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

Definition

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， $F \subseteq Q$

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 Extended 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

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

即只要存在一条运行路径结束于接收状态即可认为接受该 string

NFA with ϵ-transitions

允许状态间根据 ϵ 转换的 NFA

定义 Closure of states:

- $CL(q)$ = set of states you can reach from state q following only arcs labeled ϵ
  - Basis. $q \in CL(q)$
  - Induction. 如果 $p \in CL(q)$ 且有一条从 p 到 r 的边标号为 ϵ ， 则 $r \in CL(q)$
- $CL(S) = \bigcup_{q \in S} CL(q)$

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 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

对于 NFA $N=(Q_N,\Sigma,\delta_N,q_0,F_N)$，目标是构造一个 DFA $D=(Q_D,\Sigma,\delta_D,\{q_0\},F_D)$ 满足 $L(D)=L(N)$

> Proof.
>
> D 的开始状态为一个集合，其中唯一的元素是 N 的开始状态，且由于两者接受相同的语言，故 D 与 N 的 alphabet 相同，其余部分的构造如下
>
> - Q_D 是 Q_N 的 power set，如果 $|Q_N|=n$ 那么 $|Q_D|=2^n$，但实践中一般很多状态都是不可达的，可达状态的数量大约和 n 在同一数量级
> - F_D 是满足 $S\subseteq Q_N\wedge S\cap F_N\neq\varnothing$ 的 S 的集合
> - δ_D 的定义如下，对于任意 $S\subseteq Q_N,a\in\Sigma$
> $$\delta_D(S,a)=\bigcup_{p\in S}\delta_N(p,a)$$

Critical Point: DFA 的状态为 NFA 状态的**集合**

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. $w=\epsilon$
>
> $$\delta_N(q_0,\epsilon)=\delta_D(\{q_0\},\epsilon)=\{q_0\}$$
>
> Induction. 令 $w=xa$，根据 I. H. 有
>
> $$\delta_N(q_0,x)=\delta_D(\{q_0\},x)=S$$
>
> 令 $T=\bigcup_{p\in S}\delta_N(p,a)$

> 根据 NFA transition function 的定义，有
>
> $$\hat{\delta}_N(q_0, w) = \hat{\delta}_N(q_0, xa) = \bigcup_{p \in \hat{\delta}_N(q_0, x)} \delta_N(p, a) = T$$
>
> 而根据上述证明的构造有
>
> $$\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a) = T$$
>
> 同样的，根据 DFA transition function 的定义，有
>
> $$\hat{\delta}_D(\{q_0\}, w) = \hat{\delta}_D(\{q_0\}, xa) = \delta_D(\hat{\delta}_D(\{q_0\}, x), a) = \delta_D(S, a) = T$$
>
> 则 $\hat{\delta}_N(q_0, w) = T = \hat{\delta}_D(\{q_0\}, w)$
>
> 故
>
> $$\begin{aligned} w \in L(N) &\iff \hat{\delta}_N(q_0, w) \cap F_N \neq \varnothing \\ &\iff \hat{\delta}_D(\{q_0\}, w) \cap F_N \neq \varnothing \\ &\iff \hat{\delta}_D(\{q_0\}, w) \in F_D \\ &\iff w \in L(D) \end{aligned}$$
>
> 得证 $L(N) = L(D)$

由上述双向的等价可得

L 被一个 DFA 接受 $\iff$ L 被一个 NFA 接受

Equivalence of DFA, ϵ-NFA

DFA to ϵ-NFA

Obviously, every DFA is an ϵ-NFA

ϵ-NFA to DFA

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

对于一个 ϵ-NFA $E = (Q_E, \Sigma, \delta_E, q_0, F_E)$ ，目标是构造一个 DFA $D = (Q_D, \Sigma, \delta_D, q_D, F_D,)$ 使得 $L(E) = L(D)$

> Proof.
>
> D 与 E 接受相同语言，有相同的 alphabet，其余部分的构造为
>
> - Q_D 是 Q_E 的 power set，且任意 $S \in Q_D$ 满足 $S = CL(S)$ ，换言之，S 是一个 ϵ-closed set

> - $q_D = CL(q_0)$
> - F_D 是满足 $S \in Q_D \wedge S \cap F_E \neq \varnothing$ 的 S 的集合
> - $\delta_D(S, a)$ 的定义为，对于任意 $a \in \Sigma, S \in Q_D$
>
> $$T = \bigcup_{p \in S} \delta_E(p, a)$$
> $$\delta_D(S, a) = CL(T)$$

证明其正确性只需证明对任意字符串 w 满足

$$\delta_E(q_0, w) = \delta_D(q_D, w)$$

根据 w 的长度归纳证明即可