**College of Arts and Sciences**

**Computer and Information Systems Department**

**Theory of Computation**

**COSC421**

**NFAs to DFAs App & Applications of FSMs**

**Lecturer:**

Dr. Mohammad El Abed

**Group Members:**

Ali Yassine 20190234

Rima Abou Chacra 20190150

**Date:**

November 28th, 2021

Fall 2021

Table of Contents

***ABSTRACT*** **3**

**1. INTRODUCTION**3

**2. HOW TO USE**?3

**3. URL** 4

**4. WEBSITE GUI**5

**5. TECHNOLOGIES USED**11

**6. FOUR APPLICATIONS THAT USE FSMs**13

**7. CONCLUSION**19

**8. REFERENCES**19

**9. APPENDIX**19

***ABSTRACT:***

*We are assigned to build a website where finite state machines can be drawn based on the inputs that the user enters.*

**1. INTRODUCTION:**

In this project, we aim to develop a web application that dynamically draw finite state machines (DFA&NFA), FSMs are drawn based on the user’s input that are the number of states, and the language. First the NFA is drawn and directly converted to DFA. At the end, the user can check if a string is accepted or rejected by an obtained DFA, and the characteristics of NFA and DFA are shown in another page.

At the end of this report, we will be presenting 4 applications that use finite state machines.

**2. HOW TO USE?**

When the user opens the website, he will see a Navbar, In the Navbar we can see that the website is formed of 2 pages: “the homepage”, and another page named “Learn more”, and a field named “GitHub”.

Homepage:

First, the user enters the number of states and the language, the language should be separated by “-”, and the number of states should be an integer number. The user should mandatory insert the language and the number of states if any field is empty the user is asked to fill it. then after inserting the language and the number of states, the user is supposed to click on the submit button, after clicking on the submit button, a table is created dynamically where the number of states inserted by the user represents the number of the rows of the table, and the strings of the language represent the number of columns. This table represent the transition function; In this table, the user should insert where every state is going giving a certain string, also there is a check box beside every state in order that the user specify what is/are the final state/s. If the user doesn’t specify which is/are the final state/s an error message is displayed (Error: Add a final sate). Also there is a reset button, clicking on this button everything will be cleared. When inserting the table’s input the NFA is drawn and converted dynamically to DFA that will be drawn below it.

Learn more:

This is another page that contain some information about finite state machines, general characteristics of finite automata, characteristics of NFA, and characteristics of DFA.

GitHub:

Clicking on it, the user can visit our GitHub repository.

**3. URL:**

[**http://aliyassine26.github.io/**](http://aliyassine26.github.io/)

