Chapter 2

有穷自动机

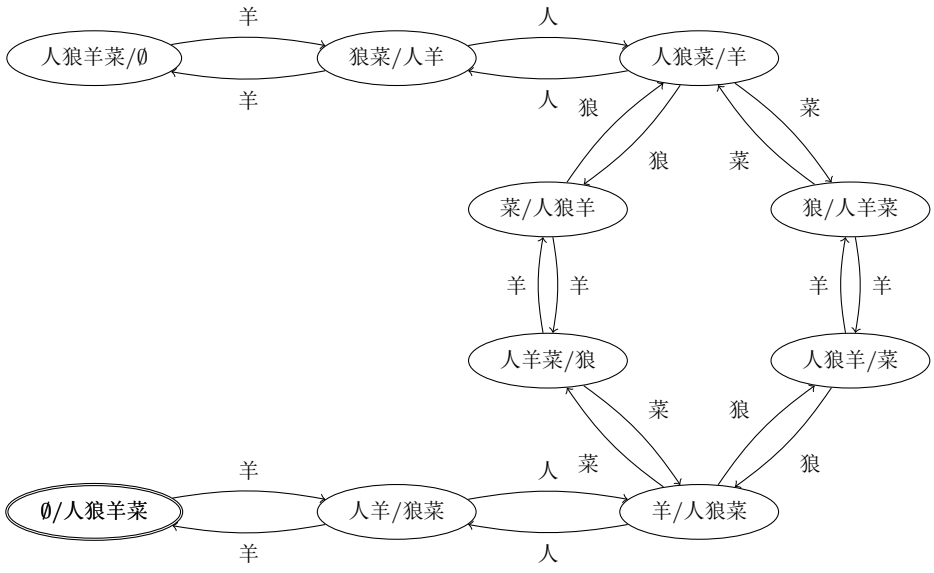
2.1 有穷状态系统

有穷自动机是具有离散输入和输出系统的一种数学模型. 系统内可以处于任一有穷个内部的格局或称“状态”. 系统的状态概括了关于过去输入的某些信息, 并为确定系统以后的行为所必须. 电梯的控制机构, 就是有穷状态系统的一个典型例子.

计算机科学中, 有穷系统的例子有很多, 常见的比如计算机的控制器、词法分析器、协议分析等.

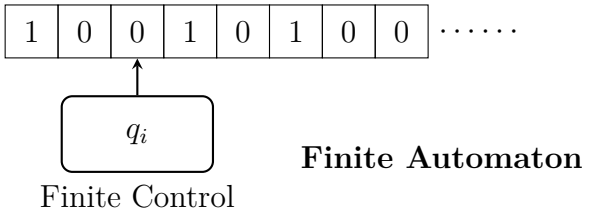
示例

狼, 羊, 菜, 人的过河问题. 一个人带着狼、山羊、白菜在一条河的左岸, 有一条船, 大小正好能装下这个人和其它三者之一. 每次只能带一件东西过河, 剩下的两件, 如果没人照顾, 狼会吃羊, 羊会吃菜. 问是否有可能安全过河, 使得羊和白菜都不会被吃掉?



2.2 确定的有穷自动机

确定型的有穷自动机具有一个有穷控制器, 一条输入带和一个读头. 有穷控制器可以记住有穷个状态, 其中一个是当前状态; 输入带上分为单元格, 每个单元格可以放置一个输入字符, 输入带放置一个输入串; 读头可以读取单元格上的字符, 并向后移动一个单元格.



一个确定的有穷自动机由一个有穷状态集和一个从状态到状态的转移集组成, 转移出现在输入时. 每个输入是字母表中的一个符号. 对每个输入符号, 从每一个状态恰有一个转移 (可以转移到这个状态本身). 有一个初始状态, 自动机从它开始. 某些状态被规定为终态或接受状态.

