Lecture Notes for CSCI 341: Theory of Computation
Set 3: Intro to Automata

Edward Talmage

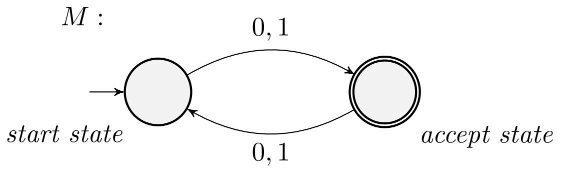
September 27, 2024

Problem 1. *Recognize* whether a string belongs to a particular language.

- This is a fundamental computation problem.
- To do this, we need a way to read through the string and evaluate some logic related to the definition of the language.

1 Finite State Automata (FSAs)

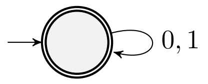
- A nearly memory-less computing device. We are trying to represent only the logic applied to the string.
- Example: $\Sigma = \{0, 1\}$



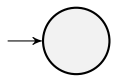
- The machine consumes one character at a time from the input and follows the matching arrow to a new circle (state).
- M *accepts* w if w drives M from its start state to an accept state.
- Define $L(M)$ as the language (set of strings) the machine accepts.
- An FSA can have any (finite) number of states, any of which may be accept states, but must have **exactly** one start state.

Examples: $(\Sigma = \{0, 1\})$

- **Exercise:** Give an FSA for $L = \Sigma^*$.

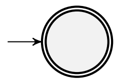


- **Exercise:** $L = \emptyset$

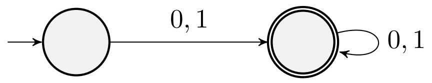


- **Exercise:** $L = \{\varepsilon\}$

Aside: ε denotes the *empty string*, which has length 0, and is identical for any alphabet. Note that $\{\}$ and $\{\varepsilon\}$ are not the same set.



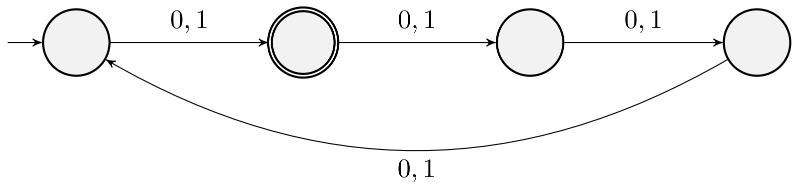
- **Exercise:** $L = \Sigma^+$



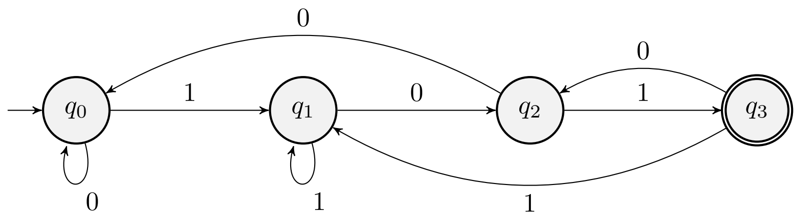
- **Exercise:** $L = \{w \mid w$ begins with 1 and ends with $0\}$

- **Exercise:** $L = \{w \mid w$ has a 1 in every position divisible by two or by three$\}$

- $L = \{w \mid |w| \equiv 1 \bmod 4\}$:



- **Exercise:** $L = \{w \mid w$ ends in $101\}$
  - Think about the logical meaning of each state.



We are particularly interested in *Deterministic Finite Automata*: FSAs in which the transitions are a function on $\{states\} \times \Sigma \to \{states\}$. That is, there is **exactly** one transition for each possible symbol from each state.

Exercise: Give a DFA that recognizes the language $L = \{w \mid$ the 3rd symbol of w from the right is a $1\}$ over $\Sigma = \{0, 1\}$.

2 Formal DFAs

Definition 1. A *Deterministic Finite Automaton (DFA)* is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$, where

- Q is a finite set (the *state set*)
- Σ is a finite set (the alphabet)
- $\delta : Q \times \Sigma \to Q$ (the *transition function*, takes a state and a symbol and gives the next state)
- $q_0 \in Q$ (*start state*)
- $F \subseteq Q$ (*accept* or *final* states)

Exercise: Write the formal, set-notation definition of the DFA accepting odd-length strings over $\{0, 1\}$.

