Chapter1 regular language(正则语言)

1.1 finite automaton(有穷自动机)

1.1.1 Formal definition of a finite automaton(有穷自动机的形式化定义)

A **(deterministic)finite automaton** is a **5-tuple(Q,Σ,δ,q_0,F)**,where

1.Q is finite set called ***states***(状态集)

2.Σ is finite set called ***alphabet***(字母表)

3.δ: $Q \times \Sigma \longrightarrow Q$ is the ***transition function***(转移函数)

4.$q_0 \in Q$ is the ***start state***(起始状态)

5.$F \subseteq Q$ is the set of ***accept states***(接受状态集)

1.1.3 Formal definition of computation(计算的形式化定义)

Let $M=(Q,\Sigma,\delta,q_0,F)$ be a finite automaton and let $w=w_1w_2......w_n$ be a string with $w_i \in \Sigma$ for all $i \in [n]$.Then M accepts w if a sequence of states $r_0,r_1,......,r_n$ in Q exists with:

1.$r_0=q_0$;

2.$\delta(r_i,w_{i+1}) = r_{i+1}$ for i=0,......n-1;

3.$r_n \in F$;

We say M recognizes A if A={w|M accepts w};

Definition of regular language

A language is called ***regular*** if some finite automaton recognizes it.

1.1.5 Regular operation(正则运算)

Let A and B be languages. We define the regular operations **union**(并）,**concatenation**(连接),and **star**(星号) as follows:

1.Union: $A \cup B$ = {x| x $\in$ A or x $\in$ B};

2.Concatenation:A $\circ$ B = {xy| x $\in$ A and y $\in$ B};

3.Star: $A^*=\{x_1x_2...x_k \mid k \geq 0$ and each $x_i \in A\}$;

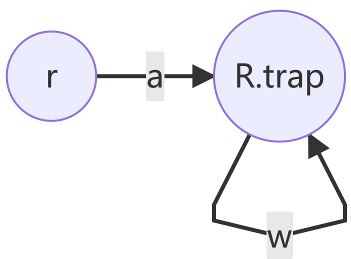
(Union)Theorem[正则语言在并运算下封闭**]**

The class of regular languages is closed under the union operation. In other words, if A_1 and A_2 are regular languages, so is A1 $\cup$ A2.

Proof:[构造一个有穷自动机识别A1∪A2]

1.assume without generality Σ_1=Σ_2:[我们可以不失一般性的假设Σ_1=Σ_2,后面都假设它们相同]

- Let a $\in$ Σ_2-Σ_1;
- We add $\delta_1(r,a)=r_{trap}$, where r_{trap} is a new state with $\delta_1(r_{trap},w)=r_{trap}$ for every w.



2.We construct M = (Q,Σ,δ,q_0,F) to recognize $A_1 \cup A_2$:

- $Q = Q_1 \times Q_2 = \{(r_1,r_2) \mid r_1 \in Q_1, r_2 \in Q_2\}$;
- $\Sigma = \Sigma_1 = \Sigma_2$;
- For each $(r_1,r_2) \in Q$ and $a \in \Sigma$ we let $\delta((r_1,r_2),a)=(\delta_1(r_1,a),\delta_2(r_2,a))$;

- $q_0 = (q_1,q_2)$;
- $F = (F \times Q_2) \cup (Q_1 \times F) = \{(r_1,r_2) \mid r_1 \in F_1 \text{ or } r_2 \in F_2\}$;

1.2 Nondeterminism(非确定性)

1.2.1 Nondeterministic finite automaton(非确定型有穷自动机)

Formal definition of nondeterministic finite automaton

A **nondeterministic finite automaton(NFA)** is a **5-tuple(Q,Σ,δ,q_0,F)**,where

1.Q is a finite set of states;

2.Σ is a finite alphabet;

3.$\delta : Q \times \Sigma\epsilon \longrightarrow P(Q)$ is the transition function, where $\Sigma\epsilon$=$\Sigma \cup \epsilon$;

$$\text{相比于确定型有穷自动机}\begin{cases}1.\epsilon\text{作为空串的使用}\\2.P(Q)\text{也可以是空集，即不是每一个字母都要有对应的状态}\end{cases}$$

4.$q_0 \in Q$ is the start state;

5.$F \subseteq Q$ is the set of accept states;

Formal definition of computation for an NFA

Let $N = (Q,\Sigma,\delta,q_0,F)$ be an NFA and let $w = y_1y_2...y_m$ be a string with $y_i \in \Sigma\epsilon$ for all i

$\in [m]$. Then N accepts w if a sequence of states $r_0,r_1,...,r_m$ in Q exists with:

- $r_0 = q_0$;
- $r_{i+1} \in \delta(r_i,y_{i+1})$ for $i = 0,...,m-1$;
- $r_m \in F$;

1.2.2 Equivalence of NFAs and DFAs(NFA与DFA的等价性)

(DFA~NFA)Theorem

Every NFA has an equivalent DFA, i.e., they recognize the same language.

