编译原理

第二章 高级语言及其语法描述

第二章 高级语言及其语法描述

- 程序语言的定义
- 高级语言的一般特性
- 程序语言的语法描述

第二章 高级语言及其语法描述

- 程序语言的定义
- 高级语言的一般特性
- 程序语言的语法描述

上下文无关文法

- 一个上下文无关文法 G 是一个四元式

  $G=(V_T, V_N, S, P)$，其中

  - V_T：终结符集合 (非空)
  - V_N：非终结符集合 (非空)，且 $V_T \cap V_N=\varnothing$
  - S：文法的开始符号，$S \in V_N$
  - P：产生式集合 (有限)，每个产生式形式为
    - $P \rightarrow \alpha$，$P \in V_N$，$\alpha \in (V_T \cup V_N)^*$
  - 开始符 S 至少必须在某个产生式的左部出现一次

上下文无关文法

- 定义：称$\alpha A\beta$直接推出$\alpha\gamma\beta$，即

$$\alpha A\beta \Rightarrow \alpha\gamma\beta$$

仅当$A \to \gamma$是一个产生式，

且α，$\beta \in (V_T \cup V_N)^*$。

- 如果$\alpha_1 \Rightarrow \alpha_2 \Rightarrow \ldots \Rightarrow \alpha_n$，则我们称这个序列是从$\alpha_1$到$\alpha_n$的一个推导。若存在一个从$\alpha_1$到$\alpha_n$的推导，则称$\alpha_1$可以推导出$\alpha_n$

上下文无关文法

- 定义：假定 G 是一个文法，S 是它的开始符号。如果 $S \overset{*}{\Longrightarrow} \alpha$，则α称是一个句型。仅含终结符号的句型是一个句子。文法 G 所产生的句子的全体是一个语言，将它记为 L(G) 。

$$L(G) = \{\alpha \mid S \overset{+}{\Longrightarrow} \alpha,\ \alpha \in V_T^*\}$$

- 例： (i*i+i) 是文法

  G(E)： $E \rightarrow i \mid E+E \mid E*E \mid (E)$

  的一个句子。

证明：

$E \Rightarrow (E)$

$\Rightarrow (E+E)$

$\Rightarrow (E*E+E)$

$\Rightarrow (i*E+E)$

$\Rightarrow (i*i+E)$

$\Rightarrow (i*i+i)$

E， (E)， (E*E+E)，…， (i*i+i) 是句型。

上下文无关文法示例

- 例：文法 $G_1(A)$：

  $A \rightarrow c|Ab$

  $G_1(A)$ 的语言？

- $L(G_1)$={c， cb， cbb， … }
- 以 c 开头，后继若干个 b

$A \Rightarrow c$

$A \Rightarrow Ab$

$\Rightarrow cb$

$A \Rightarrow Ab$

$\Rightarrow Abb$

$\Rightarrow Abbb$

$\Rightarrow \ldots$

$\Rightarrow Ab\ldots b$

$\Rightarrow cb\ldots b$

上下文无关文法示例

- 例：文法 $G_2(S)$：

  $S \rightarrow AB$

  $A \rightarrow aA \mid a$

  $B \rightarrow bB \mid b$

$G_2(S)$ 的语言？

$L(G_2)=\{a^m b^n \mid m\ ,\ \ n>0\}$

$S \Rightarrow AB$

$A \Rightarrow a$

$A \Rightarrow aA$

$\Rightarrow aaA$

$\Rightarrow aaaA$

$\Rightarrow \ldots$

$\Rightarrow a\ldots aA$

$\Rightarrow a\ldots aa$

$B \Rightarrow b$

$B \Rightarrow bB$

$\Rightarrow bbB$

$\Rightarrow bbbB$

$\Rightarrow \ldots$

$\Rightarrow b\ldots bB$

$\Rightarrow b\ldots bb$

上下文无关文法示例

- 例：给出产生语言为 $\{a^n b^n | n \geq 1\}$ 的文法

  $G_3(S)$:

  $S \rightarrow aSb$

  $S \rightarrow ab$

- 计算思维的典型方法 -- 递归
  - 问题的解决又依赖于类似问题的解决，只不过后者的复杂程度或规模较原来的问题更小
  - 一旦将问题的复杂程度和规模化简到足够小时，问题的解法其实非常简单