2.2.1 形式定义

确定型有穷自动机 (*Deterministic Finite Automaton*, DFA) A 的形式定义为五元组

$$A = (Q, \Sigma, \delta, q_0, F)$$

其中

(1) Q: *有穷状态集*;

(2) Σ: *有穷输入符号集或字母表*;

(3) $\delta : Q \times \Sigma \mapsto Q$, *状态转移函数*;

(4) q_0: *初始状态*, $q_0 \in Q$;

(5) F: *终结状态集或接受状态集*, $F \subseteq Q$.

开始时, 输入串在输入带上, 读头在第一个字符, 有穷控制器初始处于 q_0. 自动机的读头每次读入一个字符, 根据转移函数修改当前状态, 并向后移动一个单元格. 若输入串全部读入后, 处于接受状态, 那么自动机接受这个输入串, 否则拒绝该串.

示例

接受全部含有 01 子串的 0 和 1 构成的串.

首先是字母表 $\Sigma = \{0, 1\}$, 然后分析串的特点: 01 是子串, 则在扫描输入串的过程中需要记住:

(1) 当前已经发现 01, 那么串的其余部分不用再关心;

(2) 还没发现 01, 但刚刚已经读入了一个 0, 那么只要再读入 1 就符合条件了;

(3) 还没发现 01, 甚至 0 都还没出现.

刚好这三种情况可以对应三个状态, 因此

$$A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$$

其中 δ :

$$\begin{array}{ll} \delta(q_0, 1) = q_0 & \delta(q_0, 0) = q_1 \\ \delta(q_1, 0) = q_1 & \delta(q_1, 1) = q_2 \\ \delta(q_2, 1) = q_2 & \delta(q_2, 0) = q_2 \end{array}$$

2.2.2 DFA 的表示

DFA 除了使用其形式定义的五元组表示, 也可以有两种简化的表示方法, 分别为状态转移图 (*transition diagram*) 和状态转移表 (*transition table*)

DFA 的转移图

状态转移图的定义

(1) 每个状态对应一个节点, 用圆圈表示;

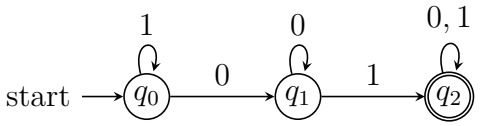
(2) 每个 $\delta(q,a)=p$ 对应一条从节点 q 到 p 的有向边, 边的标记为 a;

(3) 开始状态 q_0 有一个标有 *start*的箭头;

(4) 接受状态的节点, 用双圆圈表示.

示例

前面的例子的转移图如下



DFA 的转移表

示例

前面例子的转移表如下

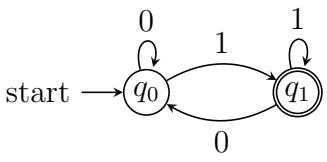
	0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
$*q_2$	q_2	q_2

2.2.3 DFA 的设计

典型的问题: 给定语言 L, 设计 DFA 使其接受 (且仅接受) 语言 L.

示例

若 $\Sigma=\{0,1\}$, 给出接受全部以 1 结尾的串的 DFA.



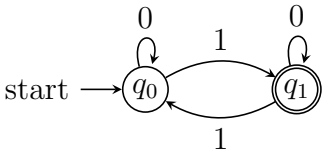
示例

$\Sigma=\{0,1\}$, 给出接受 Σ^* 的 DFA.

$\Sigma = \{0, 1\}$, 给出接受 $\{\varepsilon\}$ 的 DFA.

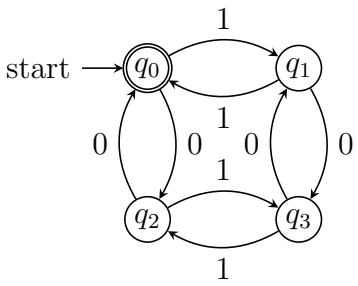
示例

$\Sigma = \{0, 1\}$, 给出接受全部含有奇数个 1 的串 DFA.