**4. WEBSITE GUI:**

1. **Desktop Version**

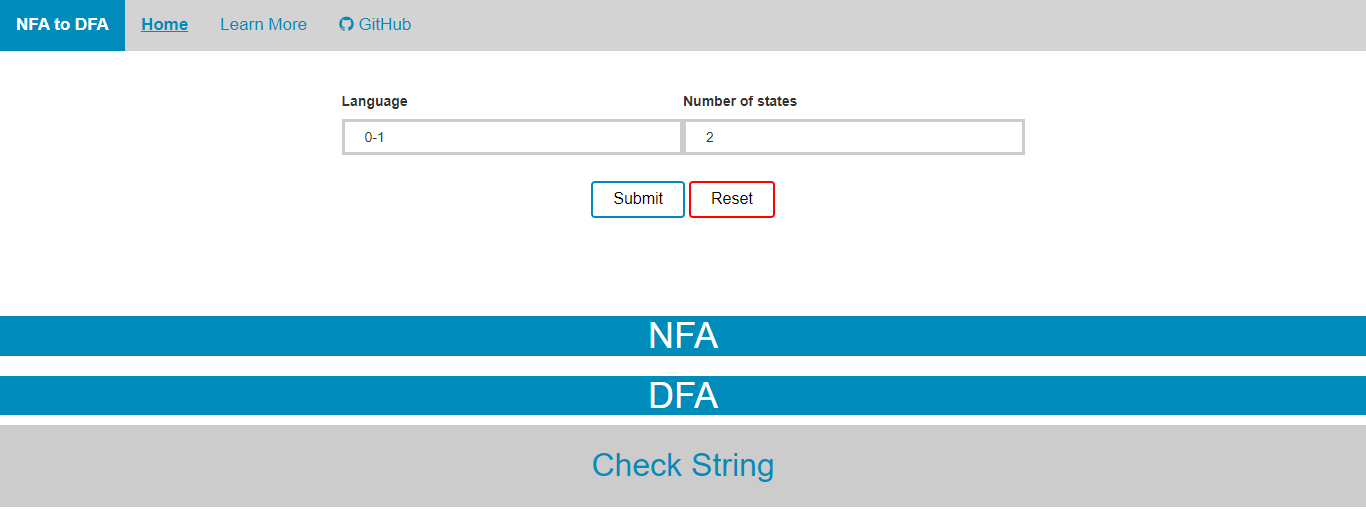


Figure 1:website GUI

Homepage:

* If any field is empty, the user is asked to fill it

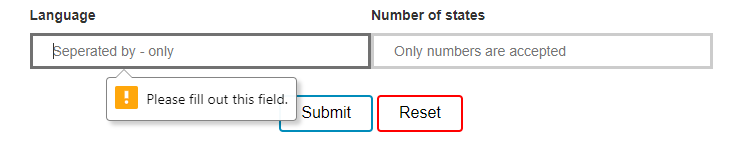


Figure 2:Empty fields

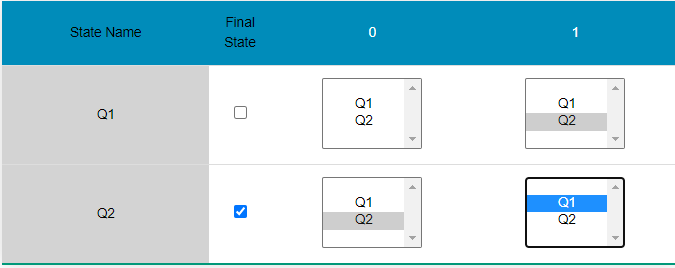
* Clicking on the submit button we can see the transition table

Figure 3:Transition table

* If the user doesn’t specify which is/are the final state/s an error message is displayed (Error: Add a final sate).

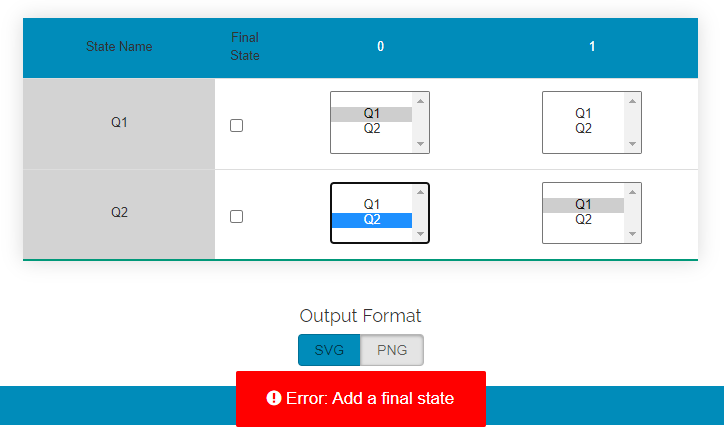


Figure 4: Error message

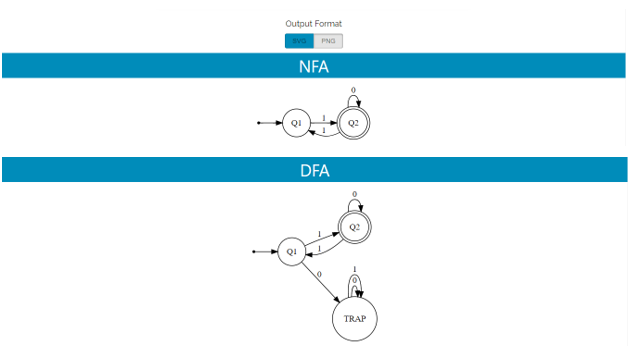
* The NFA and the DFA are drawn dynamically

Figure 5: NFA and DFA schemes

* The user can transform the drawings of NFA and DFA to the form (PNG), so the user can download of drag the image.

