编译原理

北方工业大学信息学院
School of Information Science and Technology,
North China University of Technology
束劼
shujie@ncut.edu.cn
瀚学楼1122，88801615

第三章 词法分析

第三章 词法分析

- 本章目录

3.1 对于词法分析器的要求
3.2 词法分析器的设计
3.3正规式与有限自动机
3.4词法分析器的自动产生

第三章 词法分析

- 大纲要求

1. 掌握：词法分析器的设计与实现方法，基于状态转换图的词法分析器的构造算法。
2. 理解：状态转化图的作用与画法。
3. 了解：对于词法分析器的要求；正规文法与有限自动机的等价性，正规式与有限自动机的等价性；词法分析器的自动产生工具LEX的基本作用。

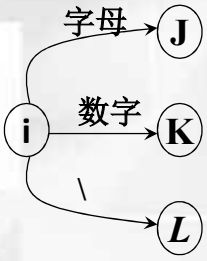
3.2.4 状态转换图的实现

3.2.4 状态转换图的实现

- 状态转换图的实现主要思想
  - 让每个状态结点对应一小段程序。
  - 对不含回路的分支结点，可以对应一个switch或一组if语句。
  - 对含回路的状态结点，可以对应一个while语句或if语句。
  - 终态结点对应一个return(code,value)语句。

3.2.4 状态转换图的实现

- 状态转换图的实现举例
  - 对不含回路的分叉结，可用一个Switch 语句或一组If-Then-Else语句实现



引进一组全局变量和过程如下：

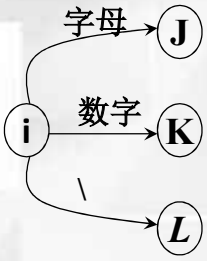
ch 是字符变量，存放最新读进的源程序字符

GetChar 子程序过程，将下一输入字符读到**ch**中，搜索指示器前移一字符位置

IsLetter( ) or IsDigit( ) 布尔函数过程，分别判断**ch**中的字符是否为字母或数字

3.2.4 状态转换图的实现

- 状态转换图的实现举例
  - 对不含回路的分叉结，可用一个Switch 语句或一组If-Then-Else语句实现




```
GetChar( );

if (IsLetter( )) {…状态J的对应程序段…;}

else if (IsDigit( )) {…状态K的对应程序段…;}

else if (ch='\') {…状态L的对应程序段…;}

else {…错误处理…;}
```

3.2.4 状态转换图的实现

- 状态转换图的实现举例
  - 对含回路的状态结，可对应一段由While结构或If语句构成的程序。

退出While循环

```
GetChar( );
While (IsLetter( ) or IsDigit( ))
    GetChar( );
…状态j的对应程序段…
```


3.2.4 状态转换图的实现

- 状态转换图的实现举例
  - 终态结点，一般对应一个形如return(code，value)的语句。

    其中,code是单词种别编码；value或是单词符号的属性值，或无定义。

  - 带*的终态结点，意味着多读了一个不属于现行单词符号的字符，必须把搜索指示器退回一个字符位置。可以由预先定义的子程序过程Retract来完成。

3.2.4 状态转换图的实现

- 状态转换图的实现举例

3.2.4 状态转换图的实现

- 状态转换图的实现举例

引进一组全局变量和过程如下：

strToken 字符数组，存放构成单词符号的字符串。

GetBC 子程序过程，检查**ch**中的字符是否为空白。若是，则调用**GetChar**直至**ch**中进入一个非空白字符。

Concat 子程序过程，将**ch**中的字符连接到**strToken**之后。例如，假定，**strToken**原来的值为“AB”,而**ch**中存放着‘C’,经调用**Concat**后，**strToken**的值就变为“ABC”。

3.2.4 状态转换图的实现

- 状态转换图的实现举例

  引进一组全局变量和过程如下：

  Reserve 整型函数过程，对**strToken**中的字符串查找保留字表，若它是一个保留字则返回它的编码，否则返回0值（假定0不是保留字的编码）。

  Retract 子程序过程，将搜索指示器回调一个字符位置，将**ch**置为空白字符。

  InsertId 整型函数过程，将**strToken**中的标识符插入符号表，返回符号表指针。

3.2.4 状态转换图的实现

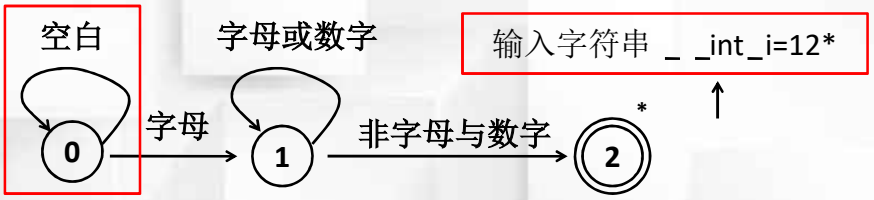
- 状态转换图的实现举例

  引进一组全局变量和过程如下：

  InsertConst 整型函数过程，将**strToken**中的常数插入常数表，返回常数表指针。

3.2.4 状态转换图的实现

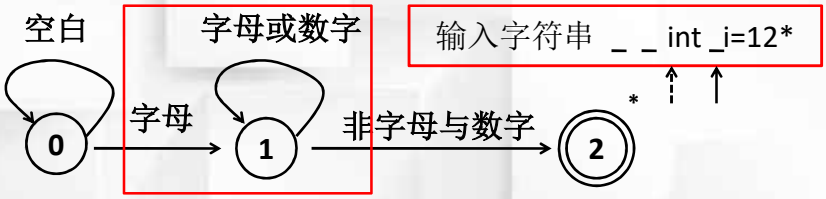
- 状态转换图的实现举例




```
int code, value;
start:   strToken : =" ";   /*置strToken为空串 */
GetChar( );   //读取下一个字符到ch
GetBC( ); //是空白，调用GetChar( )，否则下一步
```


3.2.4 状态转换图的实现

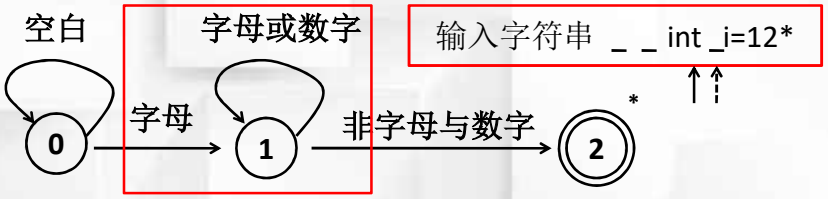
- 状态转换图的实现举例




```
if( IsLetter() ) //判断是否字母     strToken : ="int";
begin
    while (IsLetter() or IsDigit() ) //判断字母或数字
    begin
            Concat();   //加入strToken中
            GetChar();
    end;
```

3.2.4 状态转换图的实现

- 状态转换图的实现举例

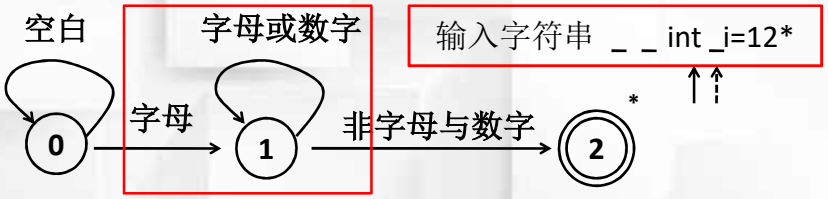


Retract();
code:=**Reserve()**;

strToken : =“int”;
Reserve (“int”);

