Introduction

A finite automaton is a formal machine that considers discrete input and output. The machine can take any configuration among a given finite set. Those configurations are usually referred to as *states*. Thus, the current state of the machine summarizes all the previous information processed and it is necessary to determine the future behaviour of the machine.

Leaving aside the theoretical interest of finite automata (they model a well-known and interesting family of languages in the Chomsky hierarchy, the family of *regular languages*), there are important reasons to study finite automata. Many widely-used algorithms are based on finite automata. For instance, the *lexical analysis* carried out by any compiler is usually based on the simulation of a finite automaton. In this process, the symbols of a source code are analyzed to detect those strings that correspond with identifiers, constants, or reserved words as well. To do so, during the process, it is enough to consider a finite amount of information.

Automata Theory is also used to design efficient string processors. An example of this is the use of finite automata to detect the positions where a certain pattern can be found in a text x. This problem is known as *String Matching* o *Pattern Matching* and will be studied in Practices three and four.

This Practice is focused to study the representation of three models of finite automata as well as to the implementation of some operations over those models.

Representation of finite automata

A finite automaton $A = (Q, \Sigma, \delta, q_0, F)$ will be represented as a list:

$$aut = \{st, alf, trans, ini, fin\}$$

where:

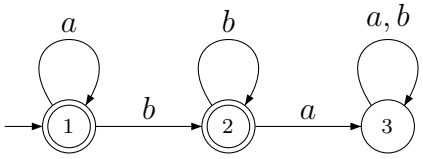
- st is a list that contains the identifiers (integers, symbols, lists, etc.) of the *set of states* (Q)
- alf is a list with the *alphabet* symbols (Σ)
- $trans$ a list with the representation of the *transition function* of the automaton (δ).

  Each transition of the automaton is represented as a list:

  $$\{OriginState, Symbol\ or\ \{\}, DestState\}$$

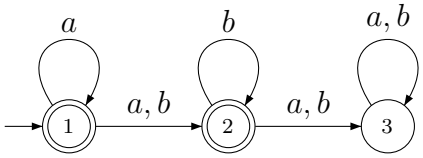
- ini is the identifier of the *initial state* (q_0)
- fin is a list with the identifiers of the *final states* (F)

Examples



its *Mathematica* representation:

$$A = \{\{1,2,3\},\{a,b\},\{\{1,a,1\},\{1,b,2\},\{2,a,3\},\{2,b,2\},\{3,a,3\},\{3,b,3\}\},1,\{1,2\}\}$$



its *Mathematica* representation:

$$A = \{\{1,2,3\},\{a,b\},$$
$$\{\{1,a,1\},\{1,a,2\},\{1,b,2\},\{2,a,3\},\{2,b,3\},\{2,b,2\},\{3,a,3\},\{3,b,3\}\},$$
$$1,\{1,2\}\}$$

Exercises

Exercise 1

Design a Mathematica module that, on input of a FA A, returns whether A is deterministic or not.

Exercise 2

For any given automaton A (deterministic or not), the automaton is *codeterministic* if it has only one final state and there exist no pair of transitions (q, a, p) and (r, a, p) (two transitions that reach the same state with the same symbol). Implement a Mathematica module that, on input of a DFA A, returns whether A is codeterministic or not.

Exercise 3

Implement a Mathematica module that, on input of a DFA A, returns whether A is complete or not.

Exercise 4

Implement a Mathematica module that, on input of a DFA A and a string x, return $True$ if x is in $L(A)$ and $False$ otherwise.

Exercise 5

Implement a Mathematica module that, given two input DFAs A_1 and A_2, and a word x, return whether the word is in the language $L(A_1) \cup L(A_2)$.

Exercise 6

Implement a Mathematica module that, given two input DFAs A_1 and A_2, and a word x, return whether the word is in the language $L(A_1) \cap L(A_2)$.

Exercise 7

Implement a Mathematica module that, given an input NFA $A = (Q, \Sigma, \delta, q_0, F)$, a set $C \subseteq Q$, and a symbol $a \in \Sigma$, return $\delta(C, a)$, or the set of states that are reached when the symbol a is analyzed in A taking into account the set C.

Example: Given the following NFA:

$$A = \{\{1,2,3\},\{a,b\},\\ \{\{1,a,1\},\{1,a,2\},\{1,b,2\},\{2,a,3\},\{2,a,1\},\{2,b,3\},\{3,a,2\},\{3,b,3\}\},\\ 1,\{1,2\}\}$$

the set $\{1,3\}$ and the symbol a, the module should return the set $\{1,2\}$.

Given the same NFA, the set $\{2,3\}$ and the symbol b, the module should return the set $\{3\}$.

Exercise 8

Implement a Mathematica module that, given an NFA A and a string x, return $True$ if x is in $L(A)$ and $False$ otherwise.

Hint: It is recommended to use of the module implemented in Exercise 7.

Exercise 9

A DFA is said to hold property P if, for any symbol in the alphabet, all the transistions labelled with that symbol reach the same state.

Implement a Mathematica module that, given an input DFA, output $True$ if the automaton hold the property P.

Example:

The following automaton:

$$A = \{\{1,2,3\},\{a,b\},\\ \{\{1,a,2\},\{1,b,3\},\{2,a,2\},\{2,b,3\},\{3,a,2\},\{3,b,3\}\},\\ 1,\{2\}\}$$

hold the property, but the following one:

$$A = \{\{1,2,3,4\},\{a,b\},\\ \{\{1,a,2\},\{1,b,4\},\{2,a,4\},\{2,b,3\},\{3,a,4\},\{3,b,2\},\{4,a,4\},\{4,b,3\}\},\\ 1,\{2\}\}$$

does not hold it (for instance, states 2 and 4 have incomming transitions labelled with the same symbol a).

Exercise 10

Given a complete DFA A and a word u, it is said that u represents the automaton A if all the states, included the initial one, are reached from some other state when u is analyzed.

Implement a Mathematica module that, given a complete DFA and a word as input, return $True$ if the word represents the automaton and $False$ otherwise.

Example:

Given:

$$A = \{\{1,2,3,4\},\{a,b\},\\ \{\{1,a,2\},\{1,b,1\},\{2,a,3\},\{2,b,2\},\{3,a,4\},\{3,b,3\},\{4,a,1\},\{4,b,4\}\},\\ 1,\{2\}\}$$

the word $u = bba$ represents the automaton because, in this case, $\delta(1,u) = 2$, $\delta(2,u) = 3$, $\delta(3,u) = 4$ and $\delta(4,u) = 1$.

Exercise 11

A word u is said to synchronize a DFA A if the analysis of u from any state always reach the same state (regardless it is final or not).

Implement a Mathematica module that, given a complete DFA and a word u as input, return $True$ if the word u synchronizes the automaton and $False$ otherwise.

Example:

Given:

$$A = \{\{1,2,3,4\},\{a,b\},\\ \{\{1,a,2\},\{1,b,2\},\{2,a,2\},\{2,b,3\},\{3,a,3\},\{3,b,4\},\{4,a,4\},\{4,b,1\}\},\\ 1,\{1\}\}$$

the word $u = abbbabbba$ sinchronizes the automaton because, for any state q in Q, $\delta(q,u) = 2$.