第六讲

✧ 正规语言的性质与运算

正规语言的性质与运算

- 针对正规语言的 *Pumping* 引理
- 有关正规语言的几个判定性质
- 关于正规语言的封闭运算

针对正规语言的 Pumping 引理

- 正规语言应满足的一个必要条件
- 可用于判定某些语言不是正规语言

针对正规语言的 Pumping 引理

✧ DFA 的 "Pumping" 特性

设 DFA $D = (Q, \Sigma, \delta, q_0, F)$, $|Q|=n$.

对于任一长度不小于 n 的字符串 $w = a_1a_2\ldots a_m$, 其中 $m \geq n$, $a_k \in \Sigma$ $(1 \leq k \leq m)$, $q \in Q$, 考察如下状态序列

$p_0=q$

$p_1=\delta^*(q, a_1)$

$p_2=\delta^*(q, a_1a_2)$

…

$p_n=\delta^*(q, a_1a_2\ldots a_n)$

$p_{n+1}=\delta^*(q, a_1a_2\ldots a_{n+1})$

…

$p_m=\delta^*(q, a_1a_2\ldots a_m)$

由Pigeonhole 原理, $p_0, p_1, p_2, \ldots, p_n$ 中至少有两个状态是重复的, 即存在 i, j, $0 \leq i < j \leq n$, $p_i=p_j$.

✧ "pumping" 特性:
任一长度不小于状态数目的字符串所标记的路径上, 必然出现重复的状态.

针对正规语言的 Pumping 引理

✧ DFA 的 “Pumping” 特性

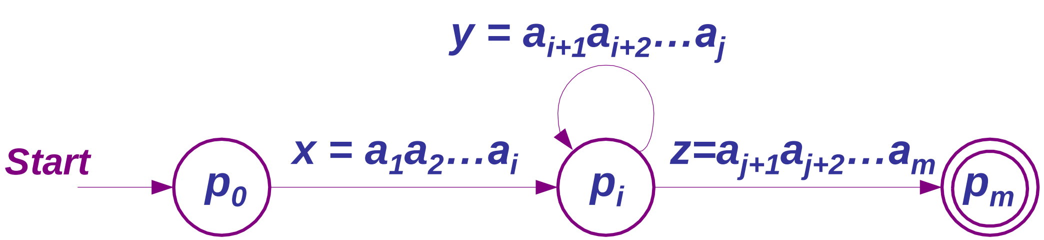
– “pumping” 特性：如前,设 DFA $D = (Q, \Sigma, \delta, q_0, F)$, $|Q|=n$, $w = a_1a_2\ldots a_m$ $(m \geq n)$, 则存在 i, j, $0 \leq i < j \leq n$, $p_i = p_j$, 其中 $p_k = \delta'(p_0, a_1a_2\ldots a_k)$, $0 \leq k \leq m$.

– 若假定 $p_0 = q_0$, $p_m \in F$, 即 $w \in L(D)$.

令 $w = xyz$, 其中：

$$x = a_1a_2\ldots a_i,\ y = a_{i+1}a_{i+2}\ldots a_j,\ z = a_{j+1}a_{j+2}\ldots a_m$$

则对任何 $k \geq 0$，都有 $xy^kz \in L(D)$. (参考下图)



针对正规语言的 Pumping 引理

✧ Pumping Lemma for Regular Language

设 L 是正规语言，则存在常数 $n \geq 1$，使得任一长度不小于 n 的字符串 $w \in L$，$|w| \geq n$，都可以分成三个部分，即 $w = xyz$，且满足下列条件：

1. $y \neq \varepsilon$.
2. $|xy| \leq n$.
3. 对任何 $k \geq 0$，都有 $xy^k z \in L$.

证明 设 L 是 DFA $D = (Q, \Sigma, \delta, q_0, F)$ 的语言.
取 $n = |Q|$ 即可.

