

# Chapter 4

## 正则语言的性质

### 4.1 证明语言的非正则性

#### 4.1.1 泵引理 (Pumping Lemma)

这里给出正则语言的一个必要条件, 即“泵引理”. 如果一个语言是正则的, 则一定满足泵引理.

示例

$L = \{0^m 1^n \mid m, n \geq 0\}$  是否是正则语言?

$L = \{0^m 1^n \mid m, n \geq 1\}$  是否是正则语言?

$L = \{0^m 1^n \mid m \geq 2, n \geq 4\}$  是否是正则语言?

$L_{01} = \{0^n 1^n \mid n \geq 0\}$  是否是正则语言?

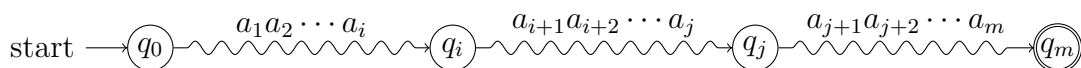
**定理 1.** (正则语言的泵引理) 若语言  $L$  是正则的, 则存在一个正整数 (常数)  $N$ , 对于任何  $w \in L$ , 只要  $|w| \geq N$ , 则可以把  $w$  分为三部分  $w = xyz$  使得

(1)  $y \neq \varepsilon$  ( $|y| > 0$ );

(2)  $|xy| \leq N$ ;

(3)  $\forall k \geq 0, xy^kz \in L$ .

证明. 如果  $L$  是正则的, 则存在 DFA  $A$ ,  $L = L(A)$ . 设  $A$  有  $n$  个状态, 对于长度不小于  $n$  的串  $w = a_1 a_2 \cdots a_m$  ( $m \geq n$ ), 定义  $q_i = \hat{\delta}(q_0, a_1 a_2 \cdots a_i)$  ( $i = 1, \cdots, m$ ),  $q_0$  是开始状态. 当  $A$  输入  $w$  的前  $n$  个字符时, 经过的状态分别是  $q_0, q_1, \cdots, q_n$  共  $n+1$  个, 根据鸽巢原理, 一定有两个状态相同, 即有满足  $0 \leq i < j \leq n$  的两个状态  $q_i = q_j$ .



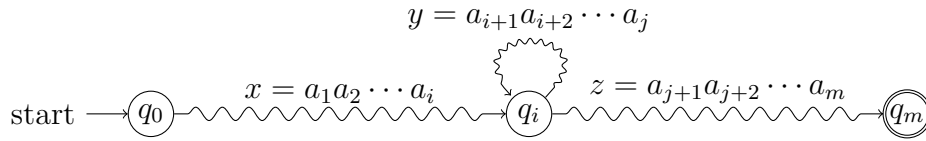
则  $w$  可以被分为三个部分:

$$x = a_1 a_2 \cdots a_i$$

$$y = a_{i+1} a_{i+2} \cdots a_j$$

$$z = a_{j+1} a_{j+2} \cdots a_m$$

如果从  $q_i$  出发, 输入  $y$ , 会到达  $q_j$ , 而因为  $q_i = q_j$ , 当输入  $y^k (k \geq 0)$ , 始终会回到  $q_i$ . 所以当 DFA  $A$  输入  $xy^kz$  时, 由  $q_0$  始终会达到  $q_m$ . 那么, 如果  $xyz \in L(A)$ , 有  $xy^kz \in L(A)$  对所有  $k \geq 0$  成立.



又因为  $i < j$  所以  $y \neq \varepsilon$  (即  $|y| > 0$ ), 并且因为  $i$  和  $j$  取自  $w$  的前  $n$  个字符, 所以  $|xy| \leq n$ .  $\square$

解释: 任何从开始状态到接受状态的路径, 如果长度超过  $n$ , 一定会经过  $n+1$  个状态, 必定有一个重复状态, 因此会形成一个循环 (loop); 那么, 这个循环可以被重复多次后, 沿原路径还会到达接收状态.

泵引理中的  $N$ , 是正则语言固有存在的, 可以近似的看做是状态的个数.

泵引理可以用来确定特定语言不在给定语言类 (正则语言) 中. 但是它们不能被用来确定一个语言在给定类中, 因为满足引理是类成员关系的必要条件, 但不是充分条件.

#### 4.1.2 泵引理的应用

pump in  
或 pump out

示例

证明  $L_{eq} = \{w \mid w \text{ 由数量相等的 } 0 \text{ 和 } 1 \text{ 构成}\}$  不是正则的.

问题 已知  $L_{01}$  不是正则, 能否用  $L_{01} \subseteq L_{eq}$  说明  $L_{eq}$  也不是正则的?

证明. 假设  $L_{eq}$  是正则的, 则一定存在正整数  $N$ , 对任何  $w \in L_{eq} (|w| \geq N)$  满足泵引理.