Proof:[证明的主要想法是抽象化DFA，DFA的每一个新状态都是Q的一个子集]

Let N = (Q,Σ,δ,q_0,F) be the NFA recognizing some language A. We construct a DFA M = (Q',Σ',δ',q_0',F') recognizing the same A.

First assume N has no ϵ arrows:

1.Q' = P(Q);

2.Σ' = Σ;

3.$\delta'(R,a) = \{q \in Q | q \in \delta(r,a)$ for some $r \in R\}$, let $R \in Q'$ and $a \in \Sigma$;

4.$q_0' = \{q_0\}$;

5.$F' = \{R \in Q' \mid R \cap F \neq \emptyset\}$;

Now we allow ϵ arrows:

For every $R \in Q'$, i.e., $R \subseteq Q$ let

E(R) = { $q \in Q$ | q can be reached from R by traveling along 0 and more ϵ arrows};

1.Q' = P(Q);

2. Let $R \in Q'$ and a$\in \Sigma$. Then we define $\delta'(R,a) = \{q \in Q \mid q \in E(\delta(r,a))$,for some $r \in R\}$;
3. $q_0' = E(\{q_0\})$;
4. $F' = \{R \in Q' \mid R \cap F \neq \emptyset\}$;

(NFA~regular language)Corollary

A language is regular if and on if some nondeterministic finite automaton recognizes it.

1.2.3 Closure under the regular operation

Closure union:

For $i \in [2]$ let $N_i = (Q_i, \Sigma_i, \delta i, q_i, F_i)$ recognize A_i. We construct an $N = (Q, Sigma, \delta, Q_0, F)$ to recognize $A_1 \cup A_2$:

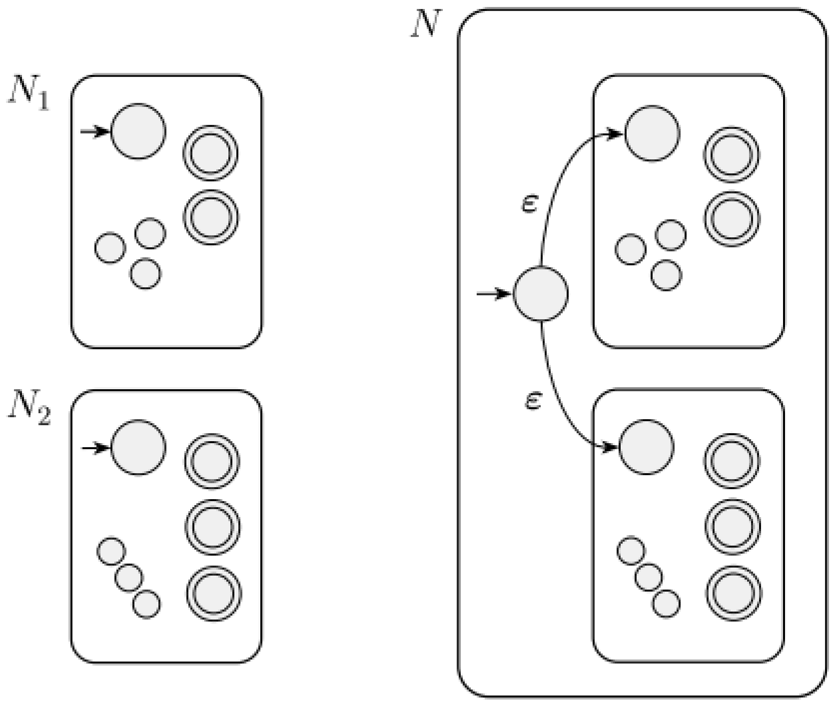
1.$Q = \{q_0\} \cup Q_1 \cup Q_2$;

2.q_0 is the start state;

3.$F = F_1 \cup F_2$;