针对正规语言的 Pumping 引理

✧ Pumping 引理的一个应用

– 用于证明某个语言 *L* 不是正规语言

Pumping 引理的条件可形式表示为:

$\exists n\, \forall w \exists x \exists y \exists z\, \forall k(n\geq 1 \wedge (w \in L \wedge |w|\geq n \rightarrow w=xyz \wedge y\neq \varepsilon \wedge |xy|\leq n \wedge (k \geq 0 \rightarrow xy^k z \in L)))$

该命题的否定形式为:

$\forall n \exists w\, \forall x\, \forall y\, \forall z \exists k(n\geq 1 \rightarrow (w \in L \wedge |w|\geq n \wedge (w=xyz \wedge y\neq \varepsilon \wedge |xy|\leq n \rightarrow k \geq 0 \wedge xy^k z \notin L)))$

– 证明步骤

1. 考虑任意的 $n\geq 1$.
2. 找到一个满足以下条件的串 $w\in L$ (长度至少为 n).
3. 任选满足 $w=xyz \wedge y\neq\varepsilon \wedge |xy|\leq n$ 的 x,y,z
4. 找到一个 $k \geq 0$, 使 $xy^k z \notin L$.

针对正规语言的 Pumping 引理

✧ Pumping 引理的一个应用

- 用于证明某个语言 L 不是正规语言
- 证明步骤
  1. 考虑任意的$n \geq 1$.
  2. 找到一个满足以下条件的串$w \in L$ (长度至少为n).
  3. 任选满足$w=xyz \wedge y \neq \varepsilon \wedge |xy| \leq n$ 的x,y,z
  4. 找到一个 $k \geq 0$, 使 $xy^kz \notin L$.
- 举例 证明语言 $L_{01} = \{ 0^k1^k \mid k \geq 0 \}$ 不是正规语言

  证明考虑任意的$n \geq 1$.

  取$w = 0^n1^n$.

  任选满足条件$w=xyz \wedge y \neq \varepsilon \wedge |xy| \leq n$ 的x,y,z

  若取 $k=0$, 则有 $xy^kz = xz \notin L_{01}$ ($\because$ xz 中0 比1 少).

针对正规语言的 Pumping 引理

- ✧ Pumping 引理不是正规语言的充分条件
  - 反例　**a, b, c** 串构成的语言

    $$L = \{a^i b^j c^k \mid i,j,k \geq 0, \text{若 } i=1 \text{ 则 } j=k\}$$

有关正规语言的几个判定性质

- 基本判定性质（*Decision Properties*）
  - 判定正规语言是否为空
  - 判定正规语言中是否包含特定的字符串
  - 判定两个正规语言是否相等

判定正规语言是否为空

✧ 以有限自动机表示正规语言

– **判定算法** 测试从初态是否可达某一终态. 先求所有可达状态的集合，若其中包含终态，则该正规语言非空， 否则为空语言。可由如下步骤递归地计算可达状态集合:

基础： 初态是可达的;

归纳： 设状态 q 是可达的，若对于某个输入符号或 ε，q 可转移到 p ，则 p 也是可达的;

– **算法复杂度** 设有限自动机的状态数目为 n，上述判定算法的复杂度为 $O(n^2)$.

判定正规语言是否为空

✧ 以正规表达式表示正规语言

- 判定算法 可由如下步骤归纳出正规表达式表示的语言是否为空：

  基础：$L(\phi)$ 为空语言，而 $L(\varepsilon)$ 和 $L(a)$ 不是；

  归纳：

  1. 设 $R=R_1+R_2$，$L(R)$ 为空 *iff* $L(R_1)$ 和 $L(R_2)$ 都为空；
  2. 设 $R=R_1R_2$，$L(R)$ 为空 *iff* $L(R_1)$ 或 $L(R_2)$ 为空；
  3. 设 $R=R_1{}^*$，$L(R)$ 非空（至少包含 ε）；
  4. 设 $R=(R_1)$，$L(R)$为空 *iff* $L(R_1)$ 为空.

- 算法复杂度 设正规表达式包含的符号数目为 n，上述判定算法的复杂度为 $O(n)$；

判定是否包含特定的字符串

- 以 DFA 表示正规语言
  - 判定算法 从初态开始，处理输入字符串 *w* ，如果可以结束于某一终态，则该正规语言中包含 *w*，否则不包含 *w* 。
  - 算法复杂度 设输入字符串 *w* 的长度 $|w|=n$，上述判定算法的复杂度为 $O(n)$.
- 以 NFA （或ε–NFA ）表示正规语言 可以将其转化为等价的 DFA，再执行上述过程；也可以直接模拟其处理字符串的过程，判定算法的复杂度为 $O(ns^2)$, 其中 n 为字符串的长度，s 为NFA （或ε–NFA ）的状态数目.
- 以正规表达式表示正规语言 将其转化为等价的 ε–NFA ，然后执行上述过程.

判定两个正规语言是否相等

✧ **判定算法** 可以采取如下步骤：

1. 先将两个正规语言的表达形式都转化为 DFA，问题转化为两个DFA是否是等价的；
2. 适当重命名，使两个DFA没有重名的状态；
3. 将两个DFA相并，构造一个新的DFA，原来的终态仍是终态，转移边不发生任何变化，取任何一个状态为初态；
4. 对新构造的DFA运用填表算法，如果原来DFA的两个初态不可区别，则这两个正规语言相等，否则不相等.

✧ **算法复杂度** 以上算法的复杂度即填表算法的复杂度，其上限为$O(n^4)$；可以适当设计填表算法的数据结构，使其复杂度降为 $O(n^2)$.

关于正规语言的封闭运算

✧ 关于正规语言的几个主要的封闭运算

- 并（*union*）
- 补（*complement*）
- 交（*intersection*）
- 差（*difference*）
- 反向（*reversal*）
- （星）闭包（*closure(star)*）
- 连接（*concatenation*）
- 同态（*homomorphism*）
- 反同态（*inverse homomorphism* ）

关于正规语言的封闭运算

- 正规语言的并（*union*）
  - 结论 若 L 和 M 为正规语言，则 $L \cup M$ 也是正规语言.
  - 证明 因为 L 和 M 为正规语言，所以存在正规表达式 R 和 S，使得 $L(R)=L$, $L(S)=M$.

    由正规表达式的定义，有

    $$L(R+S) = L(R) \cup L(S) = L \cup M$$

    所以，$L \cup M$ 为正规语言.

关于正规语言的封闭运算

✧ 正规语言的（星）闭包（*closure(star)*）

– **结论** 若 L 为字母表 Σ 上的正规语言，则 L^* 也是正规语言.

– **证明** 因为 L 为正规语言，所以存在正规表达式 R，使得 $L(R)=L$.

由正规表达式的定义，$L(R^*) = (L(R))^* = L^*$.

所以，L^*为正规语言.

关于正规语言的封闭运算

✧ 正规语言的连接（*concatenation*）

- **结论** 若 ***L*** 和 ***M*** 为正规语言，则 ***LM*** 也是正规语言
- **证明** 因为 ***L*** 和 ***M*** 为正规语言，所以存在正规表达式 ***R*** 和 ***S***，使得 ***L(R)=L, L(S)=M.***

  由正规表达式的定义，***L(RS) = L(R)L(S) =LM.***

  所以，***LM*** 为正规语言.

关于正规语言的封闭运算

✧ 正规语言的补（*complement*）

– 结论 若 L 为Σ 上的正规语言，则 $\overline{L} = \Sigma^* - L$ 也是正规语言.

– 证明 因为 L 为正规语言，所以存在

$DFA\ \ A = (Q, \Sigma, \delta, q_0, F)$，使得 $L(A)=L$.

现构造 $DFA\ \ B = (Q, \Sigma, \delta, q_0, Q - F)$，则有

$w \in L(B)$ iff $\delta'(q_0, w) \in Q - F$ iff $w \notin L(A)$.

即 $L(B) = \Sigma^* - L(A) = \Sigma^* - L = \overline{L}$. 所以，$\overline{L}$ 为正规语言.

关于正规语言的封闭运算

✧ 正规语言的交（*intersection*）

– 结论 若 L 和 M 为正规语言，则 $L \cap M$ 也是正规语言：

– 证明 因为 $L \cap M = \overline{\overline{L} \cup \overline{M}}$，所以 $L \cap M$ 为正规语言.

– 另一证明途径 设 DFA $A_L = (Q_L, \Sigma, \delta_L, q_L, F_L)$ 和 DFA $A_M = (Q_M, \Sigma, \delta_M, q_M, F_M)$ 的语言分别为 L 和 M，

构造 DFA $A = (Q_L \times Q_M, \Sigma, \delta, <q_L, q_M>, F_L \times F_M)$

其中 $\delta(<p, q>, a) = < \delta_L(p, a), \delta_M(q, a) >$.

可以证明 $L(A) = L \cap M$. 所以，$L \cap M$ 为正规语言.

关于正规语言的封闭运算

✧ 正规语言的差（*difference*）

– 结论 若 L 和 M 为正规语言，则 $L-M$ 也是正规语言.

– 证明 因为 $L-M = L \cap \overline{M}$ ，所以 $L-M$ 为正规语言.

关于正规语言的封闭运算

✧ 正规语言的反向（reversal）

– 记号 设字符串 $w=a_1a_2\ldots a_n$, 则 w 的反向（ reversal ）$w^R = a_na_{n-1}\ldots a_1$; 语言 L 的反向 $L^R = \{ w^R \mid w\in L \}$.

– 结论 若 L 为正规语言，则 L^R 也是正规语言：

– 证明 设 L 对应的正规表达式为E，使得 $L(E)=L$. 归纳于 E 的结构，可以证明存在正规表达式E^R, 使得 $L(E^R)= L^R$.

基础：若 E 为ε, ϕ, a，则 $E^R= E$:

归纳：

1. 设 $E=E_1+E_2$，令 $E^R = E_1^R+E_2^R$，可证 $L(E^R)= L^R$:
2. 设 $E=E_1E_2$，令 $E^R = E_2^RE_1^R$，可证 $L(E^R)= L^R$:
3. 设 $E=E_1^*$，令 $E^R = (E_1^R)^*$，可证 $L(E^R)= L^R$;
4. 设 $E=(E_1)$，令 $E^R = (E_1^R)$，可证 $L(E^R)= L^R$.

□

关于正规语言的封闭运算

✧ 正规语言的反向（*reversal*）

– 结论 若 L 为正规语言，则 L^R 也是正规语言：

– 另一证明途径 设有限自动机 A 的语言为 L ，即 $L(A)=L$. 通过以下步骤修改 A 的转移图，得到有限自动机 B:

1. 将 A 的转移图中所有的弧反向;
2. 将 A 的初态作为 B 的唯一终态;
3. 增加一个新的状态 p_0 作为 B 的初态，并从 p_0 到 A 的所有终态增加一条 ε–转移弧.

可以证明 $L(B)= L^R$. 所以，L^R 为正规语言.

关于正规语言的封闭运算

✧ 正规语言的同态（*homomorphism*）

- 记号 设映射 $h: \Sigma \to T^*$, 则对 $w=a_1a_2\ldots a_n \in \Sigma^*$, 定义 $h(w) = h(a_1)\, h(a_2) \ldots h(a_n)$, 称为串 w 的一个同态;

  对语言 $L \subseteq \Sigma^*$, 定义 L 的同态 $h(L) = \{\, h(w) \mid w \in L \,\}$;

- 举例 设 $h(0)=ab$, $h(1)=\varepsilon$, 则

  $$h(0101) = h(0)\, h(1)\, h(0)\, h(1) = abab$$

  对于 $L = \{\, 0^k1^k \mid k \geq 0 \,\}$

  $$h(L) = \{h(0^k1^k) \mid k \geq 0\} = \{\, (ab)^k \mid k \geq 0 \,\} = L((ab)^*)$$

- 结论 若 L 为正规语言, $h: \Sigma \to T^*$, 则 $h(L)$ 也是正规语言

✧ 正规语言的同态（*homomorphism*）

– **结论** 若 L 为正规语言，$h: \Sigma \to T^*$, 则 $h(L)$ 也是正规语言：

– **证明** 设 L 对应的正规表达式为 E，使得 $L(E)=L$. 归纳于 E 的结构，可以证明存在 $h(E)$， $L(h(E)) = h(L(E)) = h(L)$.

