CS.321 PROJECT

OUR TEAM:

NAME: Alowida Elsayed ID:200042786

NAME: Seif Adel Elbasha ID:200046355

==========

DFA MINIMIZATION VISUAL TOOL

1-Relevance

This project is directly related to the *Theory of Computation* because it applies fundamental concepts of finite automata, regular languages, and state minimization, which are core topics of the course. In class, we study how Deterministic Finite Automata (DFAs) recognize regular languages and how different DFAs can represent the same language. The project demonstrates this by implementing the formal minimization algorithm that merges equivalent states based on indistinguishability.
By building a DFA Minimization Tool, we translate theoretical ideas—such as reachable states, equivalence classes, and partition refinement—into an actual working program. This shows how the abstract mathematical models taught in the course can be applied in real computation, pattern matching, and compiler design.

==========

2-Effort

Our project was completed collaboratively by Owida and Seif, with both members contributing to the design, implementation, testing, and documentation of the DFA Minimization Tool.

- **Owida & Seif**:
  Took the lead on implementing the core algorithms of the project, including removing unreachable states and applying the partition refinement method for DFA minimization. Worked on structuring the code, handling the transition table, and ensuring correctness of the minimization process.
- **Seif & Owida**:
  Focused on input handling, testing various DFA examples, and verifying the correctness of the minimized output, also responsible for formatting the final results, improving readability, and assisting in preparing the report and explanation of the algorithm steps.

Our team members reviewed each other's work, debugged the implementation, and discussed design choices to ensure the tool was accurate and aligned with the concepts learned in the Theory of Computation course.

==========

3-Content

This project implements a complete tool for minimizing a Deterministic Finite Automaton (DFA) using concepts learned in the Theory of Computation course. The tool takes a DFA as input—defined by its states, alphabet, start state, accepting states, and transition function—and produces an equivalent DFA with the minimum possible number of states.

The project consists of three main components:

1. **Preprocessing and Validation:**
   The tool first verifies that the given DFA is well-defined and removes

any unreachable states. This step ensures that the minimization process is applied only to the useful part of the automaton.

2. **DFA Minimization Algorithm:**
   The core of the project is the implementation of the partition refinement algorithm (also known as state equivalence minimization). The algorithm begins by separating accepting and non-accepting states, then repeatedly refines these groups based on transitions until no further splitting is possible. Each resulting group represents a single state in the minimized DFA.

3. **Output Generation:**
   After minimization, the tool constructs and prints the new DFA, including the minimized set of states, the updated transitions, the new start state, and the new accepting states. The output is formatted clearly to show the final structure of the minimized automaton.

The project demonstrates the full process of taking a theoretical concept—DFA equivalence and minimization—and turning it into an operational software tool. It reinforces key ideas such as regular languages, automata behavior, state equivalence, and formal algorithms used in compilers, text processing, and pattern recognition.

==========

Code:

https://github.com/owida676-lang/cs321.git