Faculty of Computers and Information
Assiut University

Deterministic Finite Automaton (DFA) Implementation

Team Members:

Abdelrahman Samir

Seif Elnasr Amr

Marwa Ahmed Ali

Table of Contents

Introduction

Automata:

A mechanism that is relatively self-operating. An automaton is an abstract self-propelled computing device which follows a predetermined sequence of operations automatically.

Deterministic Finite Automaton:

Deterministic refers to that on each input symbol there is one and only one state to which the automaton can transition from its current state; also, in DFA null (or ε) move is not allowed, and DFA cannot change state without any input character.

A DFA consists of 5-tuple (Q, $\sum$, δ, q_0, F), where:

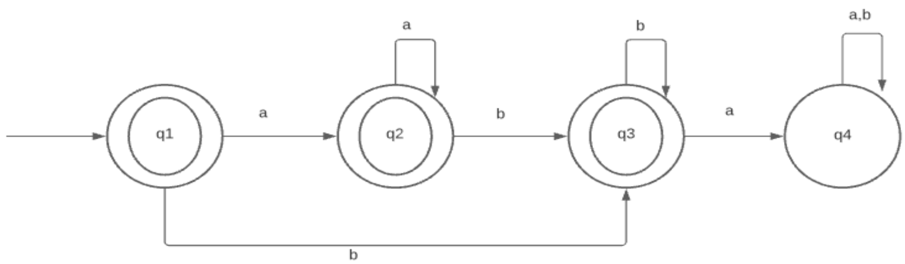
- Q is a finite set of states.
- $\sum$ is a finite set of symbols called the alphabet.
- δ is the transition function where $\delta: Q \times \sum \rightarrow Q$
- q_0 is the initial state from where any input is processed ($q_0 \in Q$).
- F is a set of final state/states of Q ($F \subseteq Q$)

Machine Design

Machine 1:

{W|W is sorted}

M1={Q, ∑, δ, q_0, F},where:



1. Q = {q1, q2, q3,q4}
2. Σ = {a, b}
3. δ is described as

	a	b	
q1	q2	q3	
q2	q2	q3	
q3	q4	q3	
q4	q4	q4	

4. q1 is the start state
5. F = {q1,q2,q3}

The machine reads inputs:

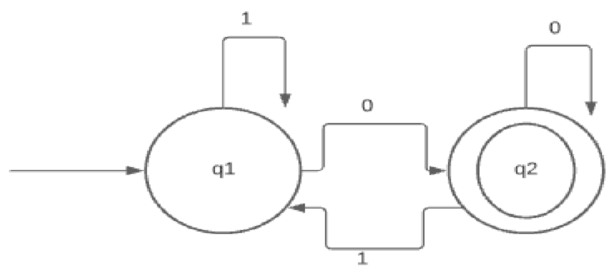
- **aaabbb:** q1 → q2 → q2 → q2 → q3→ q3 → q3 "Accept".
- **bbbbb:** q1 → q3 → q3 → q3 → q3→ q3 "Accept.
- **abba:** q1 → q2 → q3 → q3 → q4 "Reject".

Machine Design

Machine 2:

{W|W is the binary representation of an even number}

M2={Q, ∑, δ, q_0, F},where:



1. Q = {q1, q2}
2. Σ = {0, 1}
3. δ is described as

	0	1
q1	q2	q1
q2	q2	q1

4. q1 is the start state

5 .F = {q2}

The machine reads inputs:

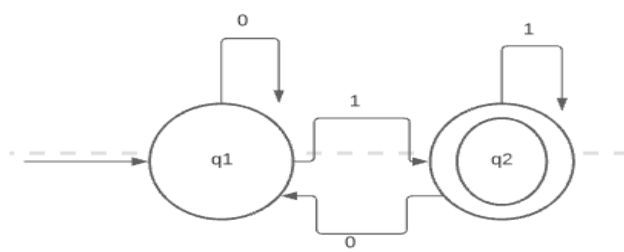
- **10110110:** q1→ q1→ q2→ q1→ q1→ q2 →q1→ q1→ q2 “Accept”.
- **010:** q1→ q2→ q1→ q2 “Accept”.
- **1011011:** q1 → q1→ q2 → q1 → q1→ q2→ q1 → q1 “Reject”.

Machine Design

Machine 3:

{W|W is the binary representation of an odd number}

M3= {Q, ∑, δ, q_0, F}, where:



1. Q = {q1, q2}
2. Σ = {0, 1}
3. δ is described as

	0	1	
q1	q1	q2	
q2	q1	q2	

4. q1 is the start state
5. F = {q2}

The machine reads inputs:

- **1001101:** q1 → q2 → q1 → q1 → q2→ q2 → q1 → q2 “Accept”.
- **001:** q1 → q1→ q1 → q2 “Accept”.
- **100110:** q1 → q2 → q1 → q1→ q2→ q2 → q1 “Reject”.