上下文无关文法示例

- 例：给出产生语言为 $\{a^m b^n \mid 1 \le n \le m \le 2n\}$ 的文法

$G_4(S)$:

$S \rightarrow ab \mid aab$

$S \rightarrow aSb \mid aaSb$

上下文无关文法

- 从一个句型到另一个句型的推导往往不唯一

  $E+E \Rightarrow i+E \Rightarrow i+i$　　$E+E \Rightarrow E+i \Rightarrow i+i$

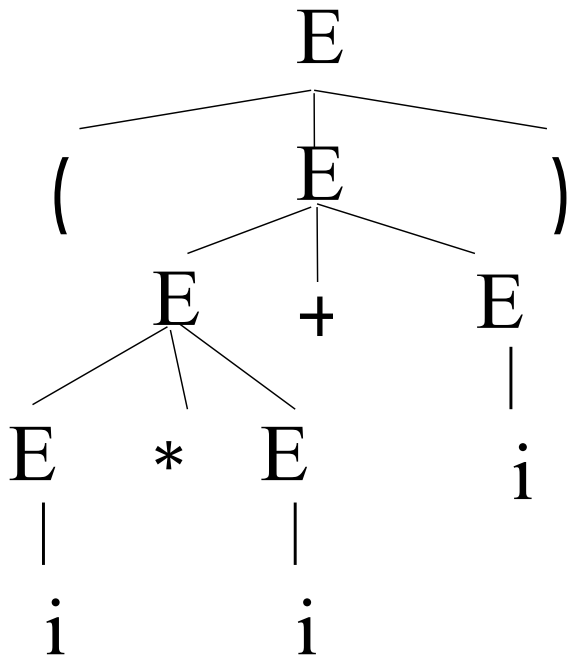
- 最左推导：任何一步$\alpha \Rightarrow \beta$都是对α中的最左非终结符进行替换

- 最右推导：任何一步$\alpha \Rightarrow \beta$都是对α中的最右非终结符进行替换

语法树与二义性 (ambiguity)

- 用一张图表示一个句型的推导，称为语法树
- (i*i+i) 的语法树

G(E)： E → i | E+E | E*E | (E)



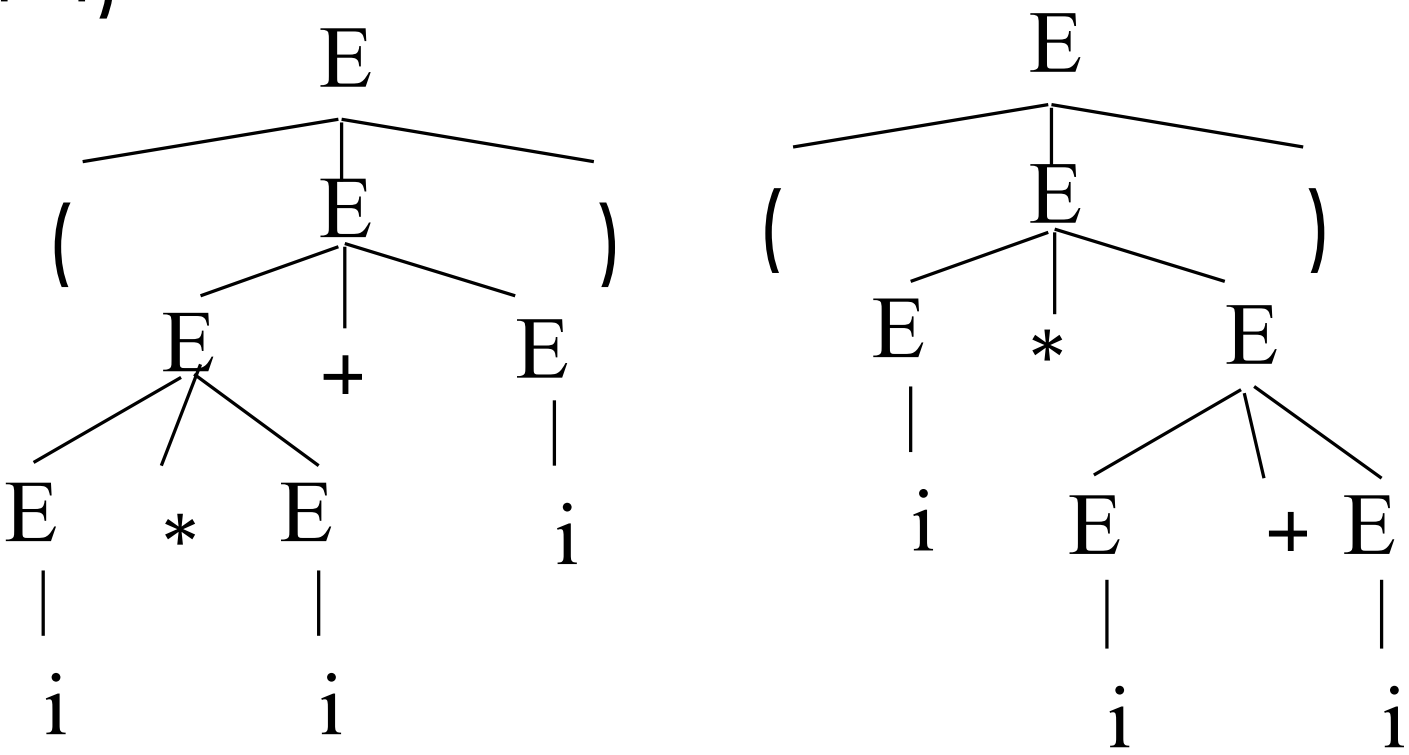
(i*i+i)

E ⇒(E)
⇒(E+E)
⇒(E*E+E)
⇒(i*E+E)
⇒(i*i+E)
⇒(i*i+i)

E ⇒(E)
⇒(E+E)
⇒(E+i)
⇒(E*E+i)
⇒(E*i+i)
⇒(i*i+i)

- 如果使用最左（右）推导，则一个最左（右）推导与语法树一一对应。

- 一个句型是否只对应唯一一棵语法树?
- (i*i+i)



文法： G(E)： $E \rightarrow i \mid E+E \mid E*E \mid (E)$

是二义的

语法树与二义性 (ambiguity)

- 定义：如果一个文法存在某个句子对应两颗不同的语法树，则说这个文法是二义的

  G(E)： $E \rightarrow i|E+E|E*E|(E)$ 是二义文法

- 语言的二义性：一个语言是二义性的，如果对它不存在无二义性的文法
  - 可能存在 G 和 G’ ，一个为二义的，一个为无二义的。但 L(G)=L(G’)
- 二义性问题是不可判定问题，即不存在一个算法，它能在有限步骤内，确切地判定一个文法是否是二义的
- 可以找到一组无二义文法的充分条件

二义文法:

G(E) : $E \rightarrow i \mid E+E \mid E*E \mid (E)$

无二义文法:

G(E) : $E \rightarrow T \mid E+T$

$T \rightarrow F \mid T*F$

$F \rightarrow (E) \mid i$

表达式 →项 | 表达式 +

项

项 → 因子 | 项 * 因子

因子 → (表达式) | i

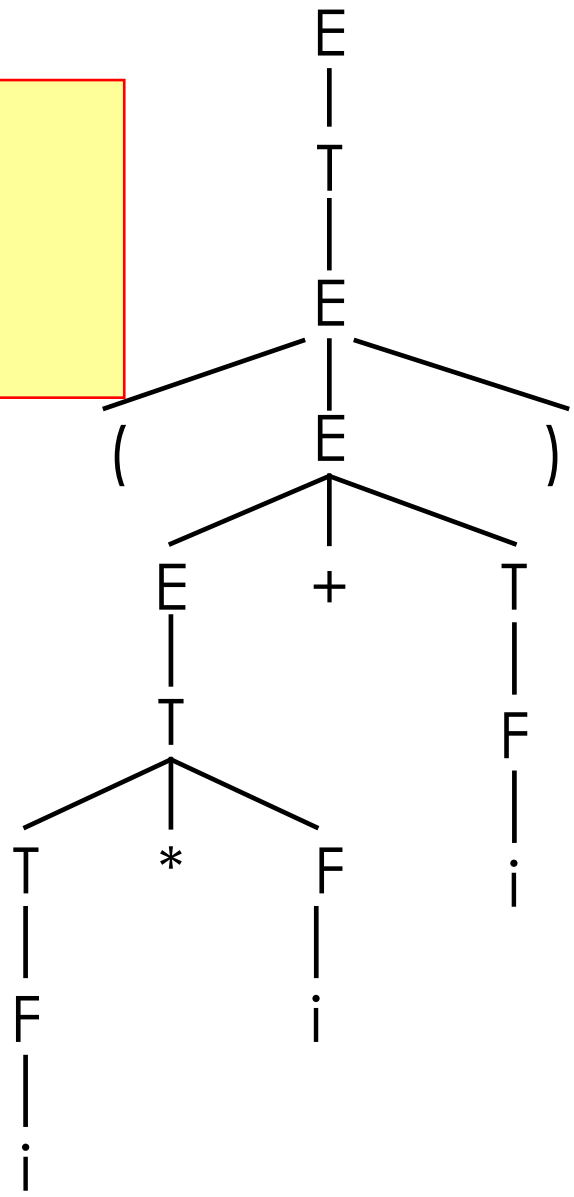
考虑句子 (i*i+i)

