International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 10 Issue: 01 | Jan 2023 www.irjet.net p-ISSN: 2395-0072

DFA CREATOR AND STRING TESTER

Shravani Phadol[1], Shubhechha Mehere[2], Prerna Divekar[3], Yash Gaikwad[4]

[1,2,3,4] *Student, Dept. of Information Technology, Vishwakarma Institute of Technology, Pune, Maharashtra, India*

Abstract - ***Machines and their working is one the vital and major concepts to understand and work on. DFA which is Deterministic Finite Automata is a very easy and understandable concept, to learn the working of different real-life machines. To visualize the working efficiency of DFA models, this web-based platform helps the end user to create various such DFAs using a very user-friendly GUI. To simplify DFA working the project is divided into the following parts: the creation of states, creating transitions, inserting states, and deleting states. After completing the design of the machine the user can test different input strings for understanding the language and working of the machine. The test results are shown simultaneously to the user. This web project focuses on the practical and hands-on implementation of different concepts of DFA including states, transitions, final states, and initial states. To simplify the understanding and working of DFA and its flow control, users can use the drag and move functionality.***

Key Words: **DFA, Automata, Machine, Web-based**

1. INTRODUCTION

DFA which is a vital part of automation theory is almost a theoretical concept. Many of its contents such as states their types, transition between two states, final states, and their symbolic representation can be understood easily by ideating the visuals. This project emphasizes the actual working of a machine. Learning these machines and their working on pen and paper becomes tough when the number of states and transitions increases proportionally. To soothe the learning process of building DFA, this web app gives GUI based platform where the addition of states and connecting them becomes easy by just clicking on buttons. The user can parallelly make a state final, add a new state, or delete an added state by using the respective buttons. Building such DFA's will help to understand the working of real-time and real-usage machines and enhance concept building. The project also has a test string functionality where the user can test, which strings are accepted by the designed DFA, which furtherly helps to identify the language accepted by the DFA. This type of interaction erases the barriers that stand up while learning the functionality of machines. DFA which is popularly used by different compilers explains the mechanism of validation which is used in different fields such as password checking. The main intent of the project is to make a cognizant understanding of the working of machines.

2. LITERATURE REVIEW

[1] In this article, the authors describe the development of a tool for visualizing and debugging deterministic finite-state machines (DFAs). The tool allows users to construct DFAs by specifying the states and transitions of the machine, and it provides a visual representation of the DFA as it processes input. The tool also includes a feature for generating code in the finite state machine (FSM) language, which allows users to execute their edited machines. The authors proposed a mechanism for visualizing and verifying the behavior of formal machines during debugging, and they claim that their tool is unique in its ability to generate code for state-based machines. This tool could be useful for educators, students, or researchers working with DFAs, as it provides a more intuitive and interactive way of constructing and understanding these machines.

[2] An Efficient Protocol for Oblivious DFA Evaluation and Applications it designs an efficient tool for the evaluation of deterministic finite automata (DFA) between one input holder and a DFA second holder. It also designs a protocol to compute the counting number of accepting states for evaluation of DFA on an input. To solve the counting variant of states it introduces a novel modification for protocol. It implements a protocol and runs the experiments for the evaluation of the DFA string.

[3] This article described a tool that allows users to experiment with finite automata and generate lists of strings in the language recognized by the automaton. The tool also has the ability to perform various conversions, such as converting an NFA (nondeterministic finite automaton) to a DFA (deterministic finite automaton). In automata theory, a finite automaton is a mathematical model used to recognize patterns within the input. It consists of a finite set of states, a set of input symbols, a transition function, a start state, and a set of accepting states. The automaton reads input symbols one at a time, transitioning from one state to another according to the transition function until it either accepts or rejects the input.

[4] Dynamic testing is a software testing technique that involves testing a system while it is executing. The goal of dynamic testing is to evaluate the behavior of a system at runtime and ensure that it is functioning as expected. Automata learning is a technique used to learn the behavior of a system by constructing a finite automaton that represents the system's behavior. This can be useful for testing purposes, as the automaton can be used to generate

test cases that cover a wide range of behaviors exhibited by the system.

