 2) = \{Q\}$ $f(Q, 0) = \Phi$

$f(Q, 1) = \Phi$ $f(Q, 2) = \{Q\}$

$S = \{B\}$

$Z = \{Q\}$



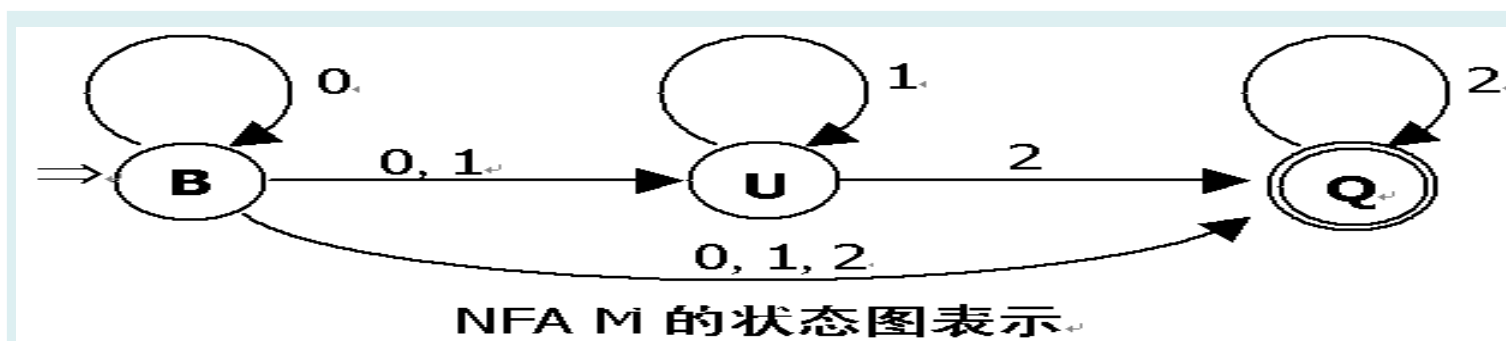
M接受的0、012和M不接受的11的识别过程

不确定有穷自动机转换函数的扩充

NFA转换函数 F 也可以扩充为 $f': \rho(K) \times \Sigma^* \rightarrow \rho(K)$ 映射, 并以 f 替代 f' 使用。设 $a \in \Sigma$, $\beta \in \Sigma^*$, $I \subset K$, 即

$$f'(I, a\beta) = \begin{cases} M(I, a) & (\beta = \varepsilon) \\ f'(M(I, a), \beta) & (\beta \neq \varepsilon) \end{cases}$$

$$\text{其中, } M(I, a) = \bigcup_{q \in I} f(q, a)$$

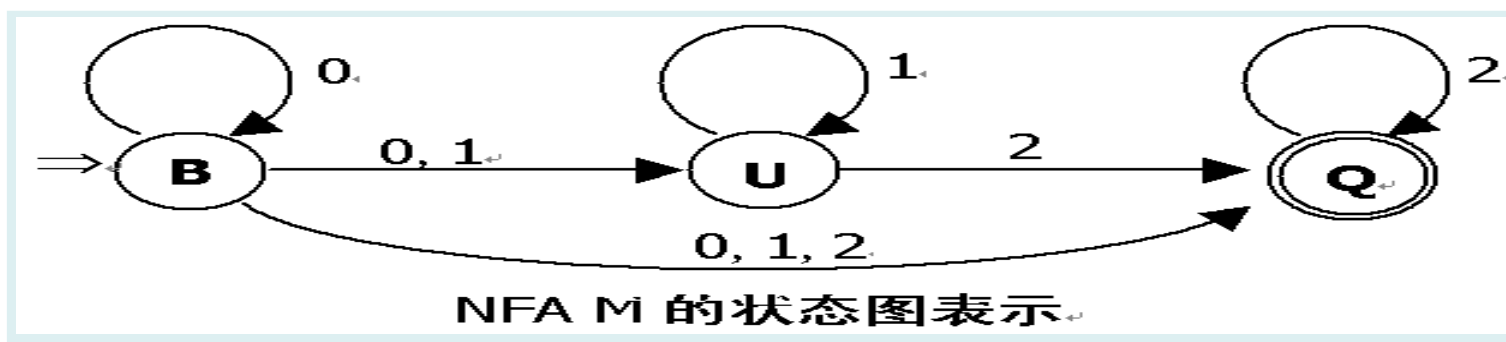


$$\begin{aligned} f'(\{B\}, 012) &= f'(M(\{B\}, 0), 12) = f'(\{B, U, Q\}, 12) \\ &= f'(M(\{B, U, Q\}, 1), 2) = f'(\{U, Q\}, 2) = M(\{U, Q\}, 2) = \{Q\} \end{aligned}$$

定义3.6 不确定有穷自动机识别的语言

设NFA $M = (K, \Sigma, f, S, Z)$, 如果 $\alpha \in \Sigma^*$, $f'(S, \alpha) \cap Z \neq \Phi$, 则称符号串 α 是NFA M 所接受(或识别)的。NFA M 所接受的符号串的集合亦记为 $L(M)$, 即

