CS 4510 Automata and Complexity January 10, 2023

Lecture 1: Introduction

Lecturer: Abrahim Ladha *Scribe(s): Akshay Kulkarni*

1 Introduction

CS 4510: Automata & Complexity Theory. This course is primarily the study of two questions:

1. What are the limits of computation? (Computability Theory: Approx. 70% of course)
2. What makes some problems easy and others hard? (Complexity Theory: Approx. 30% of course)

Fundamentally about theoretical computer science—what is a computer and how can we effectively describe its limits and capacities? Automata are tools that we can use to reason about more powerful versions.

1.1 Formal Language Theory

Let Σ be a finite set of characters called the *input alphabet*.

Examples:

1. $\Sigma = \{a, b\}, \{1\}, \{0, 1\}, \{a, \ldots, z, A \ldots, Z\}$
2. $\Sigma^2 = \{aa, ab, ba, bb\}$ (strings of length 2)
3. $\Sigma^0 = \{\varepsilon\}$ (strings of length 0)
4. $\Sigma^* = \cup_{i=1}^{\infty} = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots$ (all strings)

A *language* is any subset of strings $L \subseteq \Sigma^*$

1.2 Automata

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

1. $Q \in \{q_0, \ldots, q_k\}$ is a finite set called the *states*,
2. Σ is the *alphabet*,

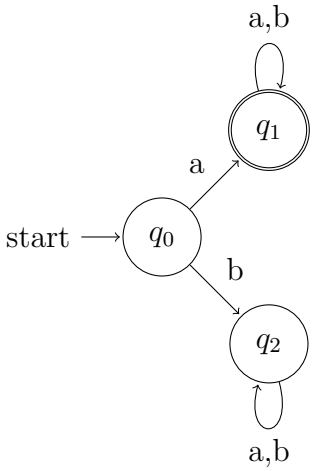
3. $\delta : Q \times \Sigma \to Q$ is the *transition function*,
4. $q_0 \in Q$ is the *start state*
5. $F \subseteq Q$ is the set of *acceptance* states.

An automata M decides a language L if:

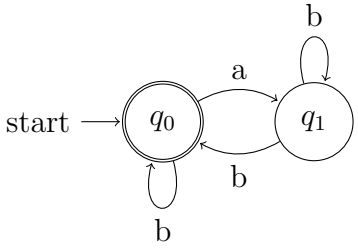
$$M(w) \textit{ accepts } L \iff w \in L \tag{1}$$
$$M(w) \textit{ rejects } L \iff w \notin L \tag{2}$$

Examples:

1. $L = \{w \in \Sigma^* \mid w \text{ begins with } a\}$



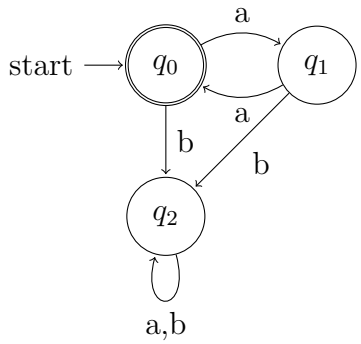
2. $L = \{w \in \Sigma^* \mid \#a(w) \text{ is even}\}$



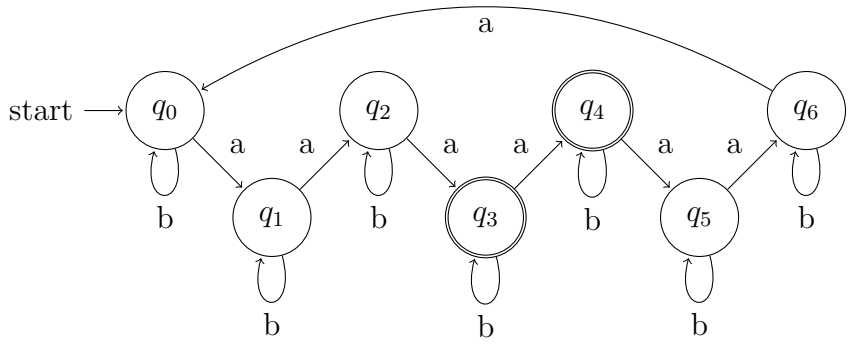
A decision problem is phrased as any yes/no question.