[5] In this article, the authors described the usage of finite state automata to detect sensitive words in a text string. The automata is trained to recognize a large number of keywords and can be used to improve the efficiency of the system. Error recognition mechanisms are also implemented to reduce the error rate and improve the performance of the system. Finite state automata is a useful tool for recognizing patterns in input and can be applied to a wide range of applications, including text processing and language recognition. In this case, the automata is used to identify sensitive words in a text string, which could be useful in the context of distance education.

[6] In automata theory, a deterministic finite automaton (DFA) is a type of finite automaton that reads input symbols one at a time and transitions from one state to another according to a fixed set of rules. A DFA can be represented using a finite state machine, which consists of a set of states, a set of input symbols, a transition function, a start state, and a set of accepting states. Binary strings can be used to represent the transition function and the set of accepted states of a DFA. Context-free grammars (CFGs) are a formalism used to describe the structure of programming languages. They consist of a set of terminal symbols, which represent the basic elements of the language, and a set of nonterminal symbols, which represent more complex structures. A CFG can be used to generate the set of all possible strings that belong to the language it describes.

3. METHODOLOGY

A) Proposed system

The main aim of designing this application is, for learners to understand the DFA concept. This system allows users to construct or design a DFA according to problem statement. Users can add states as per need and give transitions to the states. Then give input strings between the states and test the string whether it is valid or not. This application allows users to delete a state and make a final state. This application is easy to understand and easy to use for learners. This project is developed using HTML, CSS, and JavaScript.

B) Flowchart

This application consists of two major concepts that include constructing a DFA and testing a DFA. To construct a DFA four major steps are considered including add state, state transition, make final state, and delete state.

i) Design DFA

In DFA (deterministic finite automata) first, add an initial state for example q1 and then add the other states following the initial state for example q1,q2,q3, etc. Then add transitions between the initial state and the following states in the DFA. In the transition between the states add input strings for example 1,0, a, b, etc. From the initial state and following states make a final state in the DFA. Then the design of DFA is complete.

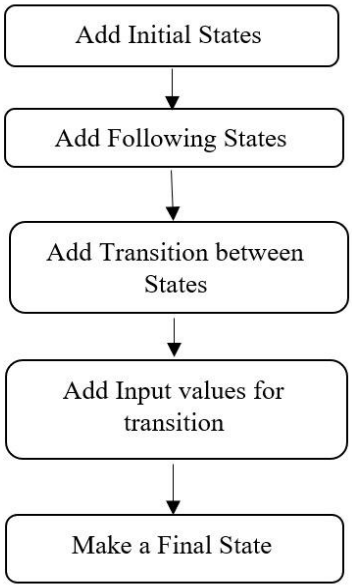

Fig 1: Design DFA

ii) Testing

First, we have to enter the input string from the initial state to other states transitions then user enters string to test whether it is valid or not for the constructed DFA. In testing if the input string is valid then it will be accepted by the DFA otherwise it will be rejected by the DFA.

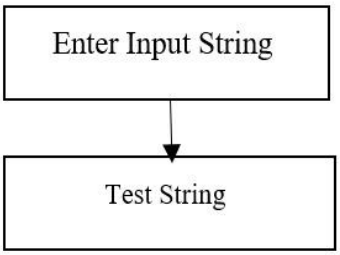

Fig 2: Testing

iii) State Transition

After constructing the DFA, add the initial states and the following states then user adds the transition from the current state to the next state and gives input strings. We have to specify valid input strings for given states and then connect the states.

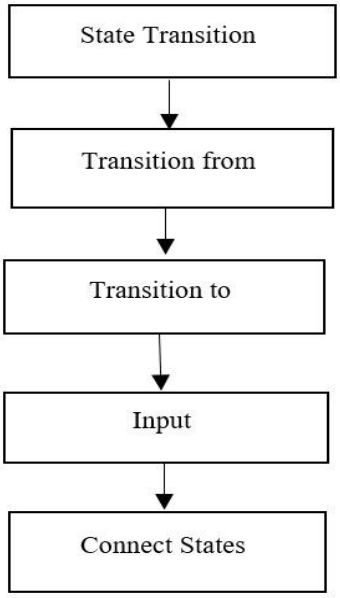

Fig3: State Transition

iv) Delete a state

After the design of DFA deleting a state is possible. Select one of the constructed states, then select a state to delete and click on delete state button.

