第三章：语法分析

文法与语言

文法的分类

文法的相关概念

内容介绍

- 文法与语言
- 文法的相关概念
- 文法的分类

1.1 语言与文法

- 文法的产生：自然语言与程序设计语言的区别
- 语言的三个基本要素：语法、语义、语用

1.1 语言与文法

- 文法能**清晰**地描述程序设计语言的语法构成易于理解。
- 文法能**自动**地构造有效的语法分析器，检查源程序是否符合语言规定的语法形式。
- 文法定义可以了解程序设计语言的**结构**，有利于将源程序转化为目标代码，以及检查出语法错误。
- 基于文法实现的语言易于**扩展**。

1.2 巴克斯范式BNF

- BNF:

语法成分::=组成部分

赋值语句::=变量=表达式

变量::=标识符|下标变量|指针变量...

表达式::=算术表达式|逻辑表达式...

1.3 上下文无关文法

- 上下文无关文法的规则：

  P(语法符号)→$X_1X_2...X_n$

  赋值语句→变量=表达式

- **文法**G定义为四元组(V_T,V_N,S,P)

V_T是有限的终极符集合

V_N是有限的非终极符集合

S是开始符，$S \in V_N$

P是产生式（规则）的集合

1.3 上下文无关文法

- 例子和简写:
- G={V_N,V_T,P,S}

  V_N={N,D} V_T={0,1,2,...,9}

  P={N→D, N→ND, D→0|1|...|9多条规则}

  S=N

基本概念(1)

- **直接推导**：如果$A \to \beta$是一个产生式，则有$\alpha_1 A \alpha_2 \Rightarrow \alpha_1 \beta \alpha_2$,其中$\Rightarrow$表示一步推导。这时称$\alpha_1 \beta \alpha_2$是由$\alpha_1 A \alpha_2$直接推导的。

  $\Rightarrow$的含义是，使用一条规则，代替$\Rightarrow$左边的符号，产生$\Rightarrow$右端的符号串。

- $A \Rightarrow^+ \beta$:表示A通过一步或多步可推导出β
- $A \Rightarrow^* \beta$:表示A通过零步或多步可推导出β

基本概念(1)

- **句型**：设有文法G，如果有$S \Rightarrow^* \beta$，则称符号串β为G的句型 。我们用SF(G)表示文法G的所有句型的集合 $\beta \in (V_T \cup V_N)^*$
- **句子**：设β为文法G的一个句型，且β只包含终极符，则称β为G的句子 $\beta \in V_T^*$
- **语言**： $L(G)=\{ \beta \mid S \Rightarrow^+ \beta , \beta \in V_T^* \}$。 文法G所定义的语言是其所有句子的集合。

1.4 基本概念与高级语言语法

- 每一种高级程序设计语言都有自己的语法
- 符合高级语言文法的程序就是该语法的句子
- 所有程序组成的集合就构成了语言。

1.4 基本概念与高级语言文法

- PASCAL语言语法例子:

<程序>⇒<首部><声明部分><程序体>

<首部>⇒program (<运行环境>)

<声明部分>⇒var <标号声明><常量声明><类型声明><过程函数声明>

<程序体>⇒begin<条件语句><循环语句><赋值语句><输入输出语句><过程调用语句>...end

1.5 验证程序是不是语法的句子

- 最基本的方法：采用类搜索算法，从开始符出发进行推导
- 用剪枝来提高效率

两种基本思想

- **自顶向下的语法分析**：从开始符出发，根据定义看是否可以推导出程序。
- **自底向上的语法分析**：从程序出发，用规则的左部替换规则的右部，一直到规约成开始符。

基本概念(2)

- **短语**

  设S是文法的开始符，有$S \Rightarrow^* \alpha_1 A \alpha_2$，如果有$A \Rightarrow^+ \beta$ 则称β是句型$\alpha_1\beta\alpha_2$的一个短语。

- **简单短语**

  设S是文法的开始符，有$S \Rightarrow^* \alpha_1 A \alpha_2$，如果有$A \Rightarrow \beta$ 则称β是句型$\alpha_1\beta\alpha_2$的一个简单短语。

- **句柄**

  句型中的最左简单短语称为句柄。

1.6 例子

E →E+T| T

T → T *F| F

F → (E) | i （标识符）

这个文法的开始符就是E，非终极符是E T F

终极符就是+ * ( )

- 问：(T+i)*F+i 是不是文法的句型，有哪些短语，简单短语，句柄是哪个？

1.6 例子

E⇒ E+T
⇒ T+T
⇒ T*F+T
⇒ F*F+T
⇒ (E) * F+T
⇒ (E + T) * F+T
⇒ (T + T) * F + T
⇒ (T + F) * F +T
⇒ (T + i) * F+ T
⇒ (T + i) * F + i

1.6 例子

- 短语

(T+i)*F+i，整个的肯定是短语由E推导出来

(T+i)*F 是由T推出来的

两个i；　　（T+i）；T；T+i;

- 简单短语

T　两个i

- 句柄

T

基本概念(3)

设有文法G

- **直接左递归**：若有$E \to E\alpha$形式的规则，则称E是直接左递归。
- **直接右递归**：若有$E \to \alpha E$形式的规则，则称E是直接右递归。
- **左递归**：若有$E \Rightarrow^{+} E\alpha$，则称E是左递归。
- **右递归**：若有$E \Rightarrow^{+} \alpha E$，则称E是右递归。
- **递归**：若有$E \Rightarrow^{+} \alpha_1 E \alpha_2$

基本概念(3)

最左（右）推导:

如果进行推导时选择的是句型中的最左(右）非终极符，则称这种推导为最左(右)推导，并用符号$\Rightarrow_{lm}$（$\Rightarrow_{rm}$）表示最左（右）推导。

左（右）句型:

用最左推导方式导出的句型，称为左句型，而用最右推导方式导出的句型，称为右句型(规范句型)。

结论:

每个句子都有相应的最右和最左推导（但对句型此结论不成立）

2. 文法分类

- **0型文法**: 也称为短语文法，其产生式具有形式: $\alpha \to \beta$，其中$\alpha,\beta \in (V_T \cup V_N)^*$，并且$\alpha$至少含一个非终极符
- **1型文法**: 也称为上下文相关文法。它是0型文法的特例，要求$|\alpha| \le |\beta|$ ($S \to \lambda$例外，但S不得出现于产生式右部)。

或者是: $\alpha A_1 \beta \to \alpha A_2 \beta$

$a^n b^n c^n \quad n>=1$: $S \to aSBC|aBC$ $CB \to HB$

$HB \to HC$ $HC \to BC$ $aB \to ab$ $bB \to bb$

$bC \to bc$ $cC \to cca$

2. 文法的分类

- **2型文法**: 也称为上下文无关文法。它是1型文法的特例，即要求产生式左部是一个非终极符: $A \rightarrow \beta$
- **3型文法: 也称为正则文法**。它是2型文法的特例，即产生式的右部至多有两个符号，而且具有下面形式之一:

$A \rightarrow a$ ， $A \rightarrow a\,B$

其中$A,B \in V_N$ ， $a \in V_T$