វិទ្យាស្ថានបច្ចេកវិទ្យាកម្ពុជា

2022-2023

AUTOMATA

PROJECT: FINITE AUTOMATION (FA)

LECTURER

VALY DONA

COHORT

Tom Tito
e20200061

Veasna Dara
e20200268

Teang Sreyroth
e20200826

Sorporn Sovortey
e20200988

Ty Sopheaktra
e20200958

Content of Table

I. INTRODUCTION

The purpose of this technical report is to provide an overview of the Finite Automata project. The project aimed to develop a graphical application that allows users to work with Finite Automata, perform various operations, and save the automata data to a MySQL database. The project primarily focused on designing and implementing functionalities related to deterministic and non-deterministic finite automata. The report will detail the functions and features, data structure, database design, implementation process, and the final result achieved.

II. FUNCTION AND FEATURE

The developed application, named "Finite Automaton App," offers the following key functions and features:

- **Create Finite Automata**: Users can input the states, alphabet, transitions, initial state, and accepting states to create a finite automaton.
- **Check Deterministic**: The application can determine if the created finite automaton is deterministic or non-deterministic.
- **Check Acceptance**: Users can input a string to check if it is accepted by the finite automaton.
- **Convert NFA to DFA:** The application can convert a non-deterministic finite automaton (NFA) to a deterministic finite automaton (DFA).
- **Minimize DFA:** It allows users to minimize the created DFA using a minimization algorithm.
- **Save and Load to Database:** The application can save the finite automaton data to a MySQL database and load saved data when needed.
- **Design UI:** The user can use the application.

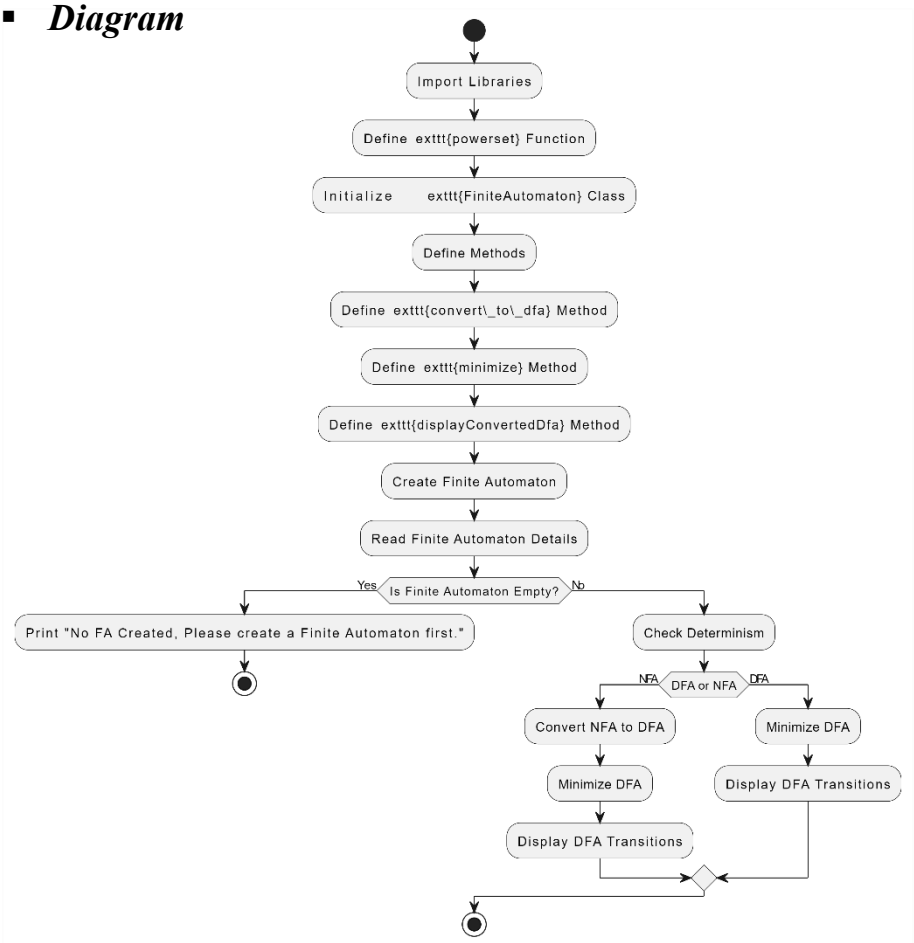
III. DATA STRUCTURE

The primary data structure used in the project is the FiniteAutomaton class. The class encapsulates the necessary attributes and methods to represent a finite automaton. The key attributes of the FiniteAutomaton class include:

- **states**: A set to store the states of the automaton.
- **alphabet**: A set to store the symbols of the alphabet.
- **transitions**: A dictionary to represent transitions between states and symbols.
- **initial_state**: A single state representing the initial state of the automaton.
- **accepting_states:** A set to store the accepting states of the automaton.

```
def __init__(self):
    self.states = set()
    self.alphabet = set()
    self.transitions = {}
    self.initial_state = None
    self.accepting_states = set()
```

- ***Diagram***



IV. DATABASE DESIGN

The application utilizes a MySQL database to store and manage Finite Automata data. The database design consists of a single table named "data," which stores the information for each Finite Automaton. The table includes the following columns:

- **id**: An auto-incrementing primary key to uniquely identify each record.
- **state**: A varchar column to store the states of the automaton as a comma-separated string.
- **alphabet**: A varchar column to store the alphabet symbols as a comma-separated string.
- **transition**: A varchar column to store the transitions in the format "from_state,symbol,to_state" for each row.
- **initial**: A varchar column to store the initial state of the automaton.
- **accepting_state**: A varchar column to store the accepting states as a comma-separated string.

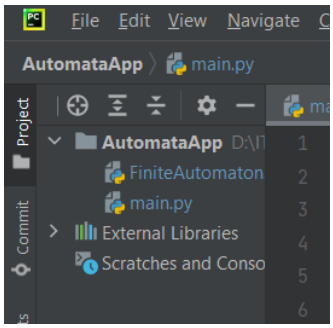
V. IMPLEMENTATION

The application was implemented using Python programming language and the Tkinter library for the graphical user interface. The FiniteAutomaton class provides essential methods for handling automata operations, including checking determinism, acceptance of input strings, conversion from NFA to DFA, and DFA minimization. The GUI elements, such as labels, entry fields, buttons, and text fields, were arranged using Tkinter to create an intuitive user interface.

◈ Tools: PyCharm, Git, GitHub

◈ Programming: Python, Library Tkinter and more modules

◉ Project Structure

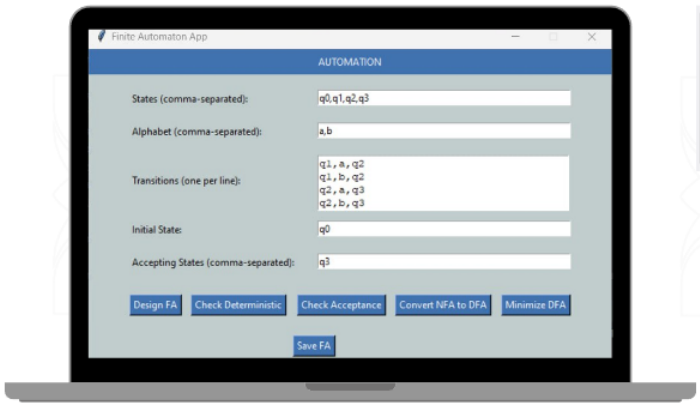


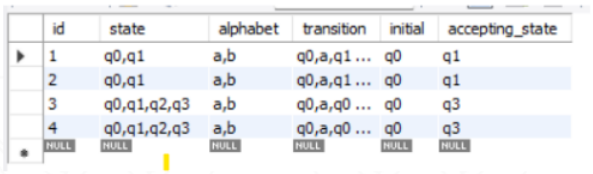
◉ Project Structure

- ○ FiniteAutomaton
- ○ FiniteAutomatonApp

VI. RESULT

The Finite Automaton App successfully achieved the project objectives. Users can create finite automata, check their determinism, acceptance of strings, convert NFA to DFA, and minimize DFA. The application allows users to save the finite automaton data to a MySQL database for future reference.





id	state	alphabet	transition	initial	accepting_state
1	q0,q1	a,b	q0,a,q1 ...	q0	q1
2	q0,q1	a,b	q0,a,q1 ...	q0	q1
3	q0,q1,q2,q3	a,b	q0,a,q0 ...	q0	q3
4	q0,q1,q2,q3	a,b	q0,a,q0 ...	q0	q3
NULL	NULL	NULL	NULL	NULL	NULL

VII. CONCLUSION AND PERSPECTIVE

- Conclusion

In conclusion, the Finite Automata project is the valuable project that provide us to understand the theory of finite automaton such as create FA, check finite automaton, check accept string, convert NFA to DFA and minimize DFA.

- Perspective
  - It is unable to show the minimized result.
  - UI displays are ineffective.
  - It will eventually be finished by us.

VIII. ACTIVITY HISTORY RECORD

Date	Time	Location	Attendance	Topic
17-June	1PM - 3PM	Online	Everyone	Topic 1: Discussion about project Topic 2: Project setup
24-June	2PM - 3PM	ITC	Everyone	Topic 1: Discuss project progress.
4-July	1PM - 3PM	ITC	Everyone	Topic 1: Discuss theory in lesson. Topic 2: how to use Tkinter
16-July	1PM - 4PM	Online	Everyone	Topic 1: Integrated code Topic 2: Test and modify.
28-July	1PM - 5PM	ITC	Everyone	Topic 1: Project complete Topic 2: Report and slide submission.