概念

词法分析的任务：从左至右逐个字符地扫描源程序，产生一个个的单词符号，把作为**字符串的源程序**改造成为**单词符号串的中间程序。**

- 分析和识别单词及属性。包括识别语言的关键字、标识符、常数、运算符等。
- 跳过各种分隔符，如空格、回车、制表符等。
- 删除注释
- 进行词法检查
- 建立符号表

状态转换图

为识别单词而专门设计的**有向图。**

- 结点表示状态
    - 初态是一个圆圈
    - 终态是一个同心圆
- 有向弧表示状态转移
- 弧上的标记表示在射出弧的结点状态下，可能出现的输入字符。

有限自动机

是一种识别装置，能够准确识别正规文法。

确定有限自动机DFA

一个DFA是一个五元式$M=(S,\Sigma,f,s_0,Z)$

- S是一个有限集，它的每个元素称为一个状态
- Σ是一个有穷字母表，它的每个元素称为一个输入字符
- f是一个从$S\times\Sigma$至S的单值部分映射。

  $f(s,a)=s'$意味着，当现行状态为S，输入字符为a时，将转换到下一状态s'。称s'为s的一个后继状态。
- $s_0\in S$是唯一的初态
- $Z\subseteq S$是一个终态集

DFA的确定性表现在映射f是一个单值函数。它能唯一确定一个状态。

从转换图的角度来看，如果：

1. 没有标号为ε的转换
2. 对于每个状态s和每个输入符号a，有且仅有一条标号为a的离开s的边

则它是DFA。

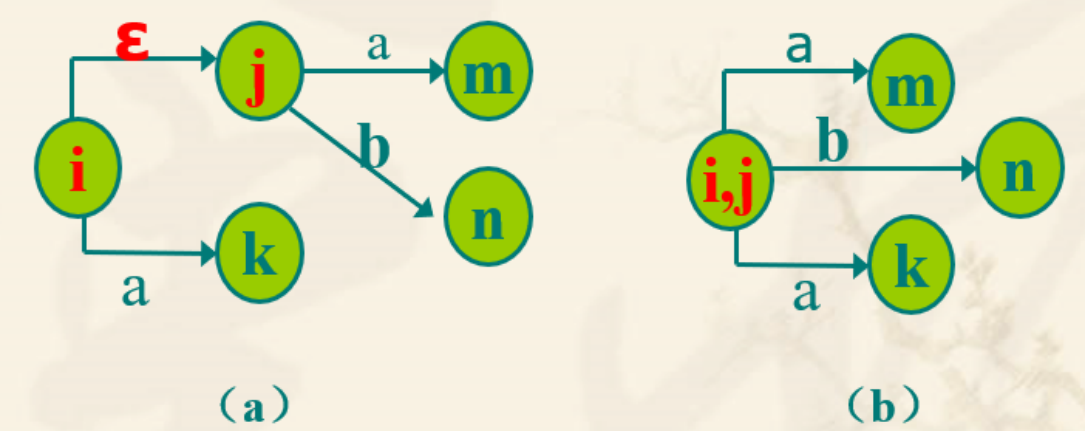
非确定有限自动机NFA

状态转换函数f是一个多值函数。

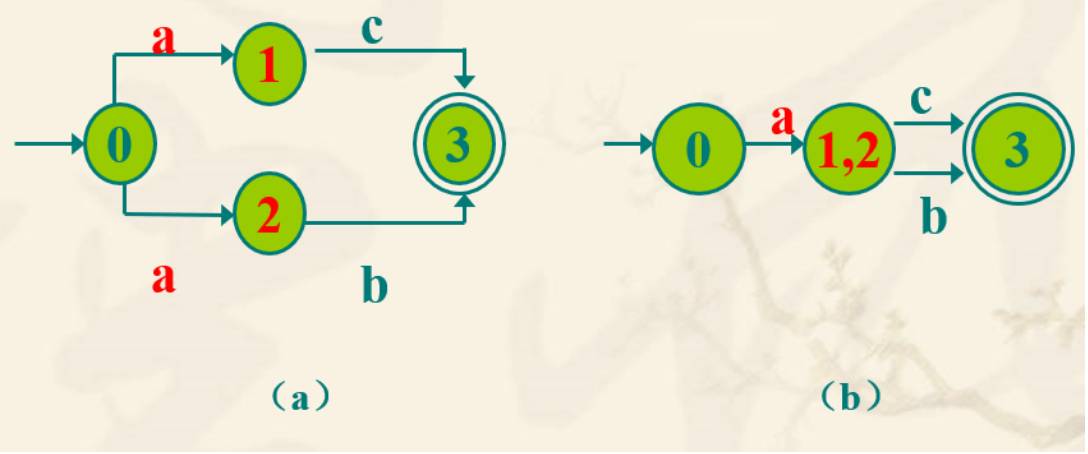
从$S \times \Sigma^*$至S的映射。

DFA到NFA的确定化

1. ε合并

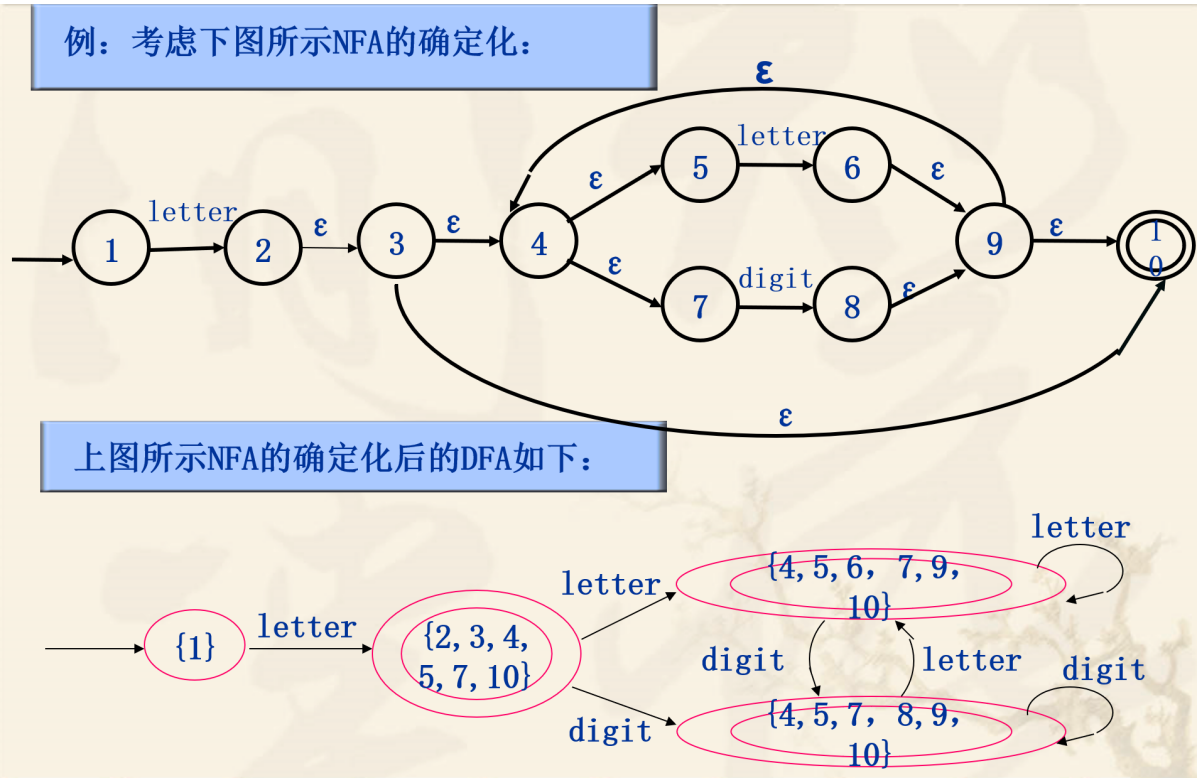


2. 状态合并



算法过程

1. 求NFA的状态转移矩阵和ε闭包
2. 建立DFA的状态转移矩阵：
    - 起始状态为 NFA的状态转移矩阵的起始状态的 ε闭包
    - 求起始状态遇到字符集中每个字符时的状态跃迁，并求闭包的并
    - 将得到的新的闭包也放入状态
3. 只要含有原来的某个结束状态，将新的也变成结束状态。



