目录

Contents:

1 上下文无关文法

考虑前面学过的, $\{0^n1^n \mid n \in \mathbb{N}\}$ 不是正则表达式的语言.

Example 1.1 (回文)**.** *if* $w \in \Sigma^*, w = w^R$, *then* w 为回文

Example 1.2 (回文语言)**.** *if* $L = \{w \in \Sigma^* \mid w = w^R\}$ *then* L 是回文语言.

运用前面的知识, 可以使用 Pump lemma 证明 L 不是正则的. 可是那么我们该如何表示该语言呢?

可以使用递归方法定义:

1. 首先 $\epsilon, 0, 1$ 都是回文.

2. if w 是回文, 那么 $0w0, 1w1$ 都是回文.

数理逻辑之中, 命题的递归定义也是类似的, 这种生成语句的规则就是上下文无关文法.

$$\left.\begin{aligned} &1.A \to \epsilon \\ &2.A \to 0 \\ &3.A \to 1 \\ &4.A \to 0A0 \\ &5.A \to 1A1 \end{aligned}\right\} \tag{1}$$

定义 1.3 (文法)**.** 文法 G 是一个 $4p$ 结构, $G = (V, T, P, S)$. 其中 V for variable, T for terminator, P for Production, S for start.

1 V 是变量的集合.

2 T 称为 terminators.

3 P 是生产规则, 形式为 $A \to \alpha \mid \beta$, 其中 α 是 ϵ 或者是 $\alpha \in T$, 或者是 $\alpha \in E$, 或者是 $\alpha \in (V \cup T)^*$, β 也是如此.

4 $S \in V$, 表示的是开始的变量, 所有句型, 语句都是从 S 开始派生的.

Example 1.4 (0,1 组成的回文)**.** $G = (V, T, P, S)$, 其中 $V = \{A\}$, $T = \{0, 1\}$, $P = A \to \epsilon \mid 0 \mid 1 \mid 0A0 \mid 1A1$, $S = A$

Example 1.5 (0^n1^n)**.** $L = \{0^n1^n \mid n \in \mathbb{N}\}$ 和上面一个完全类似, 这里就懒得说了

1.1 归约和派生

定义 1.6 (文法派生的句型)**.** $\alpha \in (V \cup T)^*$, α 可能是派生出的语句. 因为考虑到 $A \to t$, $t \in T$, 或者是 $A \to At$. 也就是说, 这个语句里面只会有 variable or terminator.

定义 1.7 (归约)**.**

定义 1.8 (派生)**.** $\alpha, \beta, \gamma \in (V \cup T)^*$, if $A \to \gamma$, then $\alpha A \beta \underset{G}{\Rightarrow} \alpha\gamma\beta$.

Example 1.9 (派生的例子)**.**

定义 1.10 (多步派生)**.**

$$\alpha_1 \overset{i}{\underset{G}{\Rightarrow}} \alpha_{1+i} \tag{2}$$

对于一般的多步派生, 记为 $\overset{*}{\underset{G}{\Rightarrow}}$

定义 1.11 (最左派生, 最右派生)**.** 对于一个句型 α, 我们考虑只对最左 (右) 的变量进行派生, 这就是最左 (右) 派生. 记为 $\underset{lm}{\Rightarrow}$ $(\underset{rm}{\Rightarrow})$

定理 1.12 (派生的**等价性**)**.** 对于任意一个派生, 都存在一个最左 (右) 派生与其对应

1.2 文法的语言

定义 1.13 (语言)**.**

$$L(G) = \{w \mid w \in T^*, S \overset{*}{\underset{G}{\Rightarrow}} w\} \tag{3}$$

Remark 1.14 (为什么称为是上下文无关文法). 我们参考 $\alpha A \beta \underset{G}{\Rightarrow} \alpha\gamma\beta$, 这个派生是否成立, 是跟 α, β 无关的, 于是称为上下文无关文法.

定义 1.15 (等价性)**.** G_1, G_2 满足 $L(G_1) = L(G_2)$, 则 G_1 等价于 G_2

定义 1.16 (句型, 左句型, 右句型)**.** G 生成的句型定义如下:

$$\{w \mid w \in (V \cup T)^*, S \overset{*}{\underset{G}{\Rightarrow}} w\} \tag{4}$$

也定义了左句型, 右句型.

Example 1.17. $L = \{a^n b^n \mid n \geq 1\}$

Example 1.18. $G = \{\}$

Example 1.19. $L = \{w \mid 0,1\ 数量相等\}$, $G = (V, T, P, S)$, 其中 $V = \{S\}, T = \{0, 1\}$, P 是下面这个:

$$S \to 0S_1 1S_2 \mid 1S_1 0S_2 \mid \epsilon$$

Example 1.20 (算术表达式)**.**

2 语法分析树

语法分析树是表示派生过程的一个树

定义 2.1 (分析树)**.** 定义为

1 每个内节点是边缘符号

2 叶子节点是 $V \cup T \cup \{\epsilon\}$ 之中的符号

3 如果说内节点的标记是 A, 那么其子节点从左到右分别为

$$X_1, X_2, \ldots, X_n$$

那么 $A \to X_1 X_2 \ldots X_n$ 是一个产生式.

