编译原理

第四章 语法分析——自上而下分析

第四章 语法分析—自上而下分析

- 语法分析器的功能
- 自上而下分析面临的问题
- LL(1) 分析法
- 递归下降分析程序构造
- 预测分析程序

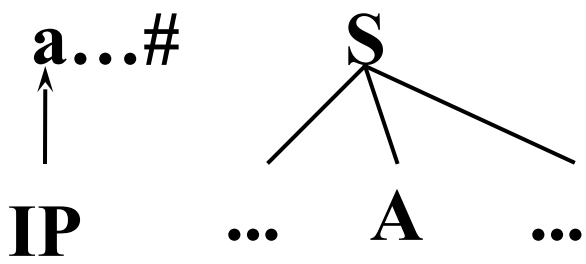
第四章 语法分析—自上而下分析

- 语法分析器的功能
- 自上而下分析面临的问题
- LL(1) 分析法
  - 消除文法的左递归
  - 克服回溯
- 递归下降分析程序构造
- 预测分析程序

4.3.2 消除回溯、提左因子

如何做到？

- 为了消除回溯必须保证
  - 对文法的任何非终结符，当要它去匹配输入串时，能够根据它所面临的输入符号准确地指派它的一个候选去执行任务，并且此候选的工作结果应是确信无疑的。
- $A \rightarrow \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$



- 令 G 是一个不含左递归的文法，对 G 的所有非终结符的每个候选α定义它的终结首符集 FIRST(α) 为：

$$FIRST(\alpha) = \{a \mid \alpha \overset{*}{\Rightarrow} a..., a \in V_T\}$$

特别是，若 $\alpha \overset{*}{\Rightarrow} \varepsilon$，则规定ε∈ FIRST(α) 。

- 如果非终结符 A 的所有候选首符集两两不相交，即 A 的任何两个不同候选α_i 和α_j

$$FIRST(\alpha_i) \cap FIRST(\alpha_j) = \phi$$

当要求 A 匹配输入串时， A 就能根据它所面临的第一个输入符号 a，准确地指派某一个候选前去执行任务。这个候选就是那个终结首符集含 a 的α。

- 提取公共左因子

  假定关于 A 的规则是

  $A \rightarrow \delta\beta_1 \mid \delta\beta_2 \mid \cdots \mid \delta\beta_n \mid \gamma_1 \mid \gamma_2 \mid \cdots \mid \gamma_m$ （其中，每个γ不以δ开头）

  那么，可以把这些规则改写成

  $A \rightarrow \delta A' \mid \gamma_1 \mid \gamma_2 \mid \cdots \mid \gamma_m$

  $A' \rightarrow \beta_1 \mid \beta_2 \mid \cdots \mid \beta_n$

- 经过反复提取左因子，就能够把每个非终结符（包括新引进者）的所有候选首符集变成为两两不相交

4.3.3 LL(1) 分析条件

- $E \rightarrow TE'$
  $E' \rightarrow +TE' \mid \varepsilon$
  $T \rightarrow FT'$
  $T' \rightarrow *FT' \mid \varepsilon$
  $F \rightarrow (E) \mid i$