Finishing our formal definitions: Take a string $w = a_0a_1a_2 \ldots a_n$, where each $a_i \in \Sigma$.

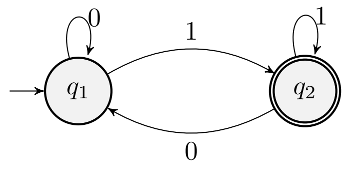
- We say M *accepts* w iff there exists a sequence of states $r_0, r_1, \ldots, r_{n+1}$ where each $r_i \in Q$, $r_0 = q_0$, for $0 \le i \le n, \delta(r_i, w_i) = r_{i+1}$, and $r_{n+1} \in F$.
  - This formalizes the notion of a string driving a machine.

  - Define $\hat{\delta} : Q \times \Sigma^* \to Q$, an extension of δ which applies it to a string instead of a single symbol, as follows:
    - For every $q \in Q, \hat{\delta}(q, \varepsilon) = q$
    - For every $u \in \Sigma^*, a \in \Sigma : \hat{\delta}(q, ua) = \delta(\hat{\delta}(q, u), a)$
  - M accepts w iff $\hat{\delta}(q_0, w) \in F$.

- The language *recognized* by a machine M is $L(M) := \{w \in \Sigma^* \mid M \text{ accepts } w\}$.
- A language L is *regular* if there exists a DFA M s.t. $L(M) = L$.

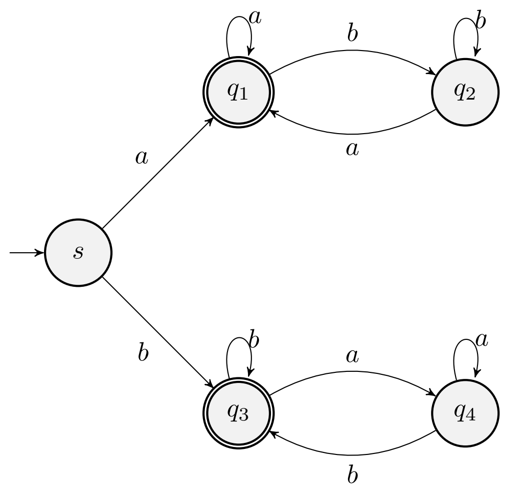
Exercise: Determine the language each of the following DFAs recognizes:

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



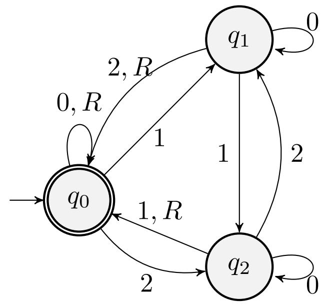
$L(M) = \{w \mid w \text{ ends in } 1\}$. Memory: last symbol (1 bit).

- $\Sigma = \{a, b\}$



Try: a, b, aa, bb, aba, bab, etc. $L(M) = \{w \mid w[0] == w[-1]\}$. Memory: 1st symbol (in the branch), most recent symbol (left/right within a branch).

- $\Sigma = \{0, 1, 2, R\}$



In general, we want to determine whether a language is regular: Can you give a DFA that recognizes it?

- Not always easy to do.
- Start with this: How can we combine known regular languages to create new regular languages?
- Use *closure properties*!

3 Closure Properties

Definition 2. Given a set S and a function $f : S \to R$, we say that S is *closed* under f if $\forall s \in S, f(s) \in S$.

- Similarly, if $g : S \times S \to T$, S is closed under g if $range(g) \subseteq S$.
- And so on for ternary-, quaternary-,... functions

Claim 1. *Let S be the set of all regular languages over a fixed alphabet Σ. S is closed under the following operations:*

1. *Union: $A \cup B := \{x \mid (x \in A) \vee (x \in B)\}$.*
2. *Concatenation: $A \circ B := \{ab \mid a \in A, b \in B\}$.*
3. *Kleene Star: $A^* := \{u_1 u_2 \dots u_k \mid u_i \in A, 1 \leq i \leq k, k \in \mathbb{Z}^{\geq}\}$*
   - *Note that if $k = 0$, we have the empty string, so $\forall A, \varepsilon \in A^*$.*

In other words, if A and B are regular languages, then $A \cup B$, $A \circ B$, and A^* are all regular languages.

