$Chapter\ 2 - HW01$

2015K8009929049 冯吕

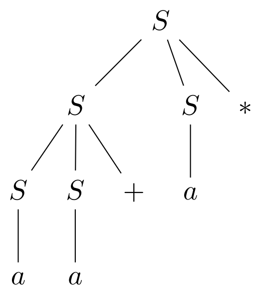
2018 年 7 月 9 日

2.2.1 解：

1) 生成串 $aa+a*$ 的过程如下，以最左推导为例：

$$S \to S\ S * \to S\ S\ +\ S * \to a\ S\ +\ S * \to aa+\ S * \to aa+a*$$

2) 该串的语法分析树如下：



3) 该文法生成的语言是运算数全为 a，带有加法和乘法运算的后缀算术表达式的集合。

4) 没有二义性。

$proof$:

- 先证明一个该文法产生串的长度的结论：设串的推导过程中使用产生式 $S \to S\ S+$ 和 $S \to S\ S*$ 的次数为 m，则串的长度为 $L = 2 \times m + 1$，且串中包含 m 个运算符和 $m+1$ 个 a；
  - 1) 当 $m = 0$ 时，仅有 $S \to a$ 一种情况，此时 $L = 1$，串由 1 个 a 和 0 个运算符构成，结论成立；
  - 2) 设当 $m < k(k \geq 1)$ 时结论成立，则当 $m = k$ 时，第一步推导必然为

    $$S \to S_1 S_2\ op$$

    op 为 + 或 $*$。设 $S_1 \to \alpha, S_2 \to \beta$，$\alpha, \beta$ 均为使用 $S \to S_1 S_2\ op$ 少于 k 次得到的串，设二者推导过程中分别使用该产生式 k_1 和 k_2 次，根据假设有：

    $$L(\alpha) = 2 * k_1 + 1, L(\beta) = 2 * k_2 + 1$$

    则串长度 $L = L(\alpha)+L(\beta)+1 = 2*(k_1+k_2+1)+1 = 2k+1$；且串中 a 的个数为 $(k_1+1)+(k_2+1) = k+1$；运算符的个数为 $k_1+k_2+1 = k$，故结论成立。

- 下面证明该文法无二义性，对串的长度做归纳。由前述证明可知，该文法产生的串长 L 可为任意非负奇数。对由该文法得到的长度为 $K = 2*k+1$ 的串 w：
  - 1) 当 $k = 0$ 时，$L = 1$，只有 $S \to a$ 一种情况，显然没有二义性;

- 2) 设当 $k < n$ 时结论成立。$S \to \omega$，根据 ω 末尾运算符可确定第一步推导使用的产生式，不妨设为：

$$S \to S_1 S_2 +$$

从后向前处理串 ω，除去末尾的运算符，找到可以由 S 推导出的最短的串 α，设 α 长度为 m_1，由前述结论可知 $m_1 = 2 * k_1 + 1$，且 α 包含 k_1 个运算符和 $k_1 + 1$ 个 a，由归纳假设可知 α 无二义性，存在唯一的最左推导 $S \to \alpha$;

设串 ω 剩余部分为 β，设 β 的长度为 m_2，同理可得 $m_2 = 2 * k_2 + 1$，β 包含 k_2 个运算符与 $k_2 + 1$ 个 a，存在唯一最左推导 $S \to \beta$，且满足 $k = k_1 + k_2$。此时串 ω 可表示成如下形式：

$$\omega = \beta\alpha +$$

故存在唯一的最左推导：

$$S \to S\ S + \to \beta S + \to \beta\alpha +$$

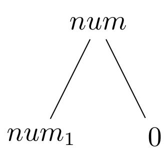
此时，仍不存在二义性。

综上所述，该文法不具有二义性。

2.2.5 解：

1)$proof$：用归纳法证明：当生成的串的语法树的节点 ≤ 3 时，生成的串有 $11, 1001$，对应的十进制的值为 $3, 9$，能够被 3 整除。假设节点数 $< n$ 的语法树生成的二进制串均能够被 3 整除，考虑节点数为 n 的语法树，它有下面两种可能的结构：

- 情形一：

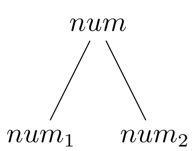


子树 num_1 的节点数 $\leq n$，因此，num_1 生成的二进制串（记为 w_1）能够被 3 整除，则 num 生成的二进制串（记为 w）：

$$w = w_1 \times 2$$

也可以被 3 整除。

- 情形二：



子树 num_1 和 num_2 的节点数均小于 n，因此，它们生成的二进制串（分别记为 w_1 和 w_2）能够被 3 整除，则 num 生成的二进制串（记为 w）：

$$w = w_1 \times 2^n + w_2$$

也能够被 3 整除。

综上，该文法生成的二进制串能够被 3 整除。

2) 不能。例如，对于二进制串 10101，它的值为 21，能够被 3 整除，但是不可以通过上面的文法推导出。

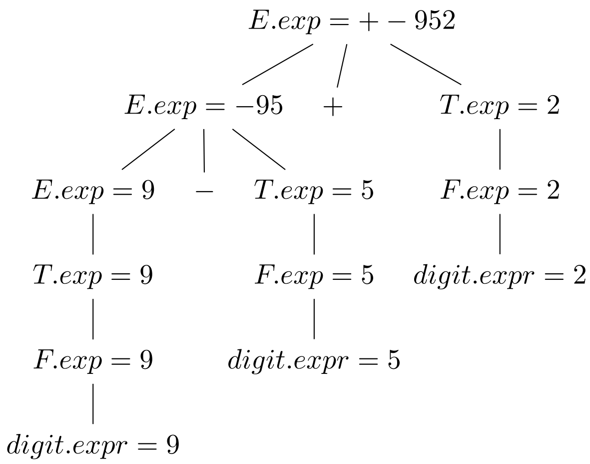
2.3.1 解：制导翻译方案如下：

```
E -> {print("+");} E1 + T
    |{print("-");} E1 - T
    |T

T -> {print("*");} T1 * F
    |{print("/");} T1 / F
    |F

F -> digit{print(digit)}
    |(expr)
```

表达式 $9-5+2$ 对应的注释语法分析树如下：



表达式 $9-5*2$ 对应的注释语法分析树如下：