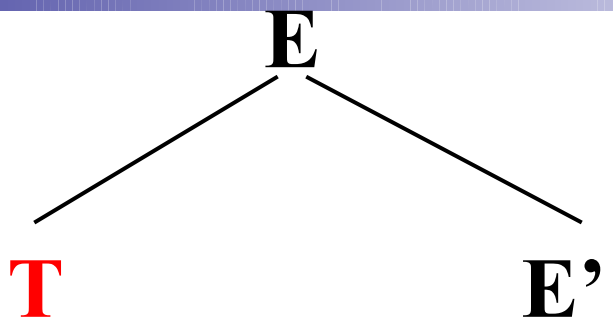
- i + i

E

i + i #

IP

- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i

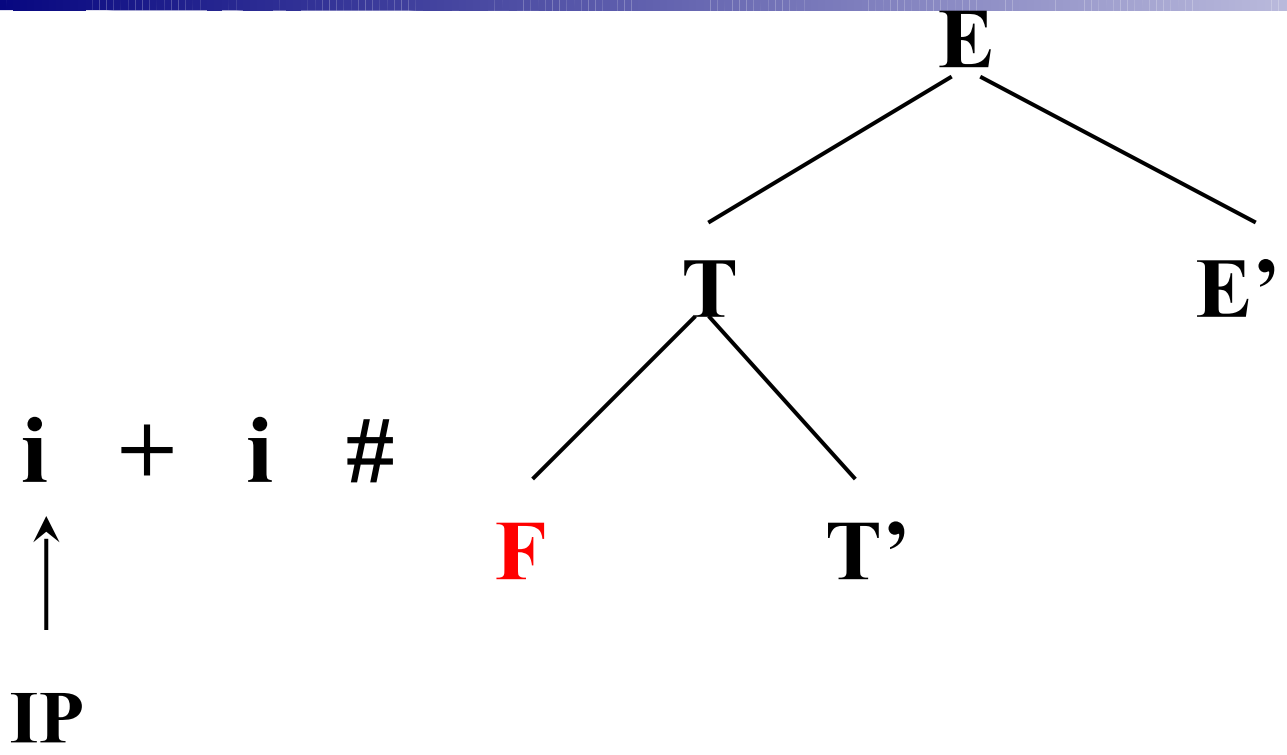


i + i #

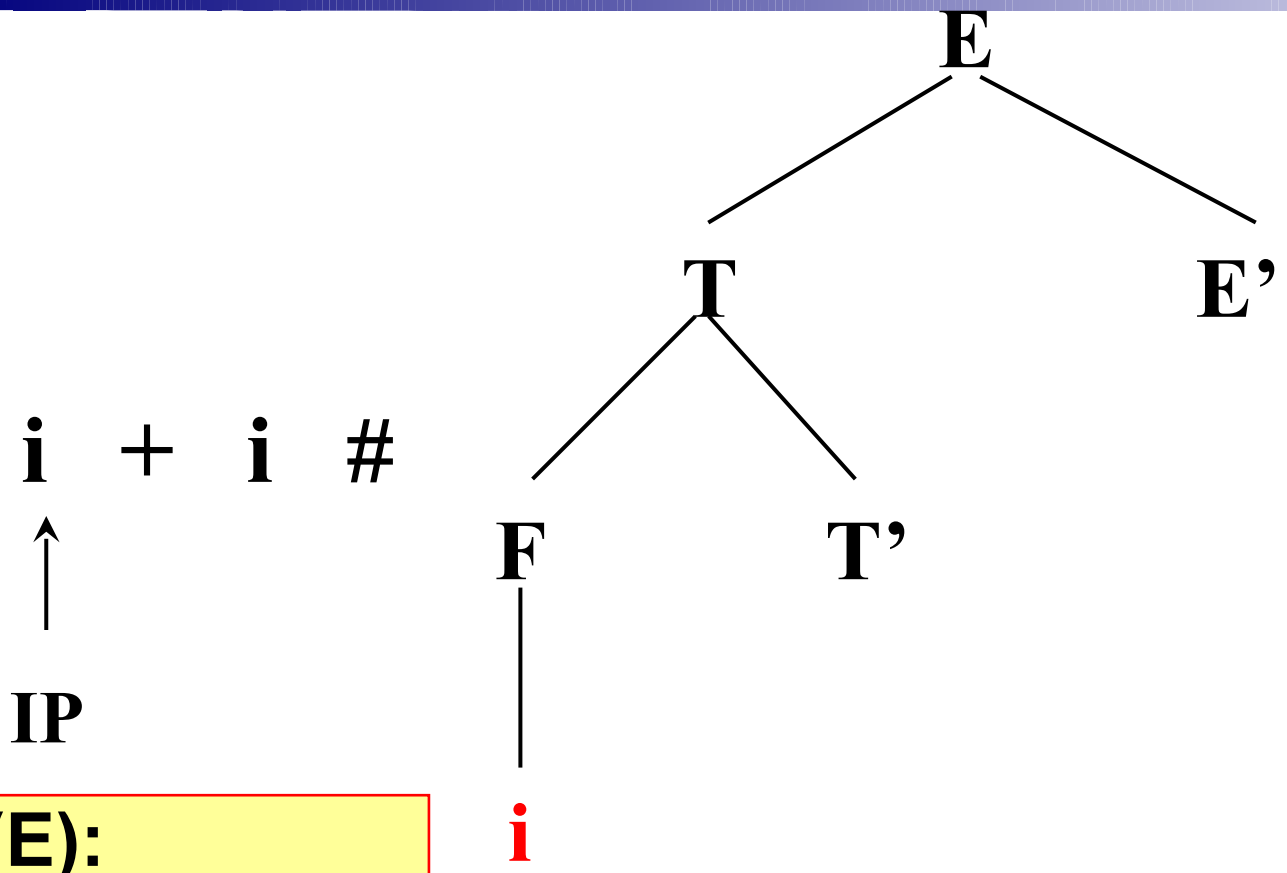
↑

IP

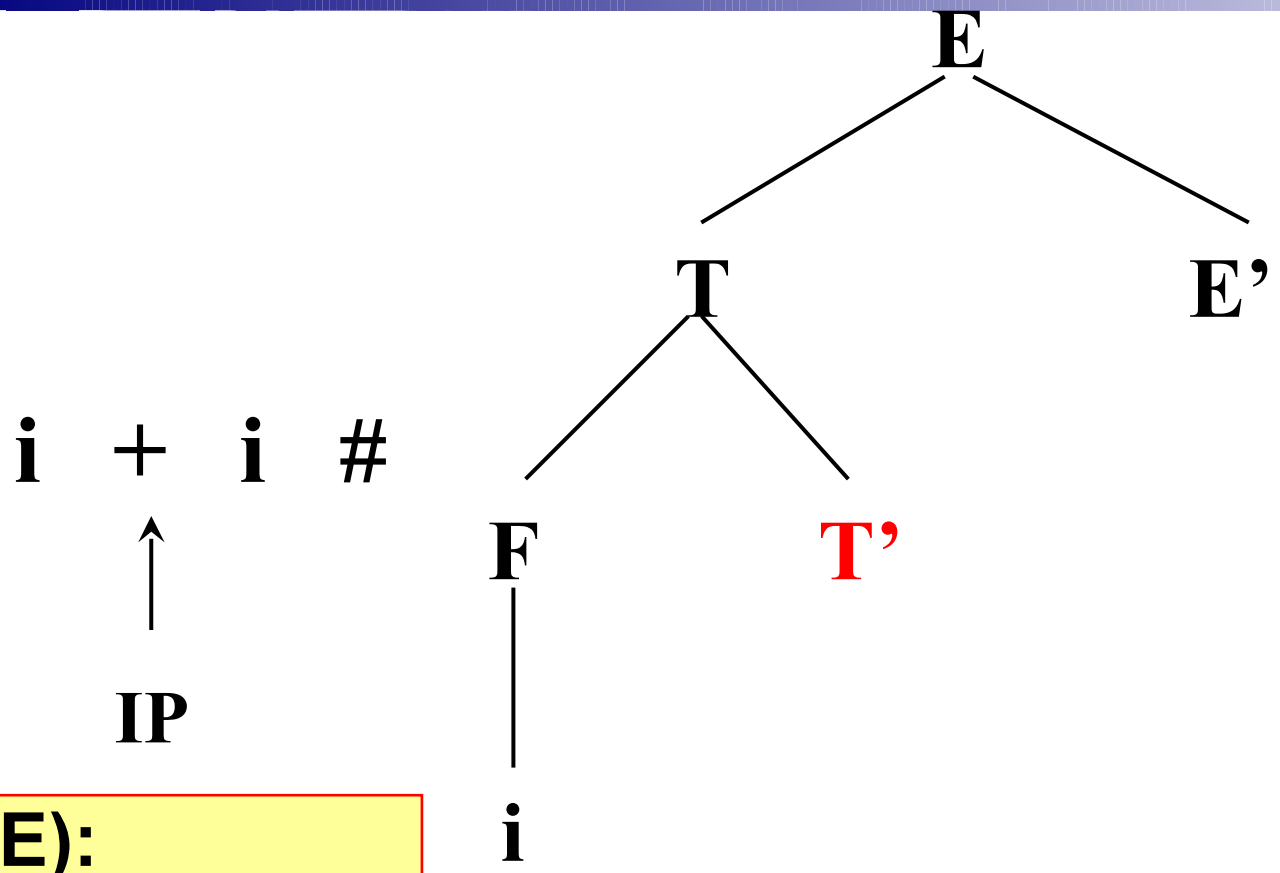
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



