Generating Music Using Markov Chains

Rajdeep Pinge (ID:201401103)*
IT461 - Stochastic Simulations course, DA-IICT

This project report contains detailed analysis about the project on Generating Music on a machine. In this Project, the focus is on the Markov Chains approach to generating music using stochastic methods. Different Approaches to generating Transition Matrices for Markov Chains have been studied in this project. We have also implemented the random initialization of new music sequence to be generated and still are able to create coherent music in fairly successful manner. Finally, necessary conclusions about the effectiveness of this approach of machine generated music have been made by analyzing the observations.

INTRODUCTION

Using machines to generate speech and music is a major area of research today. This project is a limited attempt to explore this area of Generating Music using Markov Chain approach based on the concepts of Markov Chains learnt in the course.

The aim is to try 3 different approaches based on 1st-order markov chains, 2nd-order markov chains and 3rd-order markov chains to generate the music on our computer. For this we would be using a one existing song and the MIDI standard of music so as to generate transition matrix which would help us in creation of music.

Further we will try to implement one more approach combining the previous three and try to optimize the transition matrix for space and analyze the effect of these modifications.

BACKGROUND, MOTIVATION, RELEVANCE AND IMPORTANCE

Music is pleasant because musicians understand which notes sound good together.

From a mathematical viewpoint **music** is nothing but a **sequence of 'notes'**. 'Notes' are musical sounds played at a particular 'pitch' (specific volume and frequency). A **'chord'** is a **collection of 'notes'** played simultaneously. **Song** which is a **sequence of chords** is based on the principle of **continuous transition of one chord into another**.

Musical theory lays down certain rules which dictate which notes should be used, how can they be combined and in what sequence should they be used in order to play good music. This means that the transitions are not random but rather every state has some definite probabilities of transitioning to others governed by musical aesthetics. This is what distinguishes music from noise.

PROBLEM DESCRIPTION

The aim is to make the machine listen to existing music, understand and learn it and then generate music on its own. Markov chains are a very natural way to model this process. The set of chords used to play the songs define the **possible states** of Markov chain in which our song can be at any instant of time.

The problem not only has the required structure but Markov chains will allow us capture the musically pleasant transitions. The existing songs can be used to generate the transition matrix of the Markov chain wherein the transition of chords defines the states. The number of transitions from one chord to another defines the number and correspondingly the probability of transition between those states while generating new music.

With the help of this mathematical construct, it would be possible to mimic the subjective musical aesthetics.

IMPLEMENTATION

MIDI Files

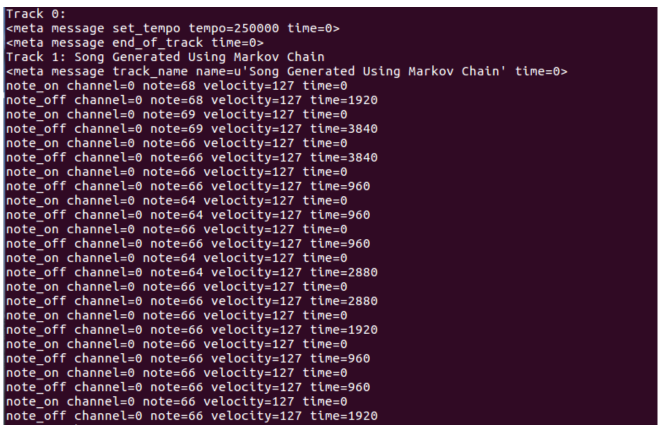
Musical Instrument Digital Interface (MIDI)[1] is a technical standard which allows electronic musical instruments and computers to connect with one another. They contain information on how to play the tune. This tune can be played by a variety of computer generated instruments.

In music the notes are divided into 'Octaves'. There are 10 full octaves and 1 incomplete octave [2]. The full octaves each contain 12 notes while the last incomplete octave contains 8 notes. So, there are a total of 128 notes. These are numbered from 0 to 127 as per the MIDI specification. Numbering the notes makes it convenient for the softwares to read and extract information from MIDI files. The enumeration also helps in the mathematical representation of the problem.

Construction of MIDI Files

MIDI files consist of one header chunk containing information about the instruments used to play the music present on that file, followed by track chunks containing the notes played and their status, duration, velocities (volume), pitch etc. There can be multiple music tracks on a single MIDI file. The value of velocity lies in the range [0, 127]. The value of pitch which is the note that is played also lies in the range [0, 127].