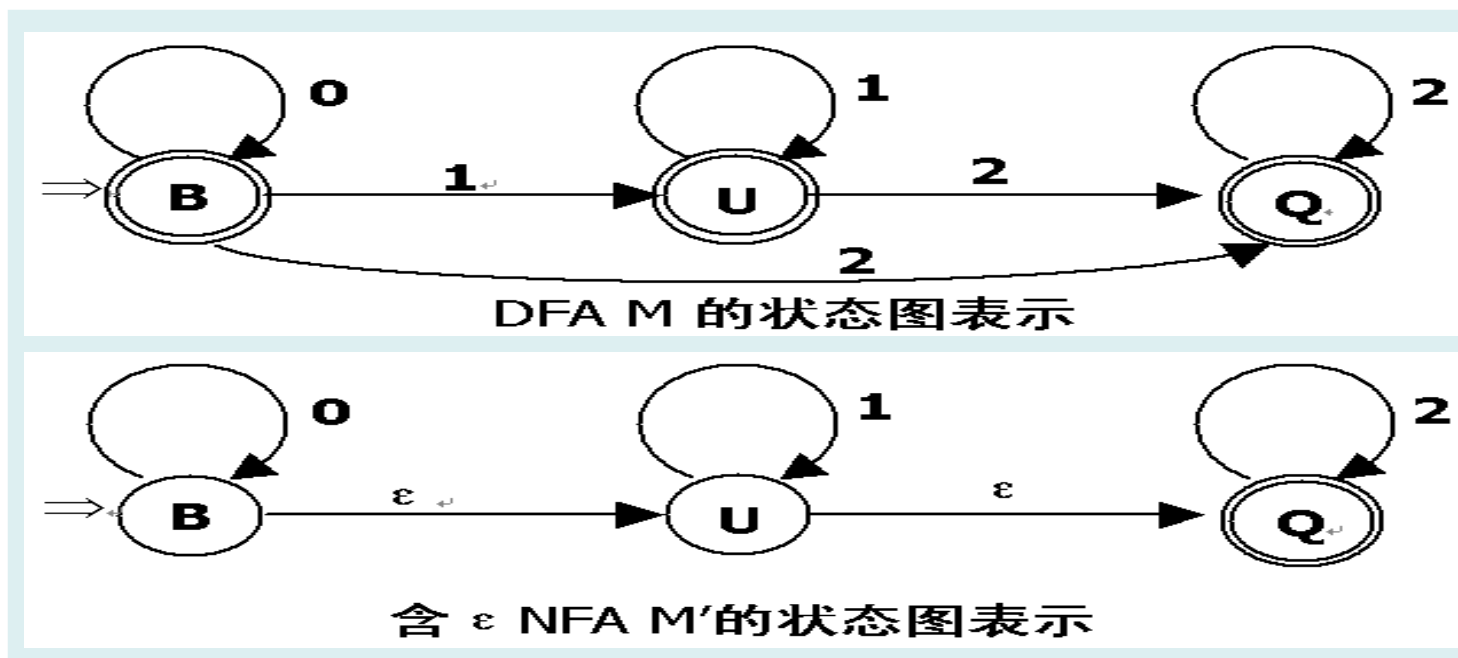
$$L(M) = \{ \alpha \mid \alpha \in \Sigma^*, f'(S, \alpha) \cap Z \neq \Phi \}.$$



$\because f'(\{B\}, 012) = f'(M(\{B\}, 0), 12) = f'(\{B, U, Q\}, 12)$
 $= f'(M(\{B, U, Q\}, 1), 2) = f'(\{U, Q\}, 2) = M(\{U, Q\}, 2) = \{Q\}$
 $f'(\{B\}, 012) \cap Z \neq \Phi$
 $\therefore 012$ 是NFA M 所接受(或识别)的

定义3.7 自动机的等价性

如果FA M_1 和FA M_2 接受相同的符号串的集合(即 $L(M_1) = L(M_2)$), 则称FA M_1 和FA M_2 是等价的。



$$\because L(M) = L(M') = \{0^n 1^m 2^k \mid n, m, k \geq 0\} = \{0\}^* \{1\}^* \{2\}^*$$

\therefore FA M 和FA M' 是等价的

状态集合的映射和闭包运算

定义 3.8 设NFA $M = (K, \Sigma, f, S, Z)$, $I \subseteq K$, $a \in \Sigma \cup \{\varepsilon\}$, 则 $M(I, a)$ 定义如下:

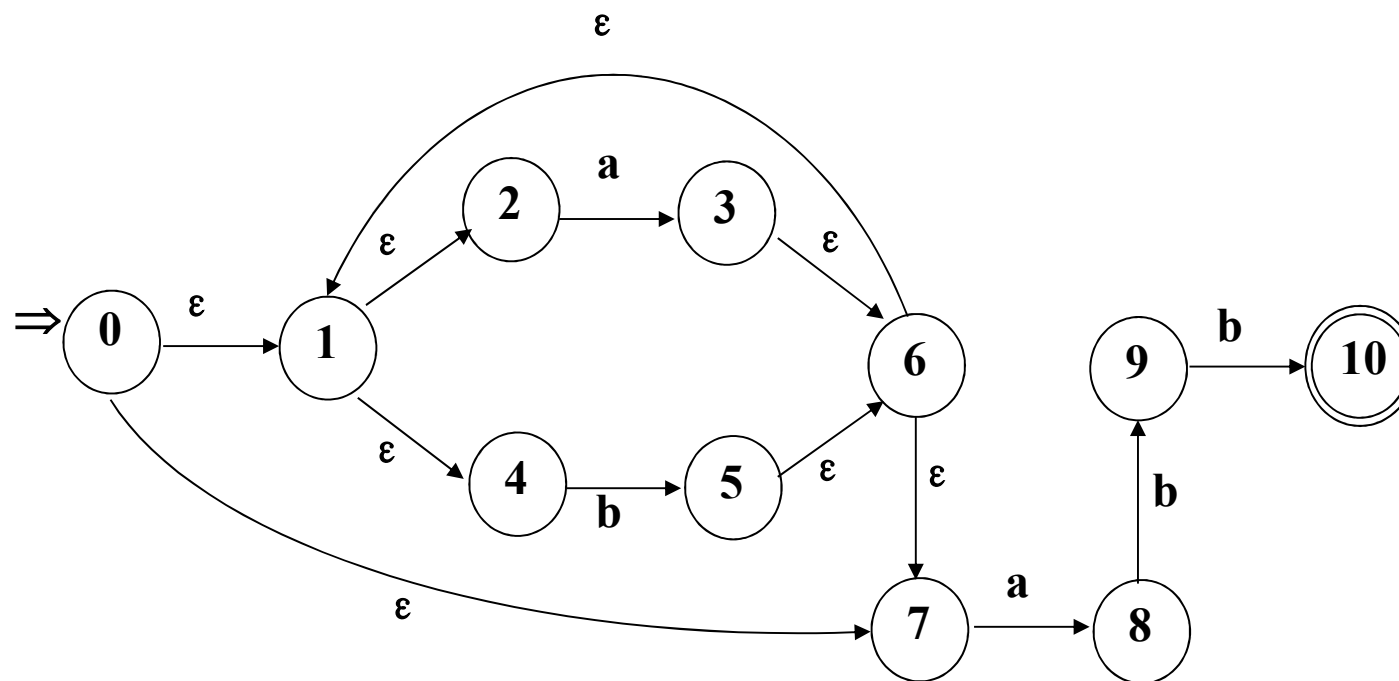
$$M(I, a) = \bigcup_{q \in I} f(q, a)$$

定义 3.9 设NFA $M = (K, \Sigma, f, S, Z)$, $I \subseteq K$, 则 $\varepsilon_closure(I)$ 定义如下:

- (1) $I \subset \varepsilon_closure(I)$
- (2) $M(\varepsilon_closure(I), \varepsilon) \subset \varepsilon_closure(I)$
- (3) 重复(2), 直到 $\varepsilon_closure(I)$, 不再扩大为止。

闭包运算举例

例 3.6 设NFA $M = (K, \Sigma, f, S, Z)$ 定义如下，给出计算 $\varepsilon_closure(\{3, 8\})$ 过程。



计算 $\varepsilon_closure(\{3, 8\})$ 过程演示。

NFA到DFA转换方法(子集法)

设 NFA $M = (K, \Sigma, f, S, Z)$ 则与之等价的DFA $M' = (K', \Sigma', f', S', Z')$, 其中,

(1) $K' = \rho(K)$ ($\rho(K)$ 是K全部子集之集合称为K之幂集)

(2) $\Sigma' = \Sigma$

(3) $f'(q, a) = \varepsilon_closure(M(q, a))$

(4) $S' = \varepsilon_closure(S)$

(5) $Z' = \{q \mid q \in K', q \cap Z \neq \Phi\}$

注解:

①从FA开始状态不存在路径到达的状态, 称为不可达状态。

②考虑舍弃不可达状态的转换状态之计算, 子集法可以简化从 $S' = \varepsilon_closure(S)$ 开始计算。

NFA到DFA转换方法(子集法)

设 NFA $M = (K, \Sigma, f, S, Z)$ ，子集法得到与其等价的DFA $M' = (K', \Sigma, f', S', Z')$ 之具体计算步骤可以是：

- ① 置 K' 为空集；
- ② 计算 M' 的开始状态 $S' = \varepsilon_closure(S)$ ， S' 作为 K' 新增状态；
- ③ 对于 K' 每一新增状态 q ，计算出每个 $a \in \Sigma$ 的转换状态 p ，即 $f'(q, a) = p = \varepsilon_closure(M(q, a))$ 。如果 $p \notin K'$ ，则 p 作为 K' 新增状态；
- ④ 重复③，直到 K' 不再出现新增状态为止；
- ⑤ 计算接受状态集 $Z' = \{q \mid q \in K', q \cap Z \neq \Phi\}$

例 3.7 将例 3.6 定义NFA $M = (K, \Sigma, f, S, Z)$ 确定化。

NFA到DFA转换过程演示。

DFA的最小化

确定有穷自动机DFA M 的简化是指寻找与之等价的、状态个数达到最小的DFA M' 。这样的DFA M' ，称为**最小化的DFA**。

定义3.10 设DFA $M = (K, \Sigma, f, S, Z)$, $p \in K, q \in K$, p 和 q 是等价的(记为 $p \equiv q$) 定义为:

$$p \equiv q \text{ iff } \forall \alpha \in \Sigma^* [f(p, \alpha) \in Z \Leftrightarrow f(q, \alpha) \in Z].$$

状态 p 和 q 等价的二个条件:

一致性条件: p 和 q 同时是可接受状态或不可接受状态。

所以, 对于 $\forall p \in Z, \forall q \in K - Z$, p 和 q 是不等价的。

蔓延性条件: 对所有输入符号, p 和 q 必须转换到等价的状态中。

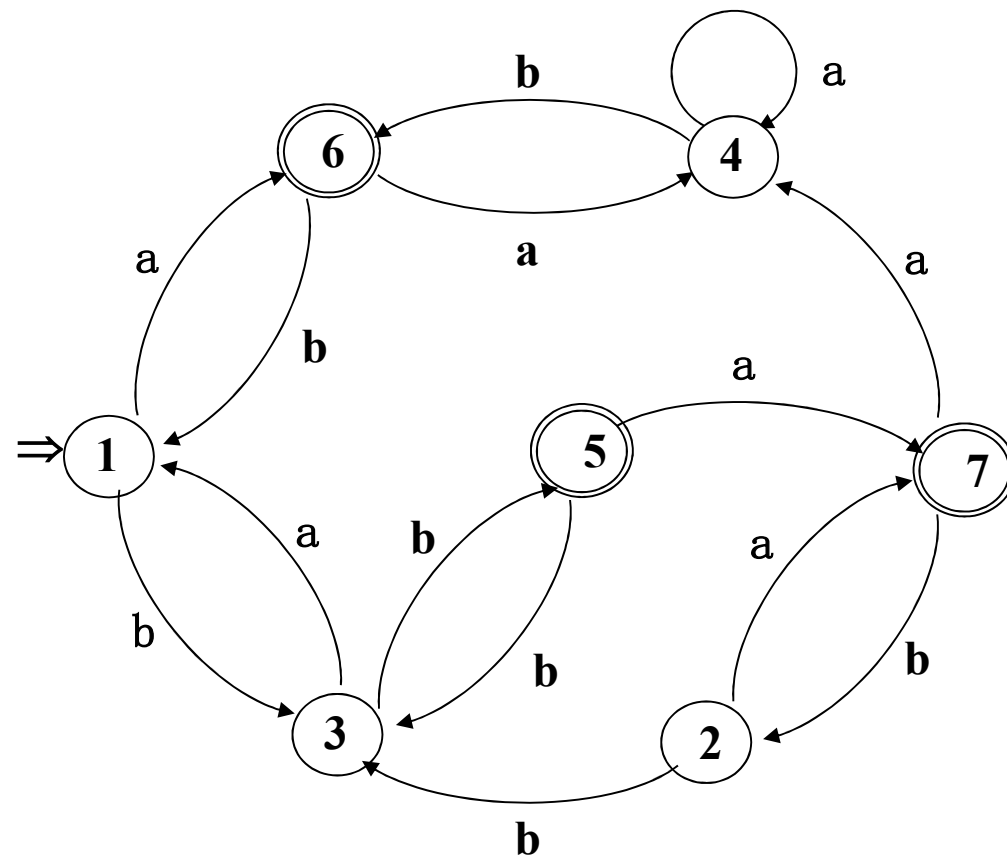
确定有穷自动机DFA M 的等价状态方法有两类:
然后将等价状态合并, 得到最小化的DFA M' 。
计算DFA M 的等价状态方法有两类:
┌ 分割法(教材采用)
└ 合并法

确定有穷自动机的最小化方法

- (1) 状态集 K 划分为两个状态子集 $\{Z, K-Z\}$ ，记为 $\Pi = \{Z, K-Z\}$ ；
 - (2) 如果 $\exists I \in \Pi \exists a \in \Sigma \exists J \in \Pi [M(I, a) \not\subset J]$ 即状态子集 $I \in \Pi$ 中至少存在两个 p 和 q ，使得 $f(p, a) \in J'$ 和 $f(q, a) \in J''$ ，且 $J' \neq J''$ （状态子集 $J', J'' \in \Pi$ ），则将 I 分割成 I' 和 I'' ，即 $I' = \{r \mid \forall r \in I [f(r, a) \in J']\}$ ， $I'' = I - J'$ ；重置划分 Π ： $\Pi \leftarrow (\Pi - \{I\}) \cup \{I', I''\}$ 。
 - (3) 置重复(2)，直到满足 $\forall I \in \Pi \forall a \in \Sigma \exists J \in \Pi [M(I, a) \subset J]$ 条件为止；
 - (4) 在DFA M 基础上，对于划分 Π 的同一个状态子集中的全部状态及其相应的转换函数合并，最后所得即为最小化的DFA M' 。
-

