✧ 词法分析

词法分析

- ✧ 词法分析概述
- ✧ 词法分析程序的设计与实现
- ✧ 词法分析程序的自动构造

词法分析概述

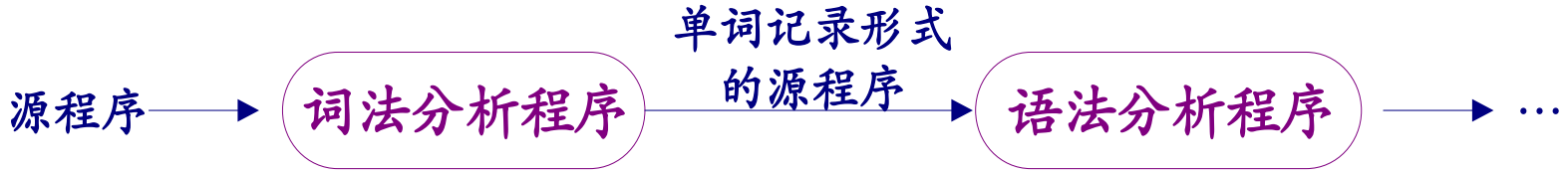
- 词法分析程序（*Lexical Analyzer*）或词法扫描程序（*Scanner*）的作用
  - 从左至右扫描构成源程序的字符流
  - 识别出有词法意义的单词（*Lexemes*）
  - 返回单词记录（由单词记号（*Token*）和单词的属性值组成），或词法错误信息
  - 除以上主要任务外，常伴有如下任务

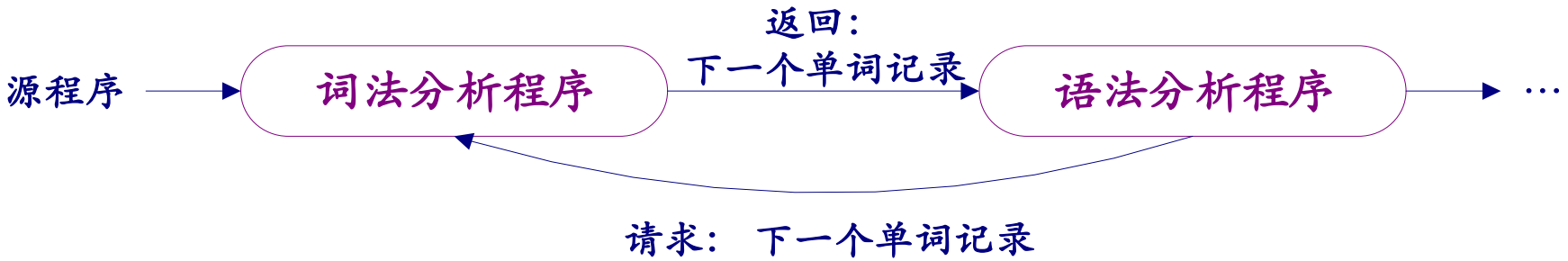
    滤掉空格，跳过注释、换行符，追踪换行标志，复制出错源程序，……

    也可能包含访问符号表的操作

词法分析概述

- 编译程序主题中如何组织词法分析程序
  - 可以作为单独的一遍
  - 较常用的方式是由语法分析程序调用
  - 基本任务都是识别单词





词法分析概述

✧ 例：Pascal 程序文本

position := initial + rate * 60;

经词法分析程序处理后，转换为下列单词序列

词法单元（对应一个单词记号）	单词属值
标识符	position
赋值算符(:=)	
标识符	initial
加算符(+)	
标识符	rate
乘算符(*)	
整数常量	60
分号(;)	

词法分析程序的设计与实现

- 常用的单词描述工具
  - 扩展巴克斯范式（EBNF）
  - 状态转换图
  - 正规表达式
  - 有限状态自动机

✧ 实例：某语言词法分析程序的设计

– 单词类别（种别）的 EBNF 描述

<无符号整数> ::= <数字>{<数字>}
<标识符> ::= <字母>{<字母>|<数字>}
<字母> ::= *a* | *b* | … | *X* | *Y* | *Z*
<数字> ::= 0 | 1 | 2 | … | 8 | 9
<保留字> ::= const | var | procedur | begin | end | odd
| if | then | call | while | do | read | write
<运算符> ::= + | - | * | / | = | # | < | <= | > | >= | :=
<界符> ::= (|) | , | ; | .

