Review of Genetic Musical Accompaniment

Gregory Hughes
Computer Science
WVU Tech
gshughes@mix.wvu.edu

Mardigon Toler
Computer Science
WVU Tech
mmtoler@mix.wvu.edu

I. Introduction

In the pursuit of finding an algorithm to aid in musical accompaniment, certain benefits of Artificial Intelligence come to mind. While it is entirely possible for performers to play alongside of pre-recorded pieces of music, it can leave a lot to be desired. Pre-recorded music has the disadvantage of becoming stale after it is played many times, and musicians may wish to be able to improvise. Recordings would have no ability to adapt to this sort of behavior.

An artificially intelligent agent may be able to create musical accompaniment based on the performance of musician. A genetic algorithm is a type of artificial intelligence that may be applicable here. These algorithms work by creating a population of individuals, each of which has a chromosome representing pieces of a proposed solution. Like in the natural process of evolution, these individuals can reproduce over time to replace the current population, and fitter individuals (those for which a problem-specific fitness function will have the largest relative output) stand a better chance of reproducing. Individuals also have a random chance to mutate, an event which modifies their chromosomes unpredictably. In this review, we look at a 2012 article about genetic music generation and compare it to an earlier work and to a project of our own.

II. Polyphonic Accompaniment using GA and Music Theory

Liu and Ting's Artificial Intelligence, described in "Polyphonic accompaniment using genetic algorithm with music theory," was designed to create polyphonic accompaniment for a piece of music by applying a genetic algorithm whose fitness function is based on ideas from music theory [1]. Their AI was focused on providing main accompaniment, bass accompaniment, and chord accompaniment. [1]. In their implementation, the genetic algorithms for these three pieces of accompaniment were performed one after another, getting feedback from previous results. That is, the accompaniment output from one genetic algorithm is part of the input for the next genetic algorithm.

Their fitness function takes musical sections as input and examines it for various features, each of which has a corresponding weight assigned to it. For example, the presence of a perfect fifth interval in the input (for main accompaