第 2 章　文　　法

2.1 问题的提出

2.2 文法的定义

2.3 文法的乔姆斯基体系

2.1 问题的提出

1. 什么是语言？
2. 自然语言与程序设计语言的区别？
3. 如何判断一个句子（字符串）是否属于一个语言？

巴科斯—瑙尔范式

例2.1 在类Pascal语言中，〈语句〉是用下述一组规则定义的：

〈语句〉::=〈条件语句〉|〈当语句〉|〈复合语句〉|〈赋值语句〉

〈条件语句〉::= if〈布尔表达式〉then〈语句〉else〈语句〉

〈当语句〉::= while〈布尔表达式〉 do〈语句〉

〈复合语句〉::=begin〈语句表〉end

〈语句表〉::=〈语句〉|〈语句〉；〈语句表〉

〈赋值语句〉： :=〈变量〉:=〈算术表达式〉

〈布尔表达式〉： :=〈算术表达式〉〈关系运算符〉〈算术表达式〉

〈关系运算符〉： := < | > | ≤ | ≥ | = | ≠

〈算术表达式〉： :=〈常量〉|〈变量〉|（〈算术表达式〉〈算术运算符〉〈算术表达式〉）

〈算术运算符〉： := +|-|*|/

〈常量〉： := 0|1|2|3|4|5|6|7|8|9

〈变量〉::= a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z

以上这种表示法称为巴科斯—瑙尔范式（Backus-Naur Forms），简记为BNF。

问题的提出

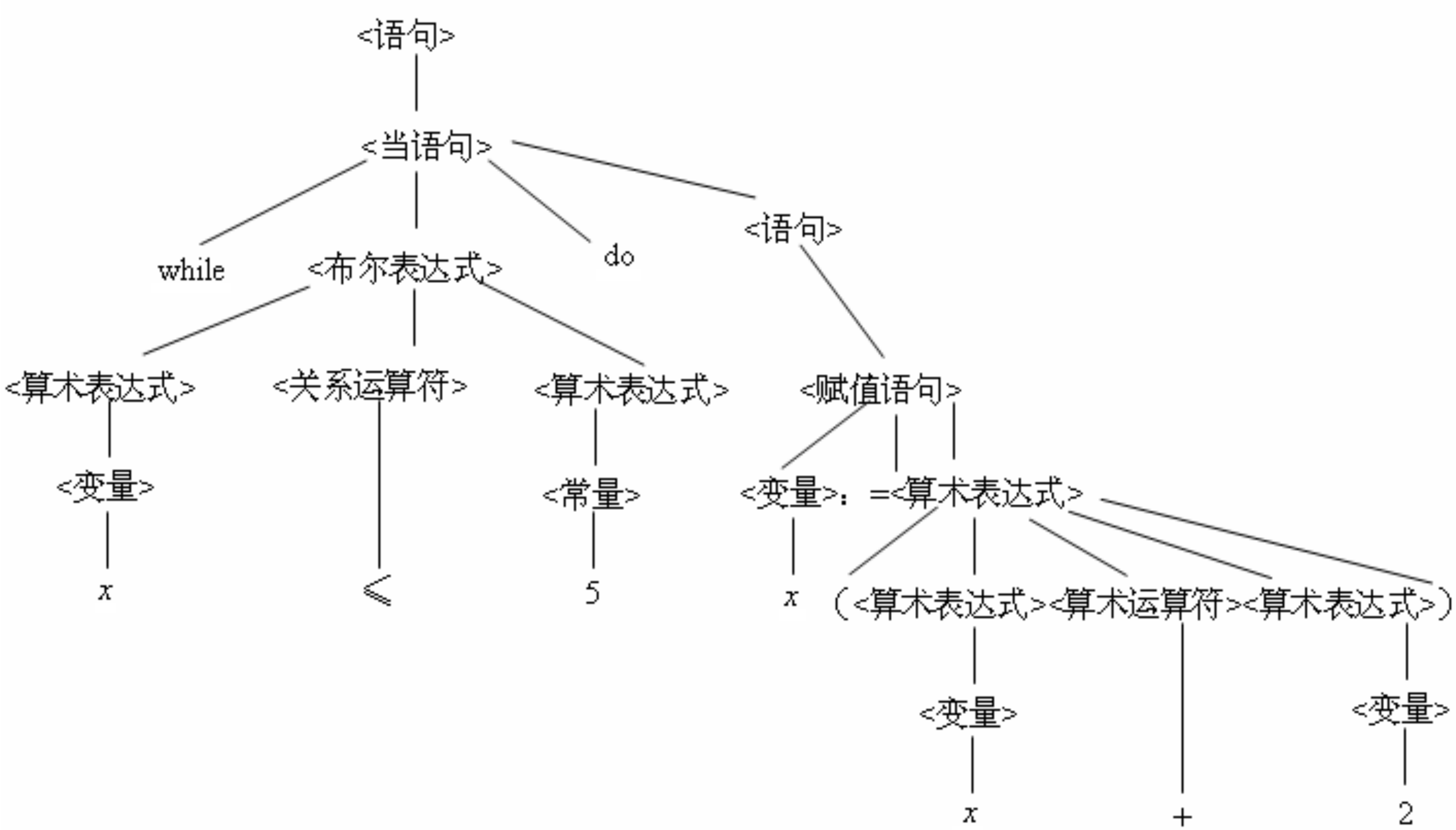
例2.2 根据例2.1中的各规则，我们指出下述的字符串

while x≤5 do x := (x+2)

是一个合法的语句。

- 它符合<当语句>的结构；
- x≤5是<布尔表达式>的一种；
- x := （x+1）是<赋值语句>的一种（从而也是<语句>的一种）；

语句的语法树/剖析树



<语句>
<当语句>
while
<布尔表达式>
do
<语句>
<算术表达式>
<关系运算符>
<算术表达式>
<赋值语句>
<变量>
<常量>
<变量>：=<算术表达式>
x
≤
5
x
（<算术表达式><算术运算符><算术表达式>）
<变量>
<变量>
x
+
2

2.1 问题的提出

对语言的研究主要包括三个方面：

1. 表示(representation)—— 无穷语言的表示。
2. 有穷描述(finite description) ——研究的语言要么是有穷的，要么是可数无穷的，这里主要研究可数无穷语言的有穷描述。
3. 结构(structure)——语言的结构特征。

解决办法：**文法**，文法可以描述语言的结构特征，而且可以产生语言的所有句子。

2.1 问题的提出---解决办法（文法）

- 所谓文法是用来定义语言的一个数学模型。
- 表示语言的方法：
  1. 若语言L是有限集合，可用列举法
  2. 若L是无限集合（集合中的每个元素有限长度），用其他方法。

     ① 方法一：文法产生系统，由定义的文法规则产生出语言的每个句子

     ② 方法二：机器识别系统，当一个字符串能被一个语言的识别系统接受，则这个字符串是该语言的一个句子，否则不属于该语言。

2.2 文法的定义

定义2. 1 一个文法G是一个四元组 $G = (V, T, P, S)$，其中

①V（Variables）是变元的有限集。

②T（Terminal symbols）是终结符的有限集。

③P（Productions）是产生式的有限集，其中每个产生式都是$\alpha \rightarrow \beta$的形式，其中$\alpha \in (V \cup T)^{+}$，且其中至少有一个V中的符号，$\beta \in (V \cup T)^{*}$。α称为**产生式的左部**，β称为**产生式的右部**。

④$S \in V$，称为文法G的开始符号（**Start variable**）。

例2.3 下面的四元组都是文法。

$G_1 = (\{A, B\}, \{0,1\},\{A\rightarrow 0B, B\rightarrow 1B, B\rightarrow 0\}, A)$。

$G_2 =(\{A,B,C\},\{a,b,C\},\{A\rightarrow aBC, B\rightarrow b, C\rightarrow CC,$

$C\rightarrow \varepsilon\}, A)$。

约 定

- 有关文法的例子，都遵循下述的约定：
  ①大写拉丁字母A,B,C,D,E和S等等表示变元，除非另做说明，S表示开始符号。
  ②小写拉丁字母a,b,c,d,e数字等等表示终结符。
  ③小写拉丁字母u, v, w, x, y, z等等表示终结符串。
  ④小写希腊字母α，β，γ等等表示变元和终结符共同组成的串。

- 另外我们还约定，同一个文法中如果有若干个左部相同而右部不同的产生式，如

  $\alpha \rightarrow \beta_1$，$\alpha \rightarrow \beta_2$，…，$\alpha \rightarrow \beta_n$

  则可以缩写为

  $\alpha \rightarrow \beta_1 | \beta_2 | \ldots | \beta_n$

文法与形式语言

例 2.4　在以上的约定下，当我们要写一个文法时，只写出它的产生式集合也是可以的。如我们写出：

（1） $S \to 0A1|10$

（2）$0A \to 00A1$

（3）$A \to \varepsilon$

就表示该文法

G＝（ {S,A},
{0,1},
{S→0A1, S→10, 0A→00A1, A→ε},
S）

- **定义2.2** 给出文法 G = (V, T, P, S)，我们定义两个字符串之间的一个关系“$\underset{G}{\Rightarrow}$”：若 $\alpha = \alpha_1\alpha_2\alpha_3$, $\gamma=\alpha_1\beta\alpha_3$, 并且 $\alpha_2 \to \beta$是P中的一个产生式，则有 $\alpha \underset{G}{\Rightarrow} \gamma$，此时称由α**直接推导(derives)**出γ。根据第一章关于集合上关系的闭包的定义，我们也可将 $\underset{G}{\Rightarrow}$扩充为 $\overset{*}{\underset{G}{\Rightarrow}}$，将$\alpha \overset{*}{\underset{G}{\Rightarrow}} \gamma$称为由α**推导**出γ。

- 若有$S \overset{*}{\underset{G}{\Rightarrow}} \gamma$，则称γ为**句型(sentential form)**，当$\gamma \in T^*$，则称γ为**句子(sentence)**。
- 对应于推导，还有一个重要的概念，称为“归约”(reduce)。其定义是：如果$\alpha \underset{G}{\Rightarrow} \gamma$是由α到γ的推导，则反过来称γ归约到α，记作$\gamma \underset{G}{\Leftarrow} \alpha$。

文法与形式语言

（1） $S \rightarrow 0A1|10$

（2） $0A \rightarrow 00A1$

（3） $A \rightarrow \varepsilon$

例2.5 对于例2.4中给出的文法G，我们有：

$$S \underset{G}{\Rightarrow} 0A1 \underset{G}{\Rightarrow} 00A11 \underset{G}{\Rightarrow} 000A111 \underset{G}{\Rightarrow} 000111$$

第一步直接推导用的是第（1）个产生式，第二步直接推导用的是第（2）个产生式，第三步直接推导还是用第（2）个产生式，最后一步直接推导用的是第（3）个产生式。总起来我们也可以写为

$S \overset{*}{\underset{G}{\Rightarrow}} 000111$。在这个推导中，0A1，00A11，000A111，000111都是句型，而000111又是句子。

在今后写推导式子的时候，若所指的文法是明确无误的，则可将记号 $\underset{G}{\Rightarrow}$ 或 $\overset{*}{\underset{G}{\Rightarrow}}$ 中的G省略，只写$\Rightarrow$ 或$\overset{*}{\Rightarrow}$即可。另外，如果α经过i步的直接推导到β，就可写$\alpha \overset{i}{\Rightarrow} \beta$。

文法与形式语言

定义2.3 给出文法 G=(V,T,P,S)，它所产生的语言记作 L (G)，定义如下：

$$L(G)=\{\boldsymbol{\omega}|S \overset{*}{\Rightarrow} \boldsymbol{\omega}，并且\boldsymbol{\omega} \in T^*\}。$$

换句话说，文法G产生的语言 L (G)，就是由G中开始符号S推导出来的全体终结符号串所构成的集合，也就是句子的集合。

文法→语言

例2.6 给出文法G,它有两个产生式:

$$S \rightarrow aSb$$
$$S \rightarrow ab$$

根据L(G)的定义，考虑从S的推导，若先用G中第二个产生式，则$S \Rightarrow ab$，就不能再往下推导了，此时相当于语言中$n=1$的情况。若从S出发，先用第一个产生式$n-1$次，即$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow \ldots \Rightarrow a^{n-1}S\,b^{n-1}$，最后再使用第二个产生式一次，得到$S \overset{*}{\Rightarrow} a^n b^n$，这个推导对于任何$n>1$都是对的。

再加上$n=1$的情况，即可得到$L(G)=\{a^n b^n \mid n \geq 1\}$。

思考题： 文法 $S \rightarrow aSb$，$S \rightarrow \varepsilon$ 生成什么语言？

语言→文法

例2.7 给出语言 $L=\{a^n \mid n\geq 1\}$，找出产生它的文法。

L＝{a，aa，aaa，…}，它是一个无限集。因此必须先产生出一个a来，我们首先用产生式S→a来实现。因为L是无限集，必须用递归的方法，以一个a为基础，不断地添加一个a。即再用一个产生式S→aS，与第一个产生式合起来，整个文法就是：

S→a

S→aS

当然，产生L的文法不是唯一的，我们也可以用以下两个产生式

S→a

S→Sa

还可以用文法?