Examples:

1. $L = \{a^n \mid n \text{ is even}\}$



2. $L = \{w \in \Sigma^* \mid \#a(w) \equiv 3, 4 \pmod 7\}$

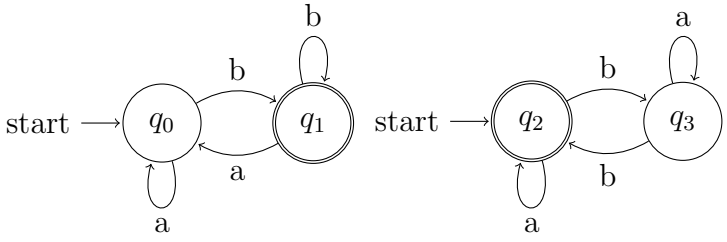


3. $L = \{w \in \Sigma^* \mid \text{int}(w) \text{ is prime}\}$
   We are not concerned with the alphabet really, not for the problems we want to study. Analogously, the primality of an integer has nothing to do with the base it is represented in. We are using languages to talk about decision problems. We have turned computational questions into ones about set membership. If an automata can determine correctly if some $w \in L$, then it certainly has the power of primality testing.

DFA's can be used to simulate two DFA's. For example, consider the languages

$$L_1 = \{w \in \Sigma^* \mid w \text{ ends with a } b\} \tag{3}$$
$$L_2 = \{w \in \Sigma^* \mid \#b(w) \text{ is even}\} \tag{4}$$

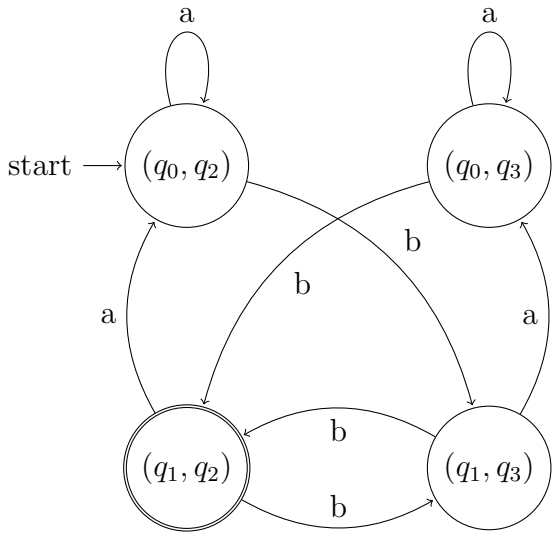
Lets make two DFAs for these languages.



We construct a DFA that simulates two DFAs, to compute $L_3 = L_1 \cap L_2$ Note that L_1 is decidable by the DFA $(\Sigma, Q_1, q_{01}, \delta_1, F_1)$ and L_2 is decidable by $(\Sigma, Q_2, q_{02}, \delta_2, F_2)$. We can use the properties of set intersection to define a DFA that accepts $L_1 \cap L_2 = (\Sigma_3, Q_3, q_{03}, \delta_3, F_3)$:

1. $Q_3 = Q_1 \times Q_2$
2. Σ is the same
3. $\delta_3((q_i, q_j), a) = (\delta_1(q_i, a), \delta_2(q_j, a))$ for $q_i \in Q_1$ and $q_j \in Q_2$
4. $q_{03} = (q_{01}, q_{02})$
5. $F_3 = F_1 \times F_2$

Out cartesian product DFA then looks like the following.



Note that if we made $F_3 = (Q_1 \times F_2) \cup (F_1 \times Q_2)$ we would have accepting states for the union of our two languages.

1.3 Regularity

A language is regular if and only if it is recognized by a DFA. We write the set of languages decidable by a DFA as $\mathscr{L}(DFA)$