4.For any $q \in Q$ and any $a \in \Sigma\epsilon$,

$$\delta(q,a) = \begin{cases} \delta_1(q,a) & q \in Q_1 \\ \delta_2(q,a) & q \in Q_2 \\ \{q_1, q_2\} & q = q_0 and a = \epsilon \\ \emptyset & q = q_0 and a \neq \epsilon \end{cases}$$



Closure concatenation:

For i $\in$ [2] let N_i = (Q_i,Σ_i,δi,q_i,F_i) recognize A_i. We construct an N = (Q,$Sigma$,δ,Q_0,F) to recognize $A_1 \circ A_2$:

construct N=(Q,Σ,δ,q_0,F) to recognize $A_1 \circ A_2$:

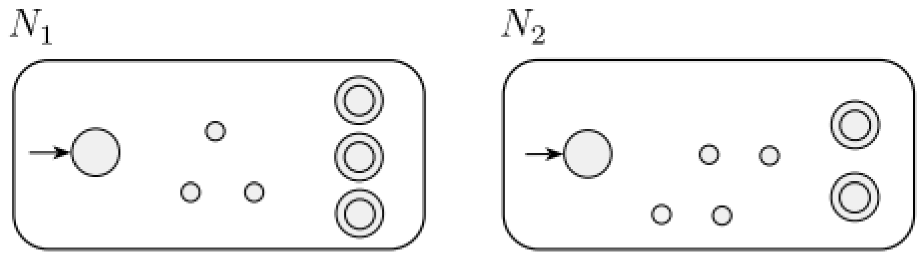
1.Q = $Q_1 \cup Q_2$

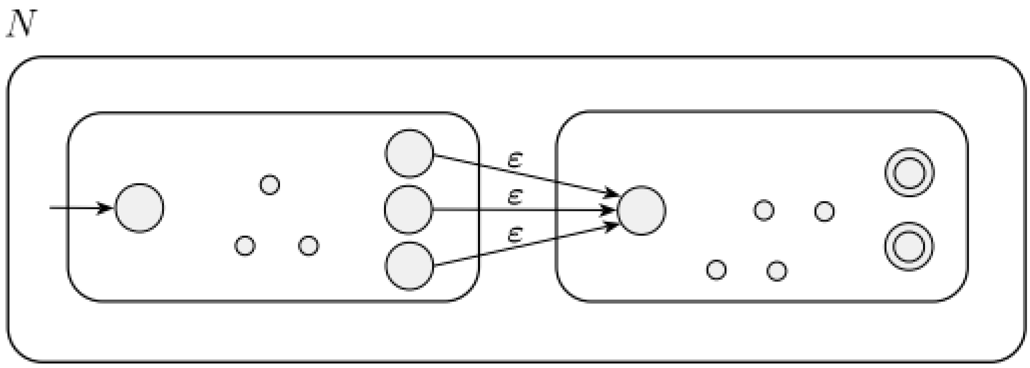
2.Σ = $\Sigma_1 + \Sigma_2$

3.q_0 = q_1

4.F = F_2

$$5.\delta(q,a) = \begin{cases} \delta_1(q,a) & q \in Q_1 and q \notin F_1 \\ \delta_2(q,a) & q \in Q_2 \\ \{q_2\} \cup \delta_1(q,a) & q \in F_1 and a = \epsilon \\ q_1 & q = q_0 and a = \epsilon \\ \delta_1(q,a) & q \in Q_1 and a \neq \epsilon \end{cases}$$





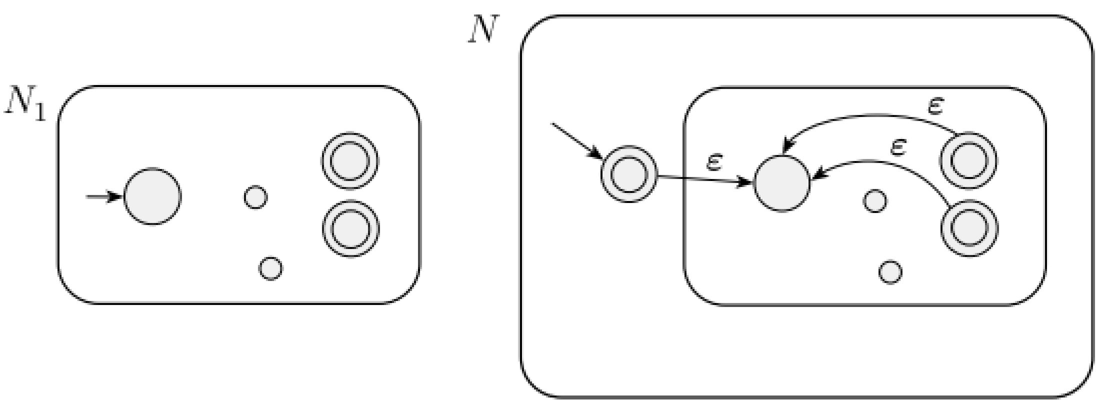
Closure star:

Let $N_1=(Q_1,\Sigma,\delta_1,q_1,F_1)$ recognizes A_1,construct $N=(Q,\Sigma,\delta,q_0,F)$ recognizes A_1*.

