

编译原理 FIRST 集、FOLLOW 集、SELECT 集求法通俗解释 & LL(1) 文法判定_多反思，多回顾，要坚持。...

1. 为什么要引入 FIRST 集的概念？

- 因为有公共左因子的问题，公共左因子是指在文法的产生式集合中，某个非终结符的多个候选式具有相同的前缀。

- 一般来说，公共左因子的产生式为

$$A \rightarrow \alpha\beta_1 \mid \alpha\beta_2$$

- 如果有公共左因子的问题，那么只能采取试探的方法来分析每一个候选式，分析的过程很可能产生回溯，回溯分析法是一种不确定的方法。
- 若所有候选式都没有公共左因子就可以选择惟一匹配的候选式，不会产生 (公共左因子引起的) 回溯。
- 为了消除回溯，对任何一个非终结符和当前的待匹配符号，期望
 - 要么**只有一个**候选式可用
 - 要么**没有**候选式可用

因此引入以下 FIRST 集合的概念:

- 对 $\alpha \in (V_T \cup V_N)^*$, 有

$$FIRST(\alpha) = \{a \mid \alpha \xRightarrow{*} a \cdots, a \in V_T\}$$

特别地，若 $\alpha \xRightarrow{*} \varepsilon$, 则 $\varepsilon \in FIRST(\alpha)$

因此对于每一文法符号 $X \in V_T \cup V_N$, 构造 $FIRST(X)$ 的方法为:
使用下列规则，直至每个 FIRST 集不再增大为止:

- 若 $X \in V_T$, 则 $FIRST(X) = \{X\}$ (意思是如果 X 是终结符，则其 $FIRST$ 集合为其自身);

• 若 $X \in V_N$, 那么 X 的产生式分以下三种情况:

• $X \rightarrow \varepsilon$

• 则 $\varepsilon \in FIRST(X)$

• $X \rightarrow a \cdots$

• 则 $a \in FIRST(X)$

• $X \rightarrow Y \cdots$, 且 $Y \in V_N$

• 则 $FIRST(Y) - \{\varepsilon\} \subseteq FIRST(X)$

• 特例: $X \rightarrow Y_1 Y_2 \cdots Y_{i-1} Y_i \cdots Y_k$

且 $Y_1, Y_2, \cdots, Y_{i-1} \in V_N$

$Y_1, Y_2, \cdots, Y_{i-1} \xRightarrow{*} \varepsilon$

• 则 $FIRST(Y_j) - \{\varepsilon\} \subseteq FIRST(X) (1 \leq j \leq i-1)$, $FIRST(Y_i) \subseteq FIRST(X)$

• 特别地, 当 $Y_1 Y_2 \cdots Y_{i-1} Y_i \cdots Y_k \xRightarrow{*} \varepsilon$

• 则 $\varepsilon \in FIRST(X)$

结论: 针对无空产生式的文法 G , 同一非终结符的任两个产生式的右部符号串的 $FIRST$ 集无交集, 即可进行确定的自顶向下分析。

2. 为什么要引入 FOLLOW 集的概念?

考虑文法 $G[S]$:

$S \rightarrow aA$

$S \rightarrow d$

$A \rightarrow bAS$

$A \rightarrow \varepsilon$

求得各终结符和符号串的 $FIRST$ 集合如下:

$FIRST(S) = \{a, d\}$

$FIRST(A) = \{b, \varepsilon\}$

$FIRST(aA) = \{a\}$

$FIRST(d) = \{d\}$

$FIRST(bAS) = \{b\}$

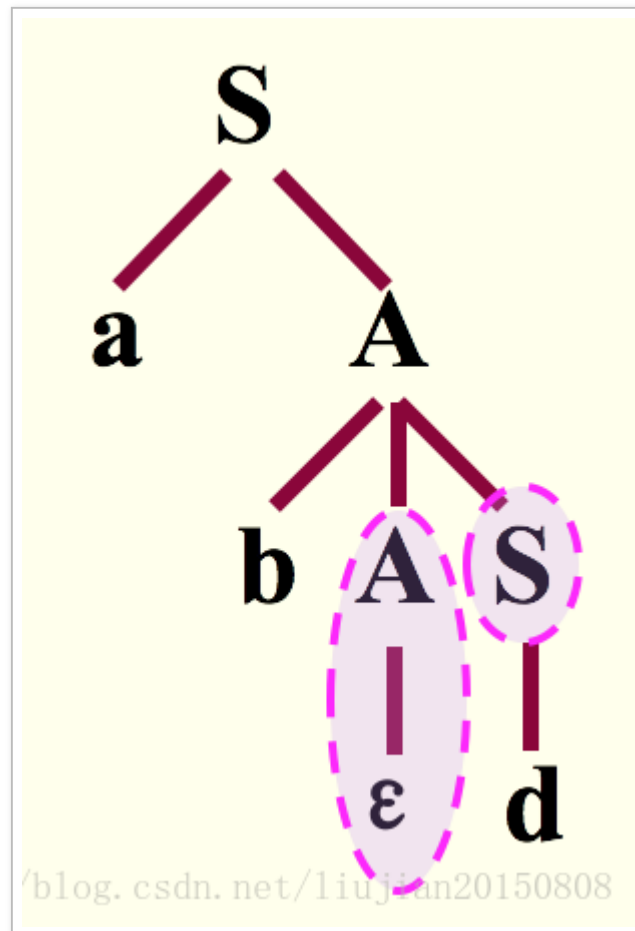
$FIRST(\varepsilon) = \{\varepsilon\}$

若输入串 $W = abd$, 则试图推导出 abd 串的推导过程为

$S \Rightarrow aA \Rightarrow abAS \Rightarrow abS \Rightarrow abd$

从以上推导过程中可以看到，在第 2 步到第 3 步的推导中，即 $abAS \Rightarrow abS$ 时，因为当前面临的输入符号为 d ，但是最左非终结符 A 的产生式右部的开始符号集都不包含 d ，但有 ϵ ，因此对于 d 的匹配自然认为只能依赖于在可能的推导过程中 A 的后面的符号，所以这时候选用产生式 $A \rightarrow \epsilon$ 向下推导。而当前 A 后面的符号为 S ， S 产生式右部的开始符号集包含了 d ，所以例子中可用 $S \rightarrow d$ 推导得到匹配。

语法树如下所示：



很显然，我们从以上叙述中可以得出：

当某一非终结符的产生式中含有空产生式时，它的非空产生式右部的开始符号集两两不相交，并与在推导过程中紧跟该非终结符右部可能出现的终结符集也不相交，则仍可构造确定的自顶向下分析。因此，引入了一个文法符号的后跟符号集合。

引入以下 FOLLOW 集的概念：

- 对 $A \in V_N$ ，有

$$\text{FOLLOW}(A) = \{a \mid S \xRightarrow{*} \cdots Aa \cdots, a \in V_T\}$$

若 $S \xRightarrow{*} \dots A$, 则规定 $\# \in \text{FOLLOW}(A)$

这里用 #作为输入串的结束符, 也称为输入串括号。

因此对于每一文法符号 $A \in V_N$, 实际上求 $\text{FOLLOW}(A)$

就是考察 A 在产生式右端的出现情况, 哪些终结符号可以跟随在 A 后面?

使用下列规则, 直至每个 FOLLOW 集不再增大为止:

- 首先, 设 S 为文法的开始符号, 把 {#} 加入 $\text{FOLLOW}(S)$ 中 (这里 #为句子括号)
- 若 $B \rightarrow \alpha A \beta$ 是一个产生式, 则 $\text{FIRST}(\beta) - \{\epsilon\} \subseteq \text{FOLLOW}(A)$
- 如果 $B \rightarrow \alpha A$ 或者 $B \rightarrow \alpha A \beta$ 且 $\beta \xRightarrow{*} \epsilon$, 则 $\text{FOLLOW}(B) \subseteq \text{FOLLOW}(A)$
 - 解释: 因为在推导过程中可能出现如下的句型序列:
 - $S \xRightarrow{*} \dots \alpha_1 B \beta_1 \dots \Rightarrow \dots \alpha_1 \alpha A \beta \beta_1 \dots \Rightarrow \dots \alpha_1 \alpha A \beta_1 \dots$

例题:

$A \rightarrow BCc \mid gDB$

$B \rightarrow bCDE \mid \epsilon$

$C \rightarrow DaB \mid ca$

$D \rightarrow dD \mid \epsilon$

$E \rightarrow gAf \mid c$

非终结符	FIRST	FOLLOW
A	{a, b, c, d, g}	{f, #}
B	{b, ϵ }	{a, c, d, g, f, #}
C	{a, c, d}	{c, d, g}
D	{d, ϵ }	{a, b, c, g, f, #}
E	{c, g}	{a, c, d, g, f, #}

对于 FIRST 集合, 解释其中的 $\text{FIRST}(A)$ 的求解。

- $A \rightarrow BCc$ 属于上述产生式的特例情况, 很显然 $B \xRightarrow{*} \epsilon$, $C \not\xRightarrow{*} \epsilon$, 因此 $(\text{FIRST}(B) - \{\epsilon\}) \cup \text{FIRST}(C) \subseteq \text{FIRST}(A)$
- $A \rightarrow gDB$ 属于上述产生式的第二种情况, 因此 $g \in \text{FIRST}(A)$
- 最后得出:

$$\text{FIRST}(A) = (\text{FIRST}(B) - \{\epsilon\}) \cup \text{FIRST}(C) \cup \{g\}$$