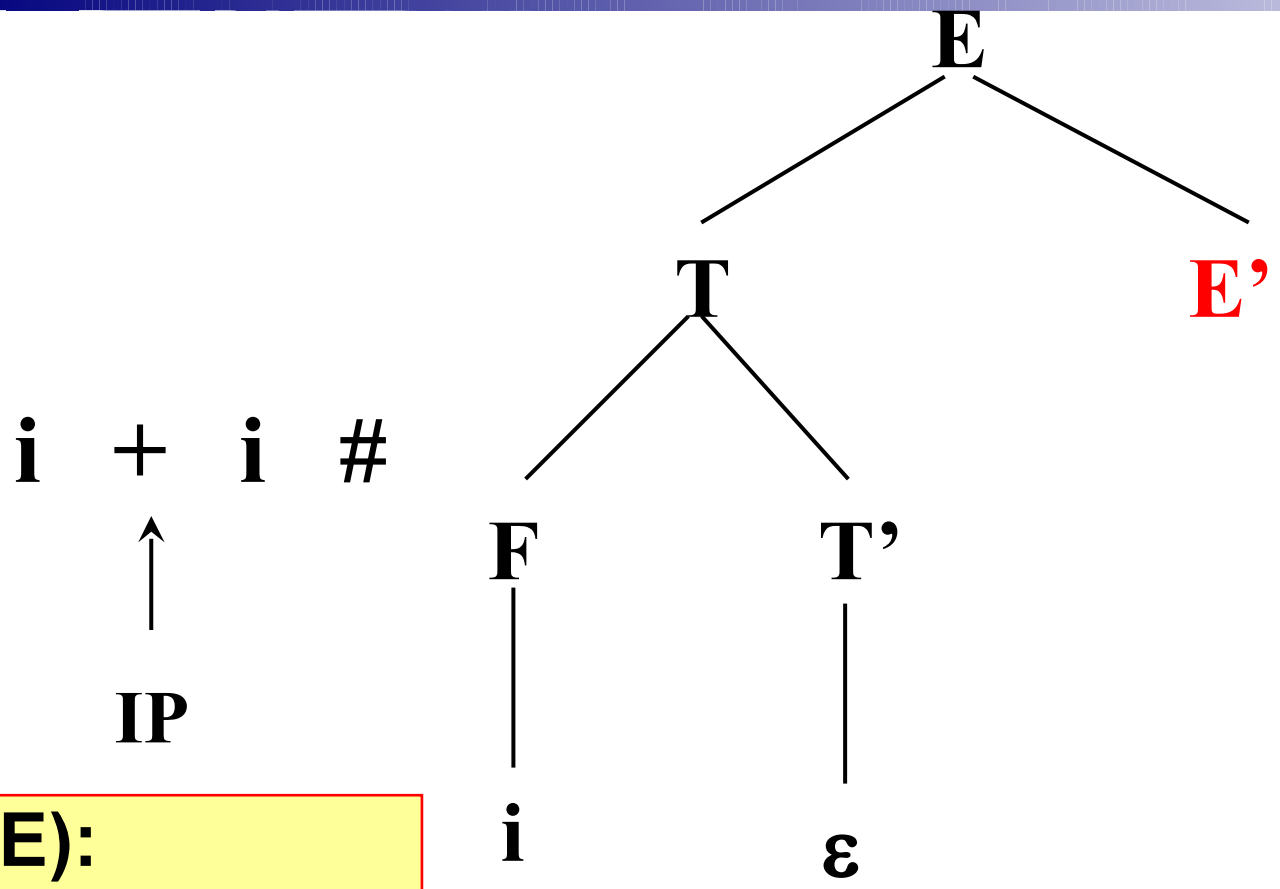
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



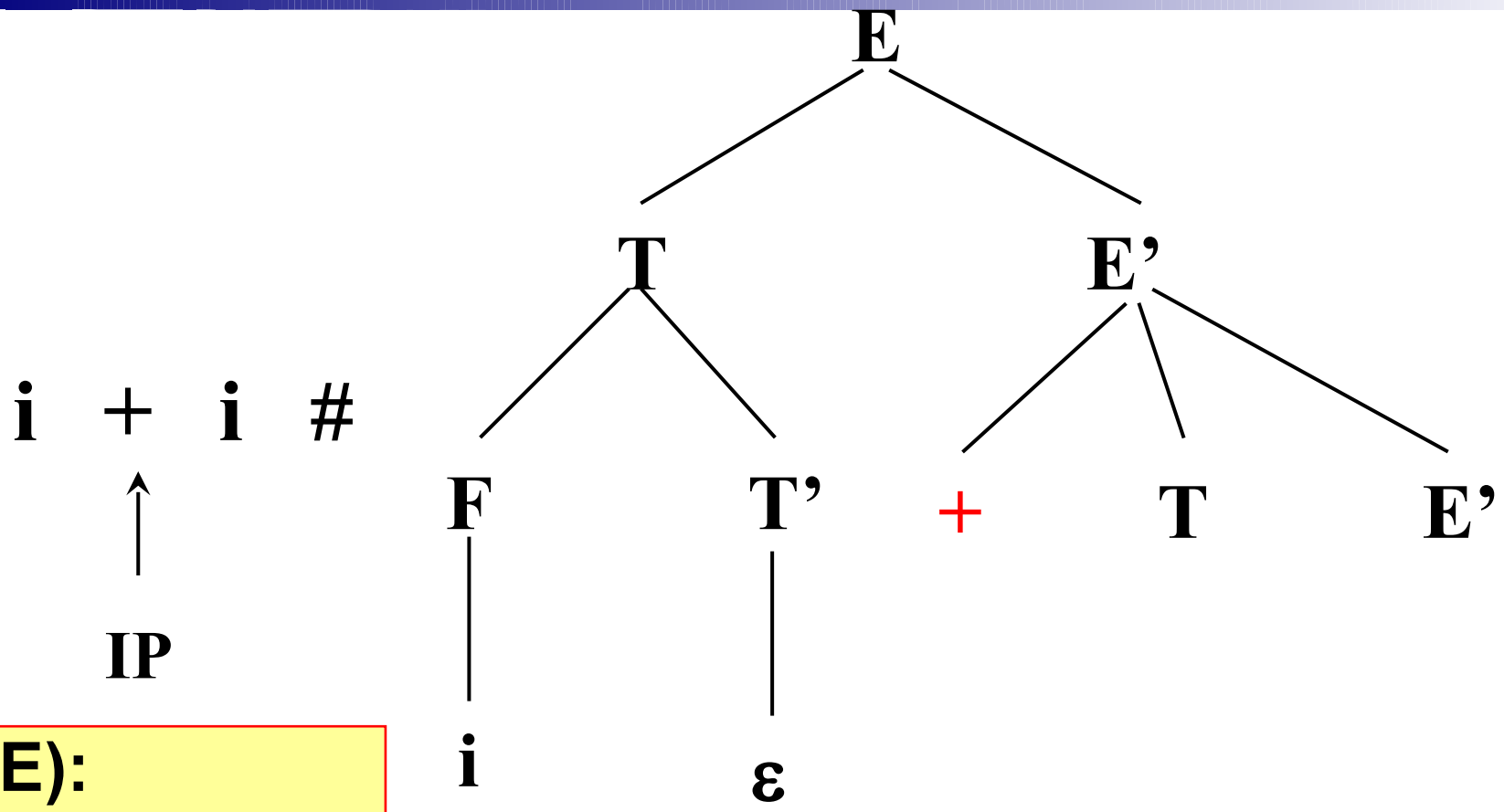
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