$E \Rightarrow T$
$\Rightarrow F$
$\Rightarrow (E)$
$\Rightarrow (E+T)$
$\Rightarrow (T+T)$
$\Rightarrow (T*F+T)$
$\Rightarrow (F*F+T)$
$\Rightarrow (i*F+T)$
$\Rightarrow (i*i+T)$
$\Rightarrow (i*i+F)$
$\Rightarrow (i*i+i)$

$G(E): \quad E \rightarrow T \mid E+T$

$T \rightarrow F \mid T*F$

$F \rightarrow (E) \mid i$



- 描述程序设计语言时，对于上下文无关文法的限制

1. 不含 P→P 形式的产生式

2. 每个非终结符 P 必须有用处

即：

$$S \overset{*}{\Rightarrow} \alpha P \beta$$

$$P \overset{*}{\Rightarrow} r \qquad r \in V_T^*$$

形式语言鸟瞰

- 乔姆斯基 (Chomsky) 是美国当代有重大影响的语言学家
- www.chomsky.info

形式语言鸟瞰

- 乔姆斯基于 1956 年建立形式语言体系，他把文法分成四种类型： 0 ， 1 ， 2 ， 3 型
- 与上下文无关文法一样，它们都由四部分组成，但对产生式的限制有所不同

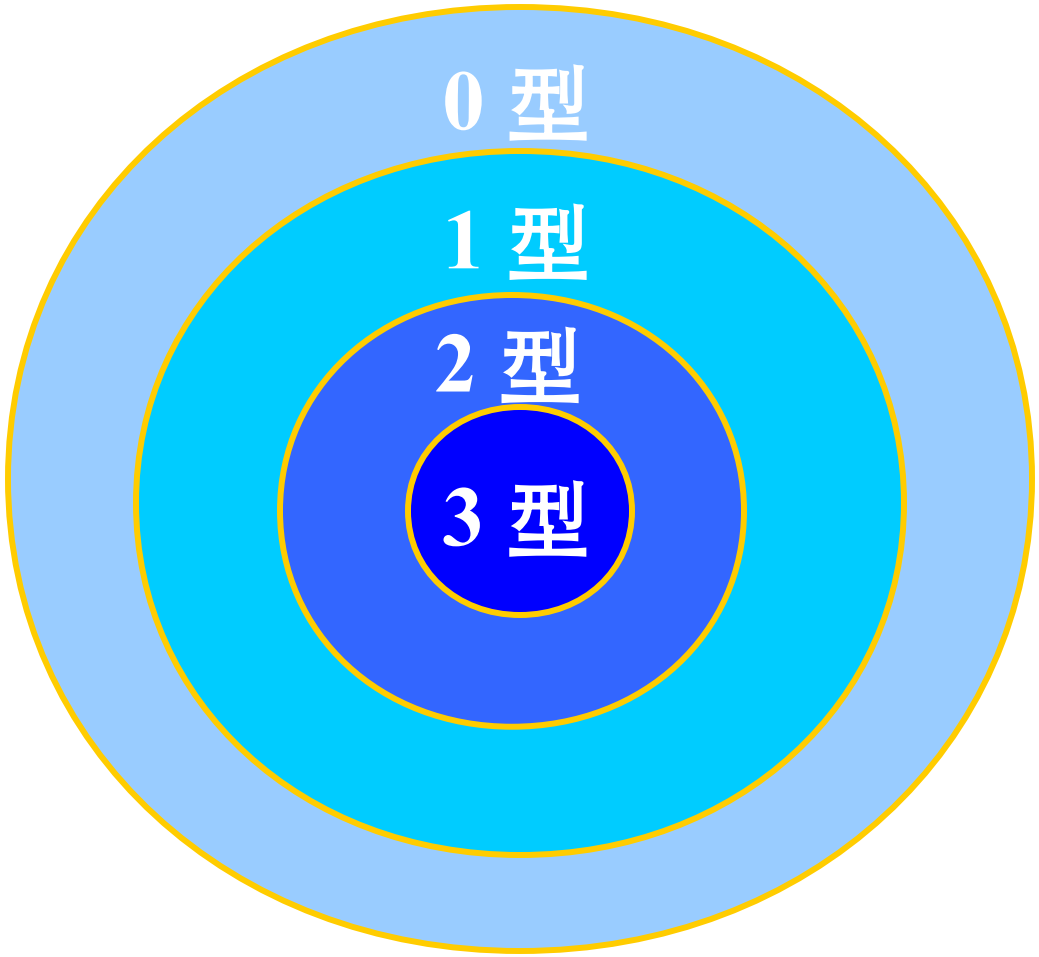
形式语言鸟瞰

- 0 型 (短语文法，图灵机)
  - 产生式形如： $\alpha \rightarrow \beta$
  - 其中：$\alpha \in (V_T \cup V_N)^*$ 且至少含有一个非终结符；$\beta \in (V_T \cup V_N)^*$
- 1 型 (上下文有关文法，线性界限自动机)
  - 产生式形如： $\alpha \rightarrow \beta$
  - 其中： $|\alpha| \leq |\beta|$ ，仅 $S \rightarrow \varepsilon$ 例外

形式语言鸟瞰

- 2 型 (上下文无关文法，非确定下推自动机)
  - 产生式形如： $A \to \beta$
  - 其中： $A \in V_N$ ； $\beta \in (V_T \cup V_N)^*$