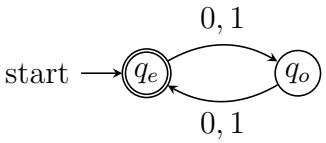
示例

$\Sigma = \{0, 1\}$, 给出接受全部含有偶数个 0 和偶数个 1 的串 DFA.



示例

$\Sigma = \{0, 1\}$, 给出一个 DFA, 其接受长度为偶数的任意串.



2.2.4 扩展转移函数

转移函数 δ 是 $Q \times \Sigma$ 上的函数, 所以只能处理 Σ 中的字符, 为了使用方便, 定义字符串上的转移函数 $\hat{\delta} : Q \times \Sigma^* \mapsto Q$, 如下

(1) $\hat{\delta}(q, \varepsilon) = q$;

(2) 若 $w = xa$, 那么 $\hat{\delta}(q, w) = \delta(\hat{\delta}(q, x), a)$.

$\hat{\delta}$ 的含义可以理解为, 从一个状态开始, 读入某个串后, 所到达的状态.

那么, 如果 $w = a_0 a_1 \cdots a_n$, 那么 $\hat{\delta}(q, w) = \delta(\delta(\cdots \delta(\hat{\delta}(q, \varepsilon), a_0) \cdots, a_{n-1}), a_n)$.

2.2.5 DFA 的语言

DFA $A = (Q, \Sigma, \delta, q_0, F)$ 接受的语言记为为 $\mathbf{L}(A)$, 定义如下:

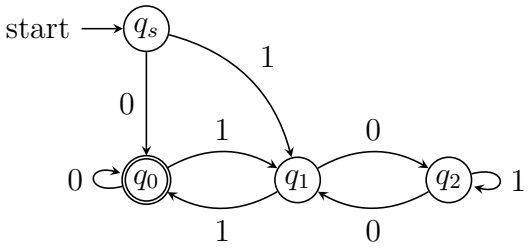
$$\mathbf{L}(A) = \{w \mid \hat{\delta}(q_0, w) \in F\}.$$

如果一个语言 L 是某个 DFA A 的语言, 即 $L = \mathbf{L}(A)$, 则称 L 是正则语言.

示例

Design a DFA that accepts all strings w over $\{0, 1\}$ such that w is the binary representation of a number that is a multiple of 3.

设 q_0, q_1, q_2 分别对应模 3 为 0, 1, 2 的状态; 此外, 因为不含空串, 设开始状态为 q_s; 对 q_0, q_1, q_2 每个当前状态, 输入 0 相当于乘 2, 输入 1 相当于乘 2 加 1, 那么可以找到相应的转移规律.



2.3 非确定的有穷自动机

下面给出非确定的有穷自动机的概念. 我们将最终证明, 被非确定的有穷自动机接受的任何集合, 都能够被确定的有穷自动机所接受. 然而, 非确定性概念无论在语言理论还是在计算理论中都起着重要的作用. 在有穷自动机的简单情况下, 透彻的理解这个概念是非常有益的. 后面, 我们将碰到确定形式和非确定形式不等价的自动机, 以及另外一些自动机, 这两种形式的等价性是一个深刻的、重要的悬而未决的问题.

修改 FA 模型, 使之对同一输入符号, 从一个状态可以有零个、一个或多个的转移. 这种新模型, 称为非确定有穷自动机. 非确定的有穷自动机具有同时处在几个状态的能力, 在处理输入串时, 几个当前状态能“并行的”跳转到下一个状态.

示例

由 0 和 1 构成的串中, 接受全部以 01 结尾的串.

2.3.1 形式定义

非确定型的有穷自动机 (*Nondeterministic Finite Automaton*, NFA) A 的形式定义为五元组

$$A = (Q, \Sigma, \delta, q_0, F)$$

其中

(1) Q: 有穷状态集;

(2) Σ: 有穷输入符号集或字母表;

(3) $\delta : Q \times \Sigma \mapsto 2^Q$, 状态转移函数;

(4) q_0: 初始状态, $q_0 \in Q$;

(5) F: 终结状态集或接受状态集, $F \subseteq Q$.

