CM0081 Formal Languages and Automata
§ 2.2 Deterministic Finite Automata

Andrés Sicard-Ramírez

Universidad EAFIT

Semester 2024-1

Preliminaries

Conventions

- The number and page numbers assigned to chapters, examples, exercises, figures, quotes, sections and theorems on these slides correspond to the numbers assigned in the textbook [Hopcroft, Motwani and Ullman (1979) 2007].
- The natural numbers include the zero, that is, $\mathbb{N} = \{0, 1, 2, \dots\}$.
- The power set of a set A, that is, the set of its subsets, is denoted by $\mathcal{P}A$.

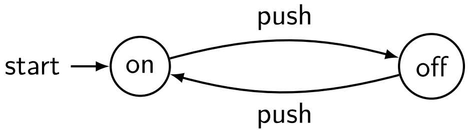
Formal Languages: Origins

Source areas [Greibach 1981, p. 14]

- Logic and recursive-function theory
- Switching circuit theory and logic design
- Modelling of biological systems (brain activity)
- Mathematical and computation linguistics
- Computer programming and the design of ALGOL and other problem-oriented languages

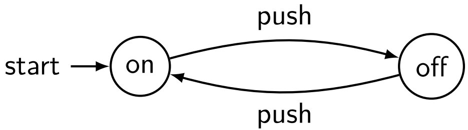
Finite Automata

Example (Modeling an on/off switch)

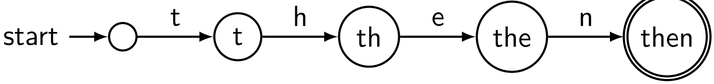


Finite Automata

Example (Modeling an on/off switch)

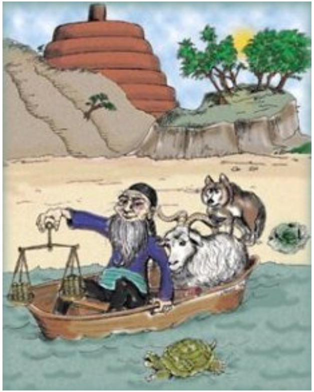


Example (Recognising the word ‘then’)



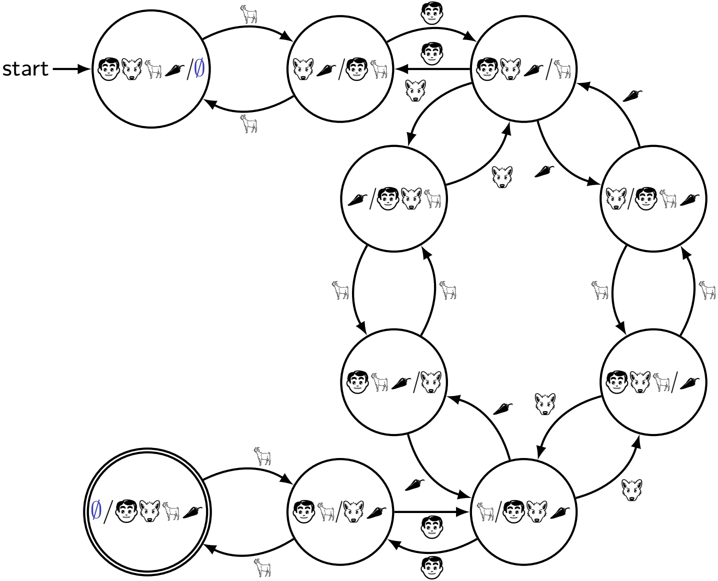
The Wolf, the Goat and the Cabbage Problem

‘A man with a wolf, goat, and cabbage is on the left bank of a river. There is a boat large enough to carry the man and only one of the other three. The man and his entourage wish to cross to the right bank, and the man can ferry each across, one at a time. However, if the man leaves the wolf and goat unattended on either shore, the wolf will surely eat the goat. Similarly, if the goat and cabbage are left unattended, the goat will eat the cabbage. Is it possible to cross the river without the goat or cabbage being eaten?’ [Hopcroft and Ullman 1979, p. 14] †



† The illustration is from the cover of [Levitin 2003].

The Wolf, the Goat and the Cabbage Problem



The Wolf, the Goat and the Cabbage Problem

Solution

In the previous automaton we can see two solutions:†

(i) 🐐, 👨, 🐺, 🐐, 🌶, 👨, 🐐.

(ii) 🐐, 👨, 🌶, 🐐, 🐺, 👨, 🐐.

†I used figures following a Haskell solution in https://iokasimov.github.io/posts/2020/08/wgc-effects .

Finite Automata

Description

*'The **finite automaton** is a mathematical model of a system, with discrete inputs and outputs. The system can be in any one of a finite number of internal configurations or "states". The state of the system summarizes the information concerning past inputs that is needed to determine the behaviour the system on subsequent inputs.' [Hopcroft and Ullman 1979, p. 13]*

Deterministic Finite Automata

Definition

A **deterministic finite automaton** (DFA) is a 5-tuple

$$(Q, \Sigma, \delta, q_0, F),$$

where

(i) Q is the finite set of states,

(ii) Σ is the alphabet of input symbols,

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

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

(v) $F \subseteq Q$ is the set of accepting (or final) states.

DFA Representations

DFAs can be represented of various equivalent ways:

(i) Transition diagram

(ii) Transition table

(iii) Detailed description

Transition Diagram

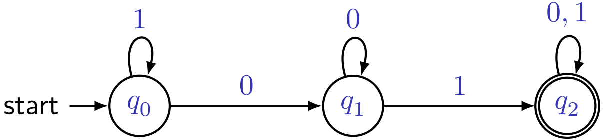
Example

Let $\Sigma = \{0, 1\}$. The following transition diagram represents a DFA that accepts the language $L = \{\, x01y \in \Sigma^* \mid x, y \in \Sigma^* \,\}$.

Transition Diagram

Example

Let $\Sigma = \{0, 1\}$. The following transition diagram represents a DFA that accepts the language $L = \{\, x01y \in \Sigma^* \mid x, y \in \Sigma^* \,\}$.

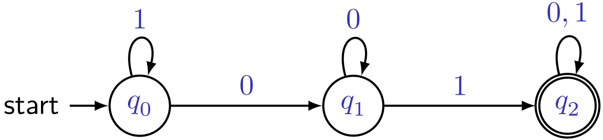


- q_0: The automaton has never seen 01, but its last input was either nonexistent or it last saw a 1.
- q_1: The automaton has never seen 01, but its most recent input was 0.
- q_2: The automaton has already seen 01.

Transition Diagram

Example

Let $\Sigma = \{0, 1\}$. The following transition diagram represents a DFA that accepts the language $L = \{\, x01y \in \Sigma^* \mid x, y \in \Sigma^* \,\}$.



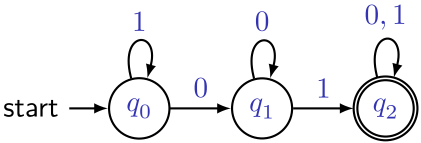
- q_0: The automaton has never seen 01, but its last input was either nonexistent or it last saw a 1.
- q_1: The automaton has never seen 01, but its most recent input was 0.
- q_2: The automaton has already seen 01.

Processing the input 0101: $\delta(q_0, 0) = \dots$

Transition Tables and Detailed Descriptions

Example

(i) Transition diagram



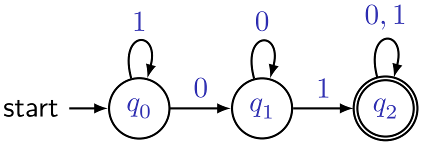
(ii) Transition table

	0	1
$\to q_0$	q_1	q_0
q_1	q_1	q_2
$*q_2$	q_2	q_2

Transition Tables and Detailed Descriptions

Example

(i) Transition diagram



(ii) Transition table

	0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
$*q_2$	q_2	q_2

(iii) Detailed description

$$Q = \{q_0, q_1, q_2\},$$

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

$$\delta(q_0, 0) = \delta(q_1, 0) = q_1,$$
$$\delta(q_0, 1) = q_0,$$
$$\delta(q_1, 1) = \delta(q_2, 0) = \delta(q_2, 1) = q_2,$$

q_0 start state,

$$F = \{q_2\}.$$

Extension of the Transition Function for DFAs

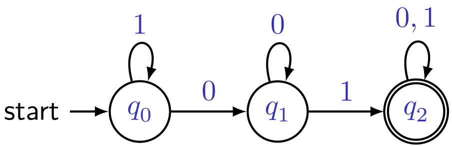
Definition

Let $D = (Q, \Sigma, \delta, q_0, F)$ be a DFA. The **extension of the transition function**, denoted by $\hat{\delta}$, is recursively defined by

$$\begin{aligned} \hat{\delta} : Q \times \Sigma^* &\to Q \\ \hat{\delta}(q, \varepsilon) &= q, \\ \hat{\delta}(q, xa) &= \delta(\hat{\delta}(q, x), a). \end{aligned}$$

Extension of the Transition Function for DFAs

Example



$$
\begin{aligned}
\hat{\delta}(q_0, 010) &= \hat{\delta}(q_0, \varepsilon 010) \\
&= \delta(\hat{\delta}(q_0, \varepsilon 01), 0) \\
&= \delta(\delta(\hat{\delta}(q_0, \varepsilon 0), 1,), 0) \\
&= \delta(\delta(\delta(\hat{\delta}(q_0, \varepsilon), 0), 1,), 0) \\
&= \delta(\delta(\delta(q_0, 0), 1,), 0) \\
&= \delta(\delta(q_1, 1,), 0) \\
&= \delta(q_2, 0) \\
&= q_2
\end{aligned}
$$

Extension of the Transition Function for DFAs

Exercise 2.2.2

Prove that $\hat{\delta}(q, xy) = \hat{\delta}(\hat{\delta}(q, x), y)$ for any state q and strings x and y. (Hint: Perform induction on y).

Extension of the Transition Function for DFAs

Exercise 2.2.2

Prove that $\hat{\delta}(q, xy) = \hat{\delta}(\hat{\delta}(q, x), y)$ for any state q and strings x and y. (Hint: Perform induction on y).

Proof by induction on y

- Basis step ($y = \varepsilon$)

$$\begin{aligned} \hat{\delta}(\hat{\delta}(q, x), \varepsilon) &= \hat{\delta}(q, x) && \text{(def. of } \hat{\delta}\text{)} \\ &= \hat{\delta}(q, x\varepsilon) && \text{(def. of concatenation)} \end{aligned}$$

Extension of the Transition Function for DFAs

Proof (continuation)

- Inductive step ($y = wa$)

$$
\begin{aligned}
\hat{\delta}(q, x \cdot wa) &= \delta(\hat{\delta}(q, xw), a) && \text{(def. of } \hat{\delta} \text{ and concatenation)} \\
&= \delta(\hat{\delta}(\hat{\delta}(q, x), w), a) && \text{(IH)} \\
&= \hat{\delta}(\hat{\delta}(q, x), wa) && \text{(def. of } \hat{\delta}\text{)}
\end{aligned}
$$

∎

Extension of the Transition Function for DFAs

Exercise 2.2.7

Let D be a DFA and q a particular state of D, such that $\delta(q, a) = q$ for all input symbols a. Show by induction on the input that for all input strings w, $\hat{\delta}(q, w) = q$.

Extension of the Transition Function for DFAs

Exercise 2.2.7

Let D be a DFA and q a particular state of D, such that $\delta(q, a) = q$ for all input symbols a. Show by induction on the input that for all input strings w, $\hat{\delta}(q, w) = q$.

Proof by induction on w

- Basis step ($w = \varepsilon$)

$$\hat{\delta}(q, \varepsilon) = q \qquad \text{(def. of } \hat{\delta}\text{)}$$

Extension of the Transition Function for DFAs

Exercise 2.2.7

Let D be a DFA and q a particular state of D, such that $\delta(q, a) = q$ for all input symbols a. Show by induction on the input that for all input strings w, $\hat{\delta}(q, w) = q$.

Proof by induction on w

- Basis step ($w = \varepsilon$)

$$\hat{\delta}(q, \varepsilon) = q \qquad \text{(def. of } \hat{\delta}\text{)}$$

- Inductive step ($w = xa$)

$$\begin{aligned} \hat{\delta}(q, xa) &= \delta(\hat{\delta}(q, x), a) && \text{(def. of } \hat{\delta}\text{)} \\ &= \delta(q, a) && \text{(IH)} \\ &= q && \text{(hypothesis)} \end{aligned}$$

Languages accepted by DFAs

Definitions

Let $D = (Q, \Sigma, \delta, q_0, F)$ be a DFA and let $w \in \Sigma^*$ be a string.

(i) The string w is **accepted** by D iff $\hat{\delta}(q_0, w) \in F$.

Languages accepted by DFAs

Definitions

Let $D = (Q, \Sigma, \delta, q_0, F)$ be a DFA and let $w \in \Sigma^*$ be a string.

(i) The string w is **accepted** by D iff $\hat{\delta}(q_0, w) \in F$.

(ii) The string w is **rejected** by D iff $\hat{\delta}(q_0, w) \notin F$.

Languages accepted by DFAs

Definitions

Let $D = (Q, \Sigma, \delta, q_0, F)$ be a DFA and let $w \in \Sigma^*$ be a string.

(i) The string w is **accepted** by D iff $\hat{\delta}(q_0, w) \in F$.

(ii) The string w is **rejected** by D iff $\hat{\delta}(q_0, w) \notin F$.

(iii) The **language accepted** by D, denoted $\mathrm{L}(D)$, is the set of strings accepted by D, that is,

$$\mathrm{L}(D) := \left\{ w \in \Sigma^* \;\middle|\; \hat{\delta}(q_0, w) \in F \right\}.$$

Regular Languages

Definition

A language L is **regular** iff exists a DFA D such that $L = \mathrm{L}(D)$.

Regular Languages

Example

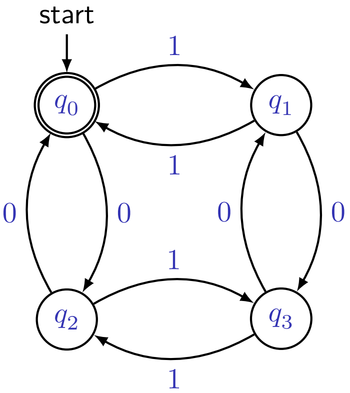
Let L be the set of words with both an even number of 0's and an even number of 1's. L is a regular language.

Regular Languages

Example

Let L be the set of words with both an even number of 0's and an even number of 1's. L is a regular language.

- q_0: Both the number of 0's seen so far and the number of 1's seen so far are even.
- q_1: The number of 0's seen so far is even, but the number of 1's seen so far is odd.
- q_2: The number of 1's seen so far is even, but the number of 0's seen so far is odd.
- q_3: Both the number of 0's seen so far and the number of 1's seen so far are odd.



Regular Languages

Question

Let Σ be an alphabet. Is $\emptyset$ a regular language?

Regular Languages

Question

Let Σ be an alphabet. Is $\emptyset$ a regular language? What about Σ^*?

Representation of DFAs

Functional Program Representation

(Adapted from [Keller 2001])

- Each state of the automaton is identified with a function from Σ^* to a truth value.
- The initial state is identified with the overall function of the automaton.

Representation of DFAs

Functional Program Representation

(Adapted from [Keller 2001])

- Each state of the automaton is identified with a function from Σ^* to a truth value.
- The initial state is identified with the overall function of the automaton.

Example

See the implementation for the representation functional of a DFA in the course homepage.

References

Greibach, S. A. (1981). Formal Languages: Origins and Directions. Annals of History of Computing 3.1, pp. 14–41. DOI: 10.1109/MAHC.1981.10006 (cit. on p. 3).

Hopcroft, J. E., Motwani, R. and Ullman, J. D. [1979] (2007). Introduction to Automata Theory, Languages, and Computation. 3rd ed. Pearson Education (cit. on p. 2).

Hopcroft, J. E. and Ullman, J. D. (1979). Introduction to Automata Theory, Languages, and Computation. Addison-Wesley (cit. on pp. 6, 9).

Keller, R. M. (2001). Computer Science: Abstraction to Implementation. URL: www.cs.hmc.edu/~keller/cs60book/ (visited on 07/02/2018) (cit. on pp. 33, 34).

Levitin, A. (2003). Introduction to the Design and Analysis of Algorithms. Addison Wesley (cit. on p. 6).