确定有穷自动机的最小化举例

例 4.8 将下列DFA M最小化。



DFA M最小化过程演示。

3.4 正规式和有穷自动机的等价性

定义 3.10 如果正规式 r 和有穷自动机 M , 有 $L(r) = L(M)$ 则称正规式 r 和有穷自动机 M 是等价的。

下面讨论正规式和有穷自动机相互等价转换的方法, 由此可以得知, 正规式和有穷自动机的语言表达能力是一样的。

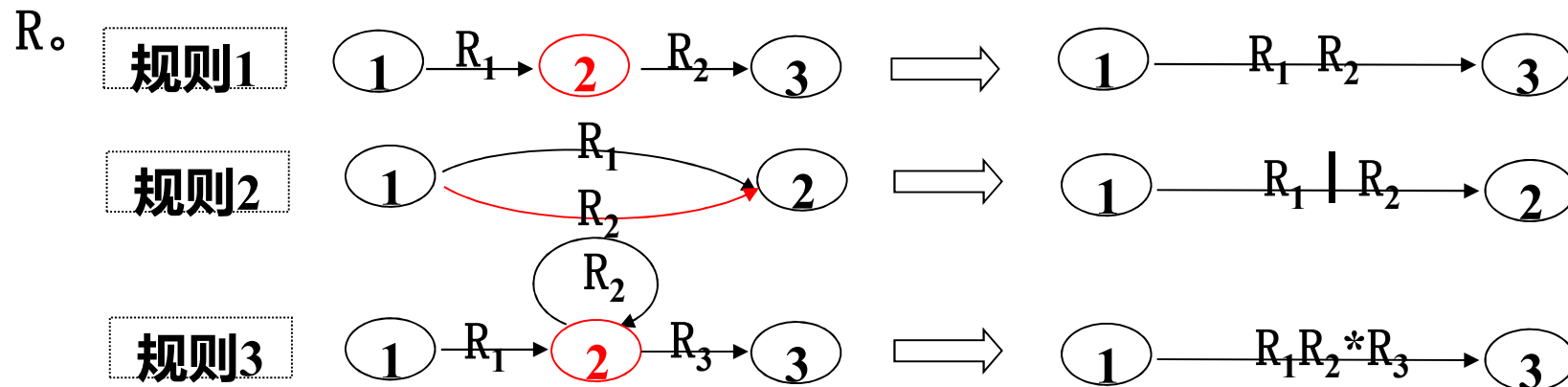
{ NFA到正规式的转换方法
正规式到NFA的转换方法

NFA到正规式的转换方法

设NFA $M = (K, \Sigma, f, S, Z)$ ，则与之等价的 Σ 上正规式 R ，可以由下列方法构造。

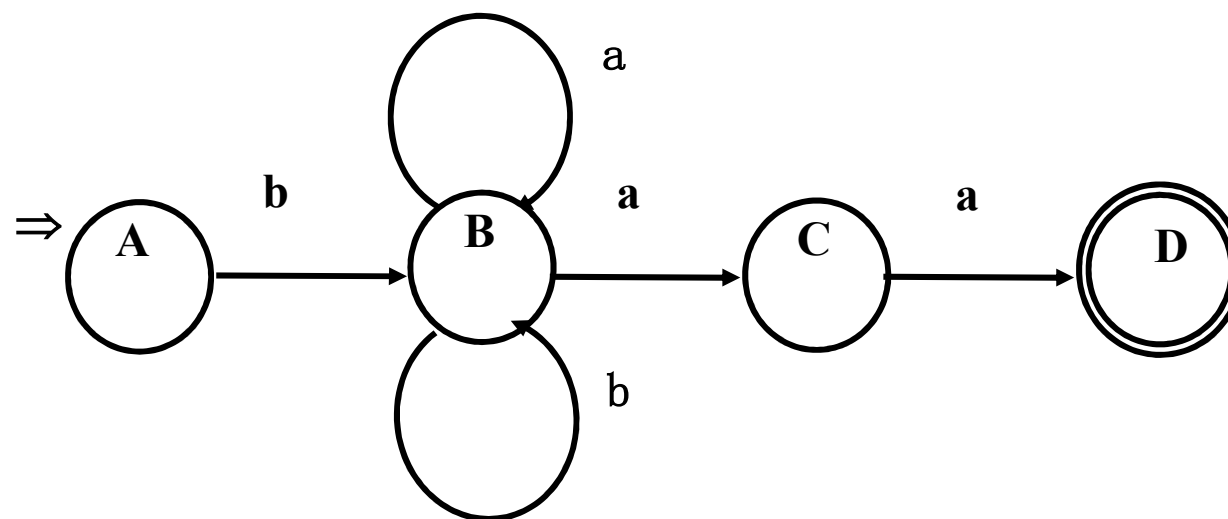
(1) 在NFA M 上，新增两个状态 X 和 Y 作为开始状态和接受状态，且将 X 经 ε 指向 M 的开始状态($\forall q \in K$ ，增加 $f(X, \varepsilon) = q$)，将 M 的开始状态经 ε 指向 Y ($\forall q \in Z$ ，增加 $f(q, \varepsilon) = Y$)。这样，得到一个与NFA M 等价的、只有唯一开始状态 X 和唯一接受状态 Y 的NFA M' ；

(2) 按下列转换规则，逐步消除NFA M' 中的状态，直到只剩下 X 和 Y 两个状态为止。弧 $\langle X, Y \rangle$ 上符号串，即为等价的 Σ 上正规式 R 。



NFA到正规式的转换方法举例

例 3.9 求与下列NFA M 等价的正规式 R 。



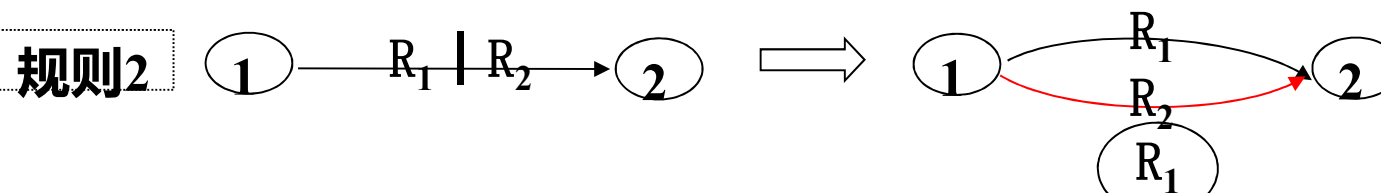
NFA M 到正规式 R 的转换过程演示。

正规式到NFA的转换方法

设 Σ 上正规式 R , 则与之等价的NFA $M = (K, \Sigma, f, S, Z)$, 可以由下列方法构造。

(1) 新增两个状态 X 和 Y 作为NFA M 的开始状态和接受状态, 且将正规式 R 作为弧 $\langle X, Y \rangle$ 上符号串。特别地, 如果 $R = \Phi$, 则保留开始状态 X 和接受状态 Y 的NFA M 即为所求。

(2) 在(1)基础上, 按下列转换规则, 逐步增加的状态, 直到弧 $\langle X, Y \rangle$ 上剩下单个符号 $a \in \Sigma \cup \{ \epsilon \}$ 为止。此刻状态图即为等价的NFA M 。



例 3.10 将正规式 $R = b(a \mid b)^*aa$ 转换成等价的NFA M 。

3.5 正规文法和有穷自动机间的转换


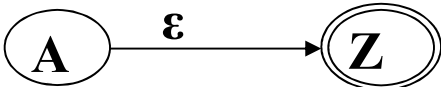

定义 4.11 如果正规文法G和有穷自动机M, 如果 $L(G) = L(M)$ 则称正规文法G和有穷自动机M是等价的。

下面讨论正规文法和有穷自动机相互等价转换的方法, 由此可以得知, 正规文法和有穷自动机的语言表达能力是一样的。

{ 右线性正规文法到NFA转换方法
左线性正规文法到NFA转换方法

右线性正规文法到NFA转换方法

设右线性正规文法 $G = (V_N, V_T, P, S)$, 则与之等价的NFA $M = (V_N \cup \{Z\}, V_T, f, \{S\}, \{Z\})$, 其中 $V_N \cap \{Z\} = \Phi$, 转换函数 f 可以由下列方法构造 :

- (1) 如果 $A \rightarrow a \in P$, 则 $f(A, a) = Z$; 
- (2) 如果 $A \rightarrow \varepsilon \in P$, 则 $f(A, \varepsilon) = Z$; 
- (3) 如果 $A \rightarrow aB \in P$, 则 $f(A, a) = B$. 