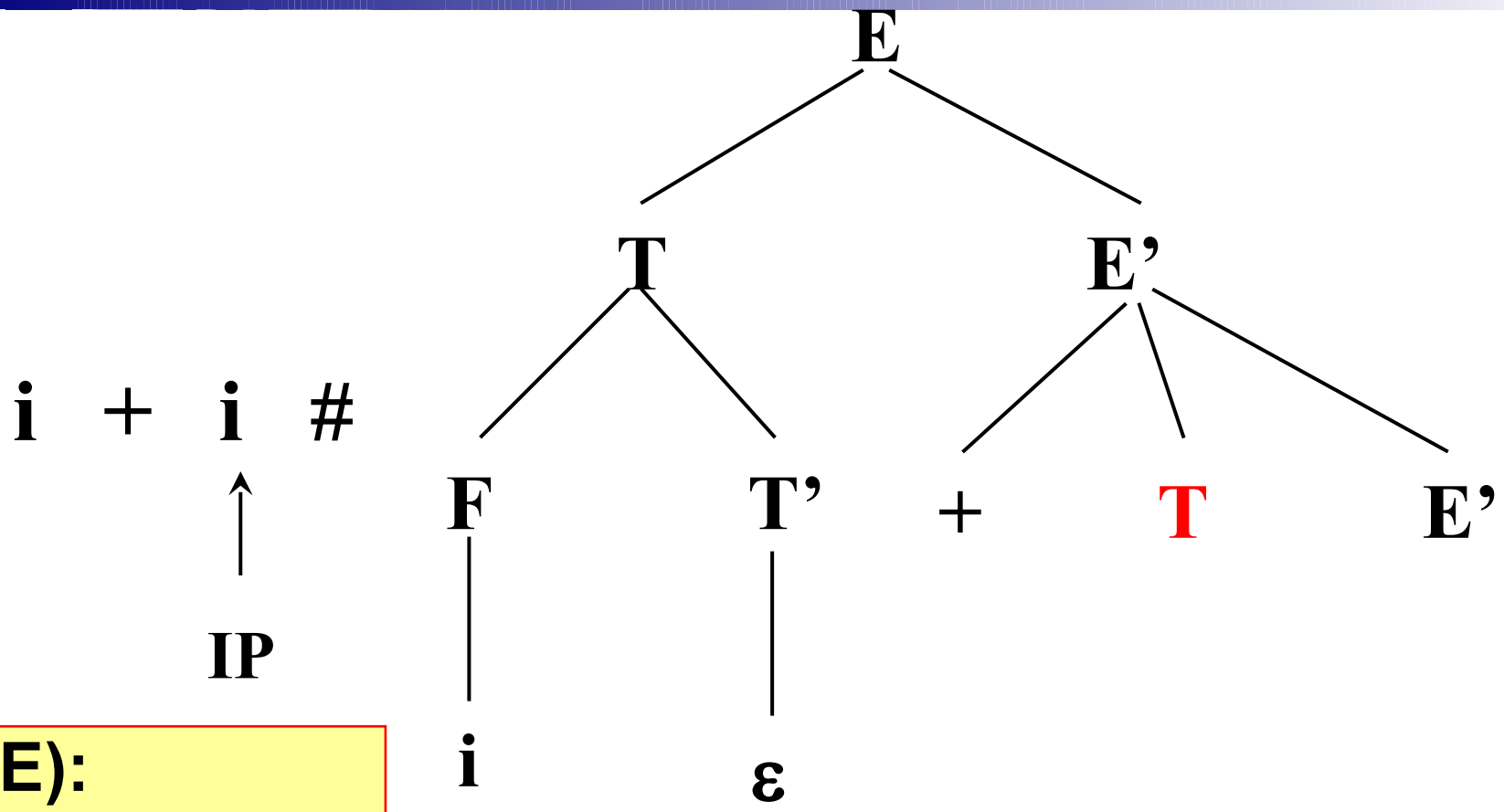
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



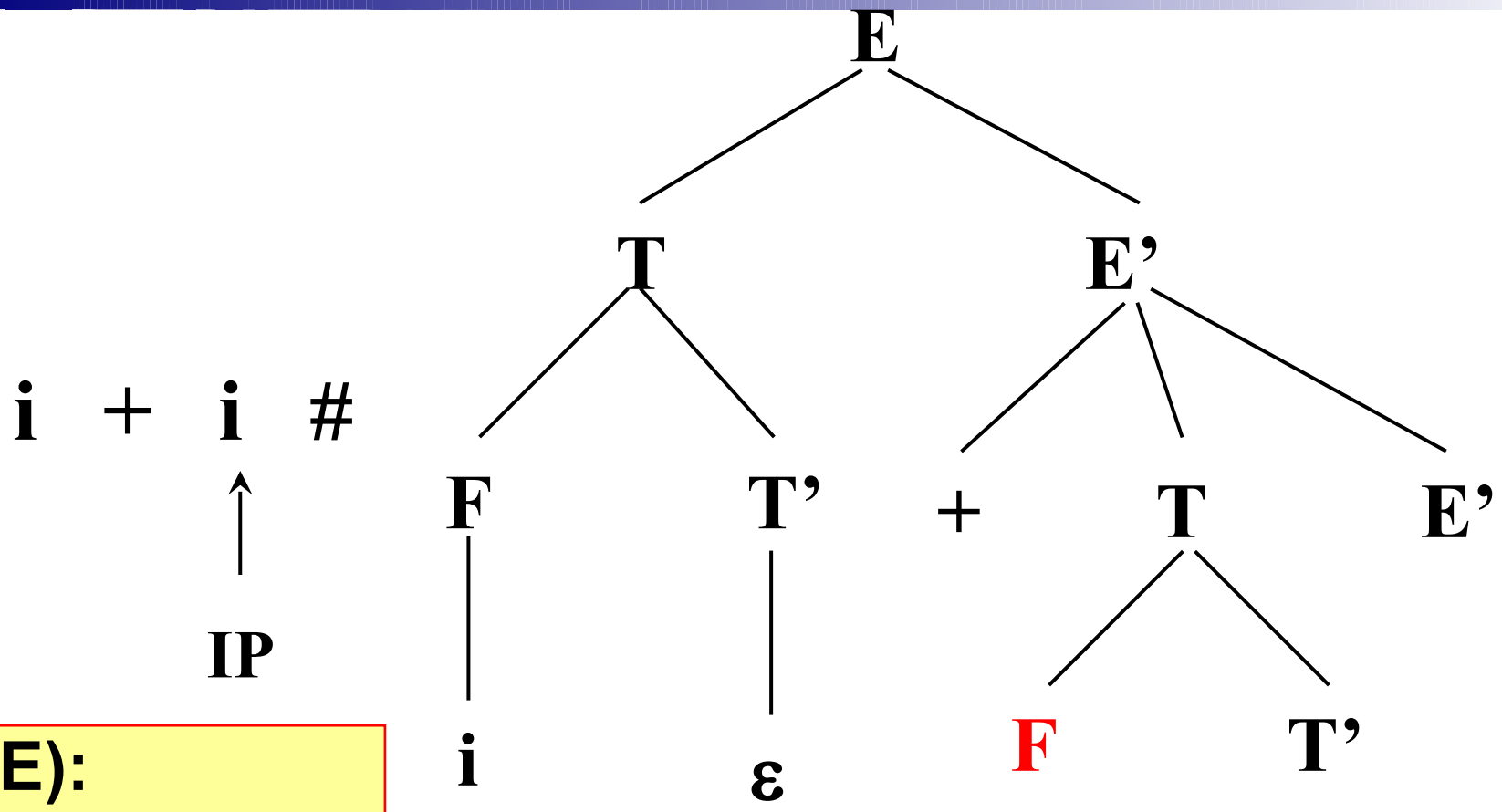
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



