Milestone 4 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 #4 Progress:

Task	Completion %	Chris	Vincent	Andrew	Keegan	To Do
NFA detection + Lambda	100%	50%	50%	0%	0%	Nothing
NFA to DFA conversion	90%	90%	0%	0%	0%	Must refine the logic to improve accuracy and add a separate conversion from minimization
DFA Minimization	90%	45%	45%	0%	0%	Refine to make sure 100% accurate
Improved animation	80%	0%	0%	0%	80%	Animation needs a “wow” factor
Refactorization	100%	0%	0%	100%	0%	Nothing

Milestone #4 Discussion (Task Details):

- NFA detection + lambda
  - We needed to implement NFA detection because you could attach multiple transitions to the starting state, and if a user tried to minimize from there, it would be incorrect. So, adding NFA detection would prevent a user from attempting to minimize an NFA. With NFAs come lambda transitions, so we needed to add that as well.
- NFA to DFA conversion
  - To follow up on the NFA detection, we added a conversion that converts an NFA to a DFA, so the user can then minimize.
- DFA minimization
  - We have the logic down, but we need to run more test cases to ensure minimization is as accurate as possible. As well as refining this logic could lead to more robust code.
- Improved animation
  - So far, the animation we have is simple and followable, which gets the job done. However, we need a “wow” factor to make this program look more visually appealing.
- Refactorization
  - This was needed because our code was messy and inconsistent. Updating and simplifying our code base was needed to continue refining the program, and it became much easier to follow along.

Milestone #4 Discussion (Team Contribution):

- **Chris:** Worked on NFA detection and NFA conversion. He will be looking into the NFA dead state and fixing bugs in the NFA logic.
- **Vincent:** Worked on NFA detection and DFA logic refinement. He will be looking into creating an executable file for BETA testing and string input to DFA creation.
- **Andrew:** Worked on refactoring. He will be working on the tutorial mode.
- **Keegan:** Worked on the canvas space and animation. He will be working on fixing the website and improving the canvas space with a few features.

Plan for Milestone #5:

Task	Chris	Vincent	Andrew	Keegan

NFA dead state and bug fixing	Logic implementation and researcher	Bug Fixer/Code Contributor and designer	Co-Lead coder and development head	Co-Lead coder and development head
Canvas GUI upgrade with quality of life features	Bug Fixer/Code Contributor and researcher	Bug Fixer/Code Contributor and researcher	Co-Lead coder and development head	Co-Lead coder and development head
Tutorial mode	Bug Fixer/Code Contributor and researcher	Bug Fixer/Code Contributor and researcher	Co-Lead coder and development head	Co-Lead coder and development head
String input converted into a DFA	Logic writer, DFA logic consultant, and bug tester	Logic writer, DFA logic consultant, and bug tester	Co-Lead code side implementor	Co-Lead code side implementor
User feedback	Bug Fixer/Code Contributor and designer	Bug Fixer/Code Contributor and designer	Co-Lead code side implementor	Co-Lead code side implementor

Milestone #5 Plan Disuccion (Task Details):

- NFA dead state and bug fixing
  - NFAs have a dead state, for whenever you cannot leave a state on a particular symbol. Having a dead state shown and clearly distinguished from other states would be useful. As well as fixing the NFA-to-DFA conversion bug (visual), this needs to happen to have a more stable program.
- Canvas GUI upgrade with quality of life features
  - The GUI needs to be upgraded to add a side panel that can be extended or collapsed, a state selector tool (which may be implemented later), and better interactivity (e.g, adding shortcuts). Along with improving animation to add the “wow” factor and giving the GUI a cleaner, more “modern” look
- Tutorial mode
  - This mode would teach a user about the flow and process of DFA, DFA minimization, and conversion from an NFA to a DFA. Which would be important because this program will be used as a teaching tool.
- String input converted into a DFA
  - This was one of our original ideas and will be implemented for a user can input a set of strings (e.g. 1101), and a DFA graph will be created. There are many ways to a DFA can be created so it will be difficult to implement but it would be a good feature to be used.
- User Feedback

- Since our program will be used as a teaching tool, it needs to be functional for everyone. So we need to put the program in the hands of users to collect any feedback to improve and fix.

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

Faculty Advisor Signature: ________________________________ Date: ________

Evaluation by Faculty Advisor

- Faculty Advisor: detach and return this page to Dr. Chan (HC 209) or email the scores to pkc@cs.fit.edu
- Score (0-10) for each member: circle a score (or circle two adjacent scores for .25 or write down a real number between 0 and 10)

Chris	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Vincent	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Andrew	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
Keegan	0	1	2	3	4	5	5.5	6	6.5	7	7.5	8	8.5	9	9.5	10

Faculty Advisor Signature: ______________________________ Date: ________

Task	**Completion %**	**Chris**	**Vincent**	**Andrew**	**Keegan**	**To Do**
Improve animation	50%	10%	20%	10%	10%	Have better interactivity, and make it smoother
Adjust minimization	50%	20%	10%	10%	10%	Minimization should be smoother (tutorial mode)
Improve GUI	80%	10%	10%	30%	30%	Adjust GUI to be more modern
Include lambda (NFAs)	0%	0%	0%	0%	0%	Need to implement NFA logic
Refined stability	100%	25%	25%	25%	25%	Make sure everything stays stable and functional