Automata Theory

Narges Khakpour

Department of Computer Science,
Linnaeus University

Table of Content

Why study automata theory?

- Automata is *an abstract computing devices*.
- It's the plural form of automaton which means "something that works automatically"
- Application of finite automata
  - Modeling and verifying finite state systems, such as communication protocols
  - Compiler for designing its lexical analyzer
  - Searching for keywords in a text
  - Etc

Basic Concepts

- alphabet: a finite, non-empty set of symbols.
- We indicate an alphabet by Σ,

Example 1

$\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ is decimal alphabet.
$\Sigma = \{a, b, c, ..., z\}$ the set of all lower case letters.

- A string: a finite sequence of symbols chosen from an alphabet,

Example 2

236 is a string over decimal alphabet.

- The empty string, ϵ, is a string with zero occurrence of symbols
- The length of a string denoted by $|w|$

Basic Concepts

- Power of an Alphabet: Σ^k is the set of strings with length k.
  $\Sigma^2 = 00, 01, ..., 10, 11, 12, ..., 99$
  $\Sigma^0 = \epsilon$
- The set of all strings is denoted by $\Sigma^* = \Sigma^0 \cup \Sigma^1 ....$
- The set of non-empty strings is represented by Σ^+.
- Concatenation: if a and b are two strings, then ab is also a string.
  $a = 23$ and $b = 54$ imply that 2354 is a string
  Is ϵa a string?

Basic Concepts

- A (possibly infinite) language L defined over Σ is a subset of Σ^*.
- Examples
  - The set of all numbers less than 1000
  - All English verbs. What is the alphabets?
  - All statements. What is the alphabets?
  - The set of Java programs
  - The set of even binary numbers: $L_b = 0, 10, 100, 110, 1000, 1010, \ldots..$
- The empty language is $\{\epsilon\} \neq \emptyset$.
- **Problem:** Does a given string s belong to a language L?
  e.g. is a Java program syntactically correct?
  or 1111110 is a member of L_b?

Table of Content

Deterministic Finite State Automata (DFA)

Definition 3

A DFA is a quintuple $A = (Q, \Sigma, \delta, q0, F)$ where

- Q is a finite set of states
- Σ is a finite alphabet (=input symbols)
- $\delta : Q \times \Sigma \to Q$ is a transition function and a transition $(q, a) \to p$ states that the DFA goes to q from p when the input action a is received,
- $q0 \in Q$ is the start state
- $F \subseteq Q$ is a set of final states

Representation of DFA-transition tables

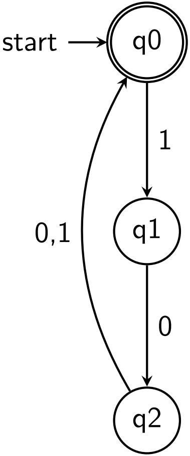
- Rows denote states and columns are the alphabet symbols
- Initial states are marked with the $\rightarrow$ symbol
- Final states are marked with the $*$ symbol
- Let $\Sigma = \{0, 1\}$

Example 4

state	0	1
* $\rightarrow$ q0		q1
q1	q2	
q2	q0	q0

Representation of DFA-transition diagrams

- Nodes are states
- Edges are transitions labeled with the corresponding action
- Initial state is marked with $\rightarrow$
- Final states are double-lined nodes



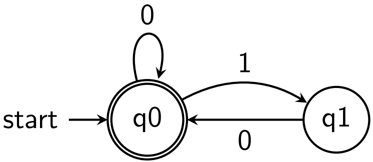
(D)FA Language

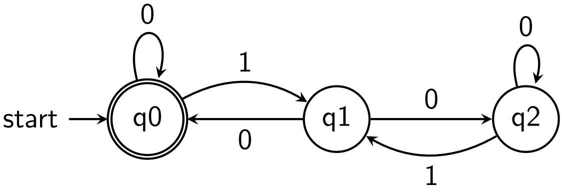
- A DFA accepts a string $s = a_1 a_2 \dots a_n$ if there is a path in its transition diagram that
  - Starts at a start state
  - Ends in a final state
  - Has the sequence of observed labels $a_1 a_2 \dots a_n$ from the start start to the final state
- The language of an automaton A is represented by $L(A)$
- $L(A)$ is the set of strings labeling the accepting paths.

Example 5

$\{100, 101, 100100, 100101, 101100, 101101, \dots\}$

Example





What are the languages?