Note that among these three operations, Kleene Star has the highest precedence, then concatenation, then union.
We will leave the proofs of the last two claims for later, since we need stronger tools we have not yet defined. We will here prove closure under union.
Idea: We need to run through machines for both A and B simultaneously. DFAs can't be in multiple places at the same time, though, so we can't do that directly.

Exercise: Can we build a DFA that tracks where you are in both individual DFAs? What would that look like?

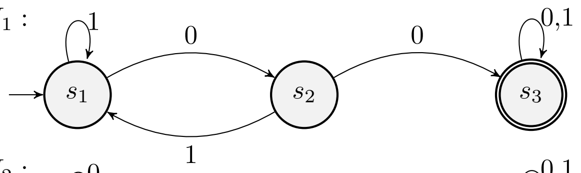
Proof. Let $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ and $M_2 = (Q_2, \Sigma, \delta_2, 1_2, F_2)$ be machines with $L(M_1) = A$ and $L(M_2) = B$. Define the machine $U = (Q, \Sigma, \delta, q_0, F)$ as follows:

- $Q = Q_1 \times Q_2$
- δ is defined as $\delta : Q \times \Sigma \to Q, \delta((u, v), a) = (\delta_1(u, a), \delta_2(v, a))$.
- $q_0 = (q_1, q_2)$
- $F = \{(r_1, r_2) \mid (r_1 \in F_1) \vee (r_2 \in F_2)\}$

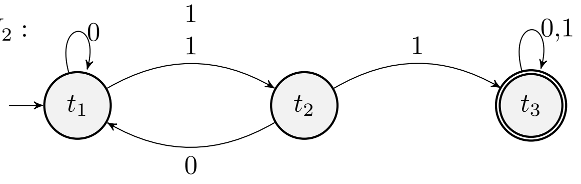
By construction, this DFA will accept a string w iff M or N accepts w. (Convince yourself that this is true.) □

Example: $(\Sigma = \{0, 1\})$

M_1 :

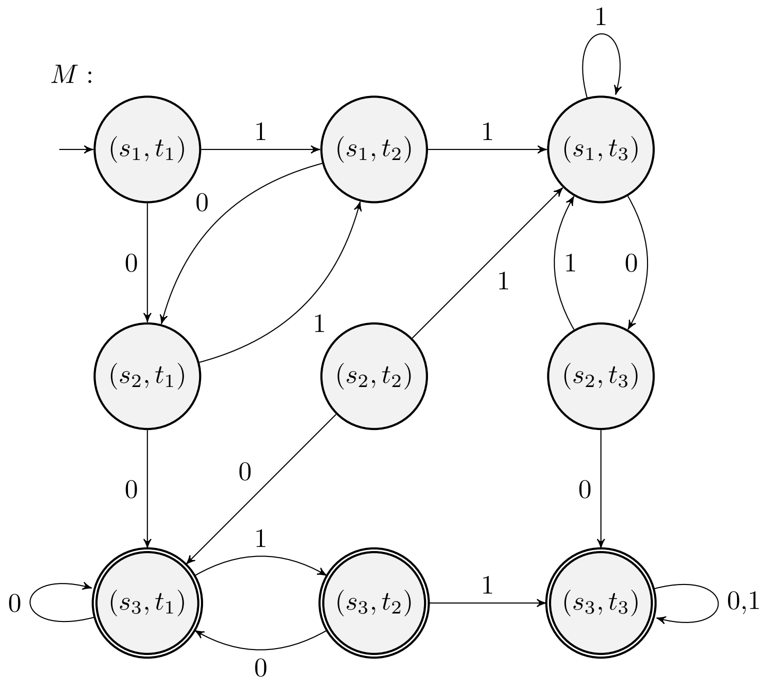


M_2 :



Exercise:

- What are the languages accepted by M_1 and M_2?
- Draw the machine which accepts $L(M_1) \cup L(M_2)$.
  - How many states will you need? How many edges? How many final states?



Exercise:

- $A = \{w \mid w$ has an even number of 1s$\}$
- $B = \{w \mid w$ contains '100'$\}$
- Build a DFA recognizing $A \cup B$.