3.2.4 状态转换图的实现

- 状态转换图的实现举例



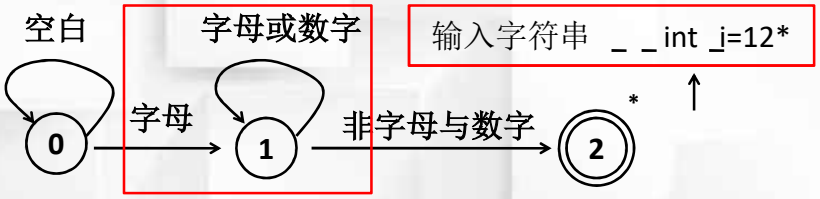
If (code=0) then
begin
value:=**Insertid(strToken)**;
return ($ID, value);
end
else Return(code, _)
End;

strToken : =“int”;

value : =“int”;

3.2.4 状态转换图的实现

- 状态转换图的实现举例



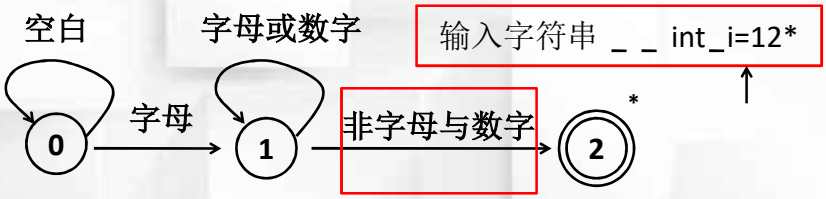
else **ProcError()** //可以设定返回程序之初

例如返回到最初

strToken : =“ ”;
GetChar( );
GetBC( );
if(**IsLetter()**)

3.2.4 状态转换图的实现

- 状态转换图的实现举例

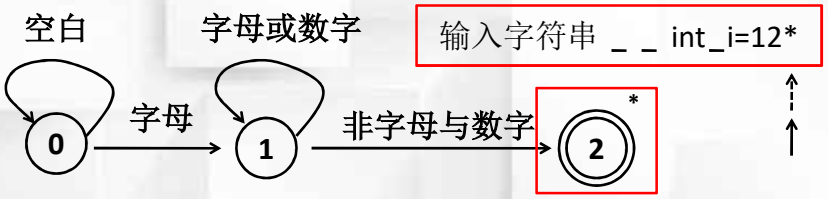


strToken : =“=”;

else if (**ch**= ‘=’) return ($ASSIGN,-);

3.2.4 状态转换图的实现

- 状态转换图的实现举例



else if (**ch**= '*')　　**strToken** : ="*";
begin
　　GetChar();
　　if (**ch**='*') return ($POWER,-);//两个星号是乘号
　　Retract(); return($STAR,-);
end

3.2.4 状态转换图的实现

- **词法分析程序的设计与实现步骤**

词法规则 ➡ 状态转换图 ➡ 词法分析程序

1. 给出描述该语言各种单词符号的词法规则，以及输出形式；
2. 画出状态转换图；
3. 设计全局变量和过程，根据状态转换图构造词法分析器。

3.3 正规式和有限自动机

3.3 正规式和有限自动机

3.3.1 正规式与正规集
3.3.2 确定有限自动机（DFA）
3.3.3 非确定有限自动机（NFA）
3.3.4 正规文法与有限自动机的等价性
3.3.5 正规式与有限自动机的等价性
3.3.6 确定有限自动机的化简

3.3.1 正规式与正规集

3.3.1 正规式与正规集

- SQL语句

  select

  Select

  sElecT

  SELECT

  selECt

SQL语句允许大小写字符任意组合

如何识别这些标识符呢？

3.3.1 正规式与正规集

- SQL语句
  select
  Select
  sElecT
  SELECT
  selECt

{S, E, L, C, T, s, e, l, c, t}

用一定的规则进行组合

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)
  T 是一组字符{A, B, …, Z, a, b,….z}，
  D是一组数字{0, 1, … 9}
  我们用下面的方式来形成任意有效的标识符

