

## 第二章作业

袁晨圃

### 2.1

**Exercise 1 (2.2):**

- 利用语言  $A = \{a^m b^n c^n \mid m, n \geq 0\}$  和  $B = \{a^n b^n c^m \mid m, n \geq 0\}$  以及例 2.20 证明上下文无关语言在交运算下不封闭
- 利用上一问和德摩根律证明上下文无关语言在补运算下不封闭

*Solution:*

- 语言  $A$  和  $B$  都是上下文无关的

证明：构造 CFG

产生  $A$  的 CFG:

$$X \rightarrow aX \mid Y$$

$$Y \rightarrow bYc \mid \varepsilon$$

产生  $B$  的 CFG:

$$X \rightarrow Xb \mid Y$$

$$Y \rightarrow aYb \mid \varepsilon$$

然而  $A \cap B = \{a^n b^n c^n \mid n \geq 0\}$ ，在例 2.20 中已经证明了它不是上下文无关的

- 1.5**

若上下文无关语言在补运算下封闭，则非上下文无关语言的补集也是非上下文无关语言，然而  $A \cap B$  的补集是  $\overline{A \cap B}$  即  $\{a^m b^n c^p \mid m, n, p \geq 0, n \neq m \text{ or } n \neq p\}$  是上下文无关的，下面给出生成该语言的 CFG.

$$X \rightarrow C_x \mid A_x$$

$$C_x \rightarrow C_x c \mid C$$

$$C \rightarrow C_a \mid C_b$$

$$C_a \rightarrow aC_a \mid aC_0$$

$$C_b \rightarrow C_b b \mid C_0 b$$

$$C_0 \rightarrow aC_0 b \mid \varepsilon$$

$$A_x \rightarrow aA_x \mid A$$

$$A \rightarrow A_b \mid A_c$$

$$A_b \rightarrow bA_b \mid bA_0$$

$$A_c \rightarrow A_c c \mid A_0 c$$

$$A_0 \rightarrow bA_0 c \mid \varepsilon$$

其中  $C_0$  产生  $\{a^n b^n \mid n \geq 0\}$ ,  $C_a$  产生  $\{a^m b^n \mid m > n \geq 0\}$ ,  $C_b$  产生  $\{a^n b^m \mid m > n \geq 0\}$ ,  $C$  产生  $\{a^m b^n \mid m, n \geq 0; m \neq n\}$ ,  $A \dots$  类似

$\forall C, D \in \text{CFL}$ , 假设命题成立则有  $\overline{C}, \overline{D}$  也是 CFL, 又 CFL 关于并封闭, 所以  $\overline{C} \cup \overline{D}$  也是 CFL, 所以  $\overline{C \cup D} = \overline{C \cap D}$  也是 CFL, 所以  $C \cap D$  也是 CFL, 第一问矛盾。

### Exercise 2 (2.6 b):

给出产生  $\{a^n b^n \mid n \geq 0\}$  的补集的 CFG

Solution:

$$S \rightarrow S_1 \mid S_2$$

$$S_1 \rightarrow aS \mid Sa \mid bS \mid Sb \mid ba \quad (\text{不符合 } a^m b^n, m, n \geq 0 \text{ 的字符串})$$

$$S_2 \rightarrow A \mid B \quad (\text{符合 } a^m b^n, m, n \geq 0 \text{ 但 } m \neq n \text{ 的字符串})$$

$$A \rightarrow aA \mid aE$$

$$B \rightarrow Bb \mid Eb$$

$$E \rightarrow aEb \mid \varepsilon \quad (\text{符合 } a^n b^n \text{ 的字符串})$$

### Exercise 3 (2.14):

用定理 2.6 中给出的过程, 把下述 CFG 转换为等价的乔姆斯基文法

$$A \rightarrow BAB \mid B \mid \varepsilon$$

$$B \rightarrow 00 \mid \varepsilon$$

Solution:

$$\#1 \ S_0 \rightarrow A$$

$$A \rightarrow BAB \mid B \mid \varepsilon$$

$$B \rightarrow 00 \mid \varepsilon$$

$$\#2 \text{ i. } S_0 \rightarrow A$$

$$A \rightarrow BA \mid AB \mid B \mid \varepsilon \mid \textcolor{red}{BAB} \mid \textcolor{red}{A}$$

$$B \rightarrow 00$$

$$\text{ii. } S_0 \rightarrow A \mid \varepsilon$$

$$A \rightarrow BA \mid AB \mid B$$

$$B \rightarrow 00$$

$$\#3 \text{ i. } S_0 \rightarrow BA \mid AB \mid B \mid \varepsilon \text{ 以下省略}$$

$$A \rightarrow BA \mid AB \mid B$$

$$B \rightarrow 00$$

$$\text{ii. } S_0 \rightarrow BA \mid AB \mid 00 \mid \varepsilon$$

$$A \rightarrow BA \mid AB \mid 00$$

$B \rightarrow 00$   
 #4 i.  $S_0 \rightarrow BA \mid AB \mid ZZ \mid \varepsilon$   
 $A \rightarrow BA \mid AB \mid ZZ$   
 $B \rightarrow ZZ$   
 $Z \rightarrow 0$

**Exercise 4** (2.18):

考虑下面的 CFG  $G$  :

$$\begin{aligned}
 S &\rightarrow SS \mid T \\
 T &\rightarrow aTb \mid ab
 \end{aligned}$$

描述并证明  $G$  是歧义的。给定一个非歧义文法  $H$  满足  $L(H) = L(G)$  并简要证明  $H$  是非歧义的。

*Solution:* 生成  $ababab$  时, 可以有两种生成语法分析树:

$$S \rightarrow SS \rightarrow TS \rightarrow abS \rightarrow abSS \rightarrow ababS \rightarrow ababab$$

$$S \rightarrow SS \rightarrow SSS \rightarrow abSS \rightarrow ababS \rightarrow ababab$$

因此  $G$  是歧义的。

非歧义文法  $H$  :

$$S \rightarrow TS \mid T$$

$$T \rightarrow aTb \mid ab$$

通过限制  $T$  只能添加至  $S$  的最左边, 避免了歧义。

**Exercise 5** (2.37):

对于语言  $A$  定义  $\text{SUFFIX}(A) = \{v \mid \exists u, \text{s.t. } uv \in A\}$  证明 CFL 在 SUFFIX 运算下封闭。

*Solution:* 设 PDA  $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$  识别  $A$ , 构造 PDA  $M' = (Q', \Sigma, \Gamma, \delta', q'_0, F')$  识别  $\text{SUFFIX}(A)$ 。

先将  $M$  复制到  $M'$

然后复制状态集  $Q$  到  $Q_\varepsilon$ ,  $Q$  中的每一个状态  $r_k$  都与一个状态  $r_{k\varepsilon}$  对应, 表示接受  $\varepsilon$  作为虚拟前缀, 模拟接受到  $r_k$  表示的字符串时  $M$  的行为, 下面介绍如何构造  $Q_\varepsilon$  内部的转移  $\delta_\varepsilon$  以及  $Q$  和  $Q_\varepsilon$  之间的转移  $\delta_0$ :

对于  $\delta$  中的每一个转移, 将输入换为  $\varepsilon$  之后添加到  $\delta_\varepsilon$

然后对于任意的  $k$ , 在  $r_{k\varepsilon}$  和  $r_k$  之间构造  $\varepsilon$  转移  $\delta_0(r_{k\varepsilon}, \varepsilon, \varepsilon) = (r_k, \varepsilon)$

令  $Q' = Q \cup Q_\varepsilon, \delta' = \delta \cup \delta_\varepsilon \cup \delta_0, q'_0 = q_{0\varepsilon}, F' = F$

这样就构造出了一个识别  $A$  中字符串的后缀的 PDA  $M'$

## 2.2

**Exercise 6 (2.11):**

用定理 2.12 中给出的过程，把下述 CFG  $G_4$  转换成 PDA

$$\begin{aligned} E &\rightarrow E + T \mid T \\ T &\rightarrow T \times F \mid F \\ F &\rightarrow (E) \mid a \end{aligned}$$

