Major Triad DFA

Ian Sniffen

Ian.Sniffen1@Marist.edu

May 16, 2019

1 Abstract

Deterministic finite automata can be made to model out whether a sequence of three notes results in a major triad. The resulting DFA is large, but can output the name of the major triad input. This DFA could be able to be expanded in the future to additionally include other types of triads to further the modeling of music theory in finite automata.

2 Introduction

Major triads consist of the first, third, and fifth notes of a major scale. These can be inverted in ways that allow for any of the notes to be the bottom note of the chord. There is a major triad for every major scale, and this can be modeled using finite automata. Each major triad could be an accepting state, and each note would progress through the DFA; if after three notes, an accepting state is reached, this will mean that a major triad has been input.

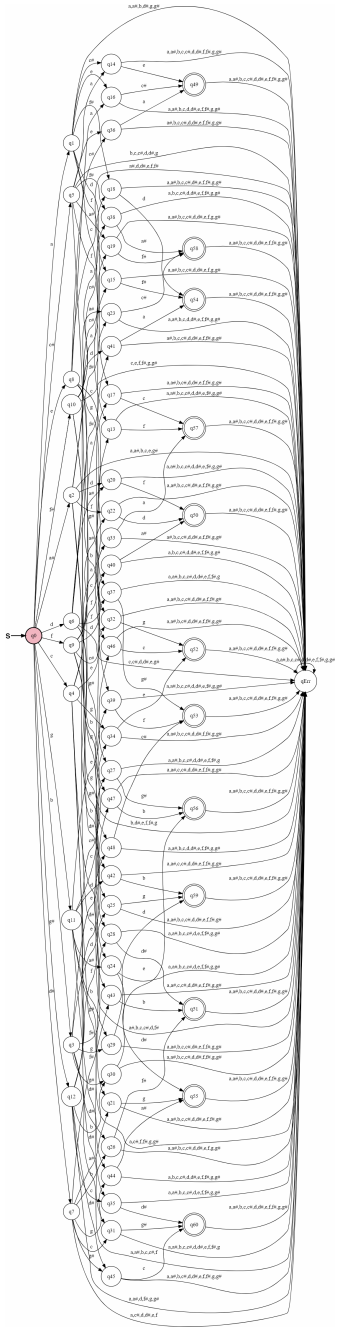
3 Detailed System Description

The program will accept three notes of user input to determine, via traversing deterministic finite automata, whether or whether not the notes of a major triad have been input. Users input the three notes, which are then used by the software to determine the acceptance of the set. The program itself is fairly simple, but the large DFA is the main part of this project. This is a large DFA, but it is separated into about five main layers: starting state, one-note states, two-note states, major triad states, and the error state. Dividing up the DFA into these layers makes it easier to understand.

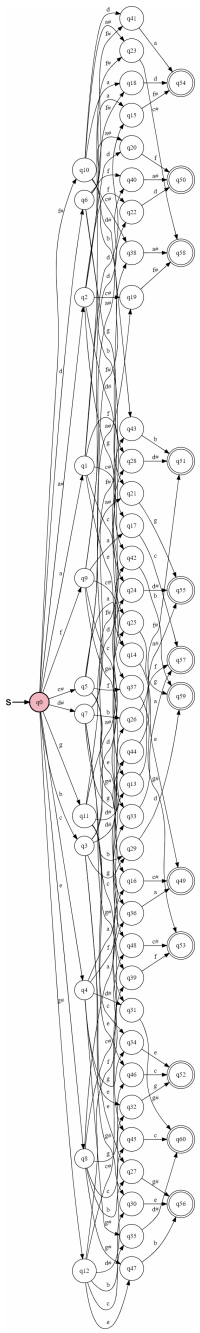
The DFA being used consists of 62 nodes, with one of those nodes being an error state. There are 12 different accepting states; these accepting states are each major scales' major triad. Starting from starting node q0, the DFA branches off into 12 different states, one for each of the possible inputs. Then, it goes into permutations of two notes For the purposes of being concise, order does not matter in these pairs or in further nodes, so "A" followed by "C" results in the same node being reached as "C" followed by "A". This also allows for

inversions to be supported by the software. If the permutation of two notes does not exist for any major triad (e.g. A and A#,) an error state is reached. Following this layer of the two-note permutations, the next layer is of the accepting states. As with the missing permutations of the two-notes, non-major triads also send the user to the error state. As order of notes does not matter, triads can be accepted regardless of the order of notes, allowing for any inverted major triads to be accepted.

The DFA is shown below.



The same DFA without the error state (for added readability) is below.



4 Requirements

The project was done in Java; a Java installation is required to run this software. I typically run the driver file from a command line for simplicity, as the driver simply needs to accept the three input notes to traverse the states.

5 Literature Survey

I was unable to find information on other application of finite automata to the classification of music theory, which leads me to believe not a lot of work has been done in regards to this. Most work with finite automata regarding music theory is related to music generation.

6 User Manual

To use the Major Triad DFA program after downloading the Java files, you run the program with the command "java driverDFA" when in the same directory as the project files. It will ask for your input notes in order. The possible inputs are: A, A#, B, C, C#, D, D#, E, F, F#, G, and G#. The program does not currently accept flats, but the equivalent sharp can be input for the same result (e.g. B flat becomes A#.) After input, the program will output the result. Accepting states will result in the major triad being reported, while any other state will result in the program reporting that the input does not result in a major triad.

7 Conclusion

The DFA created for this project successfully performs the major triad classification I set out to create. This could be expanded to include other types of triads, such as minor triads, diminished triads, and augmented triads, as well as adding support for flats. This could also be expanded to use in other software, such as sheet music software or digital audio workstations to assist in music labelling or creation. Using finite automata for music theory could also be helpful in instructing musicians in music theory.