第4章 正则表达式

定义

Basis 1: If a is any symbol, then **a** is a RE, and L(**a**) = {a}.

- Note: {a} is the language containing one string, and that string is of length 1.

Basis 2: ϵ is a RE, and $L(\epsilon) = \{\epsilon\}$.

Basis 3: $\varnothing$ is a RE, and $L(\varnothing) = \varnothing$.

优先级：* > 连接 > +

RE和DFA的等价性

RE -> ε-NFA：形式上递归构造即可，又已知ε-NFA和DFA等价

DFA -> RE：k-Path Induction

Basis: k=0. R_{ij}^0 = sum of labels of arc from i to j.

- $\varnothing$ if no such arc.
- But add ϵ if i=j.

- A k-path from i to j either:
  1. Never goes through state k, or
  2. Goes through k one or more times.

$$R_{ij}^k = R_{ij}^{k-1} + R_{ik}^{k-1}(R_{kk}^{k-1})^* R_{kj}^{k-1}.$$

正则语言的判定性质

成员性：在DFA上模拟该串

非空性：终止状态是否可达

有限性：如果DFA有n个状态，是否有长度大于等于n的串

如果有长度大于2n的串，总可以缩短到 [n, 2n-1]

Test for membership all strings of length between n and 2n-1.

- If any are accepted, then infinite, else finite.

A terrible algorithm.

Better: find cycles between the start state and a final state.

泵引理

用于判断一个语言是否是正则语言

For every regular language L

There is an integer n, such that

For every string w in L of length $\geq$ n

We can write w = xyz such that:

1. $|xy| \leq n$.
2. $|y| > 0$.
3. For all $i \geq 0$, xy^iz is in L.

Number of states of DFA for L

Labels along first cycle on path labeled w

等价性

product DFA

- Start state = $[q_0, r_0]$ (the start states of the DFA's for L, M).
- Transitions: $\delta([q,r], a) = [\delta_L(q,a), \delta_M(r,a)]$
  - δ_L, δ_M are the transition functions for the DFA's of L, M.
  - That is, we simulate the two DFA's in the two state components of the product DFA.

Make the final states of the product DFA be those states [q, r] such that exactly one of q and r is a final state of its own DFA.

Thus, the product accepts w iff w is in exactly one of L and M.

L = M if and only if the product automaton's language is empty.

Given regular languages L and M, is $L \subset M$?

乘积自动机，接受状态：[q,r]（q为接收状态而r不为接受状态），如果接受的语言为空， 则属于

DFA最小化

1）空串区分所有的接受状态和非接受状态

2）如果 q, r 在同一输入a下到达的状态 q', r' 可区分，则 q, r 可区分

3）将不可区分的状态合并成同一个状态（注意不可区分性质的传递性）

正则语言的封闭性

∩：乘积自动机

∪：等价于构造操作中的 +

连接：正则语言的构造操作

*：正则语言的构造操作

差：乘积自动机

补：Σ*本身也是正则语言

逆：构造逆的正则表达式

Basis: If E is a symbol a, ϵ, or $\emptyset$, then $E^R = E$.

Induction: If E is

- F+G, then $E^R = F^R + G^R$.
- FG, then $E^R = G^R F^R$
- F*, then $E^R = (F^R)^*$.

同态：对原正则表达式中的符号用同态函数映射的结果替换

逆同态： $h^{-1}(L) = \{w \mid h(w) \text{ is in } L\}$.

通过构造DFA：

Start with a DFA A for L.

Construct a DFA B for $h^{-1}(L)$ with:

- The same set of states.
- The same start state.
- The same final states.
- Input alphabet = the symbols to which homomorphism h applies.

The transitions for B are computed by applying h to an input symbol *a* and seeing where A would go on sequence of input symbols h(a).

Formally, $\delta_B(q, a) = \delta_A(q, h(a))$.

例子如下：