

# LR分析中的短语与句柄



什么是LR分析中的phrase 和 direct phrase以及handle？它们之间有什么关系，如何用数学表达式来定义？

在编译原理中，**LR分析**（一种自底向上的语法分析方法）中出现了术语：**phrase**、**direct phrase** 和 **handle**。它们都是描述语法分析过程中某个串与产生式之间关系的术语。下面我们来详细解释每一个，并说明它们之间的联系，并尝试用数学表达式进行定义。

## 一、基本定义

### 1. Phrase（短语）

一个**短语（phrase）**是语法分析树（派生树）中某个内部结点所对应的子树所产生的字符串。换句话说，它是能由某个非终结符推导出来的一段连续的符号串。

- 数学定义：

对于一个文法  $G = (V_N, V_T, P, S)$ ，一个句子  $w \in L(G)$ ，若有推导：

$$S \Rightarrow^* \alpha A \beta \Rightarrow \alpha \gamma \beta$$

那么  $\gamma$  就是一个**短语（phrase）**，其中  $\gamma \in (V_T \cup V_N)^*$ 。

### 2. Direct Phrase（直接短语）

直接短语是一个特殊的短语，它直接由某个产生式一次推出，而不经过多步推导。

- 数学定义：

若存在某个产生式  $A \rightarrow \gamma \in P$ ，并且在某次推导中有：

$$S \Rightarrow^* \alpha A \beta \Rightarrow \alpha \gamma \beta$$

那么  $\gamma$  是**直接短语（direct phrase）**。

- 所以，所有的 direct phrase 都是 phrase，但并非所有 phrase 都是 direct phrase。

### 3. Handle（句柄）

句柄是当前右推导中可归约的最右的一个**直接短语**。在LR分析中，找到句柄并将其归约是核心步骤。

- 数学定义：

若对某个句子  $\omega \in L(G)$  存在一个右最左推导（右推导的反向），如下：

$$S \Rightarrow^* \alpha A \beta \Rightarrow \alpha \gamma \beta = \omega$$

并且  $\gamma$  是  $A \rightarrow \gamma \in P$  的右部，那么  $\gamma$  就是句子  $\omega$  中的一个**句柄（handle）**。

## 二、三者的关系

可以用集合关系表示三者之间的关系：

- 所有的 **direct phrase** 属于 **phrase** 的子集。
- 所有的 **handle** 是某次推导中可以归约的 **direct phrase**，所以 handle 是 direct phrase 的子集。

用集合包含关系表示：

$$\text{handle} \subseteq \text{direct phrase} \subseteq \text{phrase}$$

图示关系（逻辑结构）：

markdown



## 三、举例说明

考虑文法：

less

```

S → aABe
A → Abc | b
B → d
  
```

分析串：`abbcde`

其推导过程如下（右推导）：

CSS

```

S
⇒ aABe
⇒ aAbcBe      (A → Abc)
⇒ abbcBe      (A → b)
⇒ abbcde      (B → d)
  
```

在这个过程中：

- `bbc` 是 A 的短语（因为 A 推导出 Abc 再推导出 bbc），所以它是 **phrase**。
- `bc` 是  $A \rightarrow Abc$  的一次推导，不是直接短语；但 `Abc` 是直接短语。
- `d` 是  $B \rightarrow d$  的直接短语。
- 在归约 `B → d` 时，`d` 是 **handle**。
- 在归约 `A → b` 时，`b` 是 **handle**（因为那是最后要归约的 b）。

## 四、总结表格

概念	定义简述	数学表达式或条件	与其他的关系
Phrase	某非终结符能推导出的子串	若 $S \Rightarrow^* \alpha A \beta \Rightarrow \alpha \gamma \beta$ , 则 $\gamma$ 是 phrase	最广义
Direct Phrase	一次使用某产生式推导出来的 phrase	若 $A \rightarrow \gamma \in P$ , 且 $S \Rightarrow^* \alpha A \beta \Rightarrow \alpha \gamma \beta$	phrase 的子集
Handle	当前归约的直接短语（最右的）	$\gamma$ 是当前句子中某右推导所用的 $A \rightarrow \gamma$ 的右部	direct phrase 的子集, 唯一