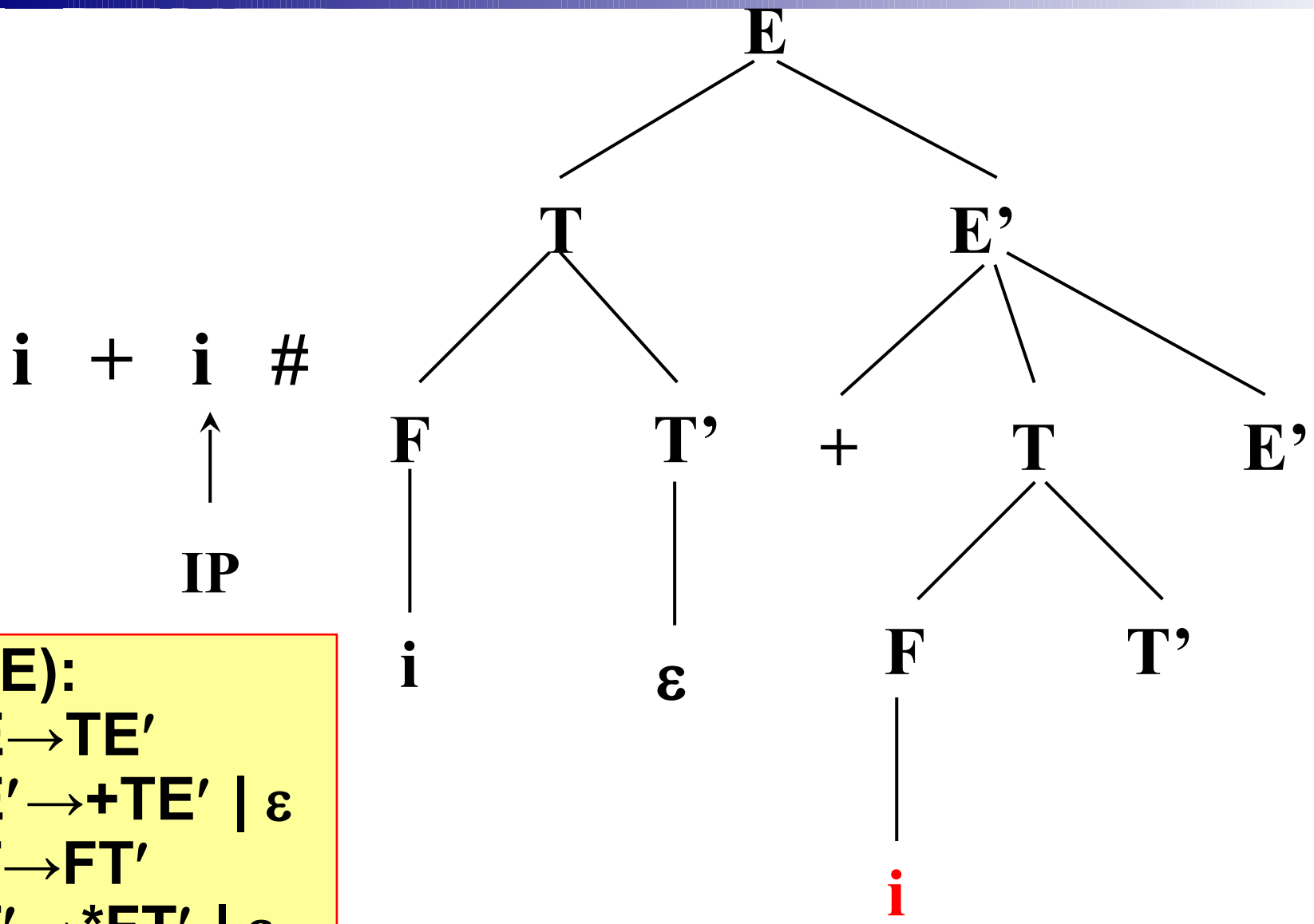
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



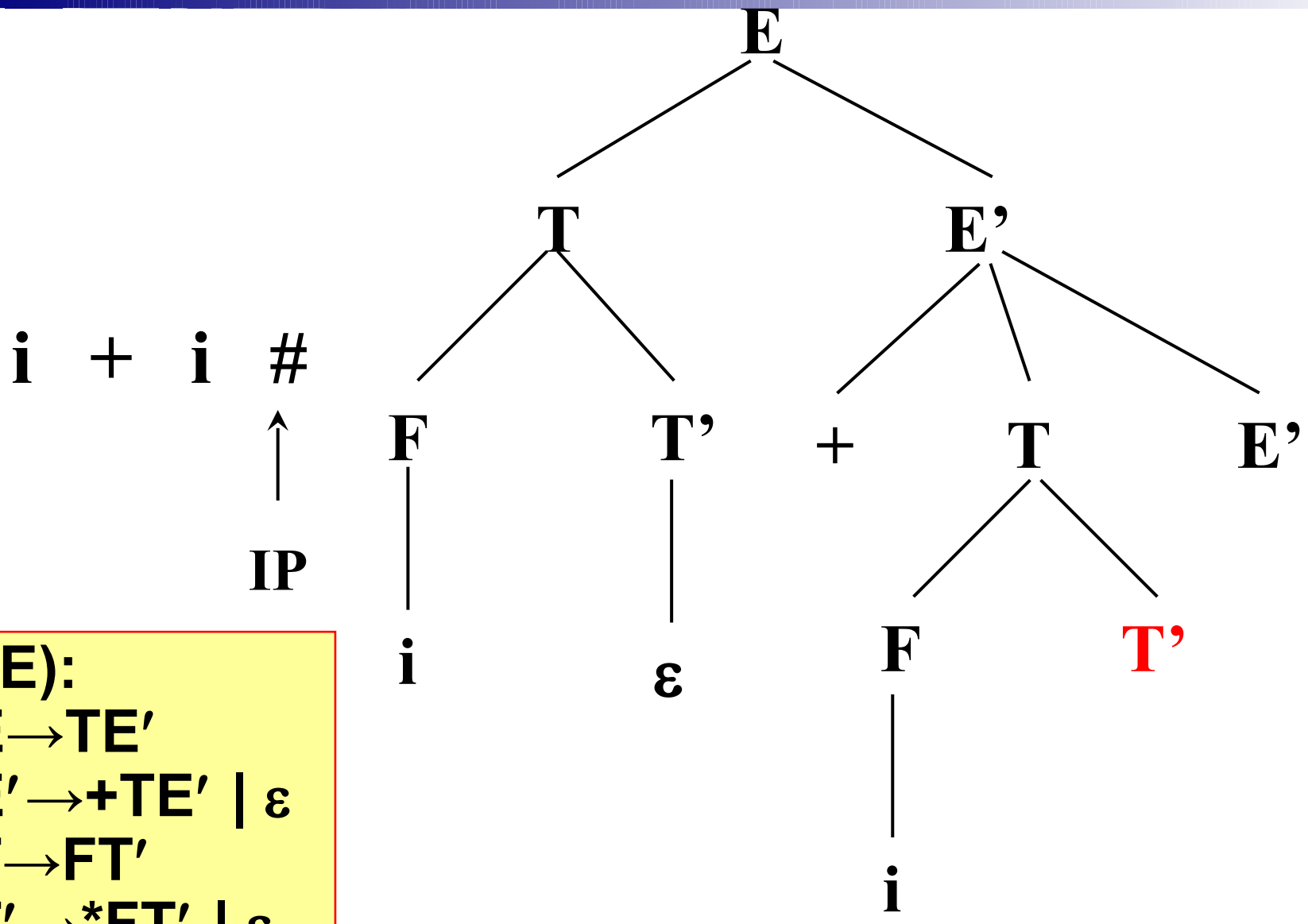
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