基础：若 E 为 ε, ϕ，取 $h(E) = E$, 显然 $L(h(E)) = h(L(E))$;

若 E 为 a，取 $h(E) = h(a)$, 有 $L(h(E)) = h(L(E)) = \{h(a)\}$;

归纳：若 $E=E_1E_2$，取 $h(E) = h(E_1)\, h(E_2)$, 有

$$
\begin{aligned}
L(h(E)) &= L(h(E_1))\, L(h(E_2)) = h(L(E_1))\, h(L(E_2)) \\
&= h(\{w_1 \mid w_1 \in L(E_1)\})\, h(\{w_2 \mid w_2 \in L(E_2)\}) \\
&= \{h(w_1) \mid w_1 \in L(E_1)\}\, \{h(w_2) \mid w_2 \in L(E_2)\} \\
&= \{h(w_1)h(w_2) \mid w_1 \in L(E_1) \wedge w_2 \in L(E_2)\} \\
&= \{h(w_1w_2) \mid w_1w_2 \in L(E_1)L(E_2)\} \\
&= h(L(E_1)L(E_2)) = h(L(E_1E_2)) = h(L(E))
\end{aligned}
$$

$E=E_1+E_2$ 和 $E=E_1^*$ 的情形类似. □

关于正规语言的封闭运算

✧ 正规语言的反同态（*inverse homomorphism*）

– **记号** 设映射 $h: \Sigma \to T^*$, 对语言 $S \subseteq T^*$, 定义 S 的反同态

$$h^{-1}(S) = \{ w \mid w \in \Sigma^* \wedge h(w) \in S \};$$

– **结论** 若 $S \subseteq T^*$ 为正规语言，$h: \Sigma \to T^*$, 则 $h^{-1}(S)$ 也是正规语言：

– **证明** 设 $S = L(A)$ ，其中 DFA $A = (Q, T, \delta, q_0, F)$.

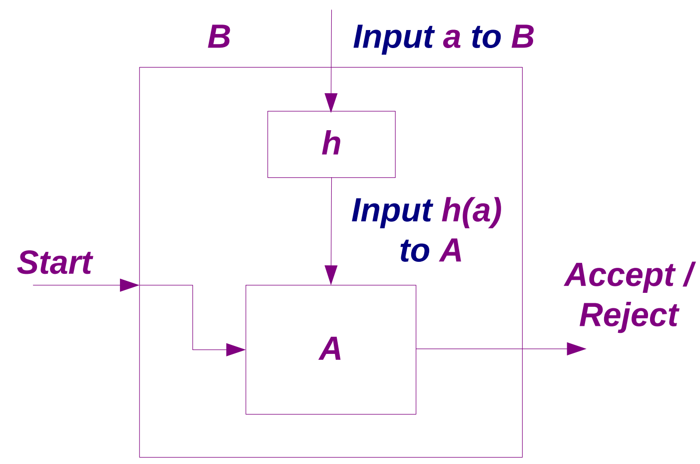
构造 DFA $B = (Q, \Sigma, \gamma, q_0, F)$,

其中 $\gamma(q, a) = \delta'(q, h(a))$.

可证 (归纳于$|w|$) 对任何w，有

$$\gamma'(q_0, w) = \delta'(q_0, h(w)).$$

所以有 $h^{-1}(S) = L(B)$.



□

关于正规语言的封闭运算

✧ 应用：证明某个语言是正规语言

– 例4.17 设 *DFA* $D = (Q, \Sigma, \delta, q_0, F)$,证明下列语言 *L* 是正规语言:

$$L = \{ w \mid w \in L(D) \wedge \forall q \in Q. \exists x_q, x \in \Sigma^*. (w = x_q x \wedge \delta(q_0, x_q) = q) \};$$

