University of Warwick
Department of Computer Science

CS255

Formal Languages

Cem Yilmaz

January 20, 2023

Contents

1 Languages

1.1 Alphabet

Definition 1.1. Alphabet

Alphabet is a non-empty finite set of symbols. For example,

$$\Sigma_1 = \{a, b, c\} \qquad \Sigma_2 = \{5, 8, 10\} \tag{1}$$

Are alphabets which contain those symbols.

Definition 1.2. Language

Language is a potentially infinite set of finite strings over an alphabet. Using our previous alphabets, for example, we could obtain

$$L_1 = \{ab, abc, aaab, ccc, ba\} \qquad L_2 = \{a, aa, aaa, aaaa, \ldots\} \tag{2}$$

We could further define

$$\Sigma^* = \{\text{ALL finite strings (also called words) over the alphabet } \Sigma\} \tag{3}$$

1.2 Deterministic Final State Automata

Definition 1.3. Deterministic Finate State Automaton

A machine M defined by the tuple

$$M = (Q, \Sigma, q_0, F, \delta) \tag{4}$$

is called the deterministic finite state automaton or a deterministic finite state machine. The Q refers to states, Σ to the alphabet, q_0 to the initial/starting state, F to the final state and δ as the state transition.

To represent a deterministic finite state automaton, consider the following diagram:

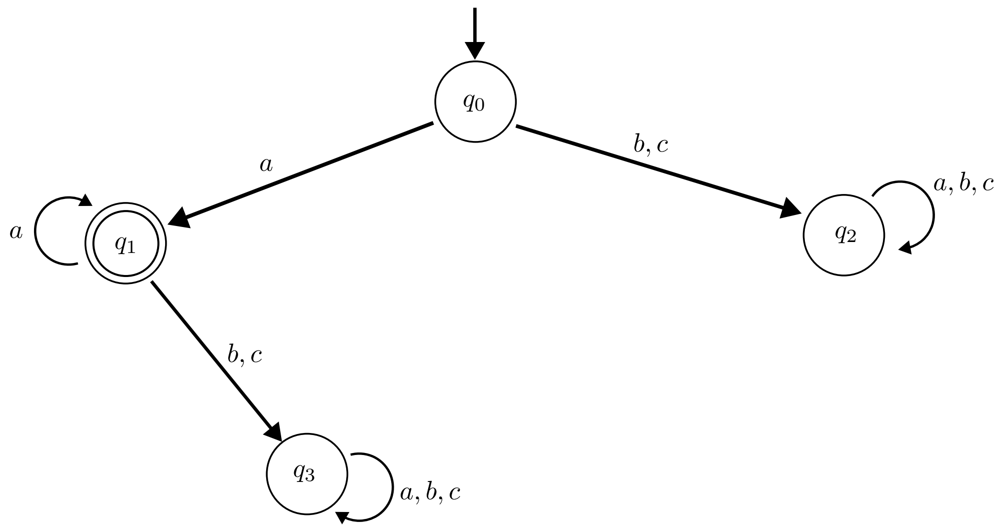

Figure 1: Exemplar deterministic finite state automata state diagram

Which represents the following table of values:

Table 1: State Transition Table

δ	a	b	c
q_0	q_1	q_2	q_2
q_1	q_1	q_3	q_3
q_2	q_2	q_2	q_2
q_3	q_3	q_3	q_3

In other words, if the input string finishes at q_1, we accept the input. If it finishes in any other node otherwise, reject.

Definition 1.4. The Empty Word

The length of $|\varepsilon| = 0$. The Language L_1 is the empty language, $L_2 = \{\varepsilon\}$ is a non-empty language. Note that Σ^* always contains ε. The role that of the empty string is to be a monoid in our system.

Definition 1.5. Monoid

Comprise of a set, an associative binary operation on the set with an identity element.

$$(\mathbb{N}_0, +, 0) \text{ is a monoid. Here, } + \text{ denotes addition.} \tag{5}$$
$$(\mathbb{N}, \times, 1) \text{ is a monoid. Here, } \times \text{ is multiplication.} \tag{6}$$
$$(\Sigma^*, \circ, \varepsilon) \text{ is a monoid. Here, } \circ \text{ denotes string concatenation.} \tag{7}$$

Definition 1.6. Transition Function

The transition function is denoted as

$$\delta(q_i, \text{string}) = q_j \tag{8}$$

k In other words, we take a state q_i, a string input, and after running the string, we get output state q_j. Note that the string can be a single letter or a bigger string. In case of a non-letter string, sometimes the δ is denoted as $\hat{\delta}$ instead. Formally,

$$\hat{\delta} : Q \times \Sigma^* \to Q \tag{9}$$

such that

$$\forall q \in Q, \hat{\delta}(q, \varepsilon) = q \tag{10}$$
$$\forall q \in Q \wedge s \in \Sigma^* \text{ s.t. } s = wa \text{ for some } w \in \Sigma^* \wedge a \in \Sigma, \hat{\delta}(q, s) = \delta(\hat{\delta}(q, w), a) \tag{11}$$

Definition 1.7. Language accepted by DFA

Consider a DFA $M = (Q, \Sigma, q_0, F, \delta)$. The language accepted or recognised by M is denoted by $L(M)$ and is defined as

$$L(M) = \{s \in \Sigma^* | \hat{\delta}(q_0, s) \in F\} \tag{12}$$

Definition 1.8. Run of a DFA

Consider a DFA $m = (Q, \Sigma, q_0, F, \delta)$. Consider a string $s = s_1 s_2 \ldots s_n$, where $s_i \in \Sigma$ for each $i \in [n]$. The run of M on the empty word ε is just the state q_0. The run of M on the word s is a sequence of states $r_0, r_1, \ldots, r_n$, where

$$r_0 = q_0 \tag{13}$$
$$\forall i \in [n], r_i = \delta(r_{i-1}, s_i) \tag{14}$$

1.3 Languages

Definition 1.9. Regular Language

A language L is called regular if it is accepted by some deterministic finite state automata (DFA)

Definition 1.10. NFA

Formally, the extended transition $\hat{\delta}$ for an NFA $(Q, \Sigma, q_0, F, \delta)$ is a function $\hat{\delta} : Q \times \Sigma^* \to \mathbb{P}(Q)$ and is defined as follows:

$$\forall q \in Q, \hat{\delta}(q, \varepsilon) = ECLOSE(q) \tag{15}$$
$$\forall q \in Q \land s \in \Sigma^* : s = wa \text{ for some } w \in \Sigma^* \land a \in \Sigma, \hat{\delta}(q, s) = ECLOSE(\cup_{q' \in \hat{\delta}(q,w)} \delta(q', a)) \tag{16}$$

It is useful to first compute the ε closure of an input and then consider the input string to see where it possible leads, repeating the process.

Corollary. *Language of NFA*

Consider an NFA $M = (Q, \Sigma, q_0, F, \delta)$. The run of M on the word s is a sequence of states $r_0, r_1, \ldots, r_n$ such that

$$r_0 = q_0 \tag{17}$$
$$\exists s_1, s_2, \ldots, s_n \in \Sigma \cup \{\varepsilon\} \text{ such that } s = s_1 s_2 \ldots s_n \land \forall i \in [n], r_i \in \delta(r_{i-1}) \in \delta(r_{i-1}, s_i) \tag{18}$$