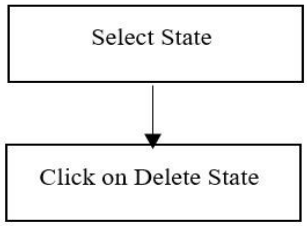

Fig 4: Delete State

v) Make A final State

Select one of the states to make a final state from the DFA and then click on make final state. This functionality allows the user to make more than one state as a final state.

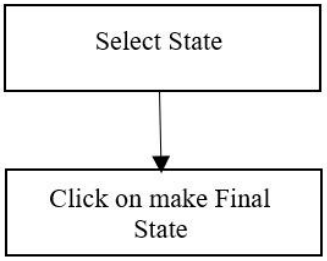

Fig 5: Make a Final State

Algorithm:

Step 1: Add initial state.

Here the user adds the starting state.

Step 2: Add the following states.

In this step, the user adds the other states.

Step 3: Add transition between states.

Here the user adds the transition between the added states. Similarly, the user gives the input values for the transitions.

Step 4: Make a state final.

Here user makes a state as final, the user can make one or more states as final.

Step 5: Test input string for designed DFA.

Here the user checks which string is accepted by the DFA and which is not

4. RESULTS AND DISCUSSIONS

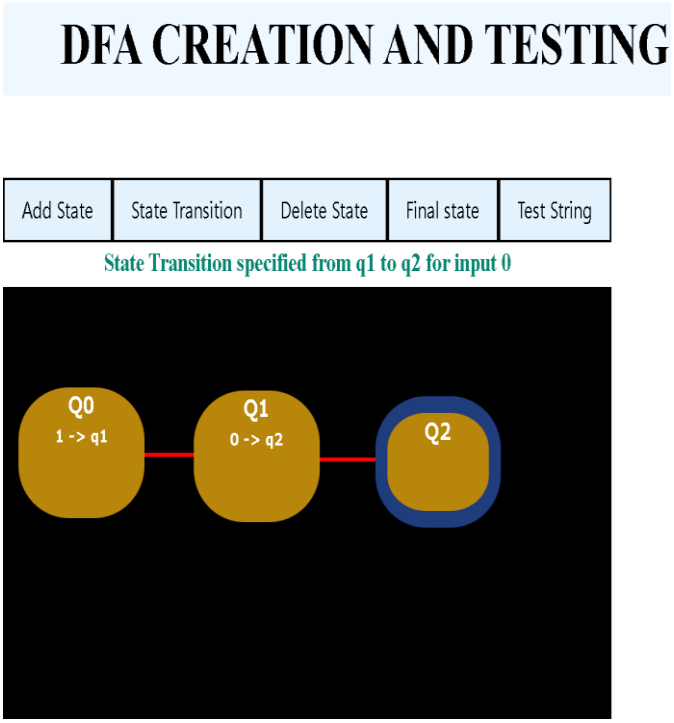

Fig 6: Constructed DFA

In fig 6: the user constructed DFA by adding states by using Add state button. Then the user adds state transition from q0 to q1 and q1 to q2 and gives input as 1, 0. The user makes a final state as Q2. By using Test string button the user checks whether the input string entered is accepted or not .

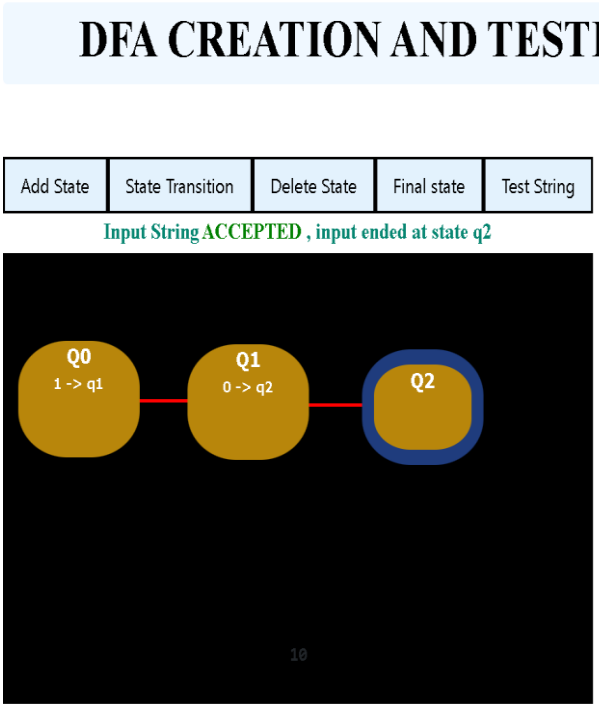

Fig 7: Output 1

In fig 7: the input string given from Q0 to Q1 is 1 and from Q1 to Q2 is 0. The input string 10 is tested and it is accepted as shown in the above figure.