T | D or T ∪ D 表示所有字符和数字的合集，总共62个长度为1的字符串

T • D 连接积，总共520个长度为2的字符串，每个字符串包含1个字符和1个数字

T^4 所有长度为4的字符串，每个字符串只包含L中的字符

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)

  T 是一组字符{A, B, …, Z, a, b,….z}，
  D是一组数字{0, 1, … 9}
  我们用下面的方式来形成任意有效的标识符

  T * 所有T中的字符形成的字符串，加上空字符{ε}

  T •(T ∪ D)* 所有以T中字符为起始字符的字符串

  D^+ 所有字符形成的字符串，字符串至少有1个或多个数字，不包含空字符{ε}

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)

  字母表∑ 包含T、 ε和Φ
  定义在∑ 上的正规式和正规集的递归定义有三个步骤：

  ① 如果ε和Φ都是∑上的正规式，它们所表示的正规集分别为{ε}和Φ；

  ② 任何a∈∑，a是∑上的一个正规式，它所表示的正规集为{a}；

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)

字母表∑ 包含T、 ε和Φ

定义在∑上的正规式和正规集的递归定义有三个步骤：

③ 假定**U**和**V**都是∑上的正规式，它们所表示的正规集分别记为**L(U)**和**L(V)**，
那么，**(U|V)**、**(U•V)**和**(U)*** 也都是正规式，
它们所表示的正规集分别为
L(U)∪L(V)、
L(U)L(V)（连接积）
和**(L(U))***（闭包）。

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)

字母表∑ 包含T、 ε和Φ

定义在∑上的正规式和正规集的递归定义有三个步骤：

仅由有限次使用上述三步骤而得到的表达式才是Σ上的正规式。仅由这些正规式所表示的字集才是Σ上的正规集。

运算符的**优先级：**先*， 后•，最后|

- 在正规式中可以省略。

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)

例题：令∑={a,b}，下面是∑上的正规式和相应的正规集。

正规式	正规集
(a|b) ← 或	{a, b}
(a|b) (a|b)	{aa, ab, ba, bb}
a^*	∑上所有字符串，包括空字符和任意多个a的字符串。{ε, a, aa, aaa, aaaa, …}

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)

例题：令∑={a,b}，下面是∑上的正规式和相应的正规集。

正规式	正规集
$(a\|b)^*$	所有字符串，包括空字符和任意多个a或b的字符串。{ε, a, b, aa, ab, ba, bb, …}
$a\|a^*b$	{a, b, ab, aab, aaab, …}字符串a, 所有字符串包含0个a或者多个a，并且以b结尾
$(aa\|bb)(a\|b)^*$	∑上所有含有两个相继的a或两个相继的b的字符串

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)

正规式的四则运算：

U|V=V|U（交换律）
U|(V|W)=(U|V)|W(结合律)
U(VW)=(UV)W(结合律)
U(V|W)=UV|UW(分配律)
(V|W)U=VU|WU(分配律)
εU=Uε=U

35

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)

运算结合性:

| 或运算，具有交换律、结合律。
•连接积，具有结合律、和对|的分配律。
( )括弧，指定优先关系

意义清楚时，括号可以省略。

36

3.3.1 正规式与正规集

- 正规式与正规集(Regular definition and regular expressions)

  想要表示一个任意的字符串

Letter_(Letter_|digit_)*

↑

连接符

3.3.2 确定有限自动机

3.3.2 有限自动机

- 有限自动机(Finite automata)

  有限自动机是词法分析的**核心**，像状态转换图一样，它能准确地**识别正规集**，即识别正规文法所定义的语言和正规式所表示的集合，但不同的是：

  ① 有限自动机是识别器，对输入的程序给出"yes" or "no"的结果；

  ② 有限自动机分两类：

  非确定有限自动机Nondeterministic finite automata (NFA)

  确定有限自动机Deterministic finite automata (DFA)