✧ 实例: 某语言词法分析程序的设计

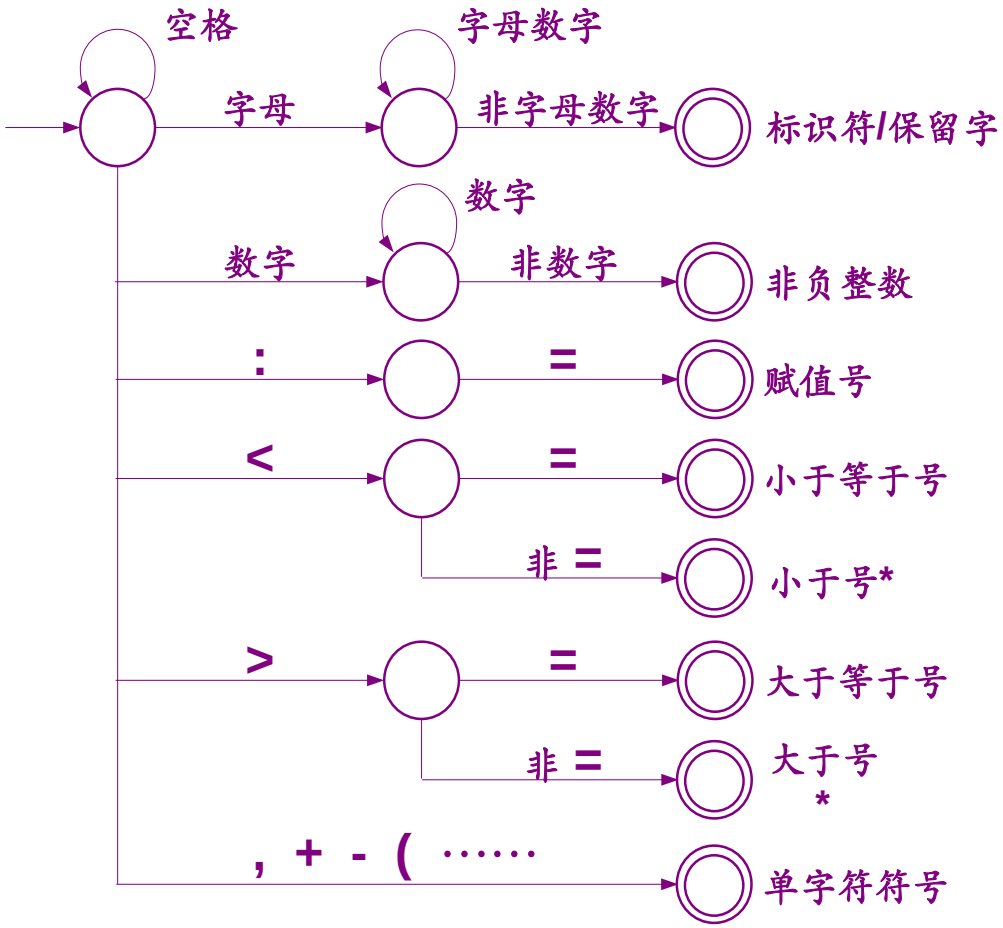
– 词法单位（单词符号）

标识符 1 个 : ident
无符号整数 1 个 : number
保留字13个 : beginsym | endsym | …
运算符11个 : plus | minus | …
界符5个 : lparen | rparen | …

<无符号整数> ::= <数字>{<数字>}
<标识符> ::= <字母>{<字母>|<数字>}
<字母> ::= *a* | *b* | … | *X* | *Y* | *Z*
<数字> ::= 0 | 1 | 2 | … | 8 | 9
<保留字> ::= const | var | procedur | begin | end | odd
| if | then | call | while | do | read | write
<运算符> ::= + | - | * | / | = | # | < | <= | > | >= | :=
<界符> ::= (|) | , | ; | .