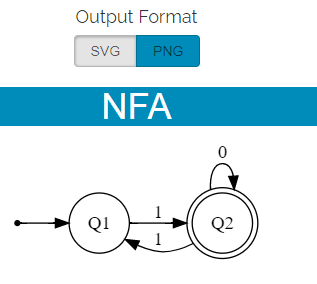


Figure 6: PNG format

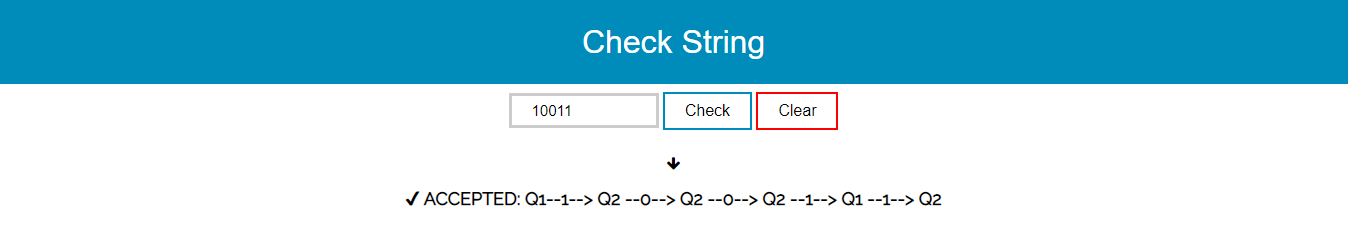
* We can check a string for example ‘10011’

Figure 7: Checking a string

Learn more:

* This page contains some information about finite state machines, general characteristics of finite automata, characteristics of NFA, and characteristics of DFA.

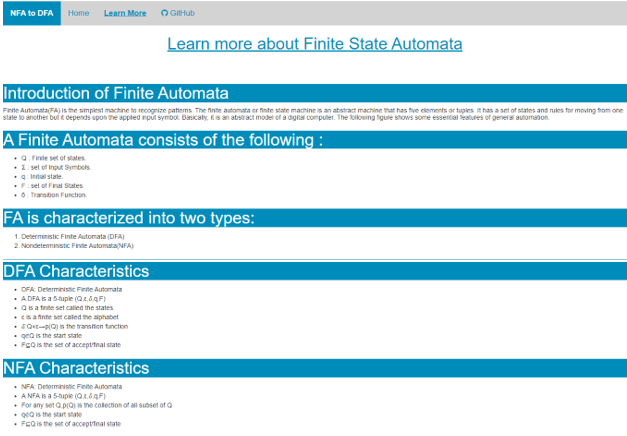


Figure 8: About FSMs

GitHub:

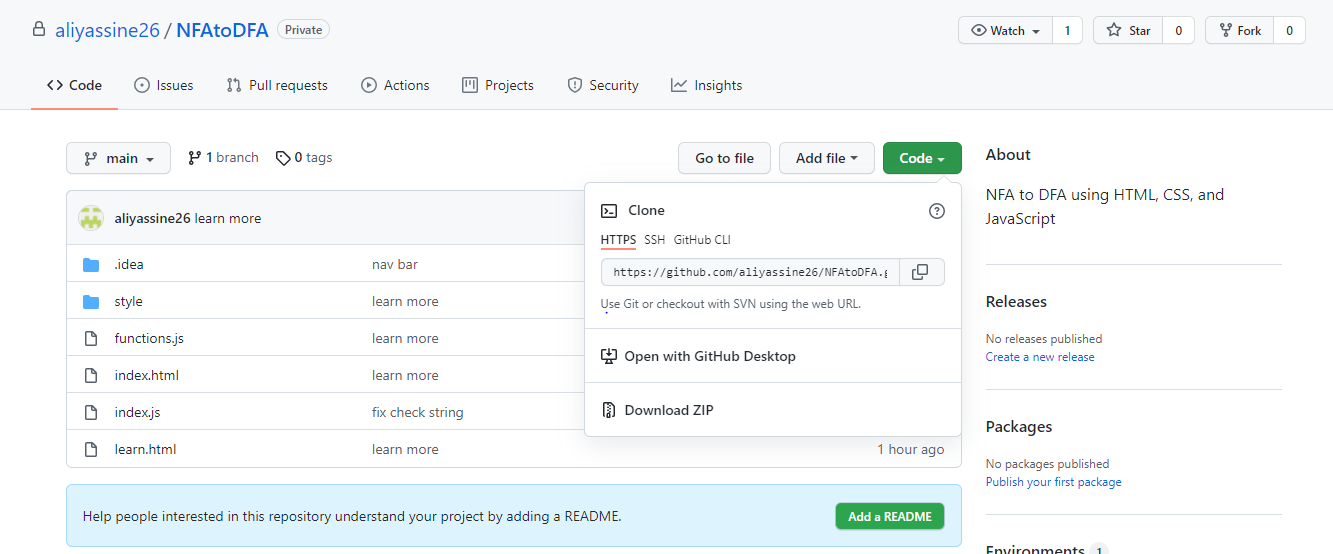
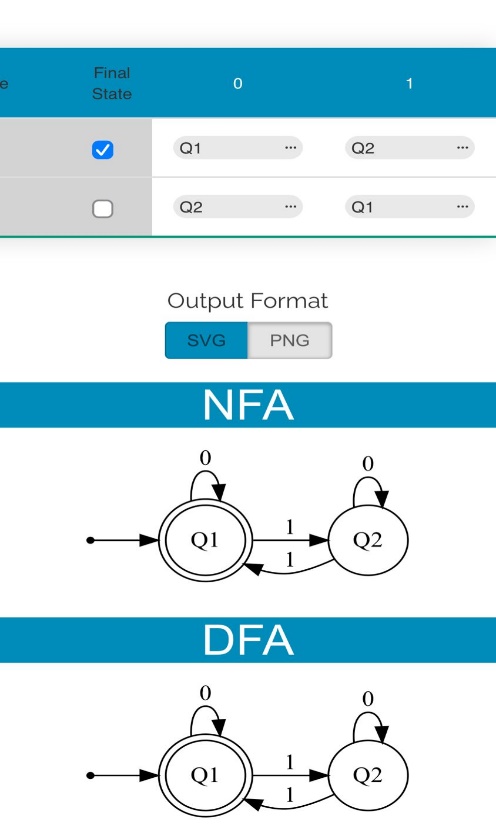


Figure 9: GitHub repository

1. **Mobile Version**

****

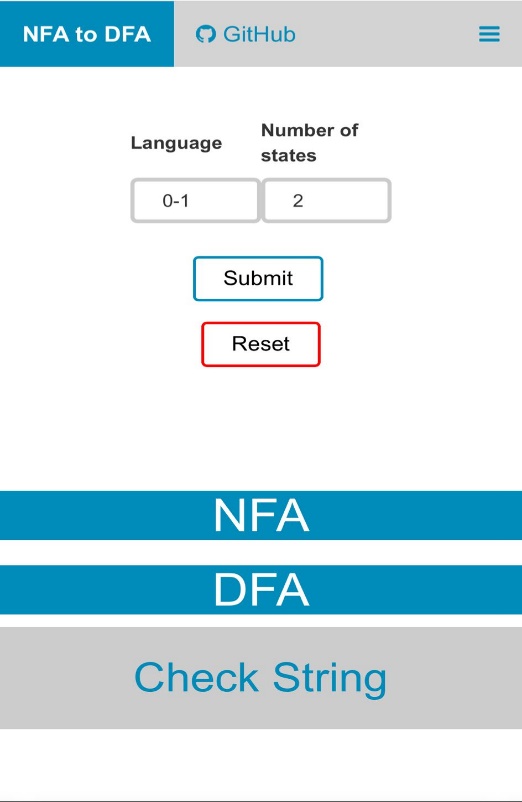
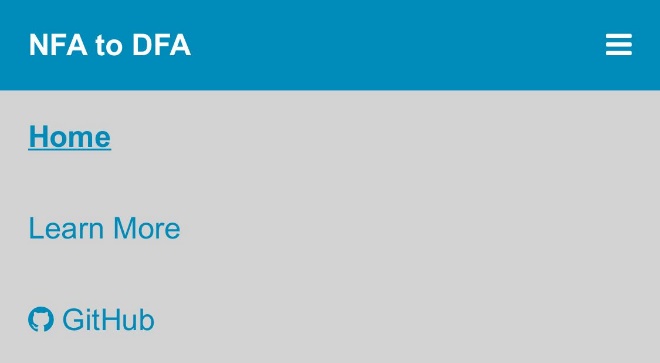
****

