Milestone 6 Progress Evaluation

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 (dluginbuhl@fit.edu)

Client: Dr.Luginbuhl

Milestone #6 Progress:

Task	Completion %	Chris	Vincent	Andrew	Keegan	To Do
Finalized Touches on user experience/ bug testing	**100%**	**25%**	**25%**	**25%**	**25%**	**N/A**
Zoom out/Zoom in (Canvas Rescale)	**100%**	**35%**	**15%**	**35%**	**15%**	**N/A**
DFA save and load	**100%**	**20%**	**20%**	**40%**	**20%**	**N/A**
Automatic Dead State Button	**100%**	**40%**	**20%**	**20%**	**20%**	**N/A**
Build from regular expression	**100%**	**25%**	**15%**	**35%**	**25%**	**N/A**
Finished Teaching mode	**100%**	**15%**	**20%**	**15%**	**40%**	**N/A**

Revamped Tutorial Mode	**100%**	**15%**	**20%**	**40%**	**15%**	**N/A**
Reorganize GUI for user experience	**100%**	**20%**	**20%**	**30%**	**30%**	**N/A**
Updated Poster	**100%**	**50%**	**20%**	**15%**	**15%**	**N/A**
Finalized“read me” file	**100%**	**20%**	**40%**	**20%**	**20%**	**N/A**
Finalized Release Ready .EXE of program	**100%**	**20%**	**40%**	**20%**	**20%**	**N/A**
User/Developer Manual	**100%**	**10%**	**75%**	**5%**	**5%**	**N/A**

Milestone #6 Discussion (Task Details):

- Finalized Touches on User Experience/Bug Test
    - Revamped features such as the automatic DFA builder and program created transitions to not overlap/maintain visual appearance consistently. Bug-tested a wide range of program features and cleaned up visual issues and frustrating elements that could impact the user experience.
- Zoom Out/Zoom In (Canvas Rescale)
    - Implemented a new feature in which the user can (with the use of the view buttons on the top of the GUI) the user can zoom in and out from the canvas, thus scaling canvas objects accordingly, allowing the user to reposition their work and expand their DFAs dynamically. The user can also zoom out using the mouse in a manner similar to a web page, where zoom occurs locally around the cursor, giving the user an unparalleled level of canvas repositioning.
- DFA Save and Load
    - A user can save a constructed DFA as a custom newflap object, which can be loaded into the canvas space edited, and traversed at will. This bespoke feature was something we wished to nail down early on, as it helps support the program’s role as a teaching tool, as shared newflap files can be used by anyone

with the program, letting teachers, students, and collaborators exchange graphs freely and easily.

- Automatic Dead State Button
    - One of the most important quality-of-life features, and one tied to an algorithmically complex process, we built a button within the GUI that allows the user to automatically construct a dead state for their DFA, saving time during the DFA construction process. All DFAs require a dead state for completion, so allowing the user to add one intuitively will greatly help them complete their work.
- Build From Regular Expression
    - Expanding on our string-based construction introduced in the last milestone, this feature lets the user write a DFA-compliant regular expression and have the program build it. This is something that often happens in real-life teaching of DFAs and thus is a vital feature for our program to demonstrate. This aspect of our program employs a high level of algorithmic complexity to fit graphs.
- Finished Teaching Mode
    - Finalized the previously implemented Teaching mode from the previous milestone, with it now incorporating animations into the teaching process via our traversal system. The user is now prompted to enter the provided random string, and the program indicates whether it works, helping to communicate the connection between construction and traversal. The dead state button makes this far easier.
- Revamped Tutorial Mode
    - A departure from the tutorial mode seen in the previous milestone, this version lets the user explore the GUI space, allowing them to click different buttons and see what they do.
- Reorganized GUI
    - Fixed the cluttered GUI by rearranging the animation controls to the bottom of the screen, and folding program features into new header segments, such as “build” which lets the user access the build from strings and regular expressions aspects of the program.
- Finished Poster
    - Completed the senior design showcase poster, with it now being ready to print
- Finished “Read Me” File
    - Finalized the on-board “read me” file, now encompasses the full scope of the newly released ready program.
- Finalized Release Ready .EXE of the program
    - Cooked up a release-ready .EXE version of the finalized program, including a custom desktop logo.
- User/Developer manual
    - Wrote and finalized a 25-page user and developer manual to let people view and explore the full scope of the development process and how people should use the program.

Milestone #6 Discussion (Team Contribution):

- **Chris: Helped focus team goals, took charge of NFA design, and dead state inclusion. Finalized the poster design and primarily revamped code features like teaching mode and tutorial.**
- **Vincent: Worked on creating the solo executable program file, code inclusion, and bugfixing. Helped write out the Read Me file, finalize bug testing, and ensure peak user experience**
- **Andrew: Assisted in code implementation and bug fixing, program QOL improvements. Lead efforts to reorganize the GUI and add several essential milestone features, such as the zooming process.**
- **Keegan: Assisted in code implementation and bug fixing, spearheaded further refactoring and GUI adjustments.**

Dates of Meeting/Client Feedback:

August 27:

- Focus is on DFAs
- User should be able to create a DFA intuitively (we'd need to nail down what that meant)
- The system should be able to check for completeness (e.g., are there transitions missing?) or correctness (is it truly a DFA) - Maybe the system should be able to complete given an incomplete DFA (e.g., fill in missing transitions – but how?)
- Eventually, it would be nice to format/prettify the DFA graphs (align nodes, improve readability)
- Be able to designate initial and final states, and maybe even designate a state as a dead state, and let the system do its magic
- Multiple symbols on a single transition between arcs in a way that is natural and readable
- Animate processing of strings on a DFA

– step-by-step, play, pause, etc.

- Minimization
- The vision is, given a set of strings that are accepted and not accepted, to let the system create a minimal DFA (in terms of states) that accepts/rejects as specified

September 10:

- Involve in the program is lambda transitions.
- The key things needed is when given strings it would create the DFA, when given strings, it would have to determine whether it accepts or rejects, and be able to create these states whenever

October 24:

- Discussed update on milestone two progress

- Expressed current work on internal program logic
- Spoke about current in-program node connecting format
- Expressed interest in further implementation of and use of the visual canvas type graphing form.
- Spoke to us about our current challenges regarding the state of the project and its code going forward.

November 10:

- Discussed the steps needed to continue the project and what should be done.

November 20:

- Discussed how minimization works by paper and the algorithm to use.

November 21:

- Discussed progress, which was a bit minimal.
- Talked briefly about minimization

January 20:

- Discussed project plan

February 20:

- Discussed what progress was made, which was the NFA logic and detection
- Figured out future steps to take

April 16:

- Finalized remarks on any changes to be made

Faculty Advisor Signature: _____ Date: _____