Oscar Reyes

Formal Languages and Computability

Abstract

This project is meant to show how DFA’s can be used to model a Dating Simulator game. It runs through a basic example of a Dating Sim where the user can make choices that lead to several different endings. Although basic this can demonstrate how to use DFA’s for even complex Dating Sims.

Introduction

For this project, I wanted to work with Dating Simulator games. Although it seems silly, dating simulators can actually be expressed very well with a DFA. Not all dating sims are the same but in general the game presents a group of girls that you meet and you have to build a relationship with them with the hopes of dating/marrying them. From there you are usually given response options to choose from that can benefit or damage your relationship with these girls. In the beginning these options lead you to select a girl from the group of girls you meet, and settles on their “route”. But some Dating Sims make it possible to end up with multiple girls or even go from one girls routes to another's randomly. All this branching back and forth can make for some complicated DFAs. Regardless of how you go about making choices you will reach one of the many possible endings. So we start off meeting these girls, which can be represented in a

DFA with an Initial State. From there the different options you are given will lead to other non-accept states. These options can be represented with the transitions between the states. Finally the different endings can be represented with accept states. Depending on how you look at it bad endings can be seen as errors or as accept states as well. Representing all this information with a DFA could help players make optimal choices. A DFA can also help game developers understand and keep track of choices, relationships, and goals within these types of games. The stories for these can become twisted and convoluted so a DFA can help keep it organized. Therefore, for this assignment I will look at some dating sims and use them to design a DFA to represent the story and progression of a new “simple” dating sim. This will allow me to show how useful DFAs can be to developers and players alike.

For this project we will be using a simple type of Dating Sim that only features one girl. This way we can keep the example very simple. There will be 4 possible endings for this girl. A perfect ending, a good ending, a neutral ending, and a bad ending. The DFA will take user input and lead to one of these states.

Detailed System Description:

In order to simplify a dating sim that can be described with a DFA, a few things will be implemented:

- The choices for each event will be simplified so that there are only three
    - Good, Neutral, and Bad (a, b, and c respectively)

- There will only be one girl with four possible endings (prefect, good, neutral, and bad)

With these things in mind, we can construct a language that will be accepted by the DFA and show routes to winning scenarios. One of the important things that we have to consider is that you are not usually able to go back and change what you said, so this DFA will strictly go forward.This means that even though we seem to accept any string that ends with an a or b, we cannot accept a string with 20 a's, as this goes beyond the scope of our game. We have to restrict it because a game has to have a set path that it goes through and cannot go back and forth.

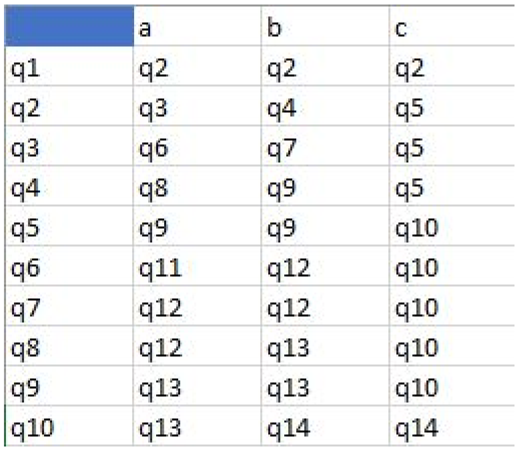
DFA Explanation:

- First transition accepts any character and goes forward, this is because the first choice is usually a freebie.
- The only way to get a perfect ending is to always make the best choices (a) so this accept state (q11) will accept only strings that have all "a"s (besides the first that is a gimme and can be any character.
- Making too many bad choices will result in an error state (bad ending)
- There are a couple of ways to get Good and Neutral endings