取  $w = 0^N 1^N$ , 则显然  $w \in L_{eq}$ ; 又因为  $|w| = 2N > N$ ,

那么有  $w = xyz$ , 且  $|xy| \leq N, |y| > 0$ ; 那么  $y$  一定是  $0^m (m = |y| > 0)$ ;

根据泵引理  $xy^2z \in L_{eq}$ , 但是  $xy^2z = 0^{N+m} 1^N$ , 则显然  $xy^2z \notin L_{eq}$ ;

因此与假设矛盾, 所以  $L_{eq}$  一定不是正则的.  $\square$

问题  $L = \{0^n x 1^n \mid n \geq 1, x \in \{0, 1\}^*\}$  是否是正则语言?  $\text{P}$

示例

证明  $L = \{0^i 1^j \mid i > j\}$  不是正则的.

证明. 假设  $L$  是正则的, 则一定存在正整数  $N$ , 对任何  $w \in L (|w| \geq N)$  满足泵引理.

取  $w = 0^{N+1} 1^N$ , 所以  $w \in L$ ; 因为  $|w| = 2N + 1 > N$ ,

那么有  $w = xyz$ , 且  $|xy| \leq N, |y| > 0$ , 这里设  $|y| = m$ , 那么有  $m > 0$ ;

根据泵引理  $xy^kz \in L (k > 0)$ ; 但是当  $k = 0$  时,  $xy^kz = xz = 0^{N+1-m} 1^N$ , 而  $N + 1 - m \leq N$ , 所以  $xz \notin L$ ;

因此与假设矛盾, 所以  $L$  一定不是正则的.  $\square$

### 示例

证明  $L = \{a^{n!} \mid n > 0\}$  不是正则的.

取  $w = a^{N!}$ , 当  $|y| = m$  时, 则  $|xy^2z| = N! + m$ , 而  $0 \leq m \leq N < N!$ , 所以  $N! < |xy^2z| = N! + m < N! + N! < N \cdot N! + N! = (N+1)!$ , 即  $|xy^2z|$  不是阶乘数.

注意: 对于有限的语言, 比如  $\emptyset, \{00, 11\}, \{0^n 1^n \mid 0 \leq n \leq 100\}$  泵引理如何解释.

### 4.1.3 泵引理只是必要条件

“正则语言的泵引理”是正则语言的必要条件, 即“正则  $\Rightarrow$  泵引理成立”, 所以“ $\neg$  泵引理成立  $\Rightarrow \neg$  正则”. 但是否有与正则语言等价的条件呢? 有, 即“Myhill-Nerode Theorem”.

### 示例

下面的语言, 每个串都可以应用泵引理, 却不是正则的

$$\{\$a^n b^n \mid n \geq 1\} \cup \{\$^k w \mid k \neq 1, w \in \{a, b\}^*\}$$

后一部分是正则的, 因此可以应用泵引理. 而前一部分不是正则的, 但每个串都可以利用  $\$$  符号泵到后一部分中.

## 4.2 正则语言的封闭性

如果语言类中的语言, 在某些特定的运算下在该语言类中保持封闭, 称为这个语言类的封闭性 (*closure property*). 正则语言的封闭性: 正则语言类中, 从其中的语言经过某些运算, 得到某个语言  $L$  还是正则的.

### 4.2.1 布尔运算下的封闭性

**定理 2.** 正则语言在并, 连接和克林闭包运算下保持封闭.

证明. 由正则表达式的定义得证. □

**定理 3.** 正则语言在补运算下封闭. 即: 如果  $L$  是字母表  $\Sigma$  上的正则语言, 即  $L \subseteq \Sigma^*$ , 则  $\bar{L} = \Sigma^* - L$  也是正则的.

证明. 设 DFA  $A = (Q, \Sigma, \delta, q_0, F)$  识别  $L$ , 即  $L = \mathbf{L}(A)$ . (注意  $A$  对不接受的输入要有 *dead state*.) 那么构造 DFA  $B = (Q, \Sigma, \delta, q_0, Q - F)$ , 则  $\bar{L} = \mathbf{L}(B)$ . 因为任何  $w \notin L$ ,  $\hat{\delta}(q_0, w) \notin F$ . □

**问题** 如何求正则表达式的补?

### 示例

证明  $L_{neq} = \{w \mid w \text{ 由数量不相等的 } 0 \text{ 和 } 1 \text{ 构成}\}$  不是正则的.

由泵引理很难直接证明  $L_{neq}$  不是正则的, 无论如何取  $w$ , 都无法将其打断为  $w = xyz$  形式, 并利用  $y$  产生不属于  $L_{neq}$  的串.

而  $L_{neq} = \overline{L_{eq}}$ , 而  $L_{eq}$  不是正则的很容易证明, 所以  $L_{neq}$  不是正则的.

