Theoretical Computer Science

Tutorial - week 3

February 4, 2021

Innopolis University

Agenda

- Recap
- A bit of theory ... again
- Examples of Finite State Automaton

Recap

- What is ...

Recap

- What is ... FSA (FSM)?

Recap

- What is ... FSA (FSM)?
- Why is F?

Recap

- What is ... FSA (FSM)?
- Why is F?
- What else is finite?
  - alphabet

Recap

- What is … FSA (FSM)?
- Why is F?
- What else is finite?
  - alphabet
  - language

Recap

- What is ... FSA (FSM)?
- Why is F?
- What else is finite?
  - alphabet
  - language
  - accepting states

Recap

- What is ... FSA (FSM)?
- Why is F?
- What else is finite?
  - alphabet
  - language
  - accepting states
- What is complete FSA?

Recap

- What is ... FSA (FSM)?
- Why is F?
- What else is finite?
  - alphabet
  - language
  - accepting states
- What is complete FSA?
- What is extended transition?

Recap

- What is ... FSA (FSM)?
- Why is F?
- What else is finite?
  - alphabet
  - language
  - accepting states
- What is complete FSA?
- What is extended transition?
- Automata and Automaton?

Finite State Automata

FSA

- Finite State Automaton is a model of computation.
- It is a simple computing device: it acts as a language acceptor.

FSA

- Finite State Automaton is a model of computation.
- It is a simple computing device: it acts as a language acceptor.

Let's see an example.

FSA - Example (intuition)

If $\Sigma = \{a, b\}$ and L_1 is defined as

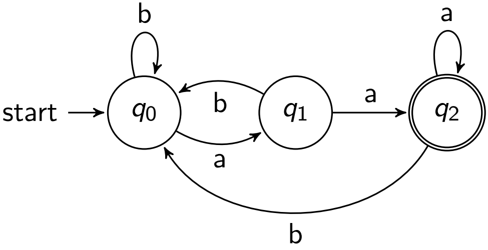
$$L_1 = \{x \in \Sigma^* \mid x \text{ ends with } aa\}$$

FSA - Example (intuition)

If $\Sigma = \{a, b\}$ and L_1 is defined as

$$L_1 = \{x \in \Sigma^* \mid x \text{ ends with } aa\}$$

Does the following FSA accepts all and only the strings represented by the language L_1?



Example (informally)

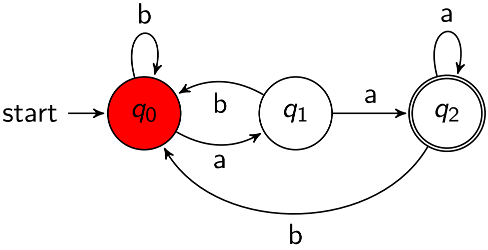
String $x = ababaa$ belongs to L_1. Meaning, $x \in \Sigma^*$ and it ends with aa. Let's see if the FSA accepts the string x

Example (informally)

String $x = ababaa$ belongs to L_1. Meaning, $x \in \Sigma^*$ and it ends with aa. Let's see if the FSA accepts the string x

q_0 is the starting point (it is graphically denoted by *start*). So the FSA starts in state q_0

$x = ababaa$



Example (informally)

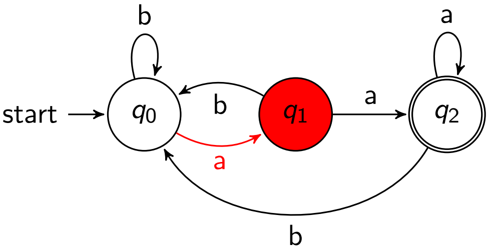
Then, we go through each character of x, following the transitions in the FSA

Example (informally)

Then, we go through each character of x, following the transitions in the FSA

$x = \mathbf{a}babaa$

From state q_0 and label a, we reach state q_1

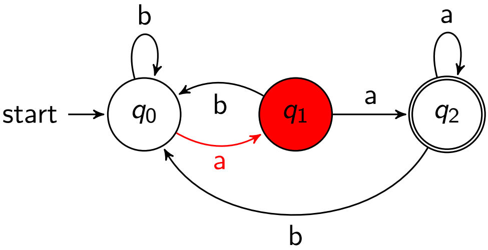


Example (informally)

Then, we go through each character of x, following the transitions in the FSA

$x = \mathbf{a}babaa$

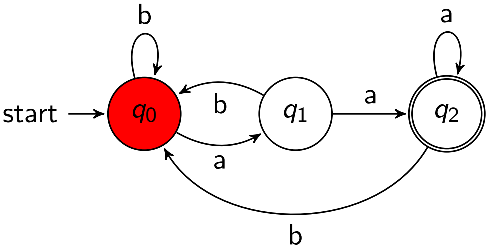
From state q_0 and label a, we reach state q_1



We repeat the process for all characters in x. If at the end we reach a final state (graphically denoted by the double circle state), we say the FSA accepts the string x.