（即满足如下条件的字符串 *w* 构成的语言：从 *DFA D* 的初态出发，经过标有*w*的路径可经过 *D* 的任何状态）

关于正规语言的封闭运算

✧ 应用：证明某个语言不是正规语言

– 例 证明如下语言不是正规语言

a，*b*，*c* 串构成的语言

$L = \{a^i b^j c^k \mid i,j,k \geq 0, 若\ i=1\ 则\ j=k\}$

– 证明思路 反证法。

假设 *L* 是正规语言，则

$L' = L \cap \{a\, b^j c^k \mid j,k \geq 0\} = \{a\, b^j c^k \mid j,k \geq 0 \wedge j=k\}$

也是正规语言。设 $h(a)=\varepsilon$, $h(b)=0$, $h(c)=1$, 则

$h(L')=\{0^n 1^n \mid n \geq 0\}$

是正规语言。但我们已知后者不是正规语言。

课后练习

✧ 必做题:

- Ex.4.1.1(e)
- Ex.4.1.2(e)
- Ex.4.1.2(f)
- Ex.4.2.1(d),(f)
- *!Ex.4.2.2
- !Ex.4.2.3
- *!! Ex.4.2.8
- !Ex.4.2.13
- Ex.4.3.4

✧ 思考题:

- Ex.4.1.2(c)
- !Ex.4.2.6
- Ex.4.3.2

课后练习

✧ 自测题:

- 语言L由所有满足如下条件的0，1串构成：0的数目二倍于1的数目。试应用Pumping引理证明L不是正规语言。
- 语言*L*由所有满足如下条件的0, 1串构成: 0的数目多于1的数目（对0和1在串中出现的次序没有限制）。试应用*Pumping*引理证明*L*不是正规语言。
- 设映射 $h: \{a,b\} \to \{0,1\}^*$ 定义为 $h(a) = \varepsilon$, $h(b) = 10$。定义 $\{a, b\}$ 上的一个正规表达式 $E = \varepsilon + (a+b)(ba)^*$。
  (1)给出一个正规表达式 E^R，使得 $L(E^R) = (L(E))^R$（后者为 $L(E)$ 的反向）
  (2) 给出一个正规表达式 $h(E)$，使得 $L(h(E)) = h(L(E))$。
  (3)试构造一个 *DFA A*，使得 $L(A) = \sim L(h(E))$。这里，$\sim$ 代表语言言的补运算。
- 假设 *A* 是字母表 Σ 上的*DFA*。给出*判定* $L(A) \neq \Sigma^*$ 的一个简要的算法思想（以自然语言叙述即可）。

课后练习

✧ **自测题:**

– 假设 *A*，*B* 是字母表 Σ 上的*DFA*。给出判定 $L(A) \cup L(B) = \Sigma^*$ 的一个简要的算法思想（以自然语言叙述即可）。

– 正规表达式 a*bb 表示 {a , b} 上的一个语言，试构造一个接受该语言的补语言的 DFA

– 左下图（a），（b）分别是 DFA A1和A2的转移表，试设计语言为L(A1)∩L(A2)的一个DFA（以转移表的形式给出）。

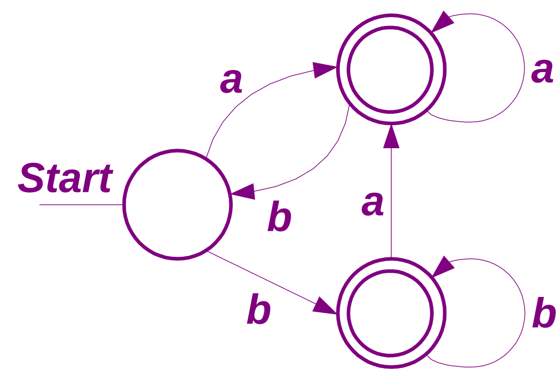
– 设映射 h：$\{0, 1\} \to \{a, b\}^*$ 定义为 $h(0) = ab$，$h(1) = ba$。右下图表示 {a, b} 上的一个 *DFA A*。试构造一个{0, 1}上的 *DFA B*，使得 $L(B) = h^{-1}(L(A))$。

	0	1
→ q_1	q_2	q_1
q_2	q_3	q_1
* q_3	q_3	q_2

(a)

	0	1
→ p_1	p_2	p_1
* p_2	p_1	p_2

(b)



That's all for today.

Thank You