NFA 与 DFA 的区别是转移函数和接受方式: NFA 转移函数一般形式 $\delta(q,a) = \{p_1, p_2, \cdots, p_n\}$; 当输入串全部读入时, NFA 所处的状态中, 只要包括 F 中的状态, 就称为接受该串.

示例

前面的例子中, NFA 的定义可以形式化的表述为

$$A = (\{q_0, q_1, q_2\}, \{0,1\}, \delta, q_0, \{q_2\})$$

其中转移函数 δ 如下:

$$\begin{array}{lll} \delta(q_0,0) = \{q_0, q_1\} & \delta(q_1,0) = \emptyset & \delta(q_2,0) = \emptyset \\ \delta(q_0,1) = \{q_0\} & \delta(q_1,1) = \{q_2\} & \delta(q_2,1) = \emptyset \end{array}$$

若表示为状态转移表:

	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
$*q_2$	$\emptyset$	$\emptyset$

2.3.2 扩展转移函数

与 DFA 类似, 将 δ 扩展到输入串. 定义转移函数 $\hat{\delta} : Q \times \Sigma^* \mapsto 2^Q$ 的扩展为

(1) $\hat{\delta}(q, \varepsilon) = \{q\}$;

(2) 若 $w = xa$, 那么 $\hat{\delta}(q,w) = \bigcup_{p \in \hat{\delta}(q,x)} \delta(p,a)$.

示例

前面的例子中, 若输入是 00101, 每一步的状态转移分别是:

(1) $\hat{\delta}(q_0, \varepsilon) = \{q_0\}$

(2) $\hat{\delta}(q_0, 0) = \delta(q_0, 0) = \{q_0, q_1\}$

(3) $\hat{\delta}(q_0, 00) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

(4) $\hat{\delta}(q_0, 001) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

(5) $\hat{\delta}(q_0, 0010) = \delta(q_0, 0) \cup \delta(q_2, 0) = \{q_0, q_1\} \cup \emptyset = \{q_0, q_1\}$

(6) $\hat{\delta}(q_0, 00101) = \delta(q_0, 1) \cup \delta(q_1, 1) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

因为 q_2 是接受状态, 所以 NFA 接受 00101.

2.3.3 NFA 的语言

定义 NFA $A = (Q, \Sigma, \delta, q_0, F)$ 的语言 $\mathbf{L}(A)$ 为

$$\mathbf{L}(A) = \{w \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset\}.$$

2.3.4 DFA 与 NFA 的等价性

每个 DFA 都是一个 NFA, 显然, NFA 接受的语言包含正则语言. 下面的定理给出, NFA 也仅接受正则语言. 证明的关键表明 DFA 能够模拟 NFA, 即, 对每个 NFA, 能够构造一个等价的 DFA. 使用一个 DFA 模拟一个 NFA 的方法是让 DFA 的状态对应于 NFA 的状态集合.

定理 1. *如果语言 L 被某个 NFA 接受, 当且仅当 L 被某个 DFA 接受.*

证明. 设 NFA N=(Q_N, Σ, δ_N, q_0, F_N) 接受 L, 那么构造 DFA D=(Q_D, Σ, δ_D, $\{q_0\}$, F_D) 如下:

(1) $Q_D = 2^{Q_N}$, 即 DFA 的状态是 NFA 的状态集;

(2) 开始状态 $\{q_0\}$;

(3) 终态 $F_D = \{S \mid S \subseteq Q_N, S \cap F_N \neq \emptyset\}$, 即包含 NFA 终态的状态集, 作为 DFA 的终态;

(4) 状态转移函数 δ_D, 对每个 $S \subseteq Q_N$ 和每个输入字符:

$$\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a)$$

即 $\delta_D(S, a)$ 的计算, 就是在 NFA 中, 从 S 中每个状态 p 在输入 a 之后所能达到的全部状态的集合.

下面用归纳法, 证明 $\mathbf{L}(D) = \mathbf{L}(N)$. 对串 w 的长度进行归纳, 往证

$$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$$

成立.

[归纳基础] 当 $|w| = 0$ 时, 即 $w = \varepsilon$ 时, 显然 $\hat{\delta}_D(\{q_0\}, \varepsilon) = \hat{\delta}_N(q_0, \varepsilon) = \{q_0\}$.

[归纳假设] 假设 $|w| = n$ ($n \geq 0$) 时, 上式成立;

[归纳递推] 那么, 当 $|w| = n + 1$ 时, 将 w 拆分成 $w = xa$ 的形式, a 是 w 最后一个符号. 由 $\hat{\delta}_N$ 的定义, 有

$$\hat{\delta}_N(q_0, w) = \hat{\delta}_N(q_0, xa) = \bigcup_{p \in \hat{\delta}_N(q_0, x)} \delta_N(p, a)$$

由 $\hat{\delta}_D$ 的定义, 有

$$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_D(\{q_0\}, xa) = \delta_D(\hat{\delta}_D(\{q_0\}, x), a)$$

由上面的 DFA D 的构造, 有

$$\delta_D(\hat{\delta}_D(\{q_0\}, x), a) = \bigcup_{p \in \hat{\delta}_D(\{q_0\}, x)} \delta_N(p, a)$$

又因为, 由归纳假设

$$\hat{\delta}_D(\{q_0\}, x) = \hat{\delta}_N(q_0, x)$$

因此 $\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_N(q_0, w)$ 成立.

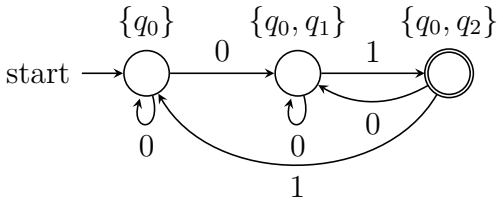
因此 $\forall w \in \mathbf{L}(N)$, 有 $\hat{\delta}_N(q_0, w) \cap F_N \neq \emptyset$, 那么 $\hat{\delta}_D(\{q_0\}, w) \in F_D$, 所以 $w \in \mathbf{L}(D)$, 即 $\mathbf{L}(N) \subseteq \mathbf{L}(D)$; 且 $\forall w \in \mathbf{L}(D)$, 有 $\hat{\delta}_D(\{q_0\}, w) \in F_D$, 那么 $\hat{\delta}_N(q_0, w) \cap F_N \neq \emptyset$, 所以 $w \in \mathbf{L}(N)$, 即 $\mathbf{L}(D) \subseteq \mathbf{L}(N)$; 因此 $\mathbf{L}(D) = \mathbf{L}(N)$.

□

示例 (子集构造法, 构造与 NFA 等价的 DFA)

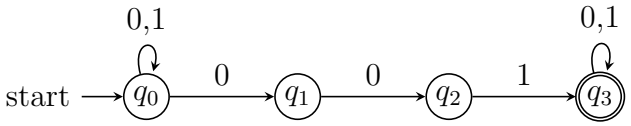
通过前面例子中 NFA 的状态转移表, 可以如下构造 DFA(横线划掉了无用状态)

	0	1
$\to \{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
~~$\{q_1\}$~~	$\emptyset$	$\{q_2\}$
~~$*\{q_2\}$~~	$\emptyset$	$\emptyset$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_2\}$
$*\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
~~$\emptyset$~~	$\emptyset$	$\emptyset$
~~$*\{q_1, q_2\}$~~	$\emptyset$	$\{q_2\}$
~~$*\{q_0, q_1, q_2\}$~~	$\{q_0, q_1\}$	$\{q_0, q_2\}$



示例

Design a NFA that accepts strings with 001 as substring.



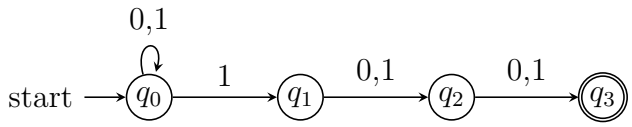
2.4 带有空转移的有穷自动机 (FA with ε-Transition)

2.4.1 ε-NFA 举例

为 FA 增加另一种扩展: 在空串 (ε) 发生状态转移.