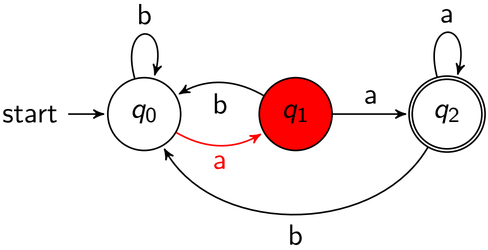
Example (informally) (1)

$x = ababaa$



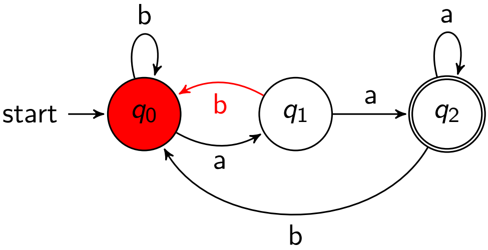
Example (informally) (2)

$x = \mathbf{a}babaa$



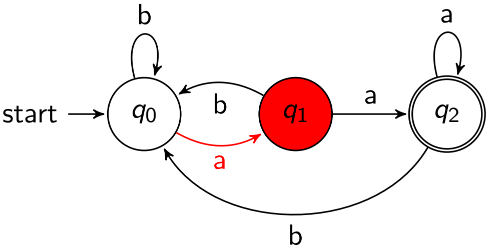
Example (informally) (3)

$x = a\mathbf{b}abaa$



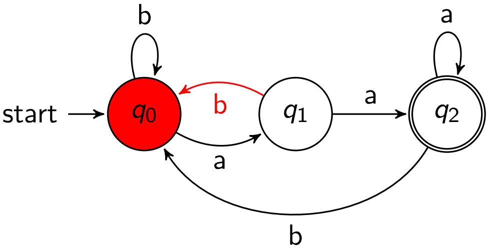
Example (informally) (4)

$x = ab\mathbf{a}baa$



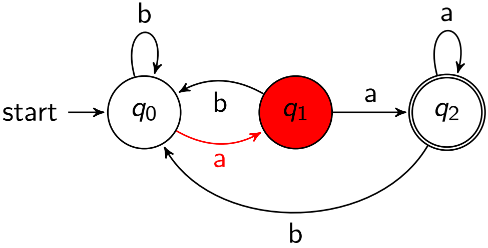
Example (informally) (5)

$x = aba\mathbf{b}aa$



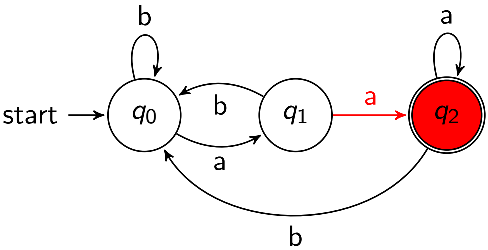
Example (informally) (6)

$x = abab\mathbf{a}a$



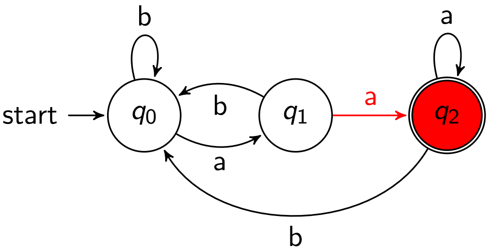
Example (informally) (7)

$x = ababa\mathbf{a}$



Example (informally) (7)

$x = ababa\mathbf{a}$



We went through all characters of x and ended up in a final state: string x belongs to language L_1.

FSA (Formal definition)

A complete Finite State Automaton

A complete Finite State Automaton is a tuple $< Q, \Sigma, q_0, A, \delta >$, where

Q is a finite set of *states*;
Σ is a finite *input alphabet*;
$q_0 \in Q$ is the *initial* state;
$A \subseteq Q$ is the set of *accepting* states;
$\delta : Q \times \Sigma \rightarrow Q$ is a total *transition* function.

For any element q of Q and any symbol $\sigma \in \Sigma$, we interpret $\delta(q, \sigma)$ as the state to which the FSA moves, if it is in state q and receives the input σ.