例 3.11 将下列右线性正规文法 G 转换成等价的NFA M 。

$G[Z]$:

$Z \rightarrow 0U \mid 1V, U \rightarrow 1Z \mid 1, V \rightarrow 0Z \mid 0$

右线性正规文法 G 到NFA M 转换过程演示。

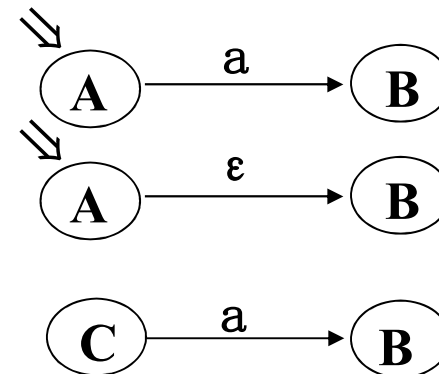
左线性正规文法到NFA转换方法

设左线性正规文法 $G = (V_N, V_T, P, S)$ ，则与之等价的NFA $M = (V_N \cup \{A\}, V_T, f, \{A\}, \{S\})$ ，其中 $V_N \cap \{A\} = \Phi$ ，转换函数 f 可以由下列方法构造：

(1) 如果 $B \rightarrow a \in P$ ，则 $f(A, a) = B$;

(2) 如果 $B \rightarrow \varepsilon \in P$ ，则 $f(A, \varepsilon) = B$;

(3) 如果 $B \rightarrow Ca \in P$ ，则 $f(C, a) = B$ 。



例 3.12 将下列左线性正规文法 G 转换成等价的NFA M 。

$G[Z]$:

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

左线性正规文法 G 到NFA M 转换过程演示。

DFA到右线性正规文法转换

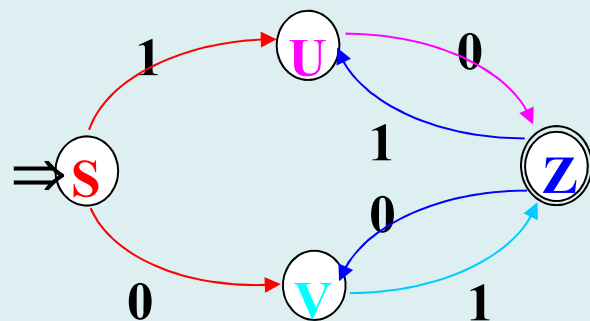
设DFA $M = (K, \Sigma, f, S, Z)$ ，则与之等价的右线性正规文法 $G = (K, \Sigma, P, S)$ ，其中规则集转换P可以由下列方法构造：

- (1) 如果 $f(B, a) = C$ ，则 $B \rightarrow aC \in P$ 。
- (2) 对接收状态 $\in Z$ ，增加 $S \rightarrow \varepsilon$

$B \xrightarrow{a} C$ $\Rightarrow B \rightarrow aC$ 再加 $B \rightarrow a$ 或 $C \rightarrow \varepsilon$

$B \xrightarrow{a} C$ $\Rightarrow B \rightarrow aC$

3.13 将下列DFA M转换成等价的右线性正规文法G。



令S为文法G开始符，得等价文法G如下。

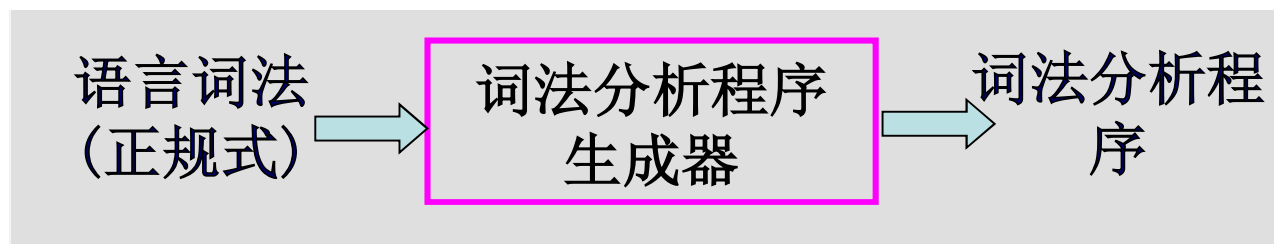
$G[S]:$

$S \rightarrow$	$1U$	$ $	$0V$
$U \rightarrow$	0	$ $	$0Z$
$V \rightarrow$	$1Z$	$ $	1
$Z \rightarrow$	$1U$	$ $	$0V$

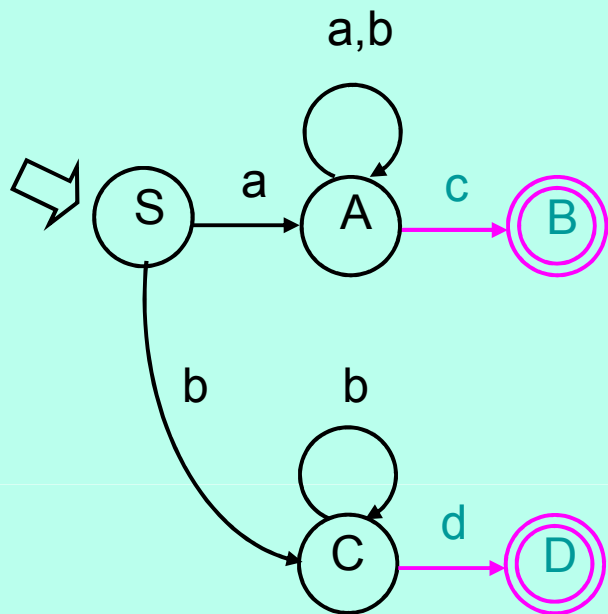
构造词法分析程序的技术线路

- (1) 依据给定的源语言之单词集，设计其正规文法或正规式；
- (2) 之后等价地转换成非确定有穷自动机；
- (3) 再通过子集法将其确定化，最终将确定有穷自动机最小化；
- (4) 最后依据最小化确定有穷自动机，设计词法分析程序。