Figure 11: Results on Mobile

Figure 10: Homepage on Mobile

****

Graphical user interface

Description automatically generated with medium confidence

Figure 12: Navigation Menu on Mobile

Figure 13: Navigation Menu on Mobile

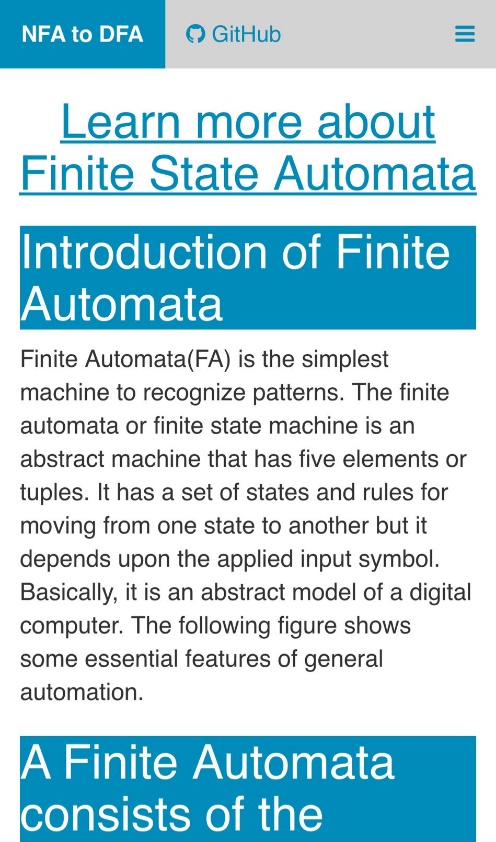
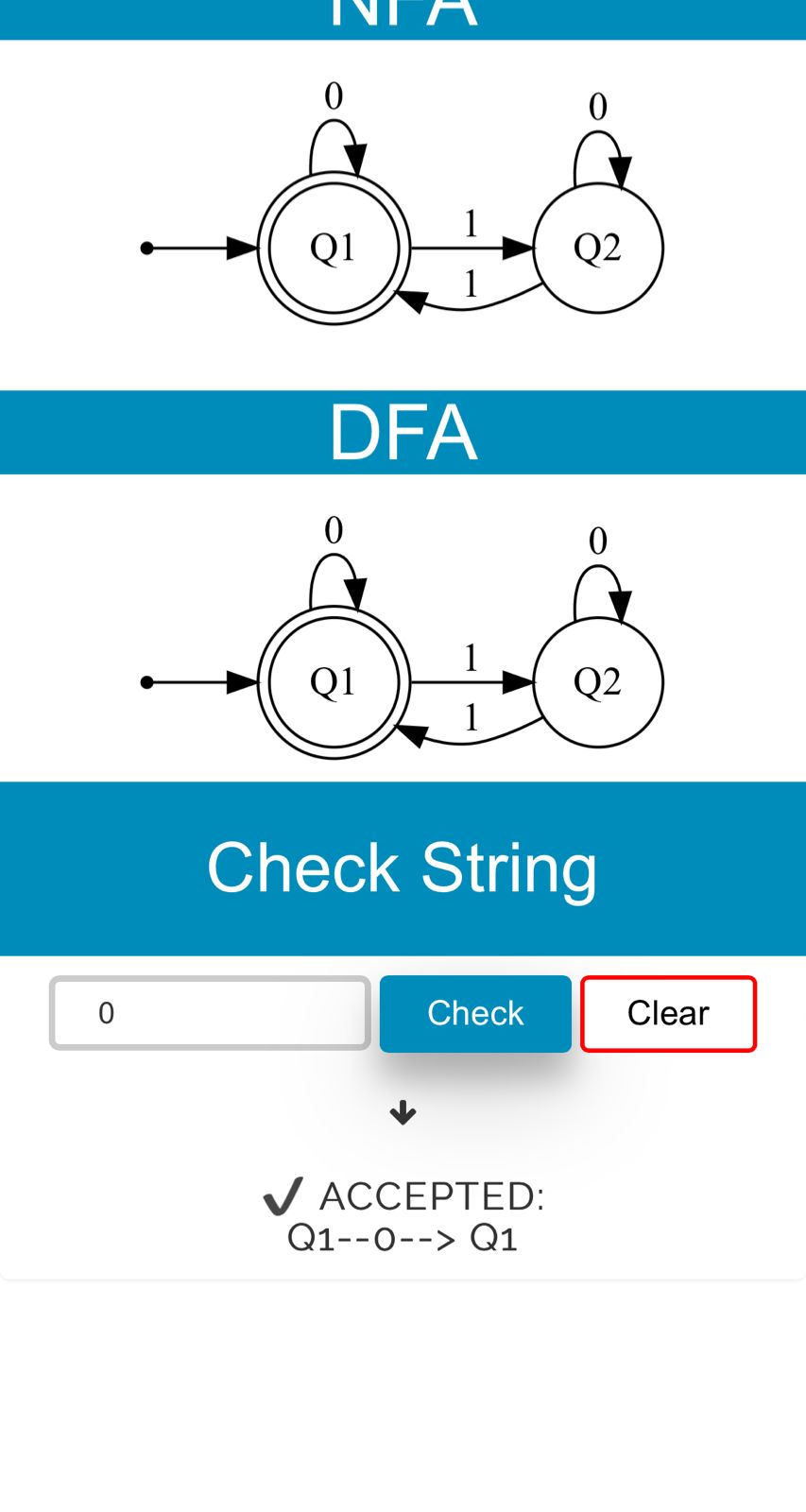
****