S→aS

S→ε

文法等价

定义2.4　对于两个不同的文法G_1＝(V_1，T_1，P_1，S_1），G_2＝(V_2，T_2，P_2，S_2)，如果L(G_1)＝L(G_2)，则称文法G_1与G_2等价。

同一个语言可以由不同的文法产生。在例2.7中已经看到，一个很简单的语言$\{a^n|n\geq 1\}$就可由两个不同的文法产生。

$S \rightarrow a$	$S \rightarrow a$
$S \rightarrow aS$	$S \rightarrow Sa$

2.3 文法的乔姆斯基体系

定义 2.5 对于文法 G＝（V，T，P，S）按产生式分为四类：

①若 P 中的产生式，不加另外的限制，则G称为0型文法，或短语结构文法（Phrase Structure Grammar, PSG）。

②若 P 中每个产生式$\alpha \rightarrow \beta$都满足条件$|\alpha| \le |\beta|$，则G称为1型文法，或上下文有关文法（ Context-Sensitive Grammar, CSG ）。

③若 P 中每个产生式都具有如下形式：

$A \rightarrow \beta$，$\beta \in (V \cup T)^*$，$A \in V$，则称G为2型文法，或上下文无关文法（Context-Free Grammar, CFG ）。

④若 P 中每个产生式都具有如下形式：

$A \rightarrow a$或$A \rightarrow aB$，$a \in T \cup \{\varepsilon\}$，$A$，$B \in V$，

则称G为3型文法，或正则文法（Regular Grammar, RG ）。

2.3 文法的乔姆斯基体系

例 2.8　给出文法G

① $S \to ACaB$

② $Ca \to aaC$

③ $CB \to DB$

④ $CB \to E$

⑤ $aD \to Da$

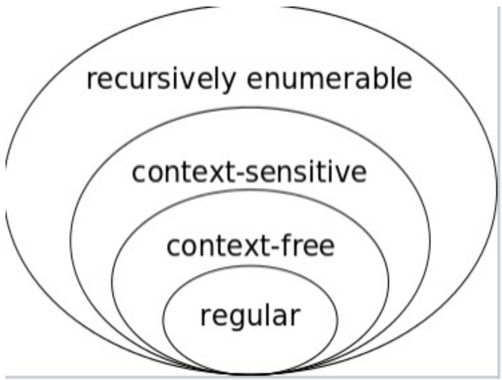
⑥ $AD \to AC$

⑦ $aE \to Ea$

⑧ $AE \to \varepsilon$

文法G是一个“真正的”0型文法，由于有产生式（4）和（8）的存在（产生式左部的长度大于右部的长度），它不是1型文法，当然更不是2型，3型文法。

2.3 文法的乔姆斯基体系**总结**



乔姆斯基文法体系

Avram Noam Chomsky

抽象模型	对应语言	相当于程序或算法
有穷自动机（**FA**）	•正则语言（**RL**）**3**型	If ,case ,goto, 无变量（内存）无数组
下推自动机（**PDA**）	前后文无关语言（**CFL**），**2**型	增加：　堆栈。仍无变量（内存）无数组
线性界限自动机（**LBA**）	前后文有关语言（**CSL**）, 1型	
图灵机（**TM**）	递归可枚举（r.e.）, 0型	输入在语言外时，可能死循环,

同济大学 TONGJI UNIVERSITY

2.3 文法的乔姆斯基体系

- 线性文法(linear grammar)
  - 设G=(V，T，P，S)，如果对于$\forall \alpha \to \beta \in P$，$\alpha \to \beta$均具有如下形式：
  - $A \to w$
  - $A \to wBx$
  - 其中A，$B \in V$，w，$x \in T^*$，则称G为线性文法。
- 线性语言(linear language)
  - L(G)叫做线性语言

2.3 文法的乔姆斯基体系

- 右线性文法(right linear grammar)
  - 设G=(V，T，P，S)，如果对于∀ $\alpha \to \beta \in P$，$\alpha \to \beta$ 均具有如下形式：
  - $A \to w$
  - $A \to wB$
  - 其中A，B∈V，w，x∈T^+，则称G为右线性文法。
- 右线性语言(right linear language)
  - L(G)叫做右线性语言。

2.3 文法的乔姆斯基体系

- 左线性文法(left linear grammar)
  - 设G=(V，T，P，S)，如果对于∀α→β∈P，α→β均具有如下形式：
  - A→w
  - A→Bw
  - 其中A，B∈V，w，x∈T^+，则称G为左线性文法。
- 左线性语言(left linear language)
  - L(G)叫做右线性语言。

可以证明：左线性文法与右线性文法等价的。

正则文法是右线型文法吗？

本章小结

1、文法作为语言的描述，不仅可以描述语言的结构特征，而且还可以产生这个语言的所有句子。从而，解决了语言的有穷描述，且提供了语言归属问题的判定方法（或计算问题的思路）。

Exp. S=‘001100’，L={x|x=0^n1^n，x∈Z^+}，问题：S属于语言L吗？

2、文法的形式化定义、派生与规约；

3、0型~3型文法的区别在于：**产生式的形式**。

4、左线性文法+右线性文法 ≠正则文法

5、文法的构造

6、上下文有关与上下文无关的区别