*Solution:* 设 PDA  $P = \{Q, \Sigma, \Gamma, \delta, q_{\text{start}}, F\}$  其中  $Q = \{q_{\text{start}}, q_{\text{loop}}, q_{\text{accept}}\}$

$$\delta(q_{\text{start}}, \varepsilon, \varepsilon) = \{(q_{\text{loop}}, E\$)\}$$

$$\delta(q_{\text{loop}}, \varepsilon, E) = \{(q_{\text{loop}}, T), (q_{\text{loop}}, E + T)\}$$

$$\delta(q_{\text{loop}}, \varepsilon, T) = \{(q_{\text{loop}}, F), (q_{\text{loop}}, T \times F)\}$$

$$\delta(q_{\text{loop}}, \varepsilon, F) = \{(q_{\text{loop}}, (E)), (q_{\text{loop}}, a)\}$$

$$\delta(q_{\text{loop}}, a, a) = \{(q_{\text{loop}}, \varepsilon)\} \text{ 其他终结符呢?}$$

$$\delta(q_{\text{loop}}, +, +) = \{(q_{\text{loop}}, \varepsilon)\}$$

$$\delta(q_{\text{loop}}, |, |) = \{(q_{\text{loop}}, \varepsilon)\}$$

$$\delta(q_{\text{loop}}, (, () = \{(q_{\text{loop}}, \varepsilon)\}$$

$$\delta(q_{\text{loop}}, ), )) = \{(q_{\text{loop}}, \varepsilon)\}$$

$$\delta(q_{\text{loop}}, +, +) = \{(q_{\text{loop}}, \varepsilon)\}$$

$$\delta(q_{\text{loop}}, \times, \times) = \{(q_{\text{loop}}, \varepsilon)\}$$

$$\delta(q_{\text{loop}}, \varepsilon, \$) = \{(q_{\text{accept}}, \varepsilon)\}$$

其中  $(q_2, w) \in \delta(q_1, k, a)$ ,  $w = w_1 w_2 \dots w_n$ ,  $n > 1$ ,  $k \in \Sigma^*$ ,  $a \in \Gamma^*$  表示创建  $n - 1$  个中间状态  $q_{\text{tmp}_i}$  , 并且有

$$\begin{aligned} (q_{\text{tmp}_1}, w_1) &\in \delta(q_1, k, a) \\ \delta(q_{\text{tmp}_1}, \varepsilon, \varepsilon) &= (q_{\text{tmp}_2}, w_2) \\ &\vdots \\ \delta(q_{\text{tmp}_{n-1}}, \varepsilon, \varepsilon) &= (q_2, w_n) \end{aligned}$$

顺序写反了，应该是  $w_n \dots w_1$

**Exercise 7 (2.58 a):**

$\Sigma = \{0, 1\}$ ,  $B$  为后一半中至少包含一个 1 的所有串的集合。即:  $B = \{uv \mid u \in \Sigma^*, v \in \Sigma^* 1 \Sigma^*, |u| \geq |v|\}$

设计一个识别  $B$  的 PDA

*Solution:* 先设计一个 CFG  $G$  生成  $B$

$$S \rightarrow CS0 \mid T$$

$$T \rightarrow R1$$

$$R \rightarrow CR \mid C$$

$$C \rightarrow 0 \mid 1$$

对于  $B$  中的字符串  $w$ ， $S \rightarrow T$  的过程会删去  $w$  所有的后缀 0，并且同时对称地去掉前面等长的字符，因为  $w$  后一半至少包含一个 1，所以最后剩下的结果一定满足  $w'1$  的形式，其中  $|w'| > 1$ ，接着  $R$  会匹配  $w'$

对于任意一个由  $G$  生成的字符串  $w$ ，因为起始变元一定会生成  $w_1Tw_2, |w_1| = |w_2|$ ，只要  $T$  后半含 1， $w_1Tw_2$  后半就会含 1，而  $T$  中后半肯定含 1（因为最后一个字符是 1 且串长  $\geq 2$ ），所以  $w \in B$

