Generating Random, Yet, Constrained Music

Muhammad Faisal Islam Faisal Islam Elgamal

May 2017

Abstract

In this project, the concept of generating music, fugues in particular, is explored through writing a set of rules translated into C++ code that generates a random musical theme. This theme is then imitated throughout the piece where certain operations are applied like inversion, transposition, scaling, and retrogression . . . etc. The imitated melodies are placed at random starting points distributed throughout the piece. Some parameters and constraints were added to prevent disorganized compositions and unpleasant sounds. Some of these parameters are: using the major scale, frequency range and maximum note duration.

1 Introduction

The basic idea of imitation and repetition of the compositional technique of the fugue is an old concept in music's history. It was first termed as a canonic style until the term *fugue* was more specifically given to the technique of imitative counterpoint. The two most pioneering composers in this style are Ludwig van Beethoven, who has written what is now considered a master-piece *"Grosse Fugue"*, and Johann Sebastian Bach who has written countless amounts of fugue works including compositions like *"The Art of the Fugue"* and *"The Well-Tempered Clavier"*.

The principle idea behind this technique is that the piece is built around a musical theme, called a *subject* in this context, such that this idea is imitated and repeated throughout the piece with occasional modifications and alterations on it.

Also in the late 20th century, a new type of music started to flourish named chance music by the great composer John Cage. John Cage used new instruments in his music like a piano using stones and nails in his compositions. John cage regarded chances and indeterminacy as pleasure. This why the first motif in the generated pieces were randomly generated ("The music of chance").

The idea of repeatability and pattern formation that is very dominant in this technique in specific prompted many attempts at making programs and artificial intelligence to generate music without human intervention. Some of these attempts have had huge successes in convincing people that it was written by a human, though none ever reached the level of sophistication and beauty of the masterpieces written by the musical legends. Though, one might think that soon enough the computational strength of computers will overcome the genius of human composers.

2 Terminology and Data Structures

2.1 Describing Notes

A note is described by its frequency and duration. The frequency of a note is described by the scale number. For computational purposes, a note duration is described by two integers n and k where the legal durations are $\frac{k}{2^n}$.

Definition 2.1 *A note duration is a pair of integers* $(k, n) \in Z \times Z$. *This duration is equivalent to* $\frac{k}{2^n}$

The pitch of a note is described by an integer i that is related to the frequency f of the represented note by $f(i) = (440) * (2)^{\frac{i}{12}}$

Listing 1: Representation of the duration

```
typedef pair<int, int> duration_t;
```

Listing 2: The data structure of a note

```
struct note_t{
    freq_t freq;
    duration_t duration;
};
```

2.2 Describing A piece of Music

A melody is simply a a sequence of notes. A piece is a sequence of melodies with a starting point associated with each melody in the sequence.

Listing 3: The data structure of a melody

```
typedef vector<note_t> melody_t;
```

Listing 4: The type definition of a piece node

```
struct piece_node_t{
    melody_t melody;
    duration_t starting;
};
```

Listing 5: The data structure of a piece

```
typedef vector<piece_node_t> piece_t;
```

3 Rules

The target of this project is to randomly generate a musical piece based on a motif by using four main methods: diatonic transposition, melody reversion, retrogression, scaling.

3.1 Diatonic Transposition

Every note has a pitch that distinguishes it from other notes. This pitch is related to a specific frequency that is produced by the musical instrument that plays the note. For example, middle c has a pitch that is identified by the frequency 261.6 Hz.

Definition 3.1 *For two given frequencies $f(i_1)$ $f(i_2)$ following the relation $f(i) = (440) * (2)^{\frac{i}{12}}$, an interval is $|i_1 - i_2|$.*

Definition 3.2 *A Chord is a simultaneous play of two or more different notes. A chord's notes can be described by a set of integers that represent the intervals between them.*

Also melodies can be described using the interval notation. In fact, equal intervals has similar sounds and share some characteristics like harmony. In order to replicate the harmony included in the main motif, a diatonic transposition function was created in our program.

Definition 3.3 *For an interval $\{a_0, a_1, a_2 \ldots a_3\}$, a diatonic transposition of value k of this interval is $\{a_0 + k, a_1 + k, a_2 + k \ldots a_3 + k\}$.*

However, in order to output a coherent and consonant piece, the value of diatonic transposition was generated using a random numbers generating function that was limited to a maximum of ± 24.