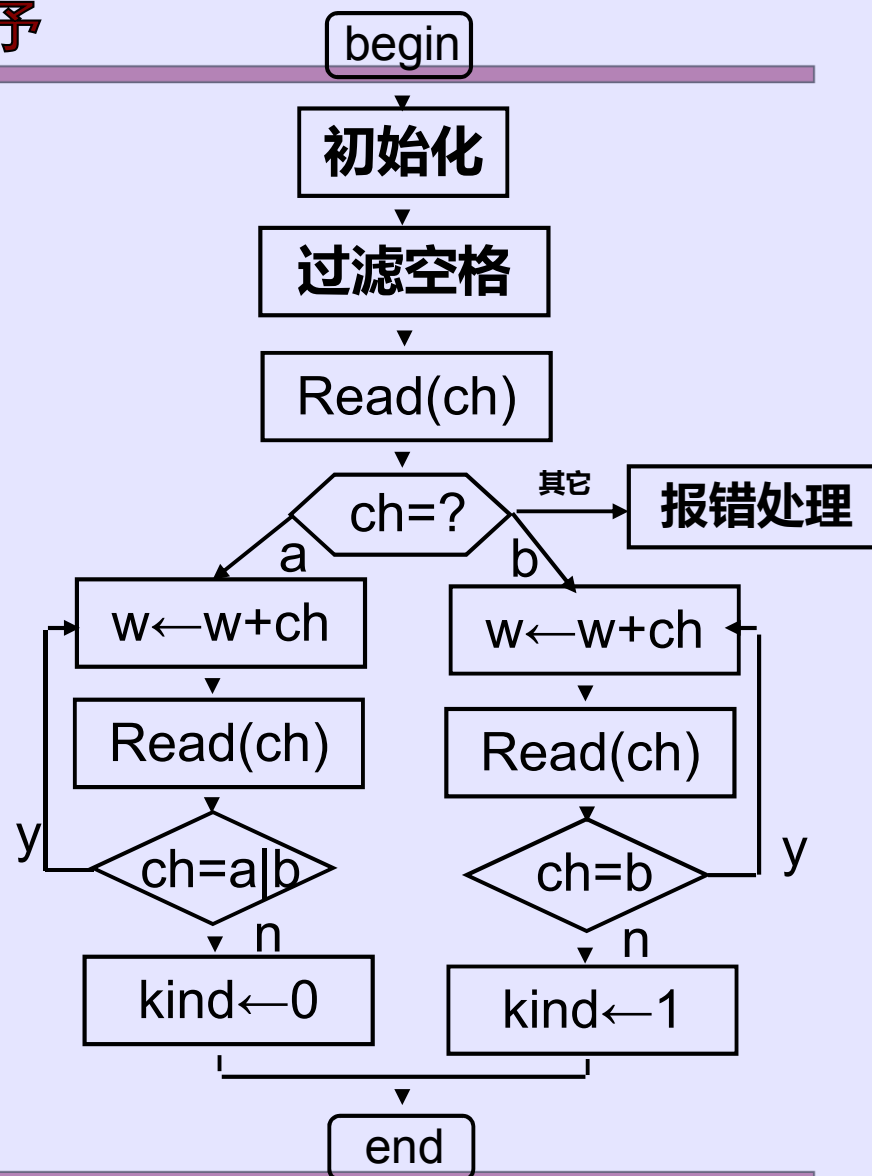
•词法分析程序生成器



根据DFA设计词法分析程序

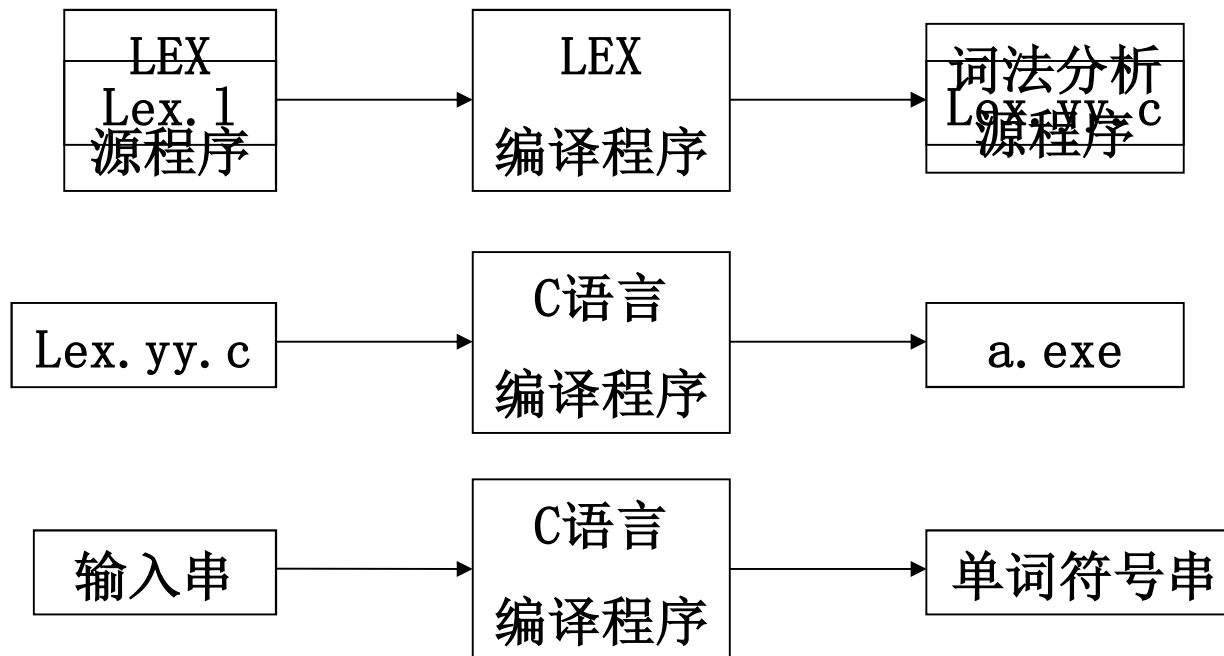


S – 单词
A – 标识符
C – 无符号整数
a – 字母
b – 数字
c – 非字母数字
d – 非数字



3.6 词法分析程序的自动构造工具（自学部分）

LEX: windows中FLEX, 词法分析程序生成工具, 实现词法分析程序的自动生成。



Lex源程序组成

% {

说明部分

% }

%%

转换规则

%%

辅助过程（用户子程序部分）

三个部分都是可选的，没有辅助过程时，第2个%%可以省略

Lex源程序举例

```
%option yylineno
%{
int yylval;
%}
chars [A-Za-z]
identifier [A-Za-z][A-Za-z0-9]*
numbers ([0-9])+
delim [ \n\t]
eq ==
as =
whitespace {delim}+
%%
{numbers}      { numcount++; yylval=atoi(yytext); }
                printf("( %d, %d) \n", 10, yylval); }
{identifier} {printf("( %d, %s) \n", 99, yytext); } ;p
{as}           {printf("( %d, %s) \n", 21, "="); } ;