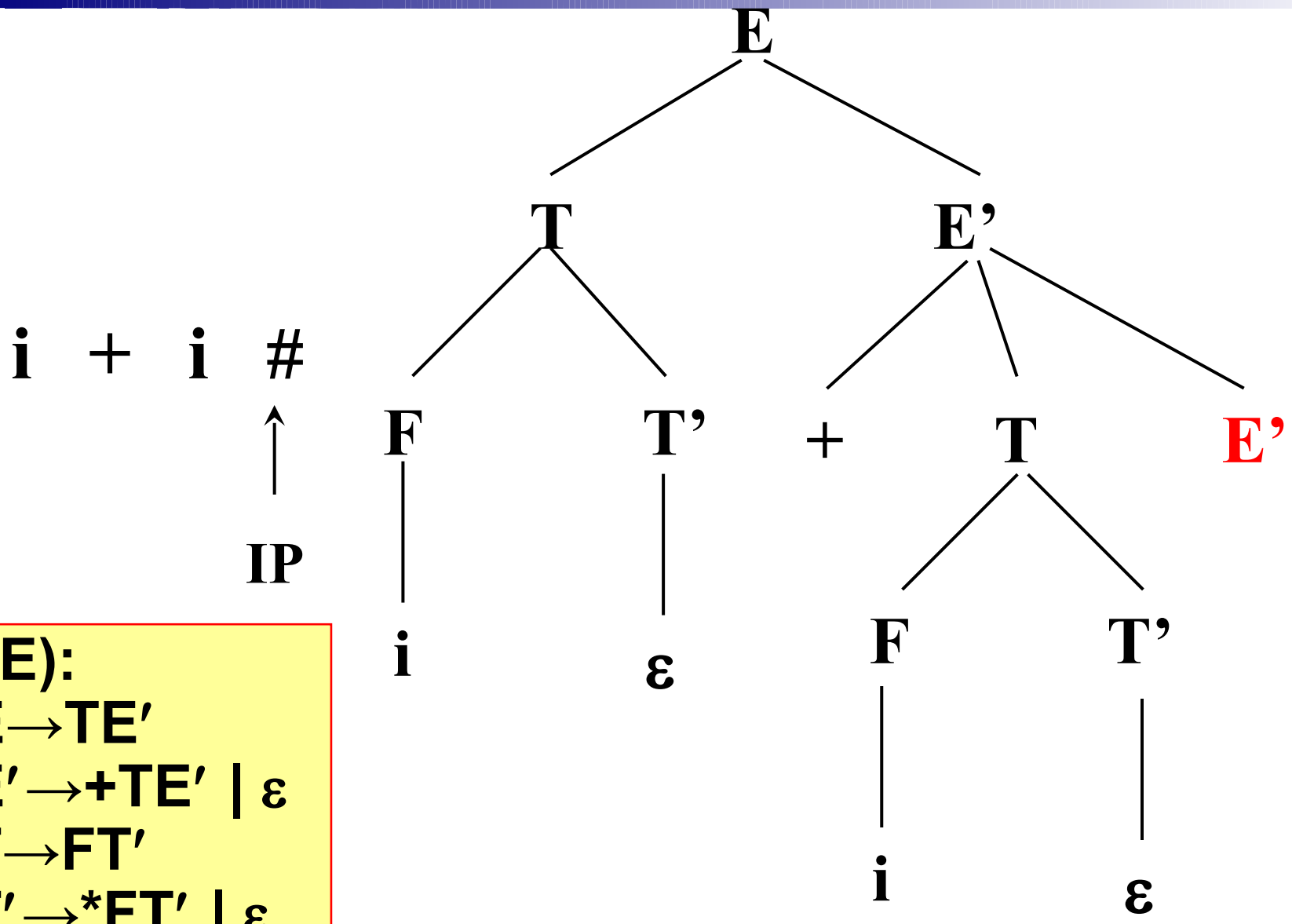
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



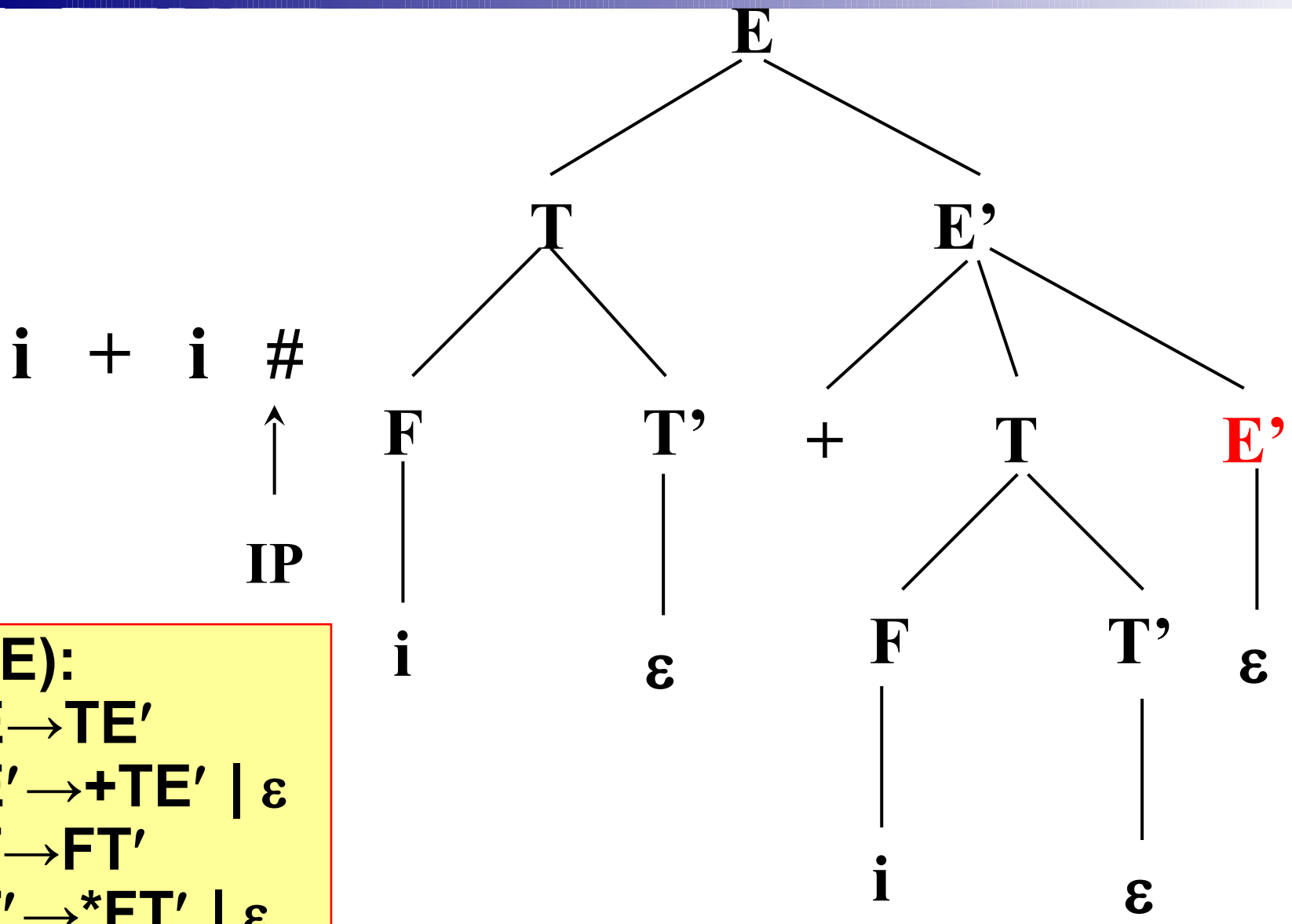
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