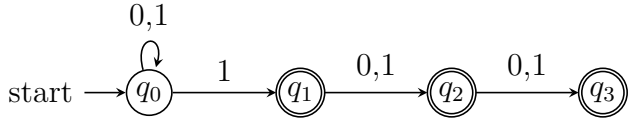
示例

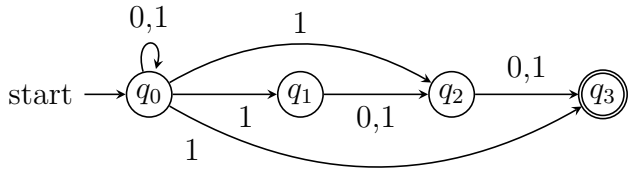
L=$\{w \mid w$ 倒数第 3 个字符是 1, $w \in \{0,1\}^*\}$



示例

L=$\{w \mid w$ 倒数 3 个字符至少有一个是 1, $w \in \{0,1\}^*\}$





2.4.2 ε-NFA 形式定义

带有空转移的非确定有穷自动机 (ε-NFA) A 的形式定义为五元组

$$A = (Q, \Sigma, \delta, q_0, F)$$

其中:

(1) Q: 有穷状态集;

(2) Σ: 有穷输入符号集或字母表;

(3) δ: $Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^Q$ 状态转移函数, 允许在空串上转移 (未读入字符就转移);

(4) q_0: 初始状态, $q_0 \in Q$;

(5) F: 终结状态集或接受状态集, $F \subseteq Q$.

与 DFA 相比, ε-NFA 的主要区别:

(1) 自动机处于某状态, 读入某个字符时, 可能有多余一个的转移;

(2) 自动机处于某状态, 读入某个字符时, 可能没有转移;

(3) 自动机处于某状态, 可能不读入字符, 就进行转移.

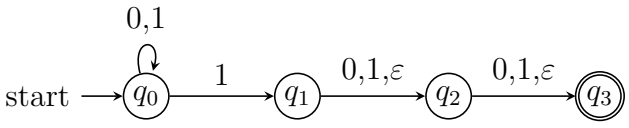
示例

前面例子的语言 L=$\{w \mid w$ 倒数 3 个字符至少有一个是 1, $w \in \{0,1\}^*\}$

可以构造 ε-NFA 为 $E = (\{q_0, q_1, q_2, q_3\}, \{0,1\}, \delta, q_0, \{q_3\})$ 其中 δ 如转移表:

	0	1	ε
$\rightarrow q_0$	$\{q_0\}$	$\{q_0, q_1\}$	$\emptyset$
q_1	$\{q_2\}$	$\{q_2\}$	$\{q_2\}$
q_2	$\{q_3\}$	$\{q_3\}$	$\{q_3\}$
$*q_3$	$\emptyset$	$\emptyset$	$\emptyset$

或状态转移图:



2.4.3 ε-闭包 (ε-Closure)

状态 q 的 ε-闭包: 从状态 q 经过全部标记 ε 的边所能到达的所有状态 (以及 q 自身) 的集合.

ε-闭包的递归定义:

(1) 状态 $q \in$ ECLOSE(q);

(2) $\forall p \in$ ECLOSE(q), 若 $r \in \delta(p, \varepsilon)$, 则 $r \in$ ECLOSE(q).