注意到空船.

如果说, 根节点是初始符号 s, 叶子节点是终结符, 那么该树是完成的, 该树的产物属于 $L(G)$.

定义 2.2 (subtree)**.** 在树 T 之中以内节点 A 为根节点的 subtree 为 T 的 subtree

定理 2.3 (the equivalence of tree)**.** 一棵树和 $L(G)$ 之中的一个语句等价.

1. G 存在 A 为根节点的树, 其产物为 α, then $A \overset{*}{\Rightarrow} \alpha$
2. 如果说 $A \overset{*}{\Rightarrow} \alpha$ then 存在 A 为根的树, 其产物为 α

证明. □

3 文法和语言的歧义性

对于 G 之中某些语句有不同的语法分析树, 则 G 是有歧义的.

比如说对于算术表达式的文法 G_{exp} 之中 $a \cdot a + a$ 之中, **有多种的阅读顺序**, 这就导致了歧义.

Example 3.1 (表示加法乘法的优先级)**.** *我们考虑 P*

$$\begin{aligned} P = \{E &\to E + T \mid T, \\ T &\to T \cdot F \mid F, \\ F &\to (E) \mid I\} \end{aligned} \tag{5}$$

定义 3.2 (固有歧义)**.** 对于一些语言, 其对应的文法都是有歧义的.

Example 3.3. *对于 $L = \{a^i b^j c^k \mid i = j \ or \ j = k\}$, L 是固有歧义的.*

4 文法的化简和设计

4.1 化简

1 消除无用符号

2 消除 ϵ 产生式 (ϵ production)

3 消除单元产生式 (unit production) :$A \to B$

4.1.1 无用符号的消除

我们将一些无用的符号进行化简, 分为两类, 一类是不可达的, 一类是不可产生的, 这两种符号都是没有用的, 这是说, 对于一个语句的派生过程, 这些符号是不会出现的. 所以说是无用符号. 下面给出定义.

定义 4.1 (Useful)**.** 对于 $G = (V, T, P, S)$, 符号 $X \in (V \cup T)$.

1. 如果 $G \overset{*}{\Rightarrow} \alpha X \beta$ then X is **reachable**.
2. if $\alpha X \beta \overset{*}{\Rightarrow} w \in T^*$ then X is **generating**.
3. If X is reachable and generating, then X is **useful**. X is useless otherwise.

Remark 4.2. T is generating, while S is reachable. You can easily prove it.

Example 4.3. *对于 $P = \{S \to AB \mid a, A \to b\}$ 可达的符号机和产生的符号分别为什么?*

证明. $T = \{a, b\}, A, S$ 是产生的, S, A, B, a, b 是可达的, □

Remark 4.4. 能够看出, 我们要找到产生的, 则需要**从后往前找**, 若是要找到可达的, 则需要**从前往后找**.

4.1.2 ϵ 产生式

如果说 $A \Rightarrow \epsilon$, 称 A 是可空的. ϵ 产生式除了贡献 L 中的 ϵ 之外没有任何作用, 于是说, 当 L 之中没有 ϵ 的时候, 我们就可以将 G 之中的 ϵ 产生式消除掉. 这 (大概) 就是原理.

定义 4.5 (可空变元)**.**

Example 4.6. *消除 $G = (\{S, A, B\}, \{a, b\}, P, S)$ 的 ϵ 产生式.*

$$\begin{aligned} P = \{S &\to AB, \\ A &\to AaA \mid \epsilon \\ B &\to BbB \mid \epsilon\} \end{aligned} \tag{6}$$

Example 4.7. $G_2 = (\{S, A, B, C\}, \{a, b\}, P, S)$

$$\begin{aligned} P = \{S &\to ABC, \\ A &\to aA \mid \epsilon, \\ B &\to bB \mid \epsilon, \\ C &\to \epsilon\} \end{aligned} \tag{7}$$

4.1.3 单元产生式的消除

If $A \Rightarrow B$ then A, B are equivalent. You may view A, B as a same variable and merge the relative productions.

Example 4.8.

$$\begin{aligned} P = \{S &\to A \mid B \mid 0S1, \\ A &\to 0A \mid 0, \\ B &\to 1B \mid 1\} \end{aligned} \tag{8}$$

证明. It is easy to note that S, A, B are equivalent. Then $P = \{S \to 0S1 \mid 0S \mid 0 \mid 1S \mid 1\}$ □

4.1.4 化简步骤

1 消除 ϵ 产生式

2 reduce the unit productions

3 Reduce the not generating productions

4 Reduce the not reachable productions