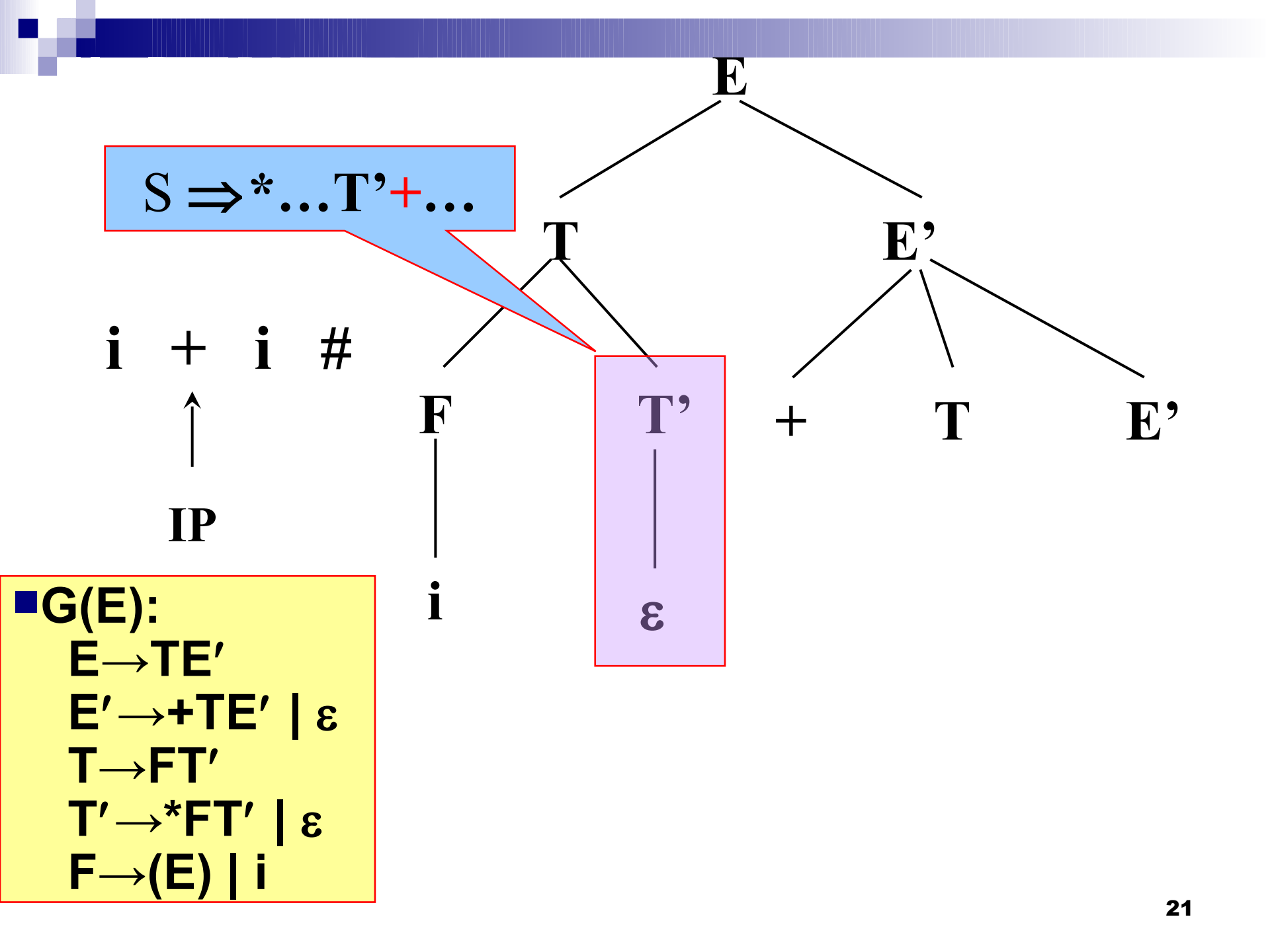
- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i



- G(E):
  - E→TE′
  - E′→+TE′ | ε
  - T→FT′
  - T′→*FT′ | ε
  - F→(E) | i

4.3.3 LL(1) 分析条件

- 假定 S 是文法 G 的开始符号，对于 G 的任何非终结符 A，我们定义 A 的 FOLLOW 集合

$$FOLLOW(A) = \{a \mid S \overset{*}{\Rightarrow} \ldots Aa \ldots, a \in V_T\}$$

特别是，若 $S \overset{*}{\Rightarrow} \cdots A$，则规定

$$\# \in \mathrm{FOLLOW(A)}$$

构造不带回溯的自上而下分析的文法条件

1. 文法不含左递归
2. 对于文法中每一个非终结符 A 的各个产生式的候选首符集两两不相交。即，若

$$A \rightarrow \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$$

则 $FIRST(\alpha_i) \cap FIRST(\alpha_j) = \phi \qquad (i \neq j)$

3. 对文法中的
首符集包含

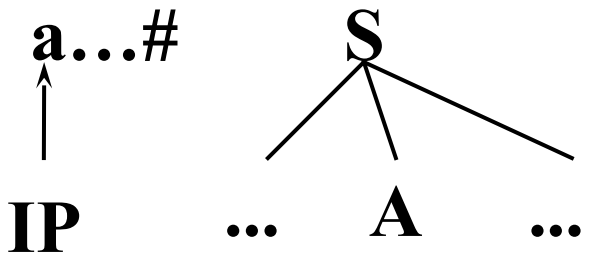
F

$$i=1,2,\ldots,n$$

第一个 L: 从左到右扫描输入串
第二个 L: 最左推导
1: 分析时每一步只需向前查看一个符号

如果一个文法 G 满足以上条件，则称该文法 G 为 LL(1) 文法。

LL(1) 分析法



- 对于一个满足上述条件的文法，可以对其输入串进行有效的无回溯的自上而下分析。假设要用非终结符 A 进行匹配，面临的输入符号为 a， A 的所有产生式为

$$A \rightarrow \alpha_1 \mid \alpha_2 \mid \cdots \mid \alpha_n$$

1. 若 a∈FIRST(α_i)，则指派α_i执行匹配任务；

2. 若 a 不属于任何一个候选首符集，则：
   (1) 若ε属于某个 FIRST(α_i) 且 a∈FOLLOW(A)，则让 A 与ε自动匹配。
   (2) 否则， a 的出现是一种语法错误。