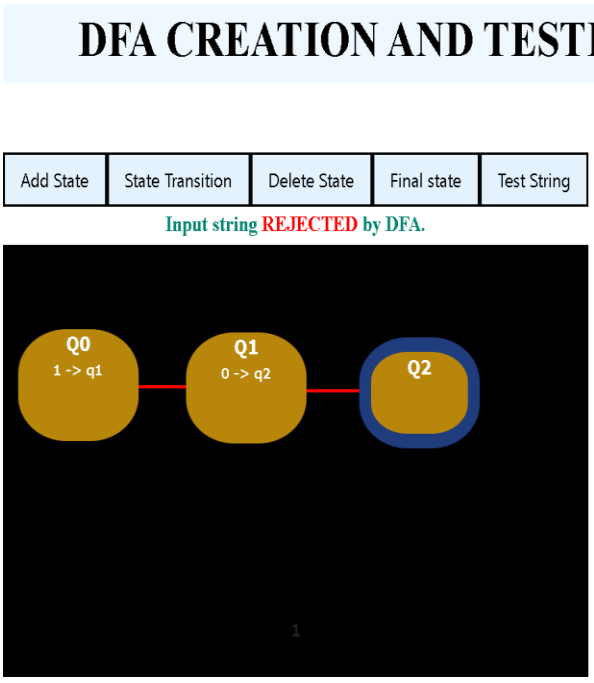

Fig 8: Output 2

In fig 8: the input string entered from Q0 to Q1 is 1 and from Q1 to Q2 is 0. The input string 1 is tested and it is rejected as Q1 is not a final state as shown in the above figure.

5. SCOPE OF PROJECT

In this project DFA Creator and String Tester, we can only design DFA and test input strings. In the future, we can add many operations and functions like an NFA generator, and PDA, construct a DFA with a given regular expression, and conversion of DFA to NFA.

6. CONCLUSION

In this paper, we described the use of a DFA Creator and String Tester for learners. Learners can understand practically how DFA works by using this application. Users can construct or design a DFA as per need, make transitions, give input string, make a final state, and test the DFA. They can practically perform operations and generate DFA from this system.

REFERENCES

[1] Visual Designing and Debugging of Deterministic Finite-State Machines in FSM [2008.09254] Visual Designing and Debugging of Deterministic Finite-State Machines in FSM (arxiv.org)

[2] An Efficient Protocol for Oblivious DFA Evaluation and Applications An Efficient Protocol for Oblivious DFA Evaluation and Applications | SpringerLink

[3] A Collection of Tools for Making Automata Theory and Formal Languages Come Alive A collection of tools for making automata theory and formal languages come alive | ACM SIGCSE Bulletin

[4] Dynamic testing via automata learning Dynamic Testing Via Automata Learning | SpringerLink

[5] Application Research of Finite Automaton in Distance Education Application research of finite automaton in distance education | IEEE Conference Publication | IEEE Xplore

[6] A General Approach to DFA Construction(PDF) A General Approach to DFA Construction (researchgate.net)

[7] Kuldeep Vayadande, Aditya Bodhankar, Ajinkya Mahajan, Diksha Prasad, Shivani Mahajan, Aishwarya Pujari and Riya Dhakalkar, "Classification of Depression on social media using Distant Supervision", ITM Web Conf. Volume 50, 2022.

[8]Kuldeep Vayadande, Rahebar Shaikh, Suraj Rothe, Sangam Patil, Tanuj Barware and Sameer Naik," Blockchain-Based Land Record System", ITM Web Conf. Volume 50, 2022.