The extended transition δ^*

A move sequence starts from an initial state and is *accepting* if it reaches one of the final states (informally explained with the previous example).
Formally, this transition is defined recursively:

the extended transition δ^*

Let $M =< Q, \Sigma, q_0, A, \delta >$ be a complete finite state automaton. We define the extended transition function

$$\delta^* : Q \times \Sigma^* \to Q$$

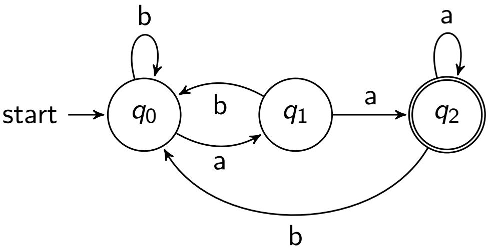
as follows:

1. For every $q \in Q$, $\delta^*(q, \epsilon) = q$
2. For every $q \in Q$, every $y \in \Sigma^*$, and every $\sigma \in \Sigma$,

$$\delta^*(q, y\sigma) = \delta(\delta^*(q, y), \sigma)$$

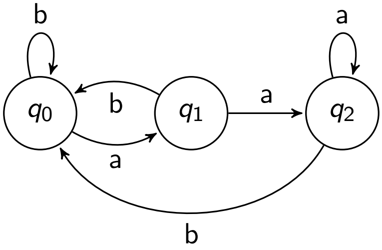
The extended transition (Example)

The complete FSA M contains the following transitions



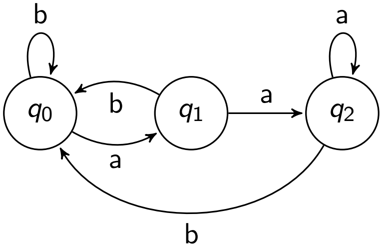
The extended transition (Example)

The complete FSA M contains the following transitions



The extended transition (Example)

The complete FSA M contains the following transitions



$$
\begin{aligned}
\delta^*(q_1, bab) &= \delta(\delta^*(q_1, ba), b) \\
&= \delta(\delta(\delta^*(q_1, b), a), b) \\
&= \delta(\delta(\delta(\delta^*(q_1, \epsilon), b), a), b) \\
&= \delta(\delta(\delta(q_1, b), a), b) \\
&= \delta(\delta(q_0, a), b) \\
&= \delta(q_1, b) \\
&= q_0
\end{aligned}
$$

Acceptance by a FSA

The extended transition function is used to determine what it means for a FSA to accept (or reject) a string or a language.

Acceptance by a FSA

The extended transition function is used to determine what it means for a FSA to accept (or reject) a string or a language. Formally:

Acceptance by a FSA

Let $M =< Q, \Sigma, q_0, A, \delta >$ be a complete FSA, and let $x \in \Sigma^*$. The string x is accepted by M if

$$\delta^*(q_0, x) \in A$$

and it is rejected by M otherwise. The language accepted by M is the set

$$L(M) = \{x \in \Sigma^* \mid x \text{ is accepted by } M\}$$

If L is a language over Σ, L is accepted by M iff $L = L(M)$

FSA

- Finite State Automaton is a model of computation.
- It is a simple computing device: it acts as a language acceptor.

FSA

- Finite State Automaton is a model of computation.
- It is a simple computing device: it acts as a language acceptor.

Why problems are about accepting strings?

FSA

- Finite State Automaton is a model of computation.
- It is a simple computing device: it acts as a language acceptor.

Why problems are about accepting strings?

- Properties of objects: ex. - number of sides

FSA

- Finite State Automaton is a model of computation.
- It is a simple computing device: it acts as a language acceptor.

Why problems are about accepting strings?

- Properties of objects: ex. - number of sides
- Functions: ex. - $2 + 2$

Example 2

Build[1] a complete FSA accepting a language L which comprises of all binary representations of integers divisible by 3.

[1] using a tool available here or here

Example 2

Build[1] a complete FSA accepting a language L which comprises of all binary representations of integers divisible by 3.

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

[1] using a tool available here or here

Example 2

Build[1] a complete FSA accepting a language L which comprises of all binary representations of integers divisible by 3.

- The alphabet $\Sigma = \{0, 1\}$
- $L = \{x \in \Sigma^* \mid \; x$ is the a binary representation of an integer, and it is divisible by 3$\}$;

[1] using a tool available here or here

Example 2

Build[1] a complete FSA accepting a language L which comprises of all binary representations of integers divisible by 3.

- The alphabet $\Sigma = \{0, 1\}$
- $L = \{x \in \Sigma^* \mid \ x$ is the a binary representation of an integer, and it is divisible by 3$\}$;

Let's consider 2 solutions:

1. L includes ϵ (for simplicity)

[1] using a tool available here or here

Example 2

Build[1] a complete FSA accepting a language L which comprises of all binary representations of integers divisible by 3.

- The alphabet $\Sigma = \{0, 1\}$
- $L = \{x \in \Sigma^* \mid \ x \text{ is the a binary representation of an integer, and it is divisible by } 3\}$;

Let's consider 2 solutions:

1. L includes ϵ (for simplicity)
2. L **does not** include ϵ

[1] using a tool available here or here

Wrap up

- What have you learnt today?
- Why problems are about accepting strings?