**定理 4** (习题 4.2.15). 若 DFA  $A_L$ ,  $A_M$  和  $A$  的定义如下

$$\begin{aligned} A_L &= (Q_L, \Sigma, \delta_L, q_L, F_L) \\ A_M &= (Q_M, \Sigma, \delta_M, q_M, F_M) \\ A &= (Q_L \times Q_M, \Sigma, \delta, (q_L, q_M), F_L \times F_M) \end{aligned}$$

其中

$$\begin{aligned} \delta : (Q_L \times Q_M) \times \Sigma &\rightarrow Q_L \times Q_M \\ \delta((p, q), a) &= (\delta_L(p, a), \delta_M(q, a)). \end{aligned}$$

则对任意  $w \in \Sigma^*$ ,

$$\hat{\delta}((q_L, q_M), w) = (\hat{\delta}(q_L, w), \hat{\delta}(q_M, w)).$$

归纳法

证明. 通过对  $w$  的归纳来证明.

(1) 归纳基础: 当  $w = \varepsilon$  时

$$\begin{aligned} \hat{\delta}((q_L, q_M), \varepsilon) &= (q_L, q_M) && \hat{\delta} \text{ 的定义} \\ &= (\hat{\delta}_L(q_L, \varepsilon), \hat{\delta}_M(q_M, \varepsilon)) && \hat{\delta}_L \text{ 和 } \hat{\delta}_M \text{ 的定义} \end{aligned}$$

(2) 归纳递推: 当  $w = xa$  时

$$\begin{aligned} \hat{\delta}((q_L, q_M), xa) &= \delta(\hat{\delta}((q_L, q_M), x), a) && \hat{\delta} \text{ 的定义} \\ &= \delta((\hat{\delta}_L(q_L, x), \hat{\delta}_M(q_M, x)), a) && \text{归纳假设} \\ &= (\delta_L(\hat{\delta}_L(q_L, x), a), \delta_M(\hat{\delta}_M(q_M, x), a)) && \delta \text{ 的定义} \\ &= (\hat{\delta}_L(q_L, xa), \hat{\delta}_M(q_M, xa)) && \hat{\delta}_L \text{ 和 } \hat{\delta}_M \text{ 的定义} \end{aligned}$$

得证. □

**定理 5.** 正则语言在交运算下封闭.

证明 1. 由  $L \cap M = \overline{\overline{L} \cup \overline{M}}$  得证. □

证明 2. 如果  $L$  和  $M$  是任意两个正则语言, 则构造识别  $L \cap M$  的 DFA 如上例所示, 则只需证明  $L(A) = L \cap M$ . 实际上, 对  $\forall w \in \Sigma^*$  有

$$\begin{aligned} w \in L \cap M &\iff w \in L \text{ 且 } w \in M \\ &\iff \hat{\delta}_L(q_L, w) \in F_L \text{ 且 } \hat{\delta}_M(q_M, w) \in F_M \\ &\iff (\hat{\delta}_L(q_L, w), \hat{\delta}_M(q_M, w)) \in F_L \times F_M \\ &\iff \hat{\delta}((q_L, q_M), w) \in F_L \times F_M \\ &\iff w \in L(A) \end{aligned}$$

因此  $L(B) = L \cap M$ . □

**示例**

已知  $L_{01}$  不是正则语言, 请证明  $L_{eq}$  不是正则的.

证明. 因为  $L_{01} = \mathbf{L}(0^*1^*) \cap L_{eq}$ , 而  $0^*1^*$  是正则的且  $L_{01}$  不是正则的, 则可知  $L_{eq}$  不是正则的.  $\square$

**问题** 为什么又可以用  $L_{eq}$  子集  $L_{01}$  的非正则, 来说明  $L_{eq}$  的非正则了?

**定理 6.** 正则语言在差运算下封闭. 如果  $L$  和  $M$  是正则语言, 那么  $L - M$  也是正则的.

证明. 因为  $L - M = L \cap \overline{M}$ , 得证.  $\square$

**示例**

Prove or disprove that if  $L_1$  and  $L_2$  are not regular, then  $L_1 \cap L_2$  is not regular.

证明. 命题错误. 因为, 若令  $L_1 = \{0^n1^n \mid n \geq 0\}$  且  $L_2 = \{a^n b^n \mid n \geq 0\}$ , 两者都不是正则语言, 但

$$L_1 \cap L_2 = \{\varepsilon\}$$

显然是正则语言.  $\square$

## 4.2.2 反转 (Reverse)

如果串  $w = a_1 a_2 \cdots a_n$ , 称  $a_n a_{n-1} \cdots a_1$  为  $w$  的反转 (reverse), 用  $w^R$  表示. 例如  $00110^R = 01100$ ,  $\varepsilon^R = \varepsilon$ . 如果  $L$  是一个语言, 则定义  $L^R = \{w^R \mid w \in L\}$ .

