Deterministic Finite Automata

S. E. Venegas-Andraca
Quantum Information Processing Group
Tecnológico de Monterrey
http://www.mindsofmexico.org/sva
svenegas@itesm.mx

August 2017

The contents of this handout are mostly based on [1].

1 Preliminary definitions

Definition 1.1. Definitions of Cartesian product, relation, function, injective function, surjective function, bijective function.

Example 1.1. Examples and counterexamples: $f(x) = x$, $f(x) = x^2$, $f(x) = \sin x$,
$f(x) = \cos x$, $f(x) = 1/x$, $f(x) = \sqrt{x}$, $f(x) = \pm\sqrt{x}$

Definition 1.2. Alphabet. Any finite set of symbols Σ is an alphabet. For any alphabet Σ, we denote by Σ^* the set of all finite strings of symbols over some fixed alphabet Σ.

Example 1.2. Examples of alphabets
1. $\Sigma_1 = \{0, 1\}$
2. $\Sigma_2 = \{a, b, c, d, e\}$
3. $\Sigma_3 = \{0, 1, x, \alpha, \beta, \Omega, r\}$

Definition 1.3. String. A string ω over an alphabet Σ is a finite sequence of symbols from that alphabet, usually written next to one another and not separated by commas.

If ω is a string over Σ then the length of ω, denoted $|\omega|$, is the number of symbols that it contains. Moreover, if $|\omega| = n$ then we can write $\omega = \omega_1\omega_2\omega_3\ldots\omega_n$, where each $\omega_i \in \Sigma$. The string of length zero is called the empty string and is written ϵ.

Example 1.3. Examples of alphabets

1. 0101000101011101 *is a string of* $\Sigma_1 = \{0, 1\}$
2. $abdcacdeedcba$ *is a string of* $\Sigma_2 = \{a, b, c, d, e\}$
3. $\alpha\Omega\beta xxxxx\Omega rrr0001110$ *is a string of* $\Sigma_3 = \{0, 1, x, \alpha, \beta, \omega, r\}$
4. $\alpha 587y245\Omega\beta xxxxxrrr0001110$ *is NOT a string of* $\Sigma_3 = \{0, 1, x, \alpha, \beta, \omega, r\}$

Definition 1.4. Concatenation. If we have string x of length m and string y of length n, the concatenation of x and y, written xy, is the string obtained by appending y to the end of x, as in $x_1x_2\ldots x_my_1y_2\ldots y_n$. To concatenate a string with itself many times we use the superscript notation. For example, to succinctly write k times the string x we may write x^k.

Definition 1.5. Language. A language is a set of strings.

2 Deterministic Finite Automata I

Deterministic Finite Automata is a model for computers with an extremely limited amount of memory. An example of a Deterministic Finite Automaton (DFA) is a machine R used to determine the parity of a sequence of characters like the one shown in Fig. (1).

Example 2.1. Automaton for a wireless communication system

Suppose that R is at one of the ends of a wireless communication system. R can receive as valid input the characters '0' and '1', and, since any communication system is subject to errors, R can receive a third character '#' which is used to state that an error has occurred in the transmission of the data stream.

So, the input for R is an arbitrarily long set of characters (also known as a string) taken from the alphabet $\Sigma = \{0, 1, \#\}$. The parity of a sequence of 0s and 1s is defined as follows: a sequence of 0s and 1s is odd if its number of 1s is odd, and it is even if its number of 1s is even.

Let us now elaborate on the properties R must have. First, we state that only strings with no errors will be suitable for parity computation. Thus, only sequences of 0*s and* 1*s will be accepted and any string having at least one error character must be rejected. Second, an empty set of characters has no* 1*s, thus we set the initial parity of a string as even.*

Let us show the behaviour of R with an example. Suppose that R receives as input the string 100110001 *(read from left to right). The first input character is '*1*' thus R goes from state 'Even' to state 'Odd'. We then read '0' and therefore we stay in state 'Odd'. The third input character is again a '0' and we remain in state 'Odd'. As fourth input we receive a '1' and then we move from state 'Odd' to state 'Even' as now we have an even number of '1's in our account. The fifth input is a '1' and consequently we move from state 'Even' to state 'Odd' as the total number of '1's we have received so far is odd. The* 6^{th}*,* 7^{th} *and* 8^{th} *characters are '0' thus the current state of machine R remains being 'Odd'. Finally, the last input character is '1' and therefore R goes from state 'Odd' to state 'Even'. The final outcome of the computation is the accept state 'Even'.*

