Models of computation

Gabriel Le Dain

March 2025

1 Finite automata

1.1 Finite automata

Definition 1.1 (Finite automaton)
A **finite automaton**, or **fintite state machine** (sometimes just **state machine**) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where

(i) Q is a finite set called the *states*

(ii) Σ is a finite set called the *alphabet*

(iii) $\delta : Q \times \Sigma \to \Sigma$ is the *transition function*

(iv) $q_0 \in Q$ is the *start state*

(v) $F \subseteq Q$ is the set of *final states* or *accept states*

Notation 1.2 (Words over an alphabet)
Let Σ be a set. The set of all words over Σ is denoted by Σ^*. This set of words always contains the empty string.

Notation 1.3 (Augmentation of an alphabet)
For any alphabet Σ, we define the **augmentation** of Σ as

$$\Sigma_\varepsilon := \Sigma \cup \{\varepsilon\}$$

where ε is a formal symbol not in Σ. We think of ε as the empty string.

Definition 1.4 (Augmentation map)
For any alphabet Σ, we define the augmentation map by

$$\text{aug} : \Sigma_\varepsilon * \to \Sigma^*$$

$$\text{aug}(\varepsilon) = \text{the empty string}$$

$$\text{aug}(a) = a \text{ for all } a \in \Sigma$$

extended to words over Σ in the obvious way.

Definition 1.5 (Language)
A language over a set Σ is some subset of Σ^*

Definition 1.6 (Accept)
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automation and $w = w_1 w_2 \dots w_n$ be a word over Σ. We say that M **accepts** w if there exist $r_0, r_1, \dots, r_n$ in Q such that

(i) $r_0 = q_0$

(ii) $r_{i+1} = \delta(r_i, w_{i+1})$ for all $i = 0, 1, \dots, n-1$

(iii) $r_n \in F$

Definition 1.7 (Recognise)
Let $M = (Q, \Sigma, \delta, q_0, F)$ be a finite automaton. We say that M **recognises** the language

$$L(M) := \{w \in \Sigma^* : M \text{ accepts } w\}$$

1.2 Regular languages

Definition 1.8 (Regular language)
We say that a language A over a set Σ is a **regular language** if there exists some finite automation M with Σ as its alphabet such that M recognises A.

Definition 1.9
Non-deterministic finite automaton A **non-deterministic finite automaton** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where

(i) Q is a finite set called the *states*

(ii) Σ is a finite set called the *alphabet*

(iii) $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ is the *transition function*

(iv) $q_0 \in Q$ is the *start state*

(v) $F \subseteq Q$ is the set of *final states* or *accept states*

Definition 1.10 (Accept)
et $M = (Q, \Sigma, \delta, q_0, F)$ be a non-deterministic finite automaton and $w = w_1 w_2 \dots w_n$ be a word over Σ. We say that M **accepts** w if there exist $r_0, r_1, \dots, r_n$ in Q such that

(i) $r_0 = q_0$

(ii) $r_{i+1} \in \delta(r_i, w_{i+1})$ for all $i = 0, 1, \dots, n-1$

(iii) $r_n \in F$

Definition 1.11 (ε-Non-deterministic finite automaton)
An **ε-non-deterministic finite automaton** is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where

(i) Q is a finite set called the *states*

(ii) Σ is a finite set called the *alphabet*

(iii) $\delta : Q \times \Sigma_\varepsilon \to \mathcal{P}(Q)$ is the *transition function*

(iv) $q_0 \in Q$ is the *start state*

(v) $F \subseteq Q$ is the set of *final states* or *accept states*

Definition 1.12 (Accept)
Let $M := (Q, \Sigma, \delta, q_0, F)$ be an ε-nondeterministic finite automaton. We say that M accepts a word $w \in \Sigma^*$ if there exists $w' := w_0 w_1 \ldots w_n \in \Sigma_\varepsilon^*$ and $r_0, r_1, \ldots, r_n \in Q$ such that $\text{aug}(w') = w$ and

(i) $r_0 = q_0$

(ii) $r_{i+1} \in \delta(r_i, w_{i+1})$ for all $i = 0, 1, \ldots, n-1$

(iii) $r_n \in F$

Definition 1.13 (Accept)
Let $M := (Q, \Sigma, \delta, q_0, F)$ be an ε-nondeterministic finite automaton. We say that M accepts a word $w \in \Sigma^*$ if there exists $w := w_1 w_2 \ldots w_n \in \Sigma^*$, $k_1, k_2, \ldots, k_n \in \mathbb{N}$ and $r_0, \ldots, r_n \in Q$ such that

(i) $r_0 = q_0, k_1 = 1$

(ii) For all $i = 0, 1, \ldots, n-1$, we have one of:

  (a) $r_{i+1} \in \delta(r_i, w_{k_{i+1}})$ and $k_{i+1} = k_i + 1$

  (b) $r_{i+1} \in \delta(r_i, \varepsilon)$ and $k_{i+1} = k_i$

(iii) $r_n \in F$

Remark
The definitions 1.13 and 1.12 are equivalent by thinking about them for a second or two. However 1.13 is more suited to implementation in a programming language.

Definition 1.14 (Regular operations)
Let Σ be some alphabet. We define three **regular operations** on languages:

(i) Union: $A \cup B$

(ii) Concatenation: $A \circ B := \{ab : a \in A, b \in B\}$

(iii) Kleene star: $A^* := \{x_a x_2 \ldots x_n \text{ where } x_1, x_2, \ldots, x_n \in A\}$

Note 1.15
Note that the definition of the Kleene star on a language above is compatible with its use to denote the set of words over some alphabet.

Proposition 1.16 (Unions of regular languages are regular)
Let A, B be regular languages over the same alphabet Σ. Then $A \cup B$ is also regular.

Proof. Since A, B are regular, they are recognised by finite automata $M^1 := (Q^1, \Sigma, \delta^1, q_0^1, F^1)$ and $M^2 := (Q^2, \Sigma, \delta^2, q_0^2, F^2)$. Define $M^3 := (Q^3, \Sigma, \delta^3, q_0^3, F^3)$ where:

- $Q^3 := Q^1 \times Q^2$
- $\delta^3((s^1, s^2), a) = (\delta^1(s^1, a).\delta^2(s^2, a))$
- $q_0^3 = (q_0^1, q_0^2)$

- $F^3 := \left(F^1 \times Q^2\right) \cup \left(Q^1 \times F^2\right)$

Then M_3 recognises $A \cup B$. $\square$

Proposition 1.17 (Intersections of regular languages are regular)
Let A, B be regular languages over the same alphabet Σ. Then $A \cap B$ is also regular.

Proof. Since A, B are regular, they are recognised by finite automata $M^1 := (Q^1, \Sigma, \delta^1, q_0^1, F^1)$ and $M^2 := (Q^2, \Sigma, \delta^2, q_0^2, F^2)$. Define $M^3 := (Q^3, \Sigma, \delta^3, q_0^3, F^3)$ where:

- $Q^3 := Q^1 \times Q^2$
- $\delta^3((s^1, s^2), a) = (\delta^1(s^1, a).\delta^2(s^2, a))$
- $q_0^3 = (q_0^1, q_0^2)$
- $F^3 := F^1 \times F^2$

Then M_3 recognises $A \cap B$. $\square$