3.3.2 确定有限自动机

- 确定有限自动机Deterministic finite automata (DFA)
- **DFA是NFA的一种特殊形式**

一个确定有限自动机(DFA) M是一个五元式

$$M = (S, \sum, \delta, s_0, F)$$

其中：

S是一个有限集，它的每个元素称为一个状态。

∑是一个有穷字母表，它的每个元素称为一个输入字符。

ε是否属于∑？ 是一个空字符串，不属于∑

3.3.2确定有限自动机

- 确定有限自动机Deterministic finite automata (DFA)

一个确定有限自动机(DFA) M是一个五元式

$M = (S, \sum, \delta, s_0, F)$

其中：

δ是一个从**S×∑** 至S的**单值映射**。
δ(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’。我们称s’为s的一个后继状态。

$s_0 \in S$，是**唯一初态**。

$F \subseteq S$，是一个终态集（可空）。

41

3.3.2确定有限自动机

- 确定有限自动机Deterministic finite automata (DFA)

•例题：DFA M=（{0,1,2,3}，{a,b}，δ，0，{3}）

δ（0,a）=1	δ（0,b）=2
δ（1,a）=3	δ（1,b）=2
δ（2,a）=1	δ（2,b）=3
δ（3,a）=3	δ（3,b）=3

42

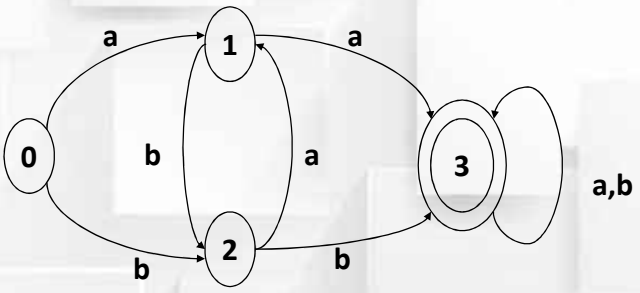
3.3.2确定有限自动机

- 确定有限自动机Deterministic finite automata (DFA)
  - •例题：

状态	a	b
0	1	2
1	3	2
2	1	3
3	3	3

3.3.2确定有限自动机

- 确定有限自动机Deterministic finite automata (DFA)
  - •例题：DFA的状态转换图



3.3.2确定有限自动机

- 确定有限自动机Deterministic finite automata (DFA)
  - ① 含有m个状态和n个输入字符
  - ② 图含有m个状态结点，从1个结点出发，**顶多有n条边**和别的结点相连接
    有限自动机的所有边都是有向边
  - ③ 每条边用∑中的1个**不同**输入字符作标记
  - ④ 整张图含有**唯一的1个初态结点**
  - ⑤ 有若干个(可以是0个)终态结点。

3.3.3 非确定有限自动机

3.3.3 非确定有限自动机

- 非确定有限自动机Nondeterministic finite automata (NFA)

一个非确定有限自动机(NFA) M是一个五元式

$$M = (S, \sum, \delta, S_0, F)$$

其中：

S是一个有限集，它的每个元素称为一个状态。

∑是一个有穷字母表，它的每个元素称为一个输入字符。包括**ε**。

3.3.3非确定有限自动机

- 非确定有限自动机Nondeterministic finite automata (NFA)

一个非确定有限自动机(NFA) M是一个五元式

$$M = (S, \sum, \delta, S_0, F)$$

其中：

δ是一个从S×（∑ ∪ **ε**）至S的**映射**。

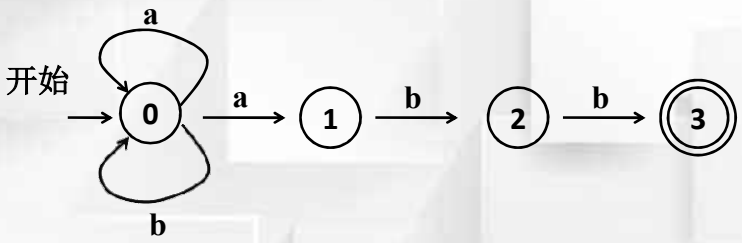
$S_0 \in S$，**是非空初态集**。

$F \subseteq S$，是一个终态集（可空）。

3.3.3 非确定有限自动机

- 非确定有限自动机Nondeterministic finite automata (NFA)