需要我为你画一张结构图或用具体文法再演示一次完整的 LR 分析过程来辅助理解吗？



对于一个文法  $G = (V_N, V_T, P, S)$ ,  $G = (V_N, V_T, P, S)$ , 一个句子  $w \in L(G)$ ,  $w \in L(G)$ , 若有推导:  $S \Rightarrow^* \alpha A \beta \Rightarrow \alpha \gamma \beta$   $\Rightarrow^* \alpha A \beta \Rightarrow \alpha \gamma \beta$  那么  $\gamma$  就是一个 \*\*短语 (phrase)\*\* , 其中  $\gamma \in (V_T \cup V_N)^*$ ,  $\gamma \in (V_T \cup V_N)^*$ .

这里的  $w$  句子有什么用？短语是在句子中讨论的还是句型中讨论的？

这个问题非常好，切中了形式语言理论中一个容易混淆的核心点：**短语 (phrase)** 是在哪个“层级”上定义的，是在句型中，还是在完整句子（句子串）中？

我们来一一澄清：

### ✓ “句子” $w \in L(G)$ 的作用是什么？

它的作用是：**确定短语所处的上下文。**

在编译原理中，我们关心的是：**某个串是不是某个文法生成的句子的一部分？它在句子中的结构是什么？**

所以：

- “短语”总是在一个句子（或至少是某个推导过程中的中间串）中讨论的。
- 我们并不是在空泛地讨论某个符号串  $\gamma$  是不是短语，而是在给定的句子中看它是不是某个非终结符的推导结果。

### ✓ 句型 vs 句子

- **句型 (sentential form)**：从文法的开始符号  $S$  经过若干步推导得到的中间结果串，可能含有终结符和非终结符。  
例如：`aABe` 是某个句子的句型。
- **句子 (sentence)**：从  $S$  推导出来的完全由终结符组成的串。  
例如：`abcde` 是某个文法生成的句子。

✱ **短语、直接短语、句柄，都是在某个“句型”或“句子”中讨论的子串。**没有具体的句型或句子，我们无法判断一个串是不是某个非终结符的“短语”。

## ✓ 举例澄清（结合你提到的数学表达）

考虑文法：

```
less
S → aABe
A → Abc | b
B → d
```

某个句子： $w = abbcde \in L(G)$

推导过程：

```
nginx
S ⇒ aABe
  ⇒ aAbcBe
  ⇒ abbcBe
  ⇒ abbcde
```

现在我们来找短语：

- 在 `abbcde` 这个句子中，子串 `bbc` 本身不是一个短语，但 `A → Abc → bbc` 所在的部分（从 `A` 推导出的子串）是 **短语**。
- 如果我们退回到 `aAbcBe` 这个句型，`Abc` 就是一个 **直接短语**（由产生式  $A \rightarrow Abc$  一步推出来）。
- 在 `abbcde` 中，`b` 就是 `A → b` 的句柄（因为它是当前句型中的某个直接短语，正在被归约）。
- 所以 `handle` 和 `direct phrase` 是相对于当前“句型”来说的，而“phrase”是更广义的，可以回溯多个推导层级。

## ✓ 总结回答

💬 短语是在“句子”中讨论的还是“句型”中讨论的？

- 短语**是定义在某个具体的句型或句子中的，即：某个字符串是否是某个非终结符在推导中所生成的部分。
- 你不能离开具体的推导串（句型/句子）来空谈一个短语。
- 给定句子  $w$  的意义在于：为我们提供判断“哪些子串是短语”的上下文。
- 所以更准确地说：短语是**在某个句型中**讨论的，而不是抽象地在语法规则中定义的。