如果 S 是状态集, 则定义:

$$\text{ECLOSE}(S) = \bigcup_{q \in S} \text{ECLOSE}(q)$$

示例

前面例子

	0	1	ε	ECLOSE()
$\to q_0$	$\{q_0\}$	$\{q_0, q_1\}$	$\emptyset$	$\{q_0\}$
q_1	$\{q_2\}$	$\{q_2\}$	$\{q_2\}$	$\{q_1, q_2, q_3\}$
q_2	$\{q_3\}$	$\{q_3\}$	$\{q_3\}$	$\{q_2, q_3\}$
$*q_3$	$\emptyset$	$\emptyset$	$\emptyset$	$\{q_3\}$

2.4.4 扩展转移函数

(1) $\hat{\delta}(q, \varepsilon) = \text{ECLOSE}(q)$

(2) 如果 $w = xa$, 则必然 $a \neq \varepsilon$, 因为 $\varepsilon \notin \Sigma$, 则

$$\hat{\delta}(q, w) = \text{ECLOSE}(\bigcup_{p \in \hat{\delta}(q,x)} \delta(p, a))$$

即: 若设 $\hat{\delta}(q, x) = \{p_1, p_2, \ldots, p_k\}$, 则从每个 p_i 经过 a 边到达的所有状态为 $\bigcup_{i=1}^{k} \delta(p_i, a)$; 再设 $\bigcup_{i=1}^{k} \delta(p_i, a) = \{r_1, r_2, \ldots, r_m\}$, 则每个 r_j 再求 ε-闭包, 所得到的状态集, 定义为 $\hat{\delta}(q, w)$; 即 $\hat{\delta}(q, w) = \text{ECLOSE}(\{r_1, r_2, \ldots, r_m\})$

例: 前面的例子中, 求 $\hat{\delta}(q_0, 10) =?$

$\hat{\delta}(q_0, \varepsilon) = \text{ECLOSE}(q_0) = \{q_0\}$

$\hat{\delta}(q_0, 1) = \text{ECLOSE}(\hat{\delta}(q_0, 1)) = \text{ECLOSE}(\{q_0, q_1\}) = \{q_0\} \cup \{q_1, q_2, q_3\} = \{q_0, q_1, q_2, q_3\}$

$\hat{\delta}(q_0, 10) = \text{ECLOSE}(\hat{\delta}(q_0, 0) \cup \hat{\delta}(q_1, 0) \cup \hat{\delta}(q_2, 0) \cup \hat{\delta}(q_3, 0)) = \text{ECLOSE}(\{q_0, q_2, q_3\}) = \{q_0, q_2, q_3\}$

2.4.5 ε-NFA 的语言

若 ε-NFA $E = (Q, \Sigma, \delta, q_0, F)$, 则定义 E 的语言 $\mathbf{L}(E)$ 为

$$\mathbf{L}(E) = \{w \mid \hat{\delta}(q_0, w) \cap F \neq \emptyset\}.$$

2.4.6 消除空转移

若有 ε-NFA E, 构造 DFA D, 使 $\mathbf{L}(D) = \mathbf{L}(E)$, 方法与子集构造法类似, 但使用 ε 闭包代替状态转移后的集合.

设 ε-NFA $E = (Q_E, \Sigma, \delta_E, q_E, F_E)$, 构造 DFA $D = (Q_D, \Sigma, \delta_D, q_D, F_D)$, 规则如下:

(1) $Q_D = 2^{Q_E} = \{S \mid S \subseteq Q_E\}$ – 但实际上只有满足 $S = \text{ECLOSE}(S)$ 是有效的;

(2) $q_D = \text{ECLOSE}(q_E)$ – 即使用 q_E 的闭包作为 D 的开始状态;

(3) $F_D = \{S \mid S \in Q_D, S \cap F_E \neq \emptyset\}$ – 只要包含 F_E, 就是 D 的接受状态;

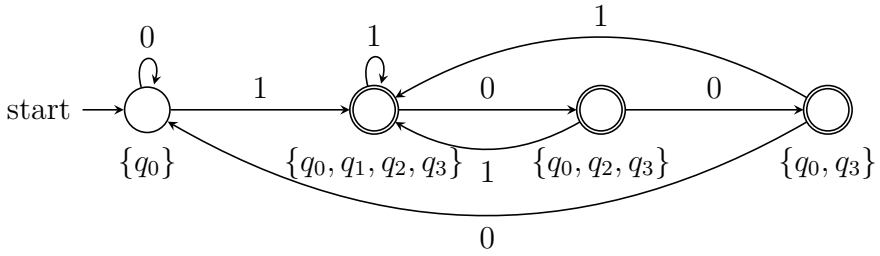
(4) δ_D: $\forall S \in Q_D$, $\delta_D(S, a) = \text{ECLOSE}(\bigcup_{p \in S} \delta_E(p, a))$.