例题：如果要用NFA识别正规集(a|b)*abb。



49

3.3.3 非确定有限自动机

- 非确定有限自动机Nondeterministic finite automata (NFA)
  - 该图含有m个状态结点
  - 每个结点有**若干条边**与别的结点相连接
  - 每条边用（∑ ∪ ε）中的一个字（**可以是相同的字**，而且**可以是空字ε**）作标记(称为输入字符)
  - 整张图**至少含有一个初态结点**
  - 有若干个(可以是0个)终态结点

50

DFA vs NFA

DFA M=（S, ∑, δ, s_0, F），其中δ是单值映射函数，s_0 是唯一初态。

同一个结点出来的箭弧上，不能有重复的字符，也不能有ε

NFA M=(S, ∑, δ, S_0, F)，其中δ 是多值映射函数， S_0 为非空初态集。

同一个结点出来的箭弧上，可以重复出现同样的字符，可以有ε

有限自动机通常表示为状态转换图，它是有限自动机的非形式化描述。

51

3.3.3 非确定有限自动机

- NFA确定输入字符串的方式

  •对于$\sum^*$ 中的任何字a，若存在一条从初态结点到某一终态结点的通路，且这条通路上所有边的标记字依序连接成的字等于a，则称a可为NFA M所识别（读出或接受）。

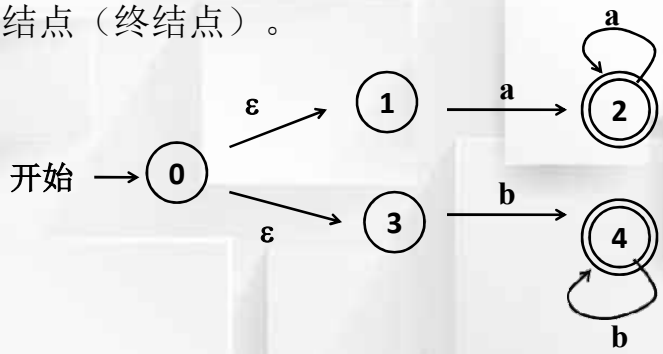
  •若M的初态结点同时又是终态结点，则空字ε可为M所识别（或接受）。NFA M所能识别的字的全体记为L(M)。

52

3.3.3 非确定有限自动机

- NFA确定输入字符串的方式

  例题：NFA确定L(aa*|bb*), 字符串aaa如果能被确定，则必定有一条通路从开始结点到某一个确定结点（终结点）。

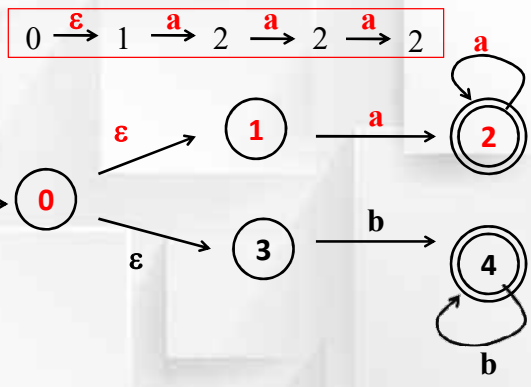


53

3.3.3 非确定有限自动机

- NFA确定输入字符串的方式

  例题：NFA确定L(aa*|bb*), 字符串aaa如果能被确定，则必定有一条通路从开始结点到某一个确定结点。



由NFA M确定的语言用L(M)表示

54

第三章 小结

第三章 小结

3.3正规式与有限自动机

3.3.1 正规式与正规集

3.3.2 确定有限自动机（DFA）

3.3.3 非确定有限自动机（NFA）

Coursework

3.2 构造一个DFA，它接受∑={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。