3.2 Melody Reversion

Melody reversion is another way to keep the same intervals between a melodic line's notes. However, it reverses each up interval to a down interval and each down interval to an up interval.

Definition 3.4 *The reversion of a melody $\{a_0, a_1, a_2, \ldots, a_3\}$ is the melody*
$\{2a_0 - a_0, 2a_0 - a_1, 2a_0 - a_2, \ldots, 2a_0 - a_3\}$.

3.3 Retrogression

Another transformation method that keeps both the used notes and the melody intervals is retrogression.

Definition 3.5 *For every set of intervals, $\{a_0, a_1, a_2, \ldots, a_n\}$, its retrograde is defined as $\{a_n, a_{(n-1)}, a_{(n-2)}, ..., a_1\}$*

3.4 Duration Scaling

One of the musical characteristics for a piece is its rhythm. In order to make a musical piece interesting, it should have a varying musical rhythm that would make the audience in some periods, calm and relaxed and in others excited and active. These can be done by using large notes duration or small notes duration respectively.

Definition 3.6 *A duration scaling by a factor k for a melody is defined as scaling the duration of each note of this melody*

Restrictions on duration scaling include not making a new modified duration to have more than 4 times its corresponding one in the main motif or less than $\frac{1}{4}$ times its corresponding one.

3.5 Ensembling

The above mentioned rules are to manipulate a motif to generate new melodic lines. These functions were used according to the following ensembling rule: Every generated melody were randomly operated by any number of the mentioned methods.

4 ABC Notation

ABC Notation is a text-based notation to write music sheets. Several tools are available online to mark-up pieces of music written in the ABC Notation. In our project, we transform the data structures described in section 2 to ABC Notation. We then use one of the available online tools[1] to transform it to Midi Format and music sheet.

4.1 Example and Explanation of ABC Notation

Several books and tutorials have been written to explain the ABC Notation. We refer the reader for [2] for a more thorough introduction to the ABC notation. We provide here a simple example and explanation of the ABC notation from [2].

Listing 6: Example of the ABC Notation (From [2])

```
X:1
T:Paddy O'Rafferty
C:Trad.
M:6/8
L:1/8
K:D
dff cee|def gfe|dff cee|dfe dBA|dff cee|def gfe|faf gfe|1 dfe dBA:|2 dfe dcB||
~A3 B3|gfe fdB|AFA B2c|dfe dcB|~A3 ~B3|efe efg|faf gfe|1 dfe dcB:|2 dfe dBA||
fAA eAA|def gfe|fAA eAA|dfe dBA|fAA eAA|def gfe|faf gfe|dfe dBA:|
```

The music sheet description starts with the title (T :), the time signature (M :), the default note length (L :), and the key (K :). Then, the notes follow, with a bar separating them to indicate an end of measure.For note duration, each note is followed by a multiple of the default note duration. For example, $A2$ above means the note A played twice as long as the default duration (i.e. $2 * \frac{1}{8} = \frac{1}{4}$).

5 Results

The code was iterated 20 times with different paramters every time in order to improve the output sound. It is found that the sound is more coherent when the first motif is one octave up or down the middle C and the generated melodies from this motif be in between two octaves up or down every note from the generated motif. After these 20 iterations, the output music was reasonably accepted. However, future verisions should consider the following characteristics which have partially applied in this project.

1. The generated notes in the first motif are better to be consonant and have small intervals.

2. Dynamics should be gradually changing throught the piece.

3. Non-uniform Probablity distribution should be used in choosing the notes, their durations and volume according to the composer's taste and interest.

The Used code in this project could be found here:
https://github.com/decltypeme/motifate-me

References

[1] Simon Becker. abc converter. http://www.mandolintab.net/abcconverter.php.

[2] John Chambers. Abc music notation tutorial. http://trillian.mit.edu/ jc/homepage.html. Accessed: 2017-05-10.

[3] Paul Walker. "Fugue". In L. Root, Deane. Grove Music Online. Oxford Music Online. Oxford University Press. May 15, 2007.

[4] Ball, Philip. "Artificial music: The computers that create melodies". BBC. Aug 8, 2014.

[5]"The Music of Chance." The Guardian. Guardian News and Media, 15 Jan. 2004. Web. 13 May 2017.