**定理 7.** 如果  $L$  是正则的, 那么  $L^R$  也是正则的.

证明的过程使用对正则表达式求反转, 规则为:

$$\begin{aligned} \varepsilon^R &= \varepsilon & (r + s)^R &= r^R + s^R \\ \emptyset^R &= \emptyset & (rs)^R &= s^R r^R \\ a^R &= a & (r^*)^R &= (r^R)^* \end{aligned}$$

下面举例说明规则的含义.

**示例**

语言

$$L = \{w \in \{0, 1\}^* \mid w \text{ ends in } 01.\}$$

正则表达式为  $(0 + 1)^*01$ . 而语言  $L$  的反转

$$L^R = \{w \in \{0, 1\}^* \mid w \text{ starts with } 10.\}$$

的正则表达式  $10(0 + 1)^*$  可以通过如下方式得到

$$\begin{aligned} ((0 + 1)^*01)^R &= ((\textcolor{red}{0} + \textcolor{red}{1})^*\textcolor{blue}{0}\textcolor{blue}{1})^R = (\textcolor{blue}{0}\textcolor{blue}{1})^R((\textcolor{red}{0} + \textcolor{red}{1})^*)^R \\ &= \textcolor{blue}{1}\textcolor{blue}{0}^R((\textcolor{red}{0} + \textcolor{red}{1})^*)^R = 10((\textcolor{red}{0} + \textcolor{red}{1})^R)^* \\ &= 10(\textcolor{red}{0}^R + \textcolor{red}{1}^R)^* = 10(0 + 1)^* \end{aligned}$$

**证明 1.** (使用正则表达式) 我们证明命题 “若有正则表达式  $E$ , 且  $L = \mathbf{L}(E)$ , 则存在正则表达式  $E^R$  使  $(\mathbf{L}(E))^R = \mathbf{L}(E^R)$ ”.

由正则表达式的定义, 对  $E$  的结构进行归纳:

首先, 如果  $E$  分别是  $\varepsilon$ ,  $\emptyset$  或  $a$ , 则分别取其反转, 有  $\varepsilon^R = \varepsilon$ ,  $\emptyset^R = \emptyset$  和  $a^R = a$ , 所以分别都存在  $E^R$  且  $(\mathbf{L}(E))^R = \mathbf{L}(E^R)$ .

其次, 对  $E$  所有可能三种递归的结构, 有:

(1) 如果  $E = E_1 + E_2$ :

$$\begin{aligned}
 (L(E))^R &= (L(E_1 + E_2))^R \\
 &= (L(E_1) \cup L(E_2))^R && \text{正则表达式的加“加”} \\
 &= \{w^R \mid w \in L(E_1) \text{ or } w \in L(E_2)\} && \text{语言的反转} \\
 &= \{w^R \mid w \in L(E_1)\} \cup \{w^R \mid w \in L(E_2)\} && \text{显然} \\
 &= (L(E_1))^R \cup (L(E_2))^R && \text{语言的反转} \\
 &= L(E_1^R) \cup L(E_2^R) && \text{归纳假设} \\
 &= L(E_1^R + E_2^R) && \text{正则表达式的“加”} \\
 &= L(E^R)
 \end{aligned}$$

(2) 如果  $E = E_1 E_2$ :

$$\begin{aligned}
 (L(E))^R &= (L(E_1 E_2))^R \\
 &= (L(E_1) L(E_2))^R && \text{正则表达式的连接} \\
 &= \{w_1 w_2 \mid w_1 \in L(E_1), w_2 \in L(E_2)\}^R && \text{语言的连接} \\
 &= \{(w_1 w_2)^R \mid w_1 \in L(E_1), w_2 \in L(E_2)\} && \text{语言的反转} \\
 &= \{w_2^R w_1^R \mid w_1 \in L(E_1), w_2 \in L(E_2)\} && \text{字符串的反转} \\
 &= \{w_2^R \mid w_2 \in L(E_2)\} \{w_1^R \mid w_1 \in L(E_1)\} && \text{语言的连接} \\
 &= (L(E_2))^R (L(E_1))^R && \text{语言的反转} \\
 &= L(E_2^R) L(E_1^R) && \text{归纳假设} \\
 &= L(E_2^R E_1^R) && \text{正则表达式的连接} \\
 &= L(E^R)
 \end{aligned}$$

(3) 如果  $E = E_1^*$ :

$$\begin{aligned}
 (L(E))^R &= (L(E_1^*))^R \\
 &= \{w_1 w_2 \dots w_n \mid n \geq 0, w_i \in L(E_1)\}^R && \text{正则表达式的闭包} \\
 &= \{(w_1 w_2 \dots w_n)^R \mid n \geq 0, w_i \in L(E_1)\} && \text{语言的反转} \\
 &= \{w_n^R w_{n-1}^R \dots w_1^R \mid n \geq 0, w_i \in L(E_1)\} && \text{字符串的反转} \\
 &= \{w_n^R w_{n-1}^R \dots w_1^R \mid n \geq 0, w_i^R \in L(E_1^R)\} && \text{归纳假设} \\
 &= \{w_1 w_2 \dots w_n \mid n \geq 0, w_i \in L(E_1^R)\} && \text{显然} \\
 &= L((E_1^R)^*) && \text{正则表达式的闭包} \\
 &= L(E^R)
 \end{aligned}$$

