四、LL(1)分析法

LL(1)分析方法也是一种自顶向下分析方法，LL 的意思是：从左(Left)到右扫描输入符号串并建立它的最左推导(Leftmost derivations)。数字 1 是指向前看一个符号来决定选择同一个非终结符不同规则。因此，我们通常把按上述方法执行语法分析任务的程序称为 LL(1)分析程序或 LL(1)分析器，使用这种方法进行语法分析，可借助于一张分析表及一个语法分析栈，在一个总控程序控制下很方便地实现。

1. LL(1)分析器的逻辑结构和工作过程

LL(1)分析器是由一个总控程序、一张分析表和一个分析栈组成，如图 4.4 所示。其中“输入”就是待分析的符号串，它以右界符＃作为结尾。分析表 M 可用一个矩阵(或二维数组)来表示。它概括了相应文法的全部信息。分析表的每一行与文法的一个非终结符 A 相关联，而每一列则与文法的一个终结符号或＃相关联。分析表元素 $M[A,a](a\in V_T\cup\{\#\})$或者指示了当前推导所应使用的产生式，或者指出了输入串中含有语法错误。分析器对每个输入串的分析在总控程序控制下进行。其步骤如下：

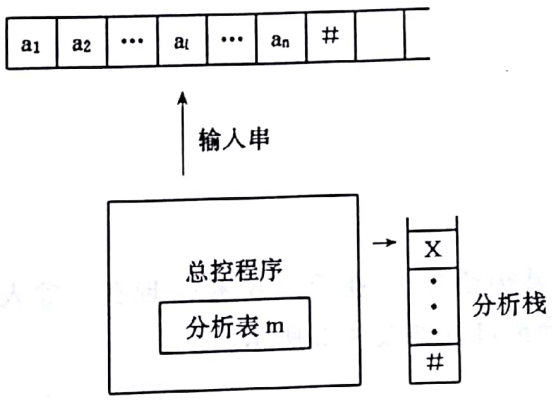
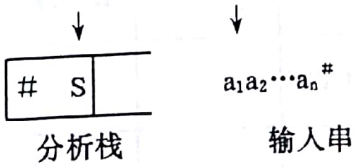


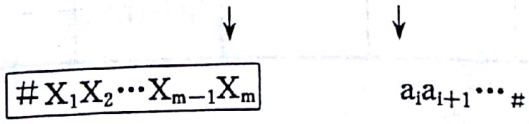
图 4.4　LL(1)分析器结构

(1)分析开始时，首先将符号＃及文法的开始符号 S 依次置于分析栈的底部，并把各指示器调整至起始位置，即初始格局为



然后，反复执行第二步所列的工作。

(2)设在分析的某一步，分析栈及余留的输入符号串处于如下的格局



其中，$X_1,X_2,\cdots,X_m$ 为分析过程中所得的文法符号，此时，可视栈顶符号 X_m 的不同情况，分别做如下的动作：

①若 $X_m \in V_N$，则以 X_m 及 a_i 组成符号对(X_m, a_i)查分析表 M，设 $M[X_m, a_i]$为一产生式，譬如说 $X_m \to UVW$，此时将 X_m 从分析栈中退出，并将 UVW 按反序推入栈中(即用该产生式推导一步)，从而得到新的格局

$$\#X_1X_2\cdots X_{m-1}WVU \qquad a_ia_{i+1}\cdots\#$$

但若 $M[X_m, a_i]$="ERROR"，则调用出错处理程序进行处理；

②若 $X_m = a_i \neq \#$，则表明栈顶符号已与当前正扫视的输入符号得到匹配，此时应将 X_m(即 a_i)从栈中退出，并将输入符号指示器向前推进一个位置；

③若 $X_m = a_i = \#$，则表明输入串已完全得到匹配，此时即可宣告分析成功而结束分析工作。

顺便提及，在上述分析过程的每一步，可视需要附加相应的语义处理工作。

例如　考虑文法 G[E]：

$E \to TE'$

$E' \to +TE' \mid \varepsilon$

$T \to FT'$

$T' \to *FT' \mid \varepsilon$

$F \to (E) \mid i$

相应的分析表如表 4.1 所示(其构造方法，在后面叙述)。现在以输入符号串 i+i*i 为例，列出利用上述算法对此符号串的分析过程如表 4.2 所示。