Figure 15: Learn more on Mobile

Figure 14: Check String on Mobile

**5. Technologies Used**



HTML (Hypertext Markup Language) is the most basic building block of the Web. It defines the meaning and structure of web content. ... HTML uses "markup" to annotate text, images, and other content for display in a Web browser.

Logo

Description automatically generated

JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. ... Incorporating JavaScript improves the user experience of the web page by converting it from a static page into an interactive one.

Logo

Description automatically generated

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML.

Logo

Description automatically generated

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers.

Graphviz is open-source graph visualization software. Graph visualization is a way of representing structural information as diagrams of abstract graphs and networks. It has important applications in networking, bioinformatics, software engineering, database and web design, machine learning, and in visual interfaces for other technical domains.Icon

Description automatically generated

**Logo

Description automatically generated**GitHub is a web-based version-control and collaboration platform for software developers. Git is used to store the source code for a project and track the complete history of all changes to that code. It allows developers to collaborate on a project more effectively by providing tools for managing possibly conflicting changes from multiple developers.

W3Schools is a freemium educational website for learning coding online. Developed in 1998, its name is derived from the World Wide Web. W3Schools offers courses covering all aspects of web development.

**Icon

Description automatically generated**

Visual Studio Code is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.

**6. FOUR APPLICATIONS THAT USE FSMs**

There are many applications that uses finite state machines (FSMs), for example, FSMs are used in vending machines, video games, traffic lights, text parsing, language processing, etc.…

**Text parsing meaning:**

Text parsing is the process of analyzing a string of symbols either in computer languages or data structures conforming to the rules of a formal grammar. It is the process of recognizing the shape of a particular input string by breaking down a certain text into its component parts of speech.