✧ 实例: 某语言词法分析程序的设计

– 词法规则的状态转换图



✧ 技术个案

- 如何区分标识符与保留字

  预设一个保留字表，通过查表来确定是否保留字

- 字符退还

  在识别双符号运算符之类的单词时，要注意到可能需要进行字符退还

  例:在读取字符‘<’后，若下一字符不是‘=’，则识别的单词是小于号‘<’，但要退还这个非‘=’字符，以保证下一次仍读到那个非‘=’字符

✧ 其他技术个案

- ……

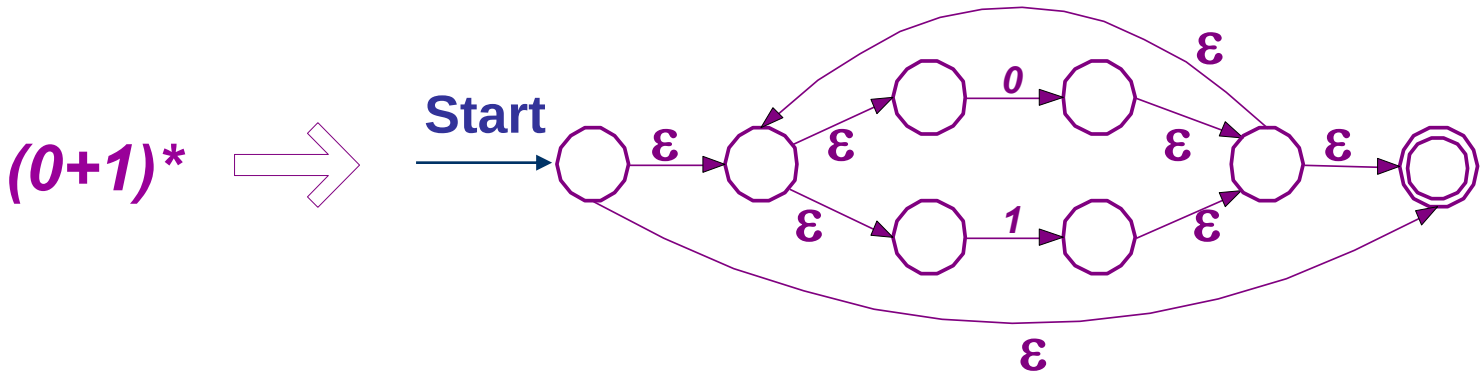
词法分析程序的自动构造

- 词法分析程序自动构造的典型过程
  - 步骤一 使用者用正规表达式作为词法规则的形式描述，每一类词法单元都对应一个正规表达式，所有正规表达式以文本方式作为自动构造工具的输入

如定义一个词法单元（0、1 串）： *(0+1)**

词法分析程序的自动构造

- 词法分析程序自动构造的典型过程
  - 步骤二 自动构造工具将每一个正规表达式转换成有限自动机的形式，比如使用 Thompson 构造法将正规表达式转换成 ε-NFA

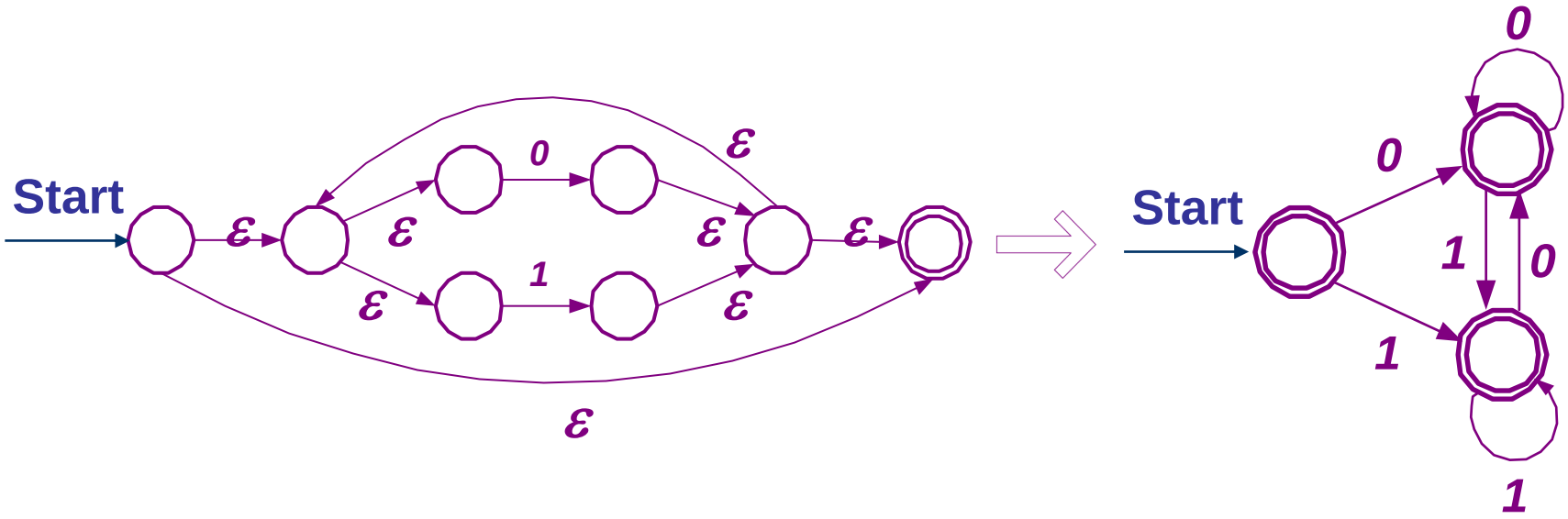


词法分析程序的自动构造

- 词法分析程序自动构造的典型过程
  - 步骤三（可选）增加一个新的开始状态，从该状态引一条 ε-转移边到上述每一个 ε-NFA 的初态，得到一个新的 ε-NFA