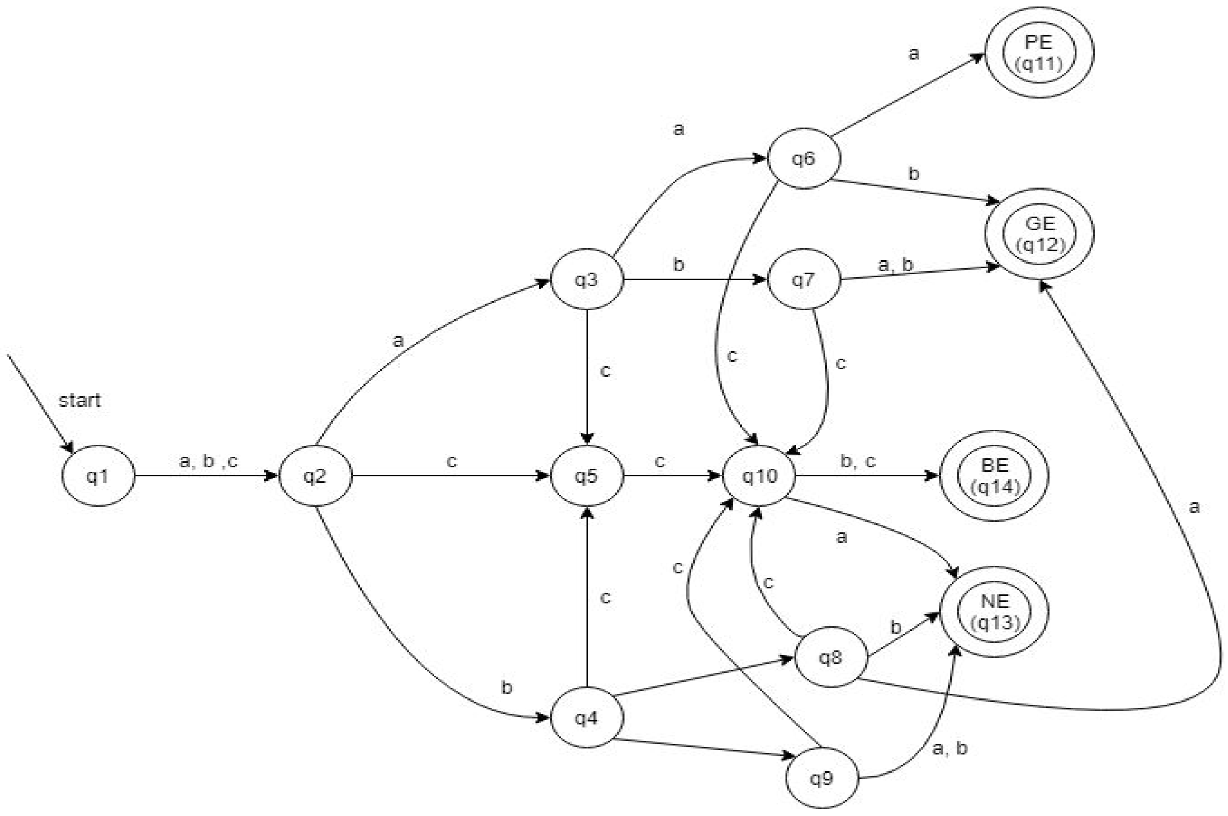
With this we have a DFA that can describe a basic dating sim. Although simple, DFAs can also be used for more complex games. For example we can account for more choices or include more girls.

Now for the system (game) that will use the DFA, it presents user with some exposition and some choices. The system then takes user input one character at a time.

The system makes sure that the input is one of the choices (a, b, or c) and not longer than a single character. It then checks the current state as well as the choice made which transitions to the next state. It then grabs text for that state and asks the player for another choice. It does this until reaching one of the accept states which will display that ending. The DFA and state transition table are as follows:



	a	b	c
q1	q2	q2	q2
q2	q3	q4	q5
q3	q6	q7	q5
q4	q8	q9	q5
q5	q9	q9	q10
q6	q11	q12	q10
q7	q12	q12	q10
q8	q12	q13	q10
q9	q13	q13	q10
q10	q13	q14	q14



Literature Survey:

There are many Visual Novel/ Dating Sim engines out there like: Twine or Ren’py. These engines make it easier to make and produce dating sim games. My project also makes it easier to do the same thing. The main difference is that my project works to make storylines easier to make and understand. Using a DFA to produce the story clears out a lot of the clutter of what path leads where. This means that a DFA can be used in conjunction with these other engines to make it even simpler to make and design Dating sims.

Conclusion:

Although the DFA described throughout the project is of a simple dating sim, It can help understand how DFA’s can be used for this purpose. This will help lead people to making DFA’s for more expansive games, including more routes, girls, choices, feature, etc. With further investigation it is very possible to use DFA’s for other types of games as well. Different levels can be described as states and moving from one level to another can be described as transitions. There are many ways to think of DFA’s and their functionality outside of the usual coding problems