• 而 $FIRST(B) = \{b, \epsilon\}$, $FIRST(C) = \{a, c, d\}$

• 故 $FIRST(A) = \{a, b, c, d, g\}$

对于 FOLLOW 集合, 有如下的计算情况:

$$FOLLOW(A) = (FIRST(f) - \{\epsilon\}) \cup \{\#\} = \{f, \#\}$$

$$\begin{aligned} FOLLOW(B) &= (FIRST(Cc) - \{\epsilon\}) \cup FOLLOW(A) \cup FOLLOW(C) \\ &= \{a, c, d\} \cup \{f, \#\} \cup FOLLOW(C) \\ &= \{a, c, d\} \cup \{f, \#\} \cup \{c, d, g\} \\ &= \{a, c, d, g, f, \#\} \end{aligned}$$

$$\begin{aligned} FOLLOW(C) &= (FIRST(c) - \{\epsilon\}) \cup (FIRST(DE) - \{\epsilon\}) \\ &= \{c\} \cup (FIRST(D) - \{\epsilon\}) \cup FIRST(E) \\ &= \{c, d, g\} \end{aligned}$$

$$\begin{aligned} FOLLOW(D) &= (FIRST(B) - \{\epsilon\}) \cup FOLLOW(A) \cup (FIRST(E) - \{\epsilon\}) \\ &\quad \cup (FIRST(aB) - \{\epsilon\}) \\ &= \{b\} \cup \{f, \#\} \cup \{c, g\} \cup \{a\} \\ &= \{a, b, c, g, f, \#\} \end{aligned}$$

$$\begin{aligned} FOLLOW(E) &= FOLLOW(B) \\ &= \{a, c, d, g, f, \#\} \end{aligned}$$

3. 为什么要引入 SELECT 集的概念?

由于从 2 中我们得出结论:

当某一非终结符的产生式中含有空产生式时, 它的非空产生式右部的开始符号集两两不相交, 并与在推导过程中紧跟该非终结符右部可能出现的终结符集也不相交, 则仍可构造确定的自顶向下分析。

因此当文法中含有形如:

$$\begin{aligned} A &\rightarrow \alpha \\ A &\rightarrow \beta \end{aligned}$$

的产生式时, 其中 $A \in V_N, \alpha, \beta \in V^*$, 若 α 和 β 不能同时推导出空, 假定

$\alpha \not\Rightarrow^* \epsilon, \beta \Rightarrow^* \epsilon$, 则当

$$\text{FIRST}(\alpha) \cap \text{FOLLOW}(A) = \emptyset$$

$$\text{FIRST}(\beta) \cap \text{FOLLOW}(A) = \emptyset$$

也即是 $\text{FIRST}(\alpha) \cap (\text{FIRST}(\beta) \cup \text{FOLLOW}(A)) = \emptyset$ 时, 对于非终结符 A 的替换仍可唯一地确定候选。为了表示和分析方便, 因此引入了 SELECT 集合。

SELECT 集合定义如下:

- 一个产生式的选择符号集 SELECT。给定上下文无关文法的产生式 $A \rightarrow \alpha, A \in V_N, \alpha \in V^*$, 若 $\alpha \overset{*}{\neq} \epsilon$, 则 $\text{SELECT}(A \rightarrow \alpha) = \text{FIRST}(\alpha)$ 。
- 如果 $\alpha \overset{*}{\Rightarrow} \epsilon$, 则 $\text{SELECT}(A \rightarrow \alpha) = (\text{FIRST}(\alpha) - \{\epsilon\}) \cup \text{FOLLOW}(A)$ 。

因此一个上下文无关文法是 LL(1) 文法的充分必要条件是, 对每个非终结符 A 的两个不同产生式, $A \rightarrow \alpha, A \rightarrow \beta$, 满足

$$\text{SELECT}(A \rightarrow \alpha) \cap \text{SELECT}(A \rightarrow \beta) = \emptyset$$

其中 α, β 不同时能 $\overset{*}{\Rightarrow} \epsilon$ 。

再次回到上述例题

$A \rightarrow BCc \mid gDB$

$B \rightarrow bCDE \mid \epsilon$

$C \rightarrow DaB \mid ca$

$D \rightarrow dD \mid \epsilon$

$E \rightarrow gAf \mid c$

非终结符	FIRST	FOLLOW
A	{a, b, c, d, g}	{f, #}
B	{b, ϵ }	{a, c, d, g, f, #}
C	{a, c, d}	{c, d, g}
D	{d, ϵ }	{a, b, c, g, f, #}
E	{c, g}	{a, c, d, g, f, #}

右部产生式	FIRST
-------	-------

右部产生式	FIRST
BCc	{a, b, c, d}
gDB	{g}
bCDE	{b}
ϵ	{ ϵ }
DaB	{a, d}
ca	{c}
dD	{d}
gAf	{g}
c	{c}

因此根据以上所求得的第一集和 FOLLOW 集，可求得各产生式的 SELECT 集合如下：

SELECT(A \rightarrow BCc) = {a, b, c, d}
SELECT(A \rightarrow gDB) = {g}
SELECT(B \rightarrow bCDE) = {b}
SELECT(B \rightarrow ϵ) = {a, c, d, g, f, #}
SELECT(C \rightarrow DaB) = {a, d}
SELECT(C \rightarrow ca) = {c}
SELECT(D \rightarrow dD) = {d}
SELECT(D \rightarrow ϵ) = {a, b, c, g, f, #}
SELECT(E \rightarrow gAf) = {g}
SELECT(E \rightarrow c) = {c}

由上可知，有相同左部产生式的 SELECT 集合的交集为空，所以文法是 LL(1) 文法。

全文完

本文由 简悦 SimpRead 转码，用以提升阅读体验，原文地址