Chris Cordaro

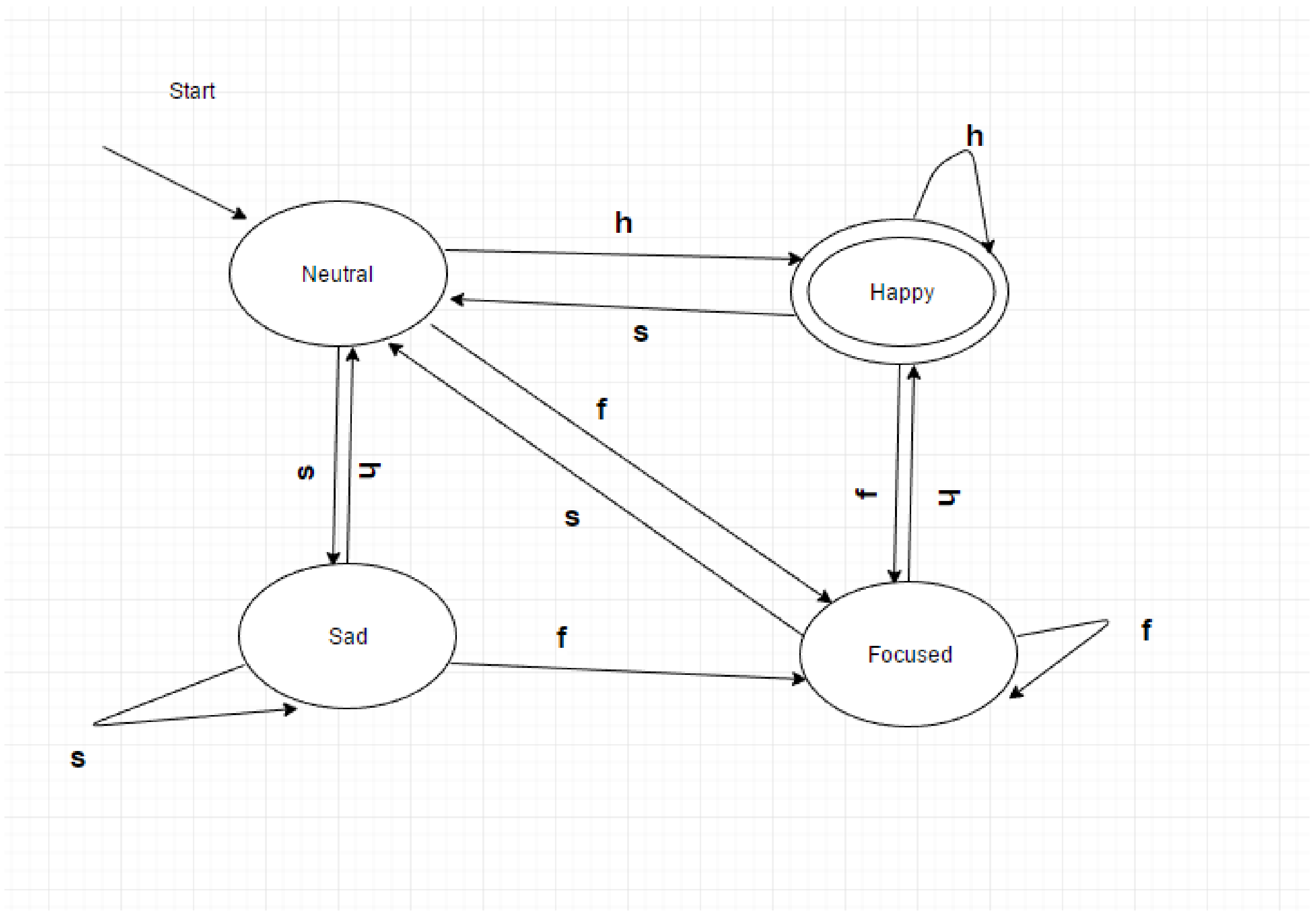
5/5/2017

DFA Music Mood Calculator

Abstract: This project is a program which given user input will determine the current mood of the user and report the song-type that he or she should listen to in order to be in a happy mood. The program implements a DFA in order to accomplish this goal. The user is prompted to enter how he or she has been feeling the past five days, and is only able to enter three values: ‘h’ for happy, ‘s’ for sad, and ‘f’ for focused. The DFA starts at a neutral state and can transition to the states ‘happy’, ‘sad’, and ‘focused’ depending on the given user input.

Introduction: The idea of this project came from researching applications of DFAs couple with my affinity towards music. I have been playing guitar since I was in fourth grade and music is a center focus in my life, so any time I can see a way to weave music into my computer science classes I will try and do so. The DFA implemented is relatively simple and can handle any combination of letters that the user inputs. The program sanitizes the user data to insure that only exactly 5 letters are entered and only the characters ‘h’, ‘f’, and ‘s’. This ensures that the DFA encounters no errors when traversing its transition table. “Happy” is considered to be the accepted state so if the DFA completes the required transitions without ending on the happy state, it will report back to the user notifying him or her as to what style of songs he or she should listen to in order to reach a state of happiness. Essentially, this DFA calculates the mood of the user based on the past five days and reports back as to how the user can reach a happy state with the help of music.

Detailed System Description: This system uses a DFA and a transition table in order to determine the current mood of the user. The implementation of the DFA is through a delta table where each transition is explicitly defined on each state for a given input. The system prompts the user to enter how they have been feeling for the past 5 days, and will continually prompt until an accepted input is given. There is a ReadMe file which lays out the rules of the input, however, if the user chooses not to read that the system provides detailed error messages regarding how the user needs to format the input.



The DFA starts at the Neutral state and utilizes the transition table in order to determine the current demeanor of the user. After the DFA has complete the five transitions it checks to see if it is at the accepted Happy state. If it is, it reports back to the user that they are currently in a happy mood and should continue to listen to happy music. If the DFA ends in the other three states, it reports back to the user telling him or her what mindset they are currently in and recommends the genre of music he or she should listen to. For example, if the DFA ends in a Sad state, it will report that the user should listen to two Happy songs in order to help reach a happy state of mind.

Requirements: The requirements to run this system is a PC that has python installed.

User Manual: Users will open the ReadMe document in order to understand what the DFA will accomplish. The user will then run the program and be greeted with the prompt. Once the user enters the correct form of data, the program will run itself and produce an output.

Conclusion: Overall, this project represents the ability for a DFA to be implemented in order to run a program which utilizes a transition table. Through the use of the transition table and input sanitation the system is able to ensure that each accepted input entered by a user will produce an output to the user. Something like this could be used in certain personality tests or dating application with some modification. It would be interesting to research how sites like match.com build their personality algorithms, although I think that the main algorithms would be hidden and protected under IP laws. The system is simple and effective in its task completion and demonstrates the ability of a DFA to complete logical step-based tasks.