需要我再结合 LR 分析步骤画个图来示意吗？或者你想基于某个文法手动分析一次句柄归约过程？



```
S ⇒ aABe
  ⇒ aAbcBe
```

$\Rightarrow abbcBe$

$\Rightarrow abbcde$

你举得这个例子对吗？我们需要的不是最右推导吗？

你说得没错，上面那个推导其实是**最左推导**（先替换最左边的非终结符  $A$ ），而在**句柄（handle）**的定义中，**我们关心的是最右推导**（总是先替换最右边的非终结符），因为句柄是那个最右的可归约的直接短语。

下面我给出同一个文法的**最右推导**过程，并在每一步标出句柄（即被归约的子串）。

## 文法

CSS

$S \rightarrow a A B e$

$A \rightarrow A b c$  (产生式  $p_1$ )

$A \rightarrow b$  (产生式  $p_2$ )

$B \rightarrow d$  (产生式  $p_3$ )

要得到句子  $w = abbcde$ ，我们做最右推导：

$$1. S \Rightarrow a A B e$$

(初始句型)

2. 替换最右非终结符  $B$  (用  $B \rightarrow d$ ) :

$$a A B e \Rightarrow a A d e$$

这里的句柄是  $d$  (对应产生式  $B \rightarrow d$  的右部)。

3. 再替换最右非终结符  $A$  (用  $A \rightarrow Abc$ , 产生式  $p_1$ ) :

$$a A d e \Rightarrow a A b c d e$$

这里的句柄是  $A b c$ 。

4. 最后一次替换最右非终结符  $A$  (用  $A \rightarrow b$ , 产生式  $p_2$ ) :

$$a A b c d e \Rightarrow a b b c d e$$

这里的句柄是  $b$ 。

5. 结果得到终结串：

$abbcde$

所以，完整的最右推导序列是：

$$S \Rightarrow a A B e \Rightarrow a A d e \Rightarrow a A b c d e \Rightarrow a b b c d e$$

在每一步所替换（归约）的大写红色部分，就是对应的**句柄**——它同时也是那一步推导用到的**直接短语**。

## 关键点小结

### 1. 最右推导 (rightmost derivation)

总是在当前串中选择“最右边的非终结符”进行替换。

### 2. 句柄 (handle)

在某个最右推导的逆过程（即归约过程）中，被归约的最右的直接短语。

### 3. 直接短语 vs. 短语

- 直接短语：在一步推导中，从某个非终结符直接展开得到的子串（如上例的  $d$ 、 $Abc$ 、 $b$ ）。
- 短语：可以是多步推导后，从某个非终结符生成的子串。

希望这个最右推导的示例，连同每步的句柄标注，能帮助你更清晰地理解 LR 分析中“句柄”为什么要用最右推导来定义！



句子一定是句型，但句型不一定是句子，这是一个句子是这是一个句型的充分条件，这样理解对吗？

是的，你的理解是对的。

- \*\*句型 (sentential form)\*\* 是从开始符号  $S$  推导出来的任意中间串，可能还含有非终结符；
- \*\*句子 (sentence)\*\* 是完全由终结符组成的推导终点。

因此：

- 每个句子在推导过程中必然是某个句型（只不过是推到“只含终结符”这一特殊句型），所以

$$\text{sentence} \implies \text{sentential form}.$$

这就是说，“**句子是句型的充分条件**”——只要是句子，就一定是句型。

- 但一个句型可能尚未完成所有推导，还含有非终结符，不一定能直接算作句子，所以

$$\text{sentential form} \not\Rightarrow \text{sentence}.$$

也就是说，“句型”本身并不足以保证它是一个句子。

从逻辑关系上看，这正是一种**子集关系**：

$$\{\text{所有句子}\} \subsetneq \{\text{所有句型}\}.$$

- 如此，属于“句子”这一集合的字符串，一定也属于“句型”这一集合（充分条件）；
- 反过来，属于“句型”的，不一定属于“句子”中（必要条件但不充分）。

小结：

- “句子  $\Rightarrow$  句型”  $\Rightarrow$  句子是句型的充分条件。
- “句型  $\Rightarrow$  句子” 不成立  $\Rightarrow$  句型不是句子的充分条件。
- 等价地，句型是句子的**必要条件**，而句子是句型的**充分条件**。