FIG. 1 shows details of a paticular MIDI file generated by our programme. All the attributes mentioned above are visible in the figure.



```
Track 0:
<meta message set_tempo tempo=250000 time=0>
<meta message end_of_track time=0>
Track 1: Song Generated Using Markov Chain
<meta message track_name name=u'Song Generated Using Markov Chain' time=0>
note_on channel=0 note=68 velocity=127 time=0
note_off channel=0 note=68 velocity=127 time=1920
note_on channel=0 note=69 velocity=127 time=0
note_off channel=0 note=69 velocity=127 time=3840
note_on channel=0 note=66 velocity=127 time=0
note_off channel=0 note=66 velocity=127 time=3840
note_on channel=0 note=66 velocity=127 time=0
note_off channel=0 note=66 velocity=127 time=960
note_on channel=0 note=64 velocity=127 time=0
note_off channel=0 note=64 velocity=127 time=960
note_on channel=0 note=66 velocity=127 time=0
note_off channel=0 note=66 velocity=127 time=960
note_on channel=0 note=64 velocity=127 time=0
note_off channel=0 note=64 velocity=127 time=2880
note_on channel=0 note=66 velocity=127 time=0
note_off channel=0 note=66 velocity=127 time=2880
note_on channel=0 note=66 velocity=127 time=0
note_off channel=0 note=66 velocity=127 time=1920
note_on channel=0 note=66 velocity=127 time=0
note_off channel=0 note=66 velocity=127 time=960
note_on channel=0 note=66 velocity=127 time=0
note_off channel=0 note=66 velocity=127 time=960
note_on channel=0 note=66 velocity=127 time=0
note_off channel=0 note=66 velocity=127 time=1920
```

FIG. 1. Detailed storage in Midi Files

The following libraries, softwares have been used in the project:

- MidiUtils library: To create MIDI files and write machine generated music in them.
- wildmidi: For playing MIDI files

Technique used

Although we have taken help of the following blog [3] to understand the problem and possible solution in detail, we have written our own code and have implemented our own modifications which will be described further in the report.

Obtaining the Transition Matrix

The goal is to obtain a transition matrix so that the machine can determine which state should the music go to from the current state. To generate matrix, the notes are required in numeric format.

To get the notes in numeric format the input file is parsed and examined for the chords of the song. The chord may contain multiple notes played at the same time. But our programme is very simplistic and generates simple music with only one note chords. Hence it is necessary to convert the multi-note chords into single-note chords. For this, at every step we obtain the notes of previous chord and present chord. We assume that every note in previous chord transitions to every note in the present chord. This gives the note-wise transitions. Accordingly, the count of transitions is maintained to estimate the probability of state transitions.

To generate transition matrix, it is important to determine the possible states first. If we use a first order markov chain, we consider only one note from the past as the state. Similarly for second and third order markov chain, we consider two and three notes from the past respectively as the state. This means that for third order, the machine should take into account the keys played in the past three instants to decide the key to be played at the next instant. As the order of the Markov chain increases, it brings more coherence to the melody creation.

Generating the Music from Transition Matrix

Once the transition matrix is ready, the generation part starts. The melody starts with a random state from the available states and searches for that state in the matrix. Once the state is found, it randomly searches for possible states that it can go to in the matrix and transitions to that state.

Note here that there would be at least one state where the music must transition from the current state otherwise the current state would not have existed in the first place in the transition matrix.

The music is generated for a fixed duration. Appropriate parameters required corresponding to each entry in the MIDI file are provided. Finally the sequence of notes is written to the MIDI file which can be played by a MIDI player software.

Modifications

There are some problems which arise during the generation process. To remove the deterministic behaviour of the programme, random initialization of the melody to be generated is done. But in 2nd and 3rd order matrices, the randomly initialized entries may not always be present because the original song may never have been in those states. In such cases, the software is unable to find the next state and is terminated.

To avoid this we use combination of the three chains (backoff method). Here we start with random notes. We first look at the third order markov chain. If the required state is not present in matrix, then we consider second order markov chain. Similarly, if state is not present in second order transition matrix, we consider first order markov chain. We are sure to find the state in the first order matrix because the notes are randomly chosen from the available notes which are guaranteed to be there in the first order matrix. Hence song can be generated with any random initialization of the notes.

Also because of increased probabilistic effect, different songs are generated each time and the possibility of deterministic behaviour is removed.

Another Modification is in the storage space used to store the matrices. As the order of the markov chain increases, the dimensions of the matrix increase by the same power.

In our example, the first order matrix is 24 x 24 while the second order matrix is 79 x 79 instead of 576 x 576. This is because the song goes to a limited number of second order states and not all of 24^2 states but still this is large enough as can be seen in the table given in FIG. 2. This further worsens the situation in 3rd order case.

