Exam No. B097426
Music 2D: Theory and Practice of Music Technology
December 14, 2016

Markov Melodies

I am fascinated by the ways in which algorithms are used in many facets of human daily life. While studying algorithmic composition in this course, I have come to view algorithms as a useful tool in the composition of music. I have very little composition experience so I decided to use an algorithmic process to guide my composition. For this project, I have created a musical composition interface using first order Markov chains and a simple synthesizer to playback the result of the composition. This report summarizes my process by detailing my project goals, the methods I used to reach these goals, and a discussion of the final output.

My primary goal for this project was to create pleasant sounding melodies using an algorithmic process. More specifically, I aimed to create an algorithmic process that produced entirely different results depending on the data input. I dreamed of a program that processed a midi file and converted its contents to an input for the algorithm. This would allow me to easily draw inspiration from any song I wanted to create songs with similar melodies. I aimed to combine multiple voices powered by my algorithm with different inputs to generate a more complex and full sound. Another goal of mine was be able to interact with my piece in real time through changes to the algorithmic process as well as sound processing.

To accomplish these goals, I broke down each large problem into a series of smaller problems and gradually built my program from the bottom up. This process involved designing the algorithmic process, creating a synthesizer to play the output of the algorithm, and creating data to serve as the input to the algorithm.

I started with the design of the algorithmic process. I was inspired by the lecture by Michael Edwards describing how first order Markov chains could be used to create a Gregorian chant. I used his _monks.pd patch as a model for the flow of my algorithmic process. I first set about creating a process to handle large amounts of input data at once. In the _monks.pd patch, the Gregorian chant probabilities were hard programmed into the patch. I wanted my patch to know nothing of specific probabilities, but rather be able to play music according to its input probability data. I created a standard transition matrix representation for pitches and rhythms and designed a program to read this data from text files. This allowed me to realize my goal of creating an algorithmic process with an output completely dependent on its input data. Despite considerable effort, I was not able to realize my goal of processing a midi file to create the transition matrices to serve as input to my program as I could not find a way to do this using pd-vanillla. A future enhancement for this project would be to create a program in a different programming language to translate midi files to transition matrices. It would certainly make the input data creation phase of the project much less tedious.

To accomplish my goal of varying the algorithmic process in real time, I added the ability to replace full rows of a transition matrix as a song is playing. I also added a setting to change the "temperature" of the pitch and rhythm probabilities for a Markov voice. This works by multiplying each probability in a row by an exponent from 0 to 11 in order to make values more similar or more different from each other. This essentially functions as a setting to make the algorithmic process more deterministic (small temperature, large exponent) or more random (large temperature, small exponent). I also added more real time algorithmic settings such as the delay between notes, the octave of a voice, the tempo, and the ability to transpose voices.

I next created a sine wave synthesizer to play my algorithm's output. I followed a tutorial online by Eduardo Mezêncio (Creating a Simple Synthesizer in Pure Data part II) to create an ADSR envelope so that I could vary the sound quality in real time. I used other elements from the sound processing section of the course taught by Dimitrius Papageorgiou such as a high and low pass filter. I aimed to allow a single voice to play multiple notes at once without distortion or clipping but was not able to accomplish this goal. This made the delay factor effect I created much less useful. In the future, I would like to improve the quality of my synthesizer by using frequency modulation.

The compositional phase of my process was very difficult, but I was grateful that my algorithm allowed me to create transition matrices rather than full compositions. I knew that my algorithmic process was very simple so would not be able to capture complex melodies. I decided to use a simple, beautiful melody as inspiration in order to generate the transition matrices. I used the English horn solo in the second movement of the New World Symphony by Antonin Dvorak. Transcribing these 14 measures into transition matrices was incredibly tedious and meant that I did not have time to construct different transition matrices for each voice.

Overall, I think that my program is able to generate simple and melodically pleasing melodies. With human intervention, it could be modified to produce a higher quality song. One major limitation with my approach is that it only considers the current pitch when deciding on the duration and next pitch to play. This means that there is isn't the concept of a phrase in the melodies and different musical ideas cannot easily develop throughout the piece. I tried to counteract this limitation with the ability to replace rows in real time and vary the temperature settings, but higher order Markov chains (where previous notes are considered) would be needed generate more interesting compositions. I am also critical of the sound quality of my synthesizer and would like to improve it in the future.

This project has given me an appreciation for the art of composition and especially computer-aided algorithmic composition. As Curtis Roads (1996) explains in *The Computer Music Tutorial*, "It takes a good composer to design algorithms that result in music that captures the imagination." I have learned that lesson firsthand.