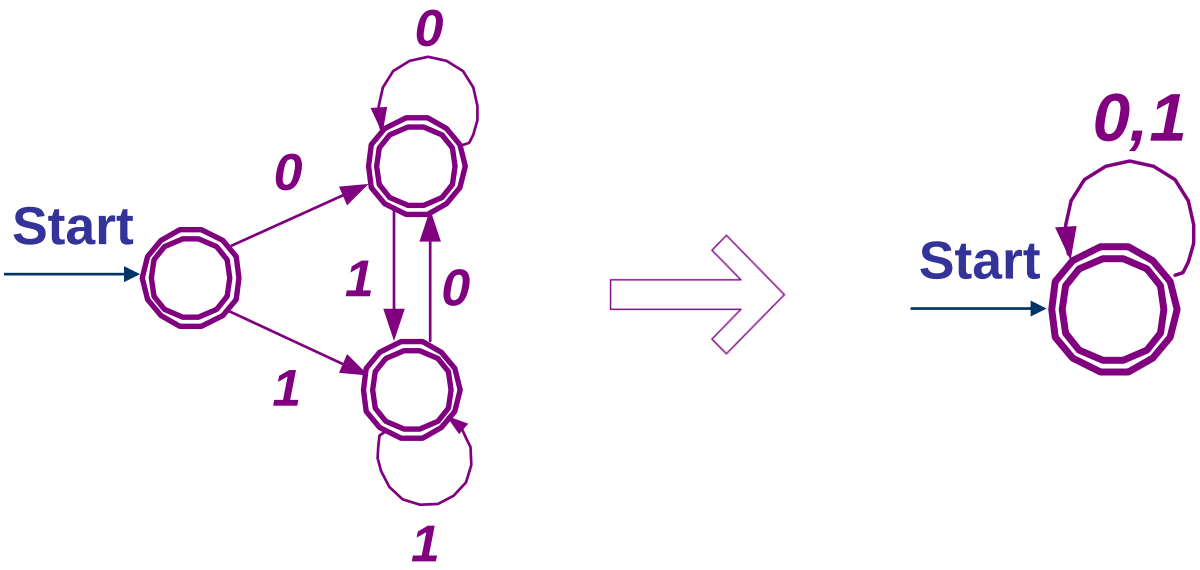
词法分析程序的自动构造

- 词法分析程序自动构造的典型过程
  - 步骤四 必要时自动构造工具会将这些 ε-NFA 确定化，比如使用子集构造法得到 DFA



词法分析程序的自动构造

- 词法分析程序自动构造的典型过程
  - 步骤五 必要时，自动构造工具会将有限自动机最小化，得到等价拥有状态数目最少的DFA



词法分析程序的自动构造

- 词法分析程序自动构造的典型过程
  - 步骤六 若执行过第3步，那么就模拟单个完整的自动机；否则，自动构造工具按照一定的控制策略生成词法分析程序中扫描程序的代码，该扫描程序可以选择对每一类词法单元所对应的有限自动机依次模拟运行，并从当前输入符号序列中识别下一个单词，然后返回相应的单词记录

词法分析程序的自动构造

- 词法分析程序自动构造的典型过程
  - 可选的正规表达式设计方法 先设计自动机

    直接设计正规表达式有时比较困难

    例如：假若想要为Java程序中所允许的注释给出正规表达式，这类注释以“/*”开始，以“*/”结束，在“/*”和“*/”之间，除了“*/”序列外，可以出现任意字符。对此类注释，某些同学可能会认为构造DFA比构造正规表达式更容易，所以可先构造DFA，然后再转换为正规表达式。

课后作业

1. 针对第 7 页中 EBNF 描述的各单词类别，设计相应的正规表达式。（非书面作业）
2. 针对第 18 页中所描述的“注释”单词类别，设计相应的正规表达式。你可以先试着直接设计这个正规表达式，若觉得复杂，就采用第18页所述的方法进行设计。如果有兴趣，还可以在首次实验开始前自行测试你的结果。（非书面作业）
3. 阅读并分析PL/0编译器的词法分析程序（课程教案中附录A）。（非书面作业）

That's all for today.

Thank You