```

编译Lex源程序

GNU Flex

Flex的前身是Lex，由伯克利实验室的Vern Paxson使用C语言重写了Lex，命名为Flex（Fast Lexical Analyzer Generator）。无论在效率上和稳定性上都优于Lex。以下为windows环境下的使用

（1）对lex文件lex.l进行编译，得到词法分析源程序lex.yy.c

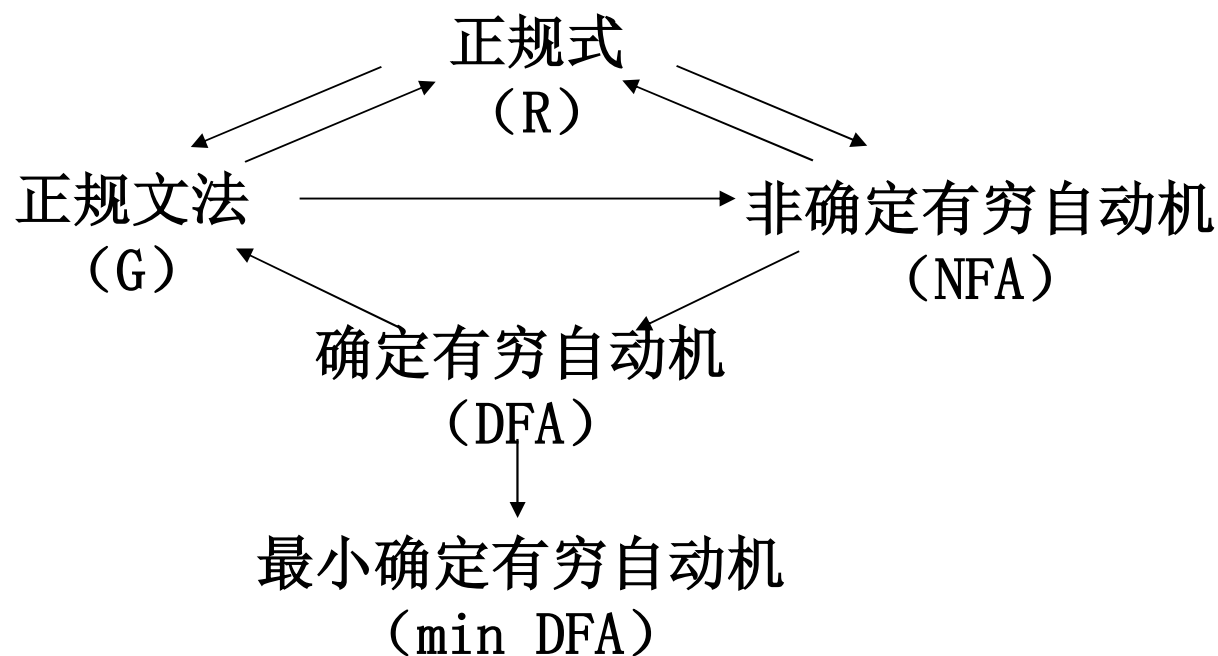
```
flex    lex.l
```

（2）使用gcc对lex.yy.c编译得到词法分析器

```
gcc    -o  scanner  lex.yy.c  -L fl
```

本章小结

本章主要介绍了词法分析程序构造的基本原理和方法，重点讨论了描述语言词法规则的3种描述工具：正规文法、正规式和有穷自动机，以及它们的相互等价地转换方法。转换方法之间关系见下图。



本章小结

提出的基本概念是正规式、非确定有穷自动机和确定有穷自动机。

构造词法分析程序的技术线路通常是：依据给定的源语言之单词集，设计其正规文法或正规式，之后等价地转换成非确定有穷自动机，再通过子集法将其确定化，最终将确定有穷自动机最小化，最后依据最小化的确定有穷自动机，设计词法分析程序。

重点掌握的内容是：

- ①设计一个定义已知语言单词集的正规文法、或正规式、或有穷自动机；
 - ②正规文法、正规式和有穷自动机的相互等价地转换方法；
 - ③正规式运算性质及其应用。
-