Machine Implementation

Machine 1:

```
5  bool isSorted(string s)
6  {
7      if (s.length() == 0)
8      {
9          return true;
10     }
11     for (int i = 0; i < s.length() - 1; i++)
12     {
13         if (s[i] == 'b' && s[i + 1] == 'a')
14         {
15             return false;
16         }
17     }
18     return true;
19 }
```

Machine 1 Sorted string

The string is sorted if and only if there isn't a 'b' followed by an 'a':

Line 7: Checks weather the string is empty i.e. (it's size equals to 0) or not.

Line 8: if the condition is true. Then the function returns true.

Line 11: for loop to check every two adjacent elements.

Line 13: if condition to check if the current character is 'b' and the character after it is 'a'.

Line 15: if the condition in line 13 is true then the function returns false i.e. (The string is not sorted) because there is a 'b' followed by an 'a'.

Line 18: If the function didn't return any value till now this means that the string is sorted there for the function returns true indicating that the string is sorted.

Machine Implementation

Machine 2:

```
20
21 bool isEven(string binary)
22 {
23     if (binary.length() == 0)
24     {
25         return false;
26     }
27     int last_bit = binary.length() - 1;
28     if (binary[last_bit] == '0')
29     {
30         return true;
31     }
32     return false;
33 }
```

Machine 2 Even Number

A number is even if and only if in its binary representation the last bit on the right is 0:

Line 23: if condition to check whether the string is empty or not.

Line 25: if the condition in line 23 is true then the function returns false.

Line 27: store the value of the last index on the right in last_bit.

Line 28: if condition to check the value in the last bit i.e. (last character on the right) equals to ‘0’.

Line 30: the function returns true if the if condition in line 28 is true indicating that the number is even.

Line 32: if the function didn’t return any value till now that means that the number is odd there for the function returns false.

Machine Implementation

Machine 3:

```
35  bool isOdd(string binary)
36  {
37      if (binary.length())
38      {
39          return false;
40      }
41      int last_bit = binary.length() - 1;
42      if (binary[last_bit] == '1')
43      {
44          return true;
45      }
46      return false;
47  }
```

Machine 3 Odd number

A number is odd if and only if in its binary representation the last bit on the right is 1:

Line 23: if condition to check whether the string is empty or not.

Line 25: if the condition in line 23 is true then the function returns false.

Line 27: store the value of the last index on the right in last_bit.

Line 28: if condition to check the value in the last bit i.e. (last character on the right) equals to ‘1’.

Line 30: the function returns true if the if condition in line 28 is true indicating that the number is odd.

Line 32: if the function didn’t return any value till now that means that the number is even there for the function returns false.

Project Code

```
#include<iostream>
#include<string>
using namespace std;

bool isSorted(string s)
{
	if (s.length() == 0)
	{
		return true;
	}
	for (int i = 0; i < s.length() - 1; i++)
	{
		if (s[i] == 'b' && s[i + 1] == 'a')
		{
			return false;
		}
	}
	return true;
}

bool isEven(string binary)
{
	if (binary.length() == 0)
	{
		return false;
	}
	int last_bit = binary.length() - 1;
	if (binary[last_bit] == '0')
	{
		return true;
	}
	return false;
}

bool isOdd(string binary)
{
	if (binary.length())
	{
		return false;
	}
	int last_bit = binary.length() - 1;
	if (binary[last_bit] == '1')
	{
		return true;
	}
	return false;

}
```

Project Code

```
int main()
{
	cout << "Enter 1 for machine 1 :{W | W is sorted}\n";
	cout << "Enter 2 for machine 2 :{W | W is the binary representation of an even number}\n";
	cout << "Enter 3 for machine 3 :{W | W is the binary representation of an odd number}\n";
	int machine;
	cin >> machine;
	cout << "Enter your string : ";
	string input;
	cin.ignore();
	getline(cin, input);
	if (machine == 1)
	{
		if (isSorted(input))
		{
			cout << "Accept\n";

		}
		else
		{
			cout << "Reject\n";
		}
	}
	else if (machine == 2)
	{
		if (isEven(input))
		{
			cout << "Accept\n";
		}
		else
		{
			cout << "Reject\n";
		}
	}
	else if (machine == 3)
	{
		if (isOdd(input))
		{
			cout << "Accept\n";
		}
		else
		{
			cout << "Reject\n";
		}
	}
	else
	{
		cout << "Invaild input. Please try again\n";
	}
}// end of main
```

Project Plan

- All team members have researched for information about **Automata** and **Deterministic Finite Automaton.**
- We have designed diagram of **DFA** on paper, used **Lucid chart** website to draw diagram, and write machine implementation in **Visual Studio Code** by using **C++** language.

Abdelrahman Samir:

- Wrote machine implementation for three machines.
- Wrote description for each machine implementation.

Marwa Ahmed Ali:

- Wrote description for each machine design.
- Took screenshot for each function.

Seif Elnasr Amr:

- Designed all diagram of **DFA** by using **Lucidchart** website.
- wrote Introduction.

References

1. Michael Sipser, Introduction to THE THEORY of COMPUTATION, 2nd Edition, page 3, page47, and page48.

2. John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, Introduction to Automata Theory, Languages, and Computation, 3rd Edition, page 1, and page 45.

3. Tutorials point
   https://www.tutorialspoint.com/automata_theory/automata_theory_introduction.htm
   27/12/2021 7:30pm.