1.$Q = Q_1 \cup \{q_0\}$

2.$F = F_1 \cup \{q_0\}$

$$3.\delta(q,a) = \begin{cases} \{q_1\} & q = q_0 and a = \epsilon \\ \delta_1(q,a) & q \in Q_1 and a \neq \epsilon \\ \{q_1\} & q \in F_1 and a = \epsilon \\ \emptyset & q = q_0 and a \neq \epsilon \end{cases}$$



1.3 Regular expressions(正则表达式)

1.3.1 Definition of regular expression

R is a regular expression if R is

1.a for some a in the alphabet Σ

2.ϵ

3.$\emptyset$

4.($R_1 \cup R_2$), where $R_1$and R_2 are regular expressions

5.$R_1 \circ R_2$， where $R_1$and R_2 are regular expressions

6.R_1*, where R_1 are regular expressions

regular expression R	language $L(R)$
a	$\{a\}$
ε	$\{\varepsilon\}$
$\emptyset$	$\emptyset$
$(R_1 \cup R_2)$	$L(R_1) \cup L(R_2)$
$(R_1 \circ R_2)$	$L(R_1) \circ L(R_2)$
(R_1^*)	$L(R_1)^*$

1.3.2 Equivalence with finite automata(与有穷自动机的等价性)

Theorem: A language is regular if and only if some regular expression describes it

Definition of a generalized nondeterministic finite automaton(广义非确定型有穷自动机的定义)

A GNFA is a 5-tuple($Q,\Sigma,\delta,q_{start},q_{accept}$),where

1.Q is the finite set of states

2.Σ is the input alphabet

3.δ : (Q-{q_{accept}})$\times$(Q-{q_{start}})$\rightarrow$R is transition function

[GNFA的特点：任意q$\in$Q-{q_{accept},q_{start}}，q有到除起始状态外所有其他状态的箭头，q_{start}只出不进，q_{accept}只进不出]

A GNFA accept a string w in Σ^* if w=$w_1 w_2 \cdots w_k$, where w_i is in Σ^*, and a sequence of states $q_0, q_1, \cdots, q_k$ exists such that:

1.$q_0 = q_{start}$

2.$q_k = q_{accept}$

3.for each i, $w_i \in L(R_i)$, where $R_i=\delta(q_{i-1},q_i)$

1.4 Nonregular languages(非正则语言)

Pumping Lemma(泵引理)

If A is a regular language, then there is a number p(the pumping length) where if s is any string in A of length at least p, then s may be divided into three pieces,s=xyz,satisfying that :

1.for each i $\geq$0,$xy^iz \in A$

2.|y|>0

3.$|xy| \leq p$

[注意]

1.若A中没有长度大于p的字符串，则A是正则语言，因为A中元素都是有限长，可以通过穷举构造有穷自动机

2.|y|>0为了限制条件1

3.条件3是在证明语言的非正则性时有时可用

Proof：

由于A是正则语言，存在有穷自动机M(Q,Σ,δ,q_0,F)识别A，取p=|Q|<∞;

1.若s∈A,|s|≥p,则计算s时遍历的状态数n=|s|+1>|Q|：$q_0, q_1, \cdots, q_n \in F$,由鸽巢原理，必存在$q_i$=$q_j$(i≤j),取x满足$\delta$($q_0$,x)依次遍历，$q_0, q_1, \cdots, q_i$,y满足$\delta$($q_i$,y)依次遍历，$q_{i+1}$,$q_{i+2}$,$\cdots$,$q_j$,z满足$\delta$($q_j$,y)依次遍历$q_{j+1}$,$q_{j+2}$,$\cdots$,$q_n$;则|y|>0，满足条件2

2.y_i：$q_i \to q_j$,满足条件1

3.由于一共有p个状态，则|xy|≤p

[我们常用pumping lemma证明一个语言是非正则的，用反证法，假设一共语言是正则的，设泵长度是p，找到一个不满足pumping lemma的字符串]