DFA的最小化

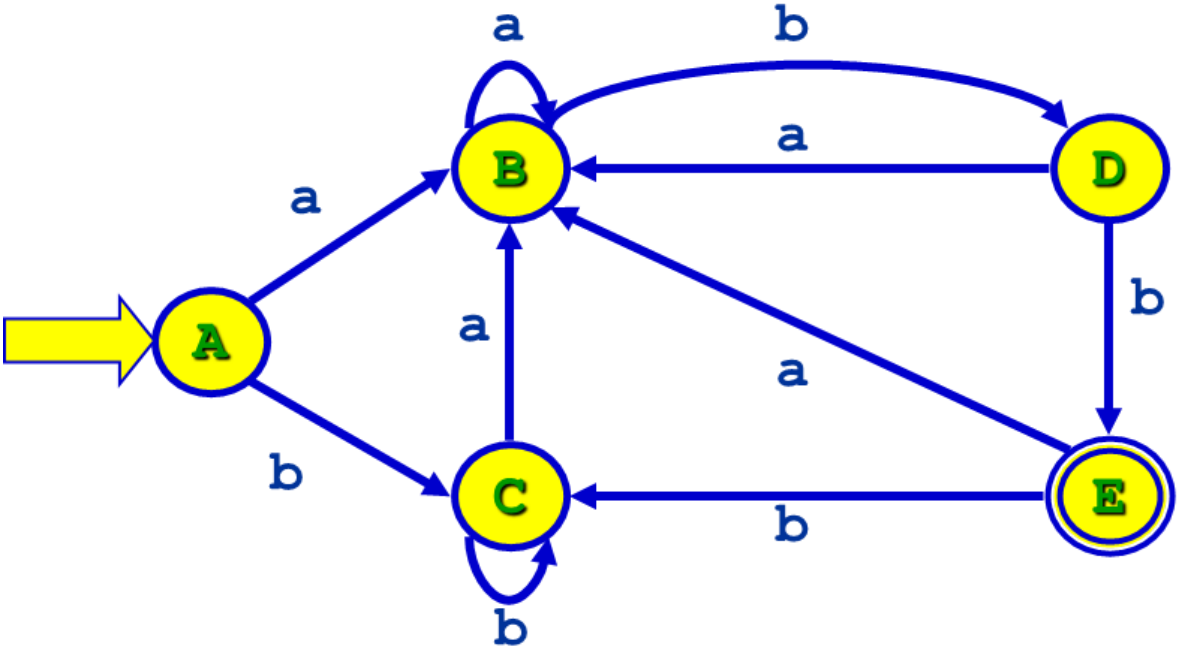
去除多余状态（不可达状态），合并等价状态。

1. 首先把Q的终态与非终态分开，分成两个子集，对每个子集中的状态考察它们是否是**可区别的**，将其中可区别的状态分裂开为不同的子集。

   不可区别的：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分划的同一子集。

2. 重复此过程，直到每个状态子集中的状态都是等价的。
3. 选取每个子集中的一个状态代表其它状态。

例：



1. 分组

非终态组{A,B,C,D}

终态组{E}

形成Π = ({A,B,C,D},{E})

2. 考察

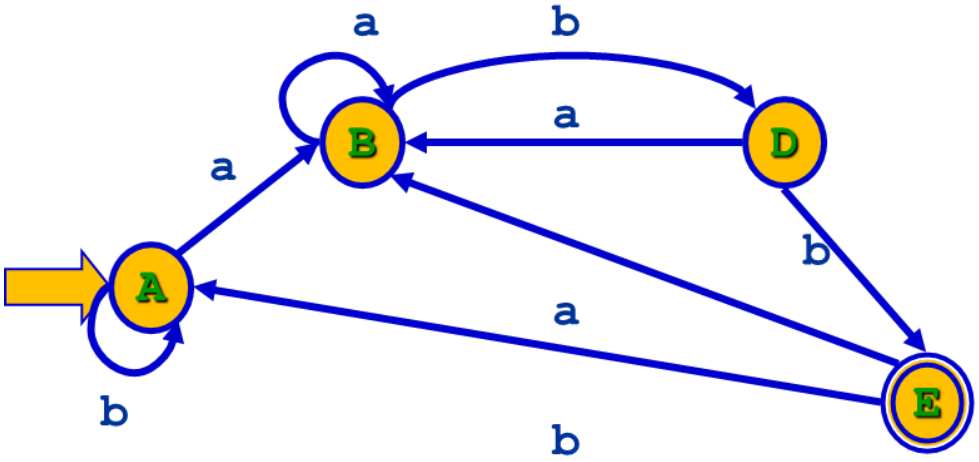
    1. 考察{E}，不能再分划。将其作为Π_{new}的一个子集
    2. 考察{A,B,C,D}

        {A,B,C,D}a = {B} $\subset$ {A,B,C,D}

        {A,B,C,D}b = {C,D,E},既不在{A,B,C,D}中，也不在{E}中。因此考虑划分{A,B,C,D}

            1. {A,B,C}b = {C,D}
            2. {D}b={E}
            3. 因此，将D单独分出来，形成

                {D},{A,B,C},放入Π_{new}
            4. 令$\Pi = \Pi_{new}$
    3. 重复上述过程，直到每个子集不可再分
    4. 在每个子集中选一个代表，并将该子集中原本和其他符号相联的弧引到新的结点。

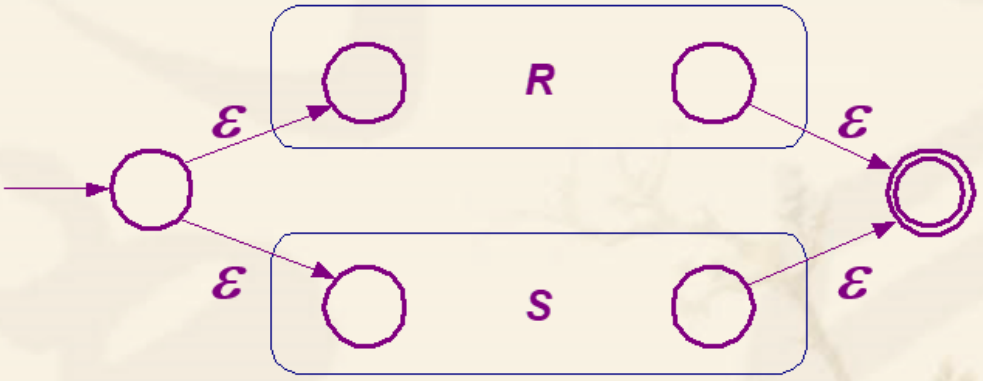


正则表达式

也称正规式，用于描述正规集（由正规式表示的语言）。

正规式到有限自动机的转换——汤普森算法

1 对于 R | S，构造为



2 对于 RS，构造为

3 对于 R*，构造为