For example:

Value1=” It is a string” //string

Value2=12345 //integer

Value3= /Desktop/home/test1.txt //path

Value4= apple, orange, banana //array

Value5= true //Boolean

**How FSMs are used in text parsing:**

In FSMs a text is parsed in a way that based on a given input the finite state machine will do the transition and go from a state to another. This act can be done using either NFA or DFA.

**Text parsing using NFA:**

In NFA, we need to follow all transitions from a state to another, and if a path fails (doesn’t reach a final state) we need to check another path until passing over all paths.

Diagram

Description automatically generatedExample:

Figure 10:text parsing using NFA

**Text parsing using DFA:**

In DFA, a text is parsed in a way that there is no backtracking and each text line is processed character by character, and when the text is fully processed and the DFA is in the final state, the parsing is successfully completed.

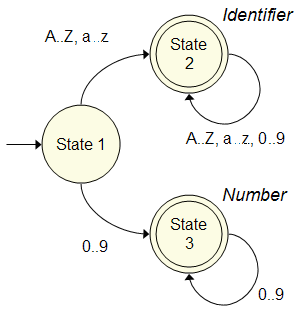
Ex:

Figure 11:parsing using DFA

**FSMs used in Super Mario Game:**

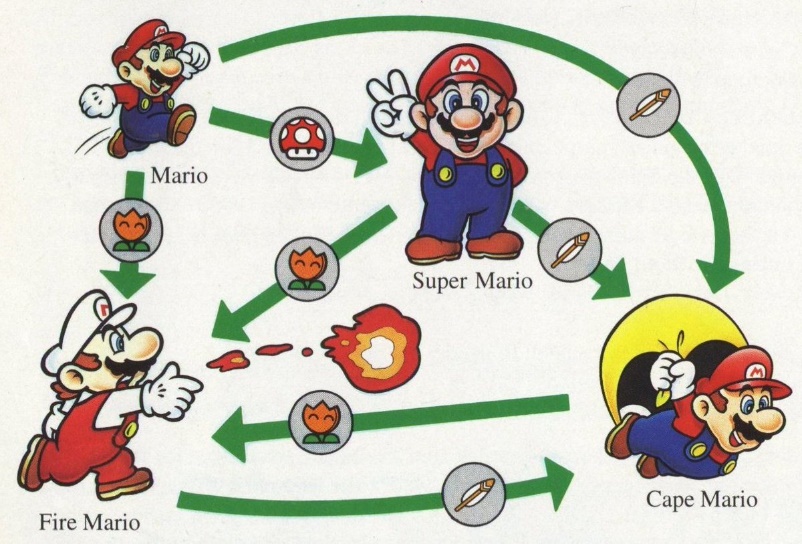
Most people are already aware of this nostalgic game. In this game, Mario changes his states and behavior based on the events which occurred, which you can see in the below image.

Figure 12: Super Mario states

**States**

1. Mario (We will refer as Small Mario hereafter)
2. Super Mario
3. Fire Mario
4. Cape Mario
5. Lost Life (Apart from the image considering this state)

**Events**

1. Got Mushroom
2. Got Fire Flower
3. Got Feather
4. Met Monster (Not shown in image)

**State Transition on Event Occurrence & Earning Coins**

The following table demonstrates how the state changes with different events.

Apart from the state change, coins are also earned on the occurrence of events.

|  |  |  |  |
| --- | --- | --- | --- |
| **Current State** | **Event Occurred** | **New State** | **Coins Earned** |
| Small Mario | Got Mushroom | Super Mario | 100 |
| Small Mario | Got Fire Flower | Fire Mario | 200 |
| Small Mario | Got Feather | Cape Mario | 300 |
| Small Mario | Met Monster | Lost Life | 0 |
| Super Mario | Got Mushroom | Super Mario | 100 |
| Super Mario | Got Fire Flower | Fire Mario | 200 |
| Super Mario | Got Feather | Cape Mario | 300 |
| Super Mario | Met Monster | Small Mario | 0 |
| Fire Mario | Got Mushroom | Fire Mario | 100 |
| Fire Mario | Got Fire Flower | Fire Mario | 200 |
| Fire Mario | Got Feather | Cape Mario | 300 |
| Fire Mario | Met Monster | Small Mario | 0 |
| Cape Mario | Got Mushroom | Cape Mario | 100 |
| Cape Mario | Got Fire Flower | Fire Mario | 200 |
| Cape Mario | Got Feather | Cape Mario | 300 |
| Cape Mario | Met Monster | Small Mario | 0 |