因此, 命题成立, 因此  $L^R$  也是正则语言. □

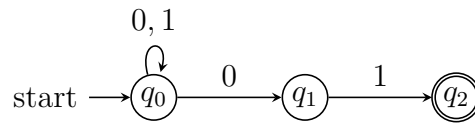
证明 2. (使用有穷自动机) 设识别正则语言  $L$  的自动机为  $A$ , 如下构造识别  $L^R$  的自动机:

- (1) 将全部的边, 调转方向.
- (2) 将初始状态  $q_0$ , 改为唯一接受状态.
- (3) 增加新初始状态  $p_0$ , 且  $\delta(p_0, \varepsilon) = F$ .

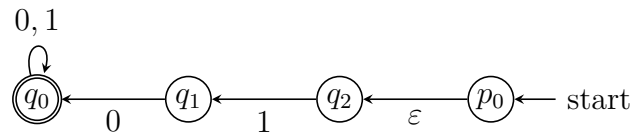
□

### 示例

续上例, 识别  $L = \{w \in \{0, 1\}^* \mid w \text{ ends in } 01.\}$  的 NFA 为

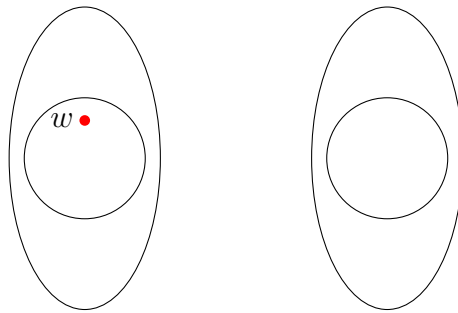


则识别其补语言  $L^R = \{w \in \{0, 1\}^* \mid w \text{ starts with } 10.\}$  的 NFA 为



### 4.2.3 同态 (Homomorphism)

设  $\Sigma$  和  $\Gamma$  为两个字母表, 首先, 定义同态为函数  $h: \Sigma \mapsto \Gamma^*$ , 即每个字符  $a \in \Sigma$ , 在  $h$  的作用下, 替换为  $\Gamma$  上的一个串. 其次, 扩展同态函数为  $h: \Sigma^* \mapsto \Gamma^*$ , 如果  $\Sigma^*$  中的串  $w = a_1 a_2 \cdots a_n$ , 则  $h(w) = h(a_1)h(a_2) \cdots h(a_n)$ . 再扩展  $h$  到语言, 若  $L$  是字母表  $\Sigma$  上一个语言, 则  $h(L) = \{h(w) \mid w \in L\}$ .



### 示例

设  $\Sigma = \{0, 1\}$ ,  $\Gamma = \{a, b\}$ , 若同态函数为  $h(0) = ab$ ,  $h(1) = \varepsilon$ , 则串 0011 在  $h$  的作用下  $h(0011) = h(0)h(0)h(1)h(1) = abab\varepsilon\varepsilon = abab$ .

对于正则表达式, 则定义  $h(E)$  为, 将  $E$  中的每个符号替换后得到的表达式, 即:

$$\begin{aligned}
 h(\emptyset) &= \emptyset & h(r + s) &= h(r) + h(s) \\
 h(\varepsilon) &= \varepsilon & h(rs) &= h(r)h(s) \\
 h(a) &= h(a) & h(r^*) &= (h(r))^*
 \end{aligned}$$

### 示例

续上例, 语言  $L = 1^*0 + 0^*1$ , 在上例的同态  $h$  的作用下, 那么  $h(L)$  为:

$$\begin{aligned}
 h(1^*0 + 0^*1) &= (h(1))^*h(0) + (h(0))^*h(1) \\
 &= (\varepsilon)^*(ab) + (ab)^*(\varepsilon) \\
 &= (ab)^*
 \end{aligned}$$

定理 8. 如果  $L$  是字母表  $\Sigma$  上的正则语言,  $h$  是  $\Sigma$  上的一个同态, 则  $h(L)$  也是正则的.

证明. 设  $E$  是正则表达式, 往证:  $h(\mathbf{L}(E)) = \mathbf{L}(h(E))$ .

