

Lecture 2: Finite Automata and Regular Languages

Lecturer: Renjie Yang

2.1 Finite automata

Definition 2.1 A finite automaton is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set called the states,
2. Σ is a finite set called the alphabet,
3. $\delta : Q \times \Sigma \rightarrow Q$ is the transition function,
4. $q_0 \in Q$ is the start state,
5. $F \subseteq Q$ is the set of accept states.

Example

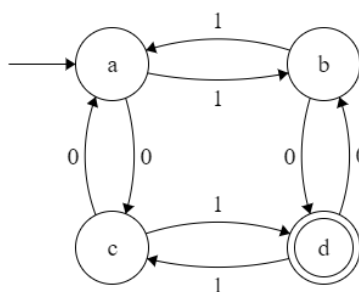


Figure 2.1: FSM example

$$Q = \{a, b, c, d\}$$

$$\Sigma = \{0, 1\}$$

$$q_0 = a$$

$$F = \{d\}$$

$$\sigma = \begin{matrix} & \begin{matrix} 0 & 1 \end{matrix} \\ \begin{matrix} a \\ b \\ c \\ d \end{matrix} & \begin{bmatrix} c & b \\ d & a \\ a & d \\ b & c \end{bmatrix} \end{matrix}$$

Definition 2.2 Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton and let $w = w_1 w_2 \dots w_n$ be a string where each w_i is a member of the alphabet Σ . Then M accepts w if a sequence of states $r_0, r_1, \dots, r_n \in Q$ exists with three conditions:

1. $r_0 = q_0$
2. $\delta(r_i, w_{i+1}) = r_{i+1}$, for $i = 0, \dots, n-1$

3. $r_n \in F$

We say that M recognizes language A if $A = \{w \mid M \text{ accepts } w\}$

Example The empty string ϵ . It is accepted by a FSM in which the start state is an accept state.

Example The empty language $\phi = \{\}$. It is recognized by a FSM with no directed path from start state to any accept state. If a FSM accepts no string, then it recognizes the empty language.

Example Let $\Sigma = \{0, 1\}$. Design a FSM that accepts any string containing 0011.

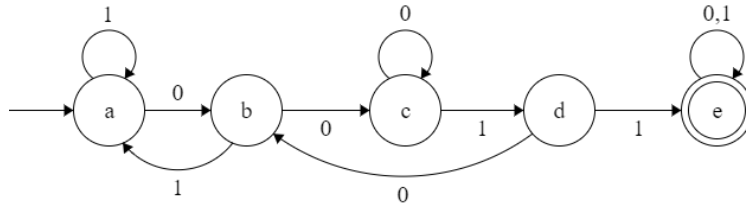


Figure 2.2: FSM example

Example The following FSM recognizes $\{w \mid w \text{ is either } 10 \text{ or } 0^+1\}$

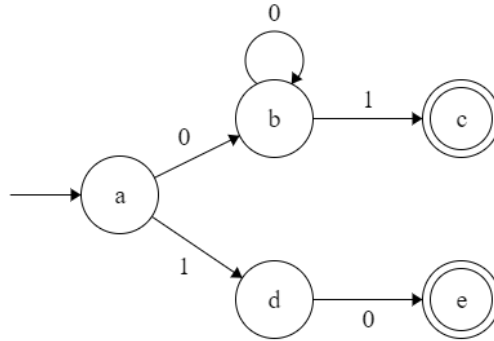


Figure 2.3: FSM example

Definition 2.3 A nondeterministic finite automaton is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

1. Q is a finite set called the states,
2. Σ is a finite set called the alphabet,
3. $\delta : Q \times \Sigma_\epsilon \rightarrow P(Q)$ is the transition function, where $\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$,
4. $q_0 \in Q$ is the start state,
5. $F \subseteq Q$ is the set of accept states.

Example Test 01010

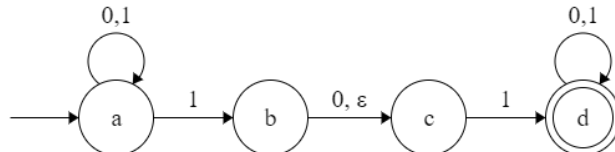


Figure 2.4: NFA example

$$Q = \{a, b, c, d\}$$

$$\Sigma = \{0, 1\}$$

$$q_0 = a$$

$$F = \{d\}$$

$$\delta = \begin{array}{c} \begin{array}{ccc} & 0 & 1 & \epsilon \\ \begin{array}{c} a \\ b \\ c \\ d \end{array} & \left[\begin{array}{ccc} \{a\} & \{a, b\} & \phi \\ \{c\} & \phi & \{c\} \\ \phi & \{d\} & \phi \\ \{d\} & \{d\} & \phi \end{array} \right] \end{array}$$