**Automatic door FSM:**

The automatic door works as follow:

If the door is closed: if no one is standing in front or behind the door, the door remains closed, and if someone is standing in front of the door the door should open, and when someone is standing behind the door or two persons, one

standing in front of the door and the other behind the door the door should remain closed.

If the door is open

if no one is standing in front or behind the door, the door should close, and if someone is standing in front of the door the door remains open, and when someone is standing behind the door or two persons one is standing in front of the door and the other behind the door the door should remain opened.

Q= {Closed, Open}

∑= {front, behind, both, neither}

δ = transition function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| State name | Front | Behind | Both | Neither |
| Closed | Open | Closed | Closed | Closed |
| Open | Open | Open | Open | Closed |

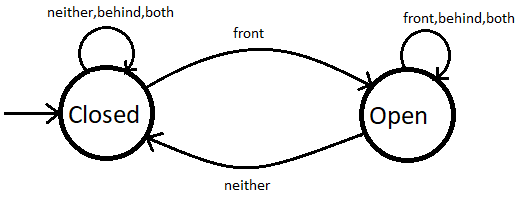


Figure 13:Automatic door FSM

**FSM used in vending machine:**

A vending machine is an automatic machine that takes certain amount of money and give people in return snacks, drinks…

This process can be done using finite state machine.

**An example of a vending machine:**

Characteristics:

This vending machine works as follows:

1) It receives coins of 1,2, &5

2) it gives drinks or snacks when the total amount of inserted money is equal or greater than 5.

3) This machine will not return money when the user inserts a higher amount into the machine.

Q= {q0, q1, q2, q3, q4, q5}

∑= {1,2,5}

L= {set of all strings over {1,2,5} that their summation is greater than or equal to 5}

Ex:

L= {5,11111,1212,221,2222,51,15…}

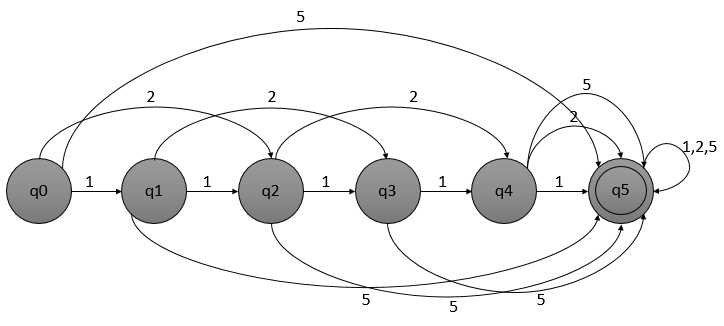


Figure 14:DFA of Language L

**7. CONCLUSION**

In conclusion, our project’ aim is developing a web application that is very helpful for users to draw FSMs and to convert NFA to DFA in the easiest way. We made the graphical user interface in the clearest way because our purpose is to facilitate the work of the users and to save their valuable time.

FSMs are very important, we use them in our day to day life, as we have mentioned they are used in many fields: electric doors, vending machine, heating system, etc…

**8. REFERENCES**

[1] El Abed, M. (2021) Theory of Computation @ Rafik Hariri University.

[2] <https://www.w3schools.com/>

[3] <https://fontawesome.com/v4.7/icons/>

[4] <https://graphviz.org/>

[5] <https://www.geeksforgeeks.org/introduction-of-finite-automata/>

[6] <https://hackernoon.com/high-performance-text-parsing-using-finite-state-machines-fsm-6d3m33j9>

[7] <https://www.c-sharpcorner.com/article/understanding-state-design-pattern-by-implementing-finite-state/>

**9. APPENDIX**