- 3 型 (正规文法，有限自动机)
  - 产生式形如： $A \to \alpha B$ 或 $A \to \alpha$ **右线性文法**
  - 其中： $\alpha \in V_T^*$ ； A ， $B \in V_N$
  - 产生式形如： $A \to B\alpha$ 或 $A \to \alpha$ **左线性文法**
  - 其中： $\alpha \in V_T^*$ ； A ， $B \in V_N$

四种类型描述能力比较



四种类型描述能力比较

- $L_5=\{a^nb^n \mid n\geq 1\}$ 不能由正规文法
  下文无关文法产生
  $G_5(S)$:
  $S \rightarrow aSb \mid ab$

- $L_6=\{a^nb^nc^n \mid n\geq 1\}$ 不能由上下文
  但可由上下文有关文法产生

$G_6(S)$: $S \rightarrow aSBC \mid aBC$

$CB \rightarrow BC$

$aB \rightarrow ab$

$bB \rightarrow bb$

$bC \rightarrow bc$

$cC \rightarrow cc$

S
⇒aSBC
⇒ aaSBCBC
⇒
aaaBCBCBC
⇒
aaaBBCCBC
⇒
aaaBBCBCC
⇒
aaaBBBCCC
⇒
aaabBBCCC
⇒

四种类型描述能

- 计算思维的典型方法
  - 理论可实现 vs. 实际可实现
  - 理论研究重在探寻问题求解的方法，对于理论成果的研究运用又需要在能力和运用中作出**权衡**

- 程序设计语言不是上下文有关语言
- $L_7=\{\alpha c\alpha \mid \alpha\in\{a,b\}^*\}$ 不能由上下文无关文法产生，甚至连上下文有关文法也不能产生，只能由0型文法产生
  - 标识符引用
  - 过程调用过程中，“形 - 实参数的对应性 "(如个数，顺序和类型一致性)
- 对于现今程序设计语言，在编译程序中，仍然采用上下文无关文法来描述其语言结构

小结

- 文法、推导
  - 文法⇔语言
  - 最左推导、最右推导
- 语法树
- 二义性
  - 文法的二义性、语言的二义性
- 乔姆斯基形式语言体系

作业

- P36-6， 7， 8， 9， 10， 11(任选 2 个小题)

第二章 高级语言及其语法描述

- 程序语言的定义
- 高级语言的一般特性
- 程序语言的语法描述