Definition 2.4 Let $M = (Q, \Sigma, \delta, q_0, F)$ be a nondeterministic finite automaton and let $w = y_1 y_2 \dots y_m$ be a string where each y_i is a member of the alphabet Σ_ϵ . Then M accepts w if a sequence of states r_0, r_1, \dots, r_m in Q exists with three conditions:

1. $r_0 = q_0$
2. $r_{i+1} \in \delta(r_i, w_{i+1})$, for $i = 0, \dots, n - 1$
3. $r_m \in F$

We say that M recognizes language A if $A = \{w \mid M \text{ accepts } w\}$

Example The following NFA accepts all strings that contain 0110.

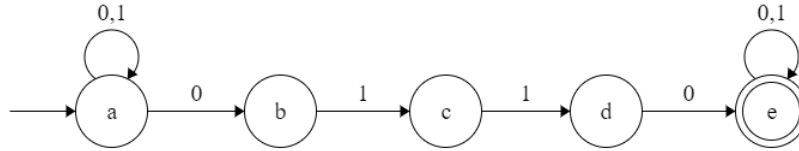


Figure 2.5: NFA example

Example The equivalence of DFA and NFA

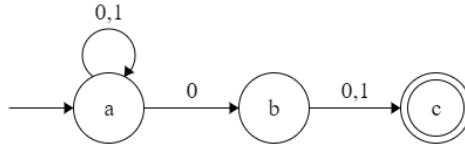


Figure 2.6: NFA example

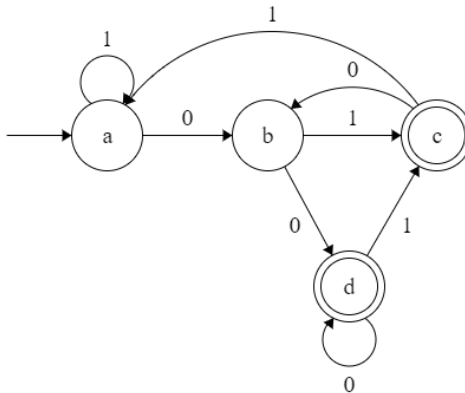


Figure 2.7: An equivalent DFA

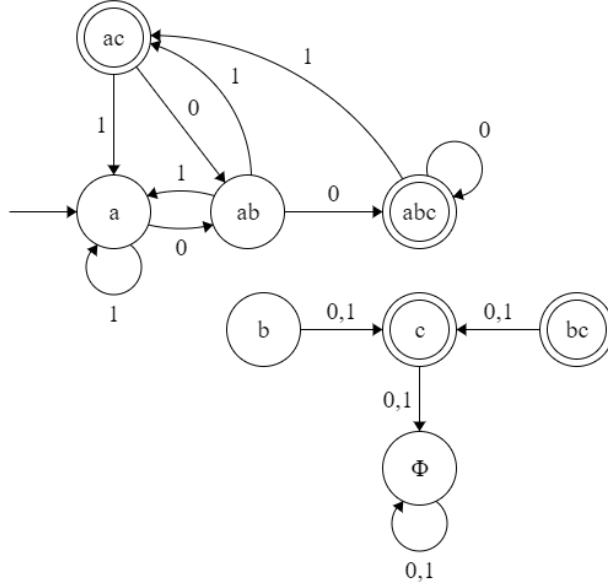


Figure 2.8: A more systematically constructed equivalent DFA

Theorem 2.5 *Every nondeterministic finite automaton has an equivalent deterministic finite automaton.*

Proof: Let $N = (Q, \Sigma, \delta, q_0, F)$ be the NFA recognizing some language A . Construct a DFA $M = (Q', \Sigma, \delta', q'_0, F')$ recognizing A as follows:

1. $Q' = P(Q)$
2. For $R \in Q'$ and $a \in \Sigma$, let

$$\delta'(R, a) = \bigcup_{r \in R} \delta(r, a) = \{q \in Q \mid q \in \delta(r, a) \text{ for some } r \in R\}$$

3. $q'_0 = \{q_0\}$
4. $F' = \{R \in Q' \mid R \text{ contains an accepted state of } N\}$

Consider the cases with ϵ arrows, define

$$E(R) = \{q \mid q \text{ can be reached from } R \text{ by traveling along 0 or more } \epsilon \text{ arrows}\}.$$

Then replace $\delta(r, a)$ by $E(\delta(r, a))$ in 2, and change q'_0 to $E(\{q'_0\})$ in 3.

At every step in the computation of M on an input, it enters a state that corresponds to the subset of states that N could be in at that point. ■

2.2 Regular languages

Definition 2.6 A language is called a regular language if some finite automaton recognizes it.

Corollary 2.7 A language is regular if and only if some nondeterministic finite automaton recognizes it.

Definition 2.8 Let A and B be languages. We define the regular operations union, concatenation, and star as follows:

- Union: $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$
- Concatenation: $A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$
- Star: $A^* = \{x_1x_2 \dots x_k \mid k \geq 0 \text{ and each } x_i \in A\}$

Example

Let $\sigma = \{a, b, c, \dots, z\}$

$A = \{aa, b\}$

$B = \{x, yy\}$

$A \cup B = \{aa, b, x, yy\}$

$A \circ B = \{aax, aayy, bx, byy\}$

$A^* = \{\epsilon, aa, b, aaaa, aab, baa, bb, aaaaaa, aaaab, aabaa, aabb \dots\}$

Theorem 2.9 The class of regular languages is closed under the union operation.

Proof: Let $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize A_1 , $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognize A_2 . Construct $M = (Q, \Sigma, \delta, q_0, F)$ to recognize $A_1 \cup A_2$.

1. $Q = \{(r_1, r_2) \mid r_1 \in Q_1, r_2 \in Q_2\}$
2. $\Sigma = \Sigma$ for both M , M_1 and M_2
3. For each $r_1, r_2 \in Q$ and each $a \in \Sigma$, let $\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$
4. $q_0 = (q_1, q_2)$
5. $F = \{(r_1, r_2) \mid r_1 \in F_1 \text{ or } r_2 \in F_2\}$

The construction of M simulates both M_1 and M_2 running on the same string simultaneously, which guarantees that it recognizes $A_1 \cup A_2$. ■

Theorem 2.10 The class of regular languages is closed under the concatenation operation.

Proof: Let $N_1 = \{Q_1, \Sigma, \delta_1, q_1, F_1\}$ recognize A_1 , and $N_2 = \{Q_2, \Sigma, \delta_2, q_2, F_2\}$ recognize A_2 , construct $N = \{Q, \Sigma, \delta, q_0, F\}$ to recognize $A_1 \circ A_2$ as follows:

1. $Q = Q_1 \cup Q_2$
2. $q_0 = q_1$
3. $F = F_2$

4. Define δ so that for any $q \in Q$ and any $a \in \Sigma_\epsilon$,

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

■

Theorem 2.11 *The class of regular languages is closed under the star operation.*

Proof: Let $N_1 = \{Q_1, \Sigma, \delta_1, q_1, F_1\}$ recognize A_1 , construct $N = \{Q, \Sigma, \delta, q_0, F\}$ to recognize A_1^* as follows:

1. $q = q_0$, a new start state
2. $Q = \{q_0\} \cup Q_1$
3. $F = \{q_0\} \cup F_1$
4. Define δ so that for any $q \in Q$ and any $a \in \Sigma_\epsilon$,

$$\delta(q, a) = \begin{cases} \delta_1(q, a), & q \in Q_1 \text{ and } q \notin F_1 \\ \delta_1(q, a), & q \in F_1 \text{ and } a \neq \epsilon \\ \delta_1(q, a) \cup \{q_1\}, & q \in F_1 \text{ and } a = \epsilon \\ \{q_1\}, & q = q_0 \text{ and } a = \epsilon \\ \phi, & q = q_0 \text{ and } a \neq \epsilon \end{cases}$$

■

2.3 Regular Expressions

Definition 2.12 *Say that R is a regular expression if R is*

1. a for some a in the alphabet Σ ,
2. ϵ ,
3. ϕ ,
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 is a regular expressions.

Example Each regular expression describes a language. Assume $\Sigma = \{a, b, c, d\}$

- a
- $abccb$
- $(ab) \cup (cd) = ab|cd$
- ab^*c
- $a(b \cup \epsilon)c = a(b|\epsilon)c = a[b]c$
- ϕ
- $a(b \cup c)\phi$
- ϕ^*

Theorem 2.13 *A language is regular if some regular expression describes it.*

Proof: Consider the 6 cases in the definition of regular expression:

1. $N = (\{q_1, q_2\}, \Sigma, \delta, q_1, \{q_2\})$, where $\delta(q_1, a) = \{q_2\}$ and $\delta(r, b) = \phi$ for $r \neq q_1$ or $b \neq a$.
2. $N = (\{q_1\}, \Sigma, \delta, q_1, \{q_1\})$, where $\delta(r, b) = \phi$ for any r and b .
3. $N = (\{q\}, \Sigma, \delta, q, \phi)$, where $\delta(r, b) = \phi$ for any r and b .

4-6 are shown in the previous section. Thus we have converted a regular expression into an NFA. ■