因此  $G$  生成  $B$

然后使用上一题的方法把 CFG 转换成 PDA 即可

法二：要求  $|u| \geq |v|$ ，则在读  $u$  的时候压栈，读  $v$  的时候弹栈

## 2.3

### Exercise 8 (2.44):

对于字母表  $\Sigma = \{1, 2, 3, 4\}$  上的语言  $C = \{w \in \Sigma^* \mid w \text{ 中, } 1 \text{ 与 } 2 \text{ 个数相同, } 3 \text{ 与 } 4 \text{ 个数相同}\}$ ，证明  $C$  不是上下文无关语言

*Solution:*  $\forall p$  考虑  $w = 1^p 3^p 2^p 4^p$

选取  $u, v, x, y, z$  的时候，因为有限制  $|vxy| \leq p$ ，所以无法做到使得  $v, y$  同时选到  $\{1, 2\}$  或者  $\{3, 4\}$ ，这样构造出的新的字符串  $uv^i xy^i z$  中，1 和 2 或者 3 和 4 的数量就会不相等。

所以不存在这样的一个适用于泵引理的  $p$

### Exercise 9 (2.58b):

$\Sigma = \{0, 1\}$ ,  $B$  为后半中至少包含一个 1 的所有串的集合。即：  $B = \{uv \mid u \in \Sigma^*, v \in \Sigma^* 1 \Sigma^*, |u| \geq |v|\}$

设计一个识别  $B$  的 CFG

*Solution:* 似乎已经在上一次作业写完了（笑

$$S \rightarrow CS0 \mid T$$

$$T \rightarrow R1$$

$$R \rightarrow CR \mid C$$

$$C \rightarrow 0 \mid 1$$

**Exercise 10 (\*2.55):**

对于字符串  $w$  和  $t$ ，如果二者字符数相等且组成的字符相同（只是字符在串中的顺序不同），则称  $w \overset{\circ}{=} t$

在此基础上给出 SCRAMBLE 定义如下：对字符串  $w$  有  $\text{SCRAMBLE}(w) = \{t \mid t \overset{\circ}{=} w\}$

对语言  $A$ ，有  $\text{SCRAMBLE}(A) = \{t \mid \exists w \in A, \text{s.t. } t \in \text{SCRAMBLE}(w)\}$

- 证明：给定字母表  $\Sigma = \{0, 1\}$ ，正则语言 SCRAMBLE 运算后成为上下文无关语言
- 当字母表包含 3 个或更多符号时会有什么样的结果？证明你的结果

*Solution:* 都会成为上下文无关语言，下面尝试给出证明

对于正则语言  $A$ ，假设有 DFA  $D$  识别它，接着把它改造成识别重排字符串的 PDA  $P$

对于  $D$  中每一个节点，除原先 DFA 的边保持不变以外，对于字母表中的每一个字符添加一个新的转移，将这个字符推入栈中，状态  $q$  不变，相当于暂时存储这个字符，先处理后面的字符以达到重新排列顺序的效果

然后对于原先 DFA 中的每一条边，添加一条新的对应边 **栈**：  $\epsilon \rightarrow \epsilon$ ，将输入转换为  $\epsilon$ ，栈顶换为原先的输入。

读入字符  $a$ ，push  $a$ ，读入字符  $b$ ，转移，pop  $a$ ，转移：这样的过程能做到读入  $ab$  实际按  $ba$  转移 i.e. 交换相邻元素

对于将原字符串  $w$  重排后的新字符串  $w'$  而言，每一次交换都可以使得逆序对数量减少，最终一定能通过有限次的相邻元素交换将逆序对数量减少至 0，即还原成原字符串  $w$ ，按照  $w$  转移到原 DFA 对应的状态上

**接受时需要让栈空**

对于两个符号可行，但是如果有三个符号，无法将  $abc$  转换为  $cab$ 。本质：入栈出栈合法序列个数是  $\text{Cat}_n$ ，不是  $n!$ ，肯定会有情况覆盖不到