示例

续前例, 得到消除 ε 转移后, 得到的 DFA 状态转移表

	0	1
$\to \{q_0\}$	$\{q_0\}$	$\{q_0, q_1, q_2, q_3\}$
$*\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_2, q_3\}$	$\{q_0, q_1, q_2, q_3\}$
$*\{q_0, q_2, q_3\}$	$\{q_0, q_3\}$	$\{q_0, q_1, q_2, q_3\}$
$*\{q_0, q_3\}$	$\{q_0\}$	$\{q_0, q_1, q_2, q_3\}$

转移图为



2.4.7 ε-NFA 与 DFA 等价性

定理 2. *如果语言 L 被某个 ε-NFA 接受, 当且仅当 L 被某个 DFA 接受.*

证明. ($\Leftarrow$) 已知 DFA $D = (Q, \Sigma, \delta_D, q_0, F)$ 只需构造 ε-NFA E 的构造函数 $\delta_E(q, \varepsilon) = \emptyset$, 且对每个 $\delta_D(q, a) = p$ 都有 $\delta_E(q, a) = p$; E 与 D 完全一样, 但没有任何空转移, 接受的语言也一样.

($\Rightarrow$) 先证明 $\hat{\delta}_E(q_E, w) = \hat{\delta}_D(q_D, w)$, 对 $|w|$ 归纳

(1) 当 $|w| = 0$ 时, 有 $\hat{\delta}_E(q_E, \varepsilon) = \text{ECLOSE}(q_E)$
由构造方法得 $q_D = \text{ECLOSE}(q_E)$, 所以 $\hat{\delta}(q_D, \varepsilon) = q_D = \text{ECLOSE}(q_E)$

(2) 假设 $|w| = n$ 时, 等式成立, 则当 $|w| = n + 1$ 时, $w = xa$, 则有 $\hat{\delta}_E(q_E, x) = \hat{\delta}_D(q_D, x)$.
设 $S = \hat{\delta}_E(q_E, x) = \hat{\delta}_D(q_D, x)$,
$\hat{\delta}_E(q_E, w) = \text{ECLOSE}(\bigcup_{p \in S} \delta_E(p, a))$,
又 $\hat{\delta}_D(q_D, w) = \delta_D(\hat{\delta}_D(q_D, x), a) = \delta_D(S, a) = \text{ECLOSE}(\bigcup_{p \in S} \delta_E(p, a))$
所以 $\hat{\delta}_E(q_E, w) = \hat{\delta}_D(q_D, w)$.
对 $\forall w \in \mathbf{L}(E)$ 有 $\hat{\delta}_E(q_E, w) \cap F_E \neq \emptyset$ 即 $\hat{\delta}_D(q_D, w) \cap F_E \neq \emptyset$ 即 $\hat{\delta}_D(q_D, w) \in F_D$ 即 $x \in \mathbf{L}(D)$
所以 $\mathbf{L}(E) \subseteq \mathbf{L}(D)$;
又 $\forall w \in \mathbf{L}(D)$ 有 $\hat{\delta}_D(q_D, w) \in F_D$ 即 $\hat{\delta}_D(q_D, w) \cap F_E \neq \emptyset$ 即 $\hat{\delta}_E(q_E, w) \cap F_E \neq \emptyset$ 即 $x \in \mathbf{L}(E)$
所以 $\mathbf{L}(D) \subseteq \mathbf{L}(E)$.
所以 $\mathbf{L}(E) = \mathbf{L}(D)$.

□

示例

Design ε-NFA for language:$\{0^k \mid k \text{ is a multiple of 2 or 3}\}$.