对  $E$  的结构进行归纳, 首先,  $\mathbf{L}(\varepsilon) = \varepsilon$ ,  $\mathbf{L}(\emptyset) = \emptyset$ , 以及若  $E = a$ , ( $a \in \Sigma$ ), 则  $h(\mathbf{L}(E)) = h(\mathbf{L}(a)) = h(\{a\}) = \{h(a)\} = \mathbf{L}(h(E))$ , 所以对  $\varepsilon$ ,  $\emptyset$  和  $\forall a \in \Sigma$  命题成立.

对  $E$  可能的三种递归结构, 分别有:

(1) 如果  $E = F + G$ :

$$\begin{aligned}
 h(\mathbf{L}(E)) &= h(\mathbf{L}(F + G)) \\
 &= h(\mathbf{L}(F) \cup \mathbf{L}(G)) && \text{正则表达式的“加”} \\
 &= h(\mathbf{L}(F)) \cup h(\mathbf{L}(G)) && h \text{ 作用在每个集合的串上} \\
 &= \mathbf{L}(h(F)) \cup \mathbf{L}(h(G)) && \text{归纳假设} \\
 &= \mathbf{L}(h(F) + h(G)) && \text{正则表达式的“加”} \\
 &= \mathbf{L}(h(F + G)) && h(F + G) \text{ 的定义} \\
 &= \mathbf{L}(h(E))
 \end{aligned}$$

(2) 如果  $E = FG$ :

$$\begin{aligned}
 h(\mathbf{L}(E)) &= h(\mathbf{L}(F)\mathbf{L}(G)) && \text{正则表达式的连接} \\
 &= h(\mathbf{L}(F))h(\mathbf{L}(G)) && \heartsuit \\
 &= \mathbf{L}(h(F))\mathbf{L}(h(G)) && \text{归纳假设} \\
 &= \mathbf{L}(h(F)h(G)) && \text{正则表达式的连接} \\
 &= \mathbf{L}(h(FG)) && h(FG) \text{ 的定义} \\
 &= \mathbf{L}(h(E))
 \end{aligned}$$

$$\heartsuit: h(a_1 \cdots a_n b_1 \cdots b_m) = h(a_1) \cdots h(b_m) = h(a_1 \cdots a_n)h(b_1 \cdots b_m)$$

(3) 如果  $E = F^*$

略. (提示: 任意  $w \in F^*$  可以看做  $w = w_1 w_2 \cdots w_n$ , 其中  $w_i \in F$ .)

□

## 示例

Prove that  $L = \{0^i 1^j 2^k \mid i, j, k \in \mathbb{Z}^+ \text{ and } i \neq j \neq k\}$  is a language not regular.

证明. 定义同态函数

$$h(0) = 0, \quad h(1) = 1, \quad h(2) = \varepsilon.$$

同态  $h$  会将  $2^k$  去掉, 因此

$$h(L) = \{0^i 1^j \mid i, j \in \mathbb{Z}^+ \text{ and } i \neq j\} = L',$$

再证明  $L'$  非正则即可。

□

## 4.2.4 逆同态 (Inverse homomorphism)

若  $h$  是字母表  $\Sigma$  到字母表  $\Gamma$  的同态, 并且  $L$  是  $\Gamma$  上的一个语言, 那么使  $h(w) \in L$  的  $w$  ( $w \in \Sigma^*$ ) 的集合, 称为语言  $L$  的  $h$  逆, 记为  $h^{-1}(L)$ , 即

$$h^{-1}(L) = \{w \mid h(w) \in L\}$$



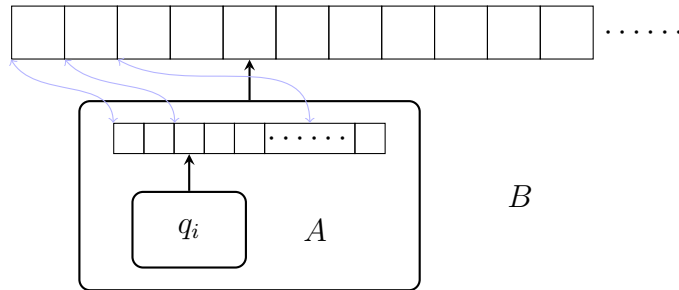
**定理 9.** 如果  $h$  是字母表  $\Sigma$  到字母表  $\Gamma$  的同态,  $L$  是  $\Gamma$  上的正则语言, 那么  $h^{-1}(L)$  也是正则语言.