如何构造 FIRST 和 FOLLOW 集合

$$FIRST(\alpha) = \{a \mid \alpha \overset{*}{\Rightarrow} a..., a \in V_T\}$$

$$FOLLOW(A) = \{a \mid S \overset{*}{\Rightarrow} ...Aa..., a \in V_T\}$$

构造 FIRST(α)

$$FIRST(\alpha) = \{a \mid \alpha \overset{*}{\Rightarrow} a..., a \in V_T\}$$

- $\alpha = X$ ， $X \in V_T \cup V_N$
- $\alpha = X_1X_2\cdots X_n$ ， $X_i \in V_T \cup V_N$

构造 FIRST(α)

$$FIRST(\alpha) = \{a \mid \alpha \overset{*}{\Rightarrow} a..., a \in V_T\}$$

- $\alpha = X$ ， $X \in V_T \cup V_N$
- $\alpha = X_1X_2\cdots X_n$ ， $X_i \in V_T \cup V_N$

构造每个文法符号的 FIRST 集合

- 对每一文法符号 $X \in V_T \cup V_N$ 构造 FIRST(X)

  连续使用下面的规则，直至每个集合 FIRST 不再增大为止：

  1. 若 $X \in V_T$，则 FIRST(X) = {X}。
  2. 若 $X \in V_N$，且有产生式 $X \rightarrow a\cdots$，则把 a 加入到 FIRST(X) 中；若 $X \rightarrow \varepsilon$ 也是一条产生式，则把ε也加到 FIRST(X) 中。

构造每个文法符号的 FIRST 集合

3.

- 若 $X \rightarrow Y\cdots$ 是一个产生式且 $Y \in V_N$，则把 FIRST(Y) 中的所有非ε - 元素都加到 FIRST(X) 中；
- 若 $X \rightarrow Y_1Y_2\cdots Y_k$ 是一个产生式，Y_1，…，Y_{i-1} 都是非终结符，
  - 对于任何 j，$1 \le j \le i-1$，FIRST(Y_j) 都含有ε (即 $Y_1\cdots Y_{i-1} \Rightarrow^* \varepsilon$)，则把 FIRST($Y_i$) 中的所有非ε - 元素都加到 FIRST(X) 中
  - 若所有的 FIRST(Y_j) 均含有ε，j = 1，2，…，k，则把ε加到 FIRST(X) 中。

构造 FIRST(α)

$$FIRST(\alpha) = \{a \mid \alpha \overset{*}{\Rightarrow} a..., a \in V_T\}$$

- $\alpha = X$ ， $X \in V_T \cup V_N$
- $\alpha = X_1X_2\cdots X_n$ ， $X_i \in V_T \cup V_N$

构造任何符号串的 FIRST 集合

- 对文法 G 的任何符号串$\alpha = X_1X_2\cdots X_n$ 构造集合 FIRST(α)

  1. 置 FIRST(α) = FIRST(X_1)\{ε} ;

  2. 若对任何 $1\le j\le i-1$ ，$\varepsilon\in$ FIRST(X_j) ，则把 FIRST(X_i)\{ε} 加至 FIRST(α) 中；特别是，若所有的 FIRST(X_j) 均含有ε， $1\le j\le n$ ，则把ε也加至 FIRST(α) 中。显然，若$\alpha=\varepsilon$则 FIRST(α) = {ε} 。

构造 FOLLOW(A)

$$FOLLOW(A) = \{a \mid S \overset{*}{\Rightarrow} \ldots Aa \ldots, a \in V_T\}$$

构造每个非终结符的 FOLLOW 集合

- 对于文法 G 的每个非终结符 A 构造 FOLLOW(A) 的办法是，连续使用下面的规则，直至每个 FOLLOW 不再增大为止：

  1. 对于文法的开始符号 S，置 # 于 FOLLOW(S) 中；
  2. 若 $A \rightarrow \alpha B \beta$ 是一个产生式，则把 FIRST(β)\{ε} 加至 FOLLOW(B) 中；
  3. 若 $A \rightarrow \alpha B$ 是一个产生式，或 $A \rightarrow \alpha B \beta$ 是一个产生式而 $\beta \overset{*}{\Rightarrow} \varepsilon$（即 $\varepsilon \in$ FIRST(β)），则把 FOLLOW(A) 加至 FOLLOW(B) 中。

- 例 4.6　对于文法 G(E)

  E→TE′
  E′→+TE′ | ε
  T→FT′
  T′→*FT′ | ε
  F→(E) | i

构造每个非终结符的 FIRST 和 FOLLOW 集合

构造每个文法符号的 FIRST 集合

- 对每一文法符号 $X \in V_T \cup V_N$ 构造 FIRST(X)

连续使用下面的规则，直至每个集合 FIRST 不再增大为止：

1. 若 $X \in V_T$，则 FIRST(X) = {X}。
2. 若 $X \in V_N$，且有产生式 $X \rightarrow a\cdots$，则把 a 加入到 FIRST(X) 中；若 $X \rightarrow \varepsilon$ 也是一条产生式，则把ε也加到 FIRST(X) 中。
3.
   - 若 $X \rightarrow Y\cdots$ 是一个产生式且 $Y \in V_N$，则把 FIRST(Y) 中的所有非ε - 元素都加到 FIRST(X) 中；
   - 若 $X \rightarrow Y_1Y_2\cdots Y_k$ 是一个产生式，Y_1，…，Y_{i-1} 都是非终结符，
     - 对于任何 j，$1 \le j \le i-1$，$FIRST(Y_j)$ 都含有ε (即 $Y_1\cdots Y_{i-1} \Rightarrow^* \varepsilon$)，则把 $FIRST(Y_i)$ 中的所有非ε - 元素都加到 FIRST(X) 中
     - 若所有的 $FIRST(Y_i)$ 均含有ε，$i = 1, 2, \cdots, k$，则把

构造每个非终结符的 FOLLOW 集合

- 对于文法 G 的每个非终结符 A 构造 FOLLOW(A) 的办法是，连续使用下面的规则，直至每个 FOLLOW 不再增大为止：

  1. 对于文法的开始符号 S，置 # 于 FOLLOW(S) 中；
  2. 若 $A \rightarrow \alpha B \beta$ 是一个产生式，则把 $FIRST(\beta) \setminus \{\varepsilon\}$ 加至 FOLLOW(B) 中；
  3. 若 $A \rightarrow \alpha B$ 是一个产生式，或 $A \rightarrow \alpha B \beta$ 是一个产生式而 $\beta \overset{*}{\Rightarrow} \varepsilon$（即 $\varepsilon \in FIRST(\beta)$），则把 FOLLOW(A) 加至 FOLLOW(B) 中。

- 例 4.6　对于文法 G(E)

  E→TE′
  E′→+TE′ | ε
  T→FT′
  T′→*FT′ | ε
  F→(E) | i

构造每个非终结符的 FIRST 和 FOLLOW 集合

FIRST(E) ={(,i}	FOLLOW(E) ={),#}
FIRST(E′)={+, ε}	FOLLOW(E′)={),#}
FIRST(T) ={(,i}	FOLLOW(T) ={+,),#}
FIRST(T′)={*, ε}	FOLLOW(T′)={+,),#}
FIRST(F) ={(,i}	FOLLOW(F) ={*,+,),#}

小结

- 构造不带回溯的自上而下分析算法
  - 消除文法的左递归
  - 提取左公共因子，克服回溯
- LL(1) 文法的条件
  - FIRST 、 FOLLOW 集合