Project Title: Improved Visualization for Formal Languages

Group Members: Chris Pinto-Font (cpintofont2021@my.fit.edu), Vincent Borrelli (vborrelli2022@my.fit.edu), Andrew Bastien (abastien2021@my.fit.edu), Keegan McNear (kmcnear2022@my.fit.edu)

Faculty Advisor: Dr. Luginbuhl (dlugunbuhl@fit.edu)

Client: Dr. Luginbuhl

Meeting time(s): MWF @ 5 pm

Goal and motivation: Design and program a user-friendly application that can visualize Deterministic Finite Automata (DFA). The user should be able to accurately plot a DFA and animate its execution as a learning aid. The current application, which our client hopes for us to replace, is clunky and confusing, it lacks some features a teacher or student would want or need in plotting and exploring DFAs. Our goal is to design an application which is user friendly, mechanically intuitive, visually striking, and highly legible.

Approach: Key features

1. Able to create a DFA intuitively. - A GUI editor allowing the user to create a DFA intuitively. The software must be able to plot a DFA, designate initial, final, and dead states, as well as track states and transitions based on an entered grammatically valid string. The DFA is created when a set of strings is given, and the system would then create a minimal DFA that accepts or rejects said string. It also should allow the user to plot a DFA manually via a system inspired by the previous outdated JFLAP application.

2. Animation of the DFA - This feature is key to be able to visualize the formation of the DFA. Such an animated process should show the step by step execution. Having an animation that can effectively show how a DFA is created is key to showing the user how a DFA is implemented. Such a tool would make the program an effective teaching tool to better communicate how one should be able to trace a DFA and read one effectively.

3. Minimization and completeness - If provided an incomplete DFA, our ideal program would be able to complete it automatically by possibly deducing missing transitions. Being able to use well-known algorithms to reduce a DFA's complexity is also essential.

4. Intuitive and improved GUI - Based on intended customer feedback, a key reason this program is needed is to provide a more intuitive and easily navigated set of tools than what was seen in a previous outdated application, thus we will build an easily used

"toolbox" component of the gui inspired by applications such as microsoft paint where in which the user can easily see what they can do and even hover their cursor over symbols to gain a better understanding of said tools.

Novel Features: Minimization and completeness. Multiple symbols on a single transition between arcs in a way that is natural and readable. Animation of DFAs in a unique and legible manner. In-Program documentation files which let the user read and learn more about the program without need to explore a separate "read me" file, which is often a confusing place for additional information for a layman user of such a program.

Algorithms and tools: The main application will be written in Python. A GUI widgets library hasn't been decided upon yet, but there are many excellent choices like Python bindings to Qt. In the unlikely situation performance is an issue, heavy processing can be offloaded to extension modules written in Cython.

Technical challenges: Our team is not accustomed to creating animations, so it will be a learning experience to fully understand and implement it. We also need to play with a User Interface to make it as simple and user-friendly as possible. As well as learning how to minimize a DFA, as we were not taught how to, but will have to understand to fully implement it in our system.

Milestone I (Sep 29):

- Compare and select technical tools for *A, B, C, ...*
- Provide small ("hello world") demo(s) to evaluate the tools for *A, B, C, ...*
- Resolve technical challenges: *X, Y, Z, ...*
- Compare and select collaboration tools for software development, documents/presentations, communication, task calendar
- Create Requirement Document
- Create Design Document
- Create Test Plan

Milestone II (Oct 27):

- GUI
- DFA logic implementation
- DFA formatting

Milestone III (Nov 24):

- DFA animation
- Proper logic for a complete and minimal DFA

Task Matrix for Milestone I:

Task	Chris	Vincent	Andrew	Keegan
Compare and select Technical Tools	Research: 25%	Research: 25%	Testing: 25%	Testing: 25%
"hello world" demos	Coding: 25%	Coding: 25%	Coding: 25%	Coding: 25%
Resolve Technical Challenges	Research: 25%	Research: 25%	Coding: 25%	Coding: 25%
Compare and select Collaboration Tools	Microsoft Teams	Google Drive	Discord	GitHub
Requirement Document	Writing: 30%	Writing: 30%	Writing: 15%	Writing: 25%
Design Document	Writing: 15%	Writing: 25%	Writing: 35%	Writing: 25%
Test Plan	Writing: 20%	Writing: 30%	Writing: 20%	Writing: 30%

Approval from Faculty Advisor:

- "I have discussed with the team and approve this project plan. I will evaluate the progress and assign a grade for each of the three milestones."
- Signature: _____ Date: _____