Markov Chain	Transition Matrix Dimensions	Total Entries
1st order	24 x 24	576
2nd order	79 x 79	6241
3rd order	137 x 137	18769

FIG. 2. Comparison of Transition Matrix Sizes

Markov Chain	Transition Matrix Dimensions (Reduced)	Total Entries
1st order	24 x 24	576
2nd order	79 x 24	1896
3rd order	137 x 24	3288

FIG. 3. Comparison of Reduced Transition Matrix Sizes

The solution to this problem is to store just the next note instead of the whole 2 element or 3 element state in the transition matrix as can be seen in the FIG. 5 and FIG. 6. In these figures, the titles of the columns contain just one note instead of the whole tuple. The next state can be generated based on the previous state since only one note changes from the previous state.

The difference in storage space can be seen by comparing the tables in FIG. 2 and FIG. 3

Creating 1st order markov chain

	61	49	68	57	54
49	1	0	0	0	0
50	0	1	0	0	0
54	0	0	5	2	2
55	0	0	0	16	0
56	0	0	0	0	7
57	5	0	0	117	0
59	8	0	0	52	0
60	0	0	0	2	0
61	27	0	0	1	0
62	5	0	0	26	0

FIG. 4. Transition Matrix using 1st-order Markov Chain

Creating 2nd order markov chain

	57	69	83	81	61
(59, 55)	8	0	0	0	0
(69, 71)	0	6	0	0	0
(83, 83)	0	0	16	10	0
(57, 61)	0	0	0	0	5
(67, 64)	0	0	0	0	0
(84, 84)	0	0	8	0	0
(57, 57)	71	0	0	0	0
(66, 65)	0	0	0	0	0
(66, 64)	0	0	0	0	0
(54, 50)	0	0	0	0	0

FIG. 5. Transition Matrix using 2nd-order Markov Chain

Creating 3rd order markov chain

	69	57	64	67	62
(69, 68, 64)	3	0	0	0	0
(54, 57, 59)	0	2	0	0	0
(64, 67, 64)	0	0	2	2	0
(57, 57, 64)	0	0	0	0	26
(56, 54, 54)	0	0	0	0	0
(68, 66, 68)	0	0	0	0	0
(54, 68, 69)	0	0	0	0	0
(71, 66, 62)	1	0	0	0	0
(59, 57, 56)	0	0	0	0	0
(83, 81, 64)	0	0	0	0	0

FIG. 6. Transition Matrix using 3rd-order Markov Chain

OBSERVATIONS

- The input files contain multi note chords while the files generated are simple uni-note chords. The music is thus pretty simple compared to the input.
- Music generated by using 1st order markov chains is very simple but has a lot of large variations. This might make it unpleasant sometimes.
- In contrast, third order music is complex but is very repetitive because of sparsity of the 3rd order transition matrix as an be seen in FIG. 6
- Backoff technique is used to remove deterministic behaviour in the system. With this method, any random starting point can be used and the music will still be generated.

CONCLUSION AND LIMITATIONS

On the whole, Markov Chain approach is a reasonable solution to generating music using a machine. It is very easy to generate simple music by tuning very few parameter. The existing music can be parsed and the generated music can be heard because of the convenience of accessing the MIDI files.

But there is a limit on the order of Markov chain that we can generate because of the storage space problem. Also managing a large matrix is difficult and slows down the programme.

Also it is difficult to generate the songs which play multiple note chords at a time.

POSSIBLE MODIFICATIONS AND EXTENSIONS

- Instead of generating random duration notes, we can store the duration of notes as played in the original song and can replicate those during music generation so as to get more realistic music.
- The volume of the notes can be varied.
- The Programme can be extended to play multiple notes, i.e. chords having multiple notes, at the same time so that more variety can be generated from the same transition matrix.
- The Markov chain can be generated by training on more than one song so as to cover a wide variety of sound levels and musical notes which will generate music more closer to real-life music.
- Further this project can be used as a stepping stone for making forays into the current research areas of music generation such as Deep Learning and Recurrent Neural Networks.

ACKNOWLEDGEMENTS

I thank my team members:

1. Aditya Joglekar 201401086
2. Rushikesh Nalla 201401106
3. Omkar Damle 201401114
4. Deep Raiya 201401221

for their contribution in doing this project. We had a wonderful experience and got to learn a lot from this project and the course of Stochastic Simulations.

* 201401103@daiict.ac.in

REFERENCES

[1] https://en.wikipedia.org/wiki/MIDI
[2] Information about midi notation: http://www.electronics.dit.ie/staff/tscarff/Music_technology/midi/midi_note_numbers_for_octaves.htm
[3] Theory: https://medium.com/@omgimanerd/generating-music-using-markov-chains-40c3f3f46405
[4] Code for reference (just for help): https://github.com/CMasanto/melody-generator