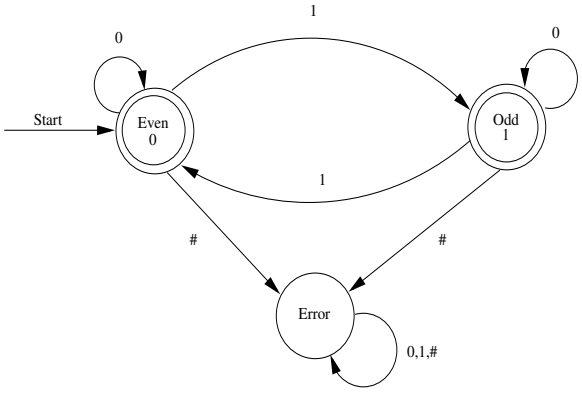

Figure 1: A finite automaton R for parity computation. Double-circled states are accept states and single-circled state is a reject state. Input strings for R are taken from the set of any combination of characters taken from the alphabet $\Sigma = \{0, 1, \#\}$. The actual state of R at any point of the computation is well-defined.

Example 2.2. More examples

1. Controller for an automatic door.

2. Automaton that accepts strings which do not have two successive b's.

Formally speaking, a DFA is a list of five objects: set of states, input alphabet, rules for moving, start state, and accept states. We use a **transition function** to define the rules for moving. If the DFA has an arrow from state x to a state y labeled with the input symbol 1, that means that, if the automaton is in state x when it reads a 1, it then moves to state y. We may denote this behaviour using the parlance of transition functions by saying that $\delta(x, 1) = y$.

Definition 2.1. Deterministic Finite Automaton. A DFA R is a 5-tuple $(Q, \Sigma, \delta, q_o, F)$, where

1. Q is a finite set called the states,
2. Σ is a finite set called the alphabet,
3. $\delta : Q \times \Sigma \to Q$ is the transition function,
4. $q_0 \in Q$ is the start state, and
5. $F \subset Q$ is the set of accept states.

The formal description of the DFA R depicted in Fig. (1) is as follows: $Q = \{\text{Even}, \text{Odd}, \text{Error}\}$, $\Sigma = \{0, 1, \#\}$, $q_0 = \text{Even}$, $F = \{\text{Even}, \text{Odd}\}$ and $L(R) = \{x \in \{0, 1\}^n\}$. The transition function is given in Table 1.

Table 1. Transition Function δ

$\delta(\text{Even}, 0) = \text{Even}$	$\delta(\text{Even}, 1) = \text{Odd}$	$\delta(\text{Even}, \#) = \text{Error}$
$\delta(\text{Odd}, 0) = \text{Odd}$	$\delta(\text{Odd}, 1) = \text{Even}$	$\delta(\text{Odd}, \#) = \text{Error}$
$\delta(\text{Error}, 0) = \text{Error}$	$\delta(\text{Error}, 1) = \text{Error}$	$\delta(\text{Error}, \#) = \text{Error}$

Definition 2.2. Computation with a Deterministic Finite Automaton. Let $R = (Q, \Sigma, \delta, q_o, F)$ be a DFA and let $w = w_1 w_2 \dots w_n$ be a string where each w_i is a member of the alphabet Σ. Then R **accepts** w if a sequence of states $r_0, r_1, \dots, r_n$ in Q exists with three conditions:

1. $r_o = q_0$,
2. $\delta(r_i, w_{i+1}) = r_{i+1}$, for $i = 0, 1, \dots, n-1$, and
3. $r_n \in F$.

Condition 1 means that the machine starts in the start state. Condition 2 states that the machine goes from state to state according to the transition function. Condition 3 says that the machine accepts its input if it ends up in an accept state. We say that R **accepts**/**recognizes** language A if $A = \{w | R \text{ accepts } w\}$, i.e. A is the set of all strings accepted by R.

Definition 2.3. Regular language. A language is called a regular language iff some finite automaton recognizes it.

Exercises

1. $A_1 = \{\omega | \omega$ has an even number of 1s$\}$
2. $A_2 = \{\omega | \omega$ contains at least one 1 and an even number of 0s follow the last 1$\}$
3. $A_3 = \{\omega | \omega$ is the empty string ϵ or ends in a 0$\}$
4. $A_4 = \{\omega | \omega$ contains at least one 1 and ends with 1$\}$
5. $A_5 = \{\omega | \omega$ starts and ends with the same symbol$\}$
6. $A_6 = \{\omega | \omega$ contains a substring 001$\}$

References

[1] M. Sipser, Introduction to the Theory of Computation, PWS (2005).