Finite Automata

Finite Automata

Finite automata is a formal system with only a finite amount of information

- Information represented by its **state**
- State changes in response to **inputs**
- Rules that tell how the state changes in response to inputs are called **transitions**

FA 的 acceptance：对一个输入的序列（input string），从起始状态开始，并按照 transition 的规则转换状态，一个输出如果被接受（**accepted**）当且仅当所有输入被读入后 FA 停留在终止状态

Language of an Automata: The set of strings accepted by an automata A is the language of A , denoted $L(A)$

不同的终止状态的集合会带来不同的 language

Deterministic Finite Automata

Alphabets, string and language

Alphabet: any finite set of symbols

String: a string over an alphabet Σ is a list, each element of which is a member of Σ

Σ^* : set of all strings over alphabet Σ

Language: a language is a subset of Σ^* for some alphabet Σ

DFA

> A DFA is represented formally by a 5-tuple, $(Q, \Sigma, \delta, q_0, F)$ ， consisting of
>
> - a finite set of states Q
> - a finite set of input symbols Σ
> - a transition function $\delta : Q \times \Sigma \to Q$
> - an initial state $q_0 \in Q$

> - a set of states F distinguished as final states $F \in Q$

transition function takes two arguments: a state and an input symbol

更为严谨的定义需要 transition function δ 为 total function，即对任意一组状态和输入，其输出都是有定义的。但一般情况下遇到输出未定义的情况，可认为 DFA 停机

DFA 也可以以图的形式表示

- 节点=状态
- 边=transition function
- 无源边 start 指向初始状态
- 接收状态用 double circles 表示

或者以 transition table 的形式表示

- 行为状态
- 列为输入
- 起始状态用箭头标注
- 接收状态用 * 标注

Extend transition function 接受一个 state 和一个 string 作为输入，递归定义如下

Basis. $\delta(q, \epsilon) = q$

Induction. $\delta(q, wa) = \delta(\delta(q, w), a)$

> Extend transition function 与 transition function 不做区分
>
> $$\hat{\delta}(q, a) = \delta(\hat{\delta}(q, \epsilon), a) = \delta(q, a)$$

Language of DFA

各种各样的 Automata 都定义语言，对于 DFA A，其定义的语言的形式化定义如下

$$L(A) = \{w : \delta(q_0, w) \in F\}$$

Regular language: a language is regular if it is the language accepted by some DFA

> Example: A Nonregular Language
>
> $L = \{0^n 1^n : n \geqslant 1\}$
>
> 使用反证法，假设存在一个 DFA 接受该语言，该 DFA 有 m 个状态。考虑字符串 $0^m 1^m$
>
> 则必然存在从起始状态到接收状态的路径

> $$(q_0, 0^m 1^m) \to (q_1, 0^{m-1} 1^m) \to \dots (q_m, 1^m) \to \dots \to (q_{2m})$$
>
> 考虑前 m 次 transition，有 $m+1$ 个 state，根据 PHP，必然有一个状态至少出现两次，假设其为 q
>
> $q_i = q_j = q, i < j$
>
> 则路径变为
>
> $$(q_0, 0^m 1^m) \to (q_1, 0^{m-1} 1^m) \to \dots \to (q, 0^{m-i} 1^m) \to \dots \to (q, 0^{m-j} 1^m) \to \dots \to (q_{2m})$$
>
> 则该 DFA 同样可接受 $0^{m-j+i}1^m$ ，矛盾

Nondeterministic Finite Automata

Nondeterminism

NFA 的 transition 结果可以是一个状态的集合

> An NFA is represented formally by a 5-tuple, $(Q, \Sigma, \delta, q_0, F)$ ，consisting of
>
> - a finite set of states Q
> - a finite set of input symbols Σ
> - a transition function $\delta : Q \times \Sigma \to P(Q)$
> - an initial state $q_0 \in Q$
> - a set of states F distinguished as final states $F \in Q$

对于 NFA， $\delta(q, a)$ 的输出是一个状态的集合。其 Extend 的递归定义

Basis. $\delta(q, \epsilon) = \{q\}$

Induction. $\delta(q, wa) = \bigcup_{p \in \delta(q,w)} \delta(p, a)$

对于 NFA A，其定义的语言如下

$$L(A) = \{w : \delta(q_0, w) \cap F \neq \varnothing\}$$

Equivalence of DFA, NFA

DFA to NFA

A DFA can be turned into an NFA that accepts the same language:

If $\delta_D(q, a) = p$, let the NFA have $\delta_N(q, a) = \{p\}$

NFA to DFA: subset construction

从 NFA 构造 DFA 可使用 subset construction，设 NFA 有状态 Q， 输入字母表 Σ ，转换函数 δ_N ，开始状态 q_0 和接收状态集 F

则与其等价的 DFA 有状态 2^Q ，输入字母表 Σ ，开始状态 $\{q_0\}$ ，以及接收状态集 $\{S : S \in 2^Q \text{ and } S \cap F \neq \varnothing\}$

Critical Point: DFA 的状态为 NFA 状态的集合

$$\delta_D(\{q_1, q_2, \ldots, q_k\}, a) = \bigcup_{i=1}^{k} \delta_N(q_i, a)$$

证明其正确性只需证明对字符串 w ，有

$$\delta_N(q_0, w) = \delta_D(\{q_0\}, w)$$

对 w 的长度归纳即可

Basis. $w = \epsilon$

$$\delta_N(q_0, \epsilon) = \delta_D(\{q_0\}, \epsilon) = \{q_0\}$$

I. H. 对比 w 短的字符串，命题成立

Ind. Step. 令 $w = xa$ ，则 $\delta_N(q_0, x) = \delta_D(\{q_0\}, x) = S$

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

则 $\delta_N(q_0, w) = T = \delta_D(S, a) = \delta_D(\{q_0\}, w)$

NFA with ϵ-transitions

允许状态间根据 ϵ 转换

定义 Closure of states:

- $CL(q)$ = set of states you can reach from state q following only arcs labeled ϵ
- $CL(S) = \bigcup_{q \in S} CL(q)$

在 ϵ-NFA 上可定义扩展的转换函数 $\hat{\delta}(q, w)$

Basis. $\hat{\delta}(q, \epsilon) = CL(q)$

Induction. $\hat{\delta}(q, xa) = \bigcup_{p \in \hat{\delta}(q, x)} CL(\delta(p, a))$

ϵ-NFA A 定义的 language 即为

$$L(A) = \{w : \hat{\delta}(q_0, w) \cap F \neq \varnothing\}$$

Equivalence of NFA, ϵ-NFA

Obviously, every NFA is an ϵ-NFA

可从一个 ϵ-NFA 构造一个接受同样语言的 NFA: remove ϵ-transitions

设一个 ϵ-NFA 有状态 Q，输入字母表 Σ，开始状态 q_0，接收状态集 F，转换函数 δ_E

构造一个 NFA 有状态 Q，输入字母表 Σ，开始状态 q_0，接收状态集 F'，转换函数 δ_N

$$\delta_N(q,a) = \bigcup_{p \in CL(q)} \delta_E(p,a)$$

即从状态 q 开始，做一次 CL，做一次 a 的转换

同理

$$F' = \{q : CL(q) \cap F \neq \varnothing\}$$

证明其正确性只需证明

$$CL(\delta_N(q_0,w)) = \hat{\delta}_E(q_0,w)$$

根据 w 的长度归纳证明即可