[9] Kuldeep Vayadande, Kirti Agarwal, Aadesh Kabra, Ketan Gangwal and Atharv Kinage," Cryptography using Automata Theory", ITM Web Conf. Volume 50, 2022

[10] Samruddhi Mumbare, Kunal Shivam, Priyanka Lokhande, Samruddhi Zaware, Varad Deshpande and Kuldeep Vayadande," Software Controller using Hand Gestures", ITM Web Conf. Volume 50, 2022

[11] Preetham, H. D., and Kuldeep Baban Vayadande. "Online Crime Reporting System Using Python Django."

[12] Vayadande, Kuldeep B., et al. "Simulation and Testing of Deterministic Finite Automata Machine." International Journal of Computer Sciences and Engineering 10.1 (2022): 13-17.

[13] Vayadande, Kuldeep, et al. "Modulo Calculator Using Tkinter Library." EasyChair Preprint 7578 (2022).

[14] VAYADANDE, KULDEEP. "Simulating Derivations of Context-Free Grammar." (2022).

[15] Vayadande, Kuldeep, Ram Mandhana, Kaustubh Paralkar, Dhananjay Pawal, Siddhant Deshpande, and Vishal Sonkusale. "Pattern Matching in File System." International Journal of Computer Applications 975: 8887.

[16] Vayadande, Kuldeep, Ritesh Pokarne, Mahalakshmi Phaldesai, Tanushri Bhuruk, Tanmay Patil, and Prachi Kumar. "Simulation Of Conway's Game Of Life Using Cellular Automata." SIMULATION 9, no. 01 (2022).

[17] Gurav, Rohit, Sakshi Suryawanshi, Parth Narkhede, Sankalp Patil, Sejal Hukare, and Kuldeep Vayadande. "Universal Turing machine simulator." International Journal of Advance Research, Ideas and Innovations in Technology, ISSN (2022).

[18] Vayadande, Kuldeep B., Parth Sheth, Arvind Shelke, Vaishnavi Patil, Srushti Shevate, and Chinmayee Sawakare. "Simulation and Testing of Deterministic Finite Automata Machine." International Journal of Computer Sciences and Engineering 10, no. 1 (2022): 13-17.

[19] Vayadande, Kuldeep, Ram Mandhana, Kaustubh Paralkar, Dhananjay Pawal, Siddhant Deshpande, and Vishal Sonkusale. "Pattern Matching in File System." International Journal of Computer Applications 975: 8887.

[20] Vayadande, Kuldeep B., and Surendra Yadav. "A Review paper on Detection of Moving Object in Dynamic Background." International Journal of Computer Sciences and Engineering 6, no. 9 (2018): 877-880.

[21] Vayadande, Kuldeep, Neha Bhavar, Sayee Chauhan, Sushrut Kulkarni, Abhijit Thorat, and Yash Annapure. Spell Checker Model for String Comparison in Automata. No. 7375. EasyChair, 2022.

[22] Vayadande, Kuldeep, Harshwardhan More, Omkar More, Shubham Mulay, Atharva Pathak, and Vishwam Talnikar. "Pac Man: Game Development using PDA and OOP." (2022).

[23] Preetham, H. D., and Kuldeep Baban Vayadande. "Online Crime Reporting System Using Python Django."

[24] Vayadande, Kuldeep. "Harshwardhan More, Omkar More, Shubham Mulay, Atahrv Pathak, Vishwam Talanikar,"Pac Man: Game Development using PDA and OOP"." International Research Journal of Engineering and Technology (IRJET), e-ISSN (2022): 2395-0056.

[25] Ingale, Varad, Kuldeep Vayadande, Vivek Verma, Abhishek Yeole, Sahil Zawar, and Zoya Jamadar. "Lexical analyzer using DFA." International Journal of Advance Research, Ideas and Innovations in Technology, www. IJARIIT. com.

[26] Manjramkar, Devang, Adwait Gharpure, Aayush Gore, Ishan Gujarathi, and Dhananjay Deore. "A Review Paper on Document text search based on nondeterministic automata." (2022).

[27] Chandra, Arunav, Aashay Bongulwar, Aayush Jadhav, Rishikesh Ahire, Amogh Dumbre, Sumaan Ali, Anveshika Kamble, Rohit Arole, Bijin Jiby, and Sukhpreet Bhatti. Survey on Randomly Generating English Sentences. No. 7655. EasyChair, 2022.