Basic Modelling: Vending Machine Operations

- The user inserts money, the amount is checked by the VM.
- If the money is enough, the operation buttons become active to choose the products.Otherwise, the money is returned back to the user.
- The user chooses the product, the product is delivered, and the change is returned.
- If the selected product is unavailable, VM will reject the service

Basic Modelling: Vending Machine Operations

- Alphabets are the valid actions/interactions

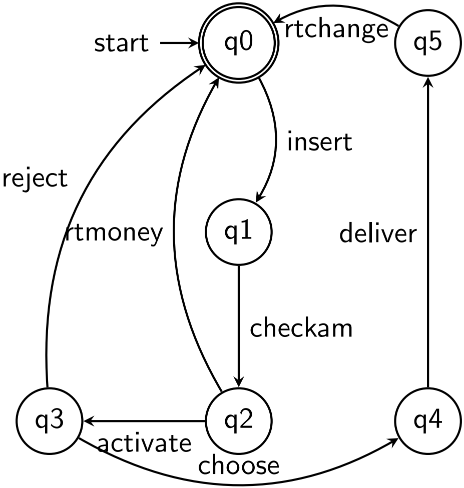
$$\Sigma = \{insert, checkam, activate, choose, rtmoney, deliver, rtchange, reject\}$$

- Its language consists of the set of valid sequence of events, e.g. the successful purchase is the following sequence:

$$insert\ checkam\ activate\ choose\ deliver\ rtchange$$

- **Problem** Can it happen that the vending machine eats the money?

Basic Modelling: Vending Machine Operations



Basic Modelling: Vending Machine Operations

Example 6

$$s_1 = \textit{insert checkam activate choose deliver rtchange}$$

$$s_2 = \textit{insert checkam activate reject}$$

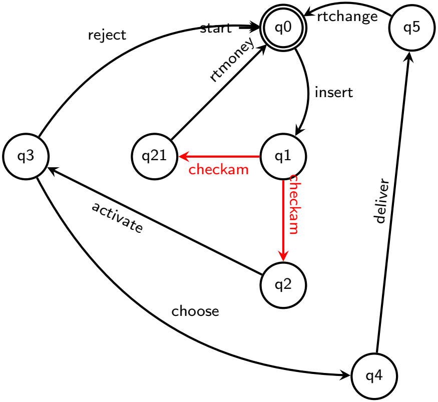
$$s_3 = \textit{insert checkam rtmoney}$$

$$L = \{s_1, s_2, s_3, s_1s_1, s_1s_2, s_1s_3, s_2s_1, s_2s_2, s_2s_3, \ldots\}$$

Table of Content

NFA

A NFA is an automaton which has several transitions with the same labels from a state.



NFA

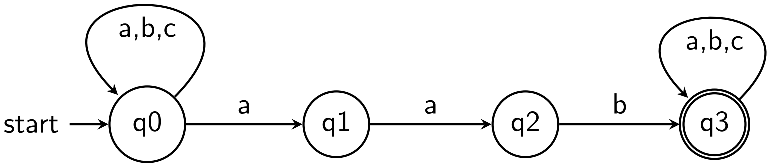
Definition 7

A NFA is a quintuple $A = (Q, \Sigma, \delta, q0, F)$ where

- Q is a finite set of states
- Σ is a finite alphabet (=input symbols)
- $\delta : Q \times \Sigma \rightarrow 2^Q$ is a transition function,
- $q0 \in Q$ is the start state
- $F \subseteq Q$ is a set of final states

The language of NFA is defined in the same way as that of DFA.

- Let $\Sigma = \{a, b, c\}$
- Design an NFA that accepts strings with *aab* as substring



How would you design it as a DFA?

FA with ϵ transitions

Definition 8

An ϵ-NFA is a quintuple $(Q, \Sigma, \delta, q0, F)$ where δ is a function from $Q \times (\Sigma \cup \{\epsilon\})$ to the powerset of Q.

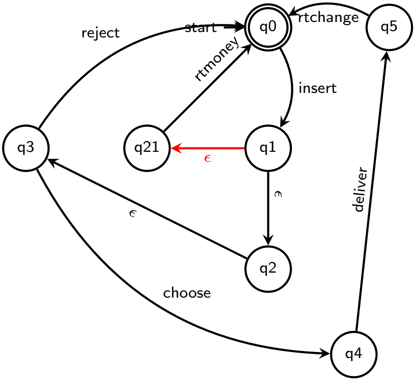
Definition 9

Let *eClosure* be a function that returns all states reachable from a state with only ϵ transitions, defined as follows:

$$q \in eClosure(q)$$

$$p \in eClosure(q) \wedge r \in \delta(p, \epsilon) \implies r \in eClosure(q).$$

NFA with ϵ transitions



We only observe our interaction with the vending machine, i.e. it is modelled as a black box component

$eClosure(q1)$?

ϵ-NFA's language

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

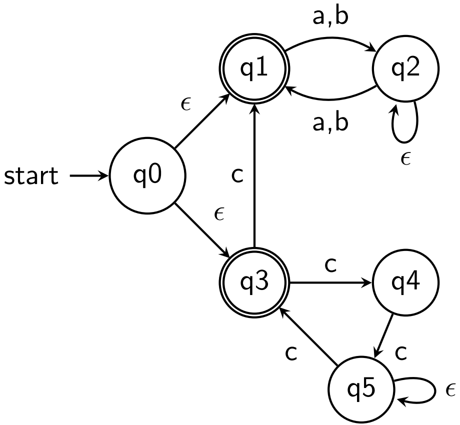


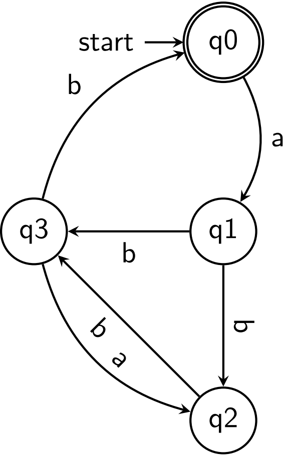
Table of Content

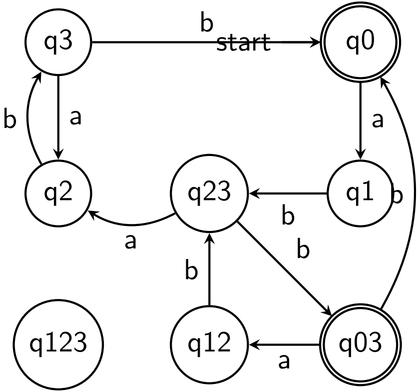
Equivalence of DFA and NFA

- NFA are usually easier to use, e.g. to model a system.
- A DFA is a NFA but not the opposite.
- We show that they are equivalent:

For any NFA N there is a DFA A such that $L(N) = L(A)$, and vise versa.

From NFA to DFA-Example

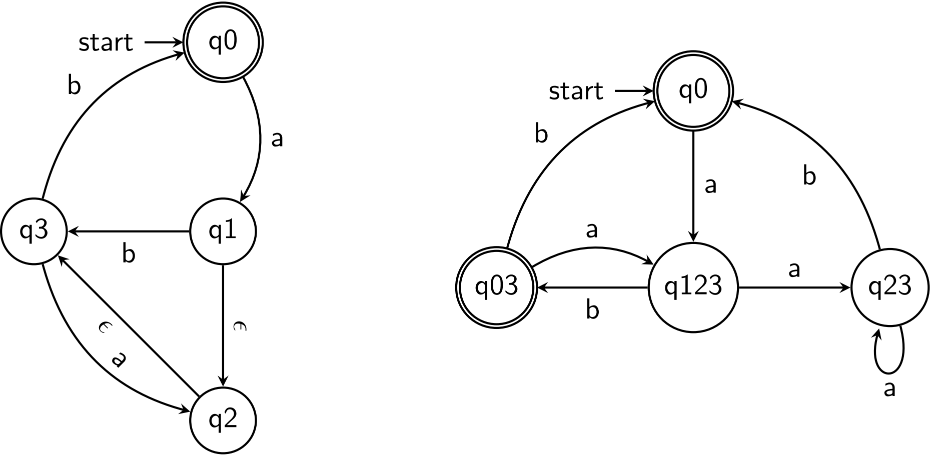




From NFA to DFA

- **Problem** Given an NFA $N = (Q_N, \Sigma, \delta_N, q_0, F_N)$, construct a DFA $D = (Q_D, \Sigma, \delta_D, q_0, F_D)$ such that $L(D) = L(N)$.
- $Q_D = \{S : S \subseteq Q_N\}$, i.e. the DFA states are all possible subsets of the states in NFA.
- A state is final in DFA, if one of its consisting states is final in NFA, i.e. $F_D = \{S \in Q_D : S \cap F_N \neq \emptyset\}$.
- For every state $S \subseteq Q_N$ and $a \in \Sigma$, $\delta_D(S, a) = \bigcup_{p \in S} \delta_N(p, a)$, the target state is the union of targets of the outgoing transitions labeled a from the states in S.

From ϵ-NFA to DFA-Example



From ϵ-NFA to DFA

Problem Given an ϵ-NFA $N = (Q_N, \Sigma, \delta_N, q_{0,N}, F_N)$, construct a DFA $D = (Q_D, \Sigma, \delta_D, q_{0,D}, F_D)$ such that $L(D) = L(N)$.

- $Q_D = \{S : S \subseteq Q_N \wedge S \in eClosure(S') \wedge S' \in Q_N\}$
- $q_{0,D} = eClosure(q_{0,N})$
- $F_D = \{S \in Q_D : S \cap F_N \neq \emptyset\}$.
- For every state $S \subseteq Q_N$ and $a \in \Sigma$,
  $\delta_D(S, a) = \bigcup\{eClosure(p) : p \in \delta_N(t, a) \text{ for some } t \in S\}$.

Table of Content

Summary and Upcoming Events

- Summary
  - Deterministic Finite Automata
  - Non-Deterministic Finite Automata
  - Converting NFA to DFA
- Next lecture on regular expressions