证明. 设接受  $L$  语言的 DFA  $A = (Q, \Gamma, \delta, q_0, F)$ , 构造识别  $h^{-1}(L)$  的 DFA

$$B = (Q', \Sigma, \delta', q_0, F),$$

其中

$$\delta'(q, a) = \hat{\delta}(q, h(a)).$$



往证  $\hat{\delta}'(q, w) = \hat{\delta}(q, h(w))$ . 对  $|w|$  归纳, 归纳基础:

$$\hat{\delta}'(q, \varepsilon) = q = \hat{\delta}(q, h(\varepsilon)) = \hat{\delta}(q, \varepsilon)$$

归纳递推: 若  $w = xa$ , 则

$$\begin{aligned} \hat{\delta}'(q, xa) &= \delta'(\hat{\delta}'(q, x), a) && \hat{\delta}' \text{ 定义} \\ &= \delta'(\hat{\delta}(q, h(x)), a) && \text{归纳假设} \\ &= \hat{\delta}(\hat{\delta}(q, h(x)), h(a)) && \delta' \text{ 构造} \\ &= \hat{\delta}(q, h(x)h(a)) && \text{参考教材习题 2.2.2} \\ &= \hat{\delta}(q, h(xa)) && h \text{ 在字符串上的扩展} \\ &= \hat{\delta}(q, h(w)) \end{aligned}$$

所以  $\forall w \in \Sigma^*$ ,  $\hat{\delta}'(q_0, w) = \hat{\delta}(q_0, h(w)) \in F$ , 即  $w$  被  $B$  接受当且仅当  $h(w)$  被  $A$  接受, 即  $B$  是识别  $h^{-1}(L)$  的 DFA, 因此  $h^{-1}(L)$  是正则的.  $\square$

### 示例

设语言  $L = \mathbf{L}((00 + 1)^*)$ , 定义同态  $h : \{a, b\} \rightarrow \{0, 1\}^*$  为  $h(a) = 01, h(b) = 10$ , 则请证明  $h^{-1}(L) = \mathbf{L}((ba)^*)$ .

证明. 往证  $h(w) \in L \iff w = (ba)^n$ .

( $\Leftarrow$ ) 若  $w = (ba)^n$  ( $n \geq 0$ ), 因为  $h(ba) = 1001$ , 因此  $h(w) = (1001)^n \in L$ .

( $\Rightarrow$ ) 若  $h(w) \in L$ , 假设  $w \notin (ba)^*$ , 则只能有四种情况, 且都会得出  $h(w) \notin L$ :

- (1)  $w$  以  $a$  开头, 则  $h(w)$  以  $01$  开头, 显然  $h(w) \notin (00 + 1)^*$ ;
- (2)  $w$  以  $b$  结尾, 则  $h(w)$  以  $10$  结尾, 显然  $h(w) \notin (00 + 1)^*$ ;
- (3)  $w$  有连续的  $a$ , 即  $w = xaay$ , 则  $h(w) = z1010v$ , 则显然  $h(w) \notin (00 + 1)^*$ ;
- (4)  $w$  有连续的  $b$ , 即  $w = xbbv$ , 则  $h(w) = z0101v$ , 则显然  $h(w) \notin (00 + 1)^*$ ;

因此  $w$  只能是  $(ba)^n$  的形式.  $\square$

## 4.3 正则语言的判定性质

正则 (或任何) 语言, 典型的 3 个判定问题:

- (1) 所描述的语言是否为空?
- (2) 某个特定的串  $w$  是否属于所描述的语言?
- (3) 语言的两种描述, 是否实际上描述的是同一语言? (即语言的等价性)

要回答这样的问题, 我们想知道, 具体的算法, 是否存在.

### 4.3.1 空性, 有穷性和无穷性 (Emptiness, finiteness and infiniteness)

正则语言的空, 有穷和无穷, 可以通过如下定理来判定.

**定理 10.** 具有  $n$  个状态的有穷自动机  $M$  接受的串的集合  $S$ :

- (1)  $S$  是非空的, 当且仅当  $M$  接受某个长度小于  $n$  的串;
- (2)  $S$  是无穷的, 当且仅当  $M$  接受某个长度为  $m$  的串,  $n \leq m < 2n$ .

证明. (1) ( $\Rightarrow$ ) 如果  $S$  非空, 则  $M$  接受某个串,  $w$  是  $M$  接受的串中长度最小的串之一, 那么一定有  $|w| < n$ , 因为如果  $|w| \geq n$ , 由泵引理  $w = xyz$ , 那么  $xz$  是  $M$  接受的一个更短的串. ( $\Leftarrow$ ) 显然.

(2) ( $\Leftarrow$ ) 如果  $w \in L(M)$  且  $n \leq |w| < 2n$ , 由泵引理,  $S$  是无穷的. ( $\Rightarrow$ , 反证法) 如果  $S$  是无穷的, 假设没有任何一个串, 长度是  $n$  到  $2n-1$ ; 那么假设  $w$  是长度  $\geq 2n$  中最短的串之一, 由泵引理  $w = xyz$ ,  $1 \leq |y| < n$ , 则  $xz \in L(M)$ ; 于是, 或者  $w$  不是长度  $2n$  及以上最短的, 或者  $xz$  长度在  $n$  到  $2n-1$  之间. 两种情况之下, 都有矛盾.  $\square$

定理的第 (1) 部分, 说明存在一个算法, 判断  $L(M)$  是否为空: 只需要检查长度小于  $n$  的串, 是否在  $L(M)$  中. (当然, 更高效的算法: 从 DFA 开始状态寻找到接受状态的路径.) 第 (2) 部分说明, 存在一个算法, 判断  $L(M)$  是否为无穷: 只需要检查长度在  $n$  到  $2n-1$  的串, 是否在  $L(M)$  中.

### 4.3.2 等价性

**定理 11.** 存在一个算法, 判定两个有穷自动机是否等价 (是否接受同一语言).

证明. 设  $M_1$  和  $M_2$  是分别接受  $L_1$  和  $L_2$  的有穷自动机, 则  $(L_1 \cap \overline{L_2}) \cup (\overline{L_1} \cap L_2)$  可以被某个有穷自动机接受  $M_3$  接受; 如果  $M_3$  接受某个串, 当且仅当  $L_1 \neq L_2$ , 由于存在算法判断  $M_3$  是否空, 因此得证.  $\square$

## 4.4 自动机最小化

### 4.4.1 状态的等价性

DFA 中, 称两个状态  $p$  和  $q$  是等价的, 如果对任意串  $w \in \Sigma^*$  满足:

$$\hat{\delta}(p, w) \in F \Leftrightarrow \hat{\delta}(q, w) \in F$$

即, 只要对任何串  $w$ ,  $\hat{\delta}(p, w)$  和  $\hat{\delta}(q, w)$  只需同时在  $F$  中或同时不在  $F$  中. 如果两个状态不等价, 则称为可区分的.

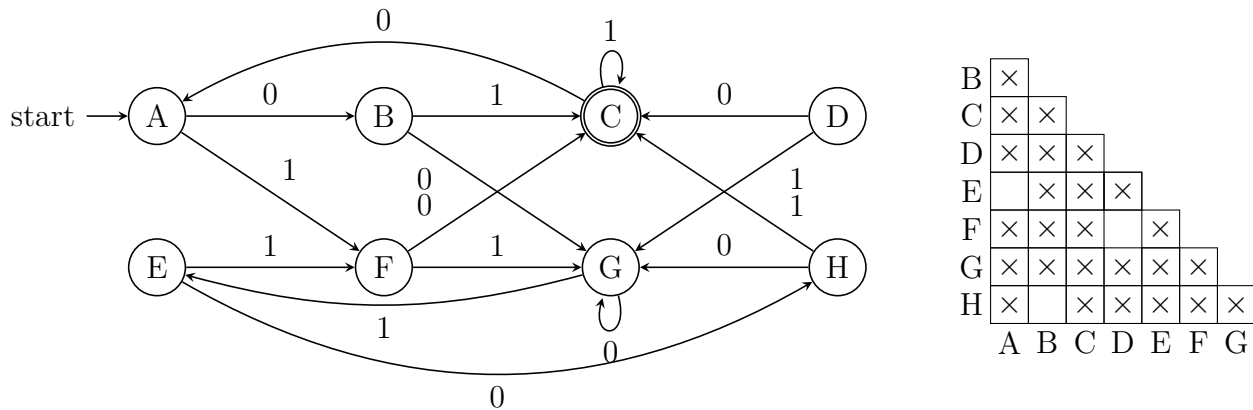
### 4.4.2 填表算法

填表算法递归的发现 DFA 中全部的可区分状态对:

基础: 如果  $p$  是接受状态而  $q$  是非接受状态, 则  $[p, q]$  对是可区分的;

归纳: 如果某个  $a \in \Sigma$ , 有  $[r = \delta(p, a), s = \delta(q, a)]$  是可区分的, 则  $[p, q]$  是可区分的.

示例



由于只有 C 是接受状态, 所以  $\{C\} \times \{A, B, D, E, F, G, H\} = \{[C, A], [C, B], [C, D], [C, E], [C, F], [C, G], [C, H]\}$  都是可区分的; 状态 B, H 通过字符 1 到接受状态, 而 A, D, E, F, G 通过字符 1 都到不接受状态, 因此  $\{B, H\} \times \{A, D, E, F, G\}$  都是可区分的; 类似的, 对字符 0, 有  $\{D, F\} \times \{A, B, E, G, H\}$  都是可区分的; 其余的再逐个检查即可.

**定理 12.** 如果不能通过填表算法区分两个状态, 则这两个状态是等价的.

### 4.4.3 DFA 最小化

根据填表算法取得的 DFA A 状态间等价性, 将状态集进行划分, 得到不同的块; 利用块构造新的 DFA B, B 的开始状态的为包含 A 初始状态的块, B 的接受状态为包含 A 的接收状态的块, 转移函数为块之间的转移; 则 B 是 A 的最小化 DFA.

注意: 不能使用同样的方法最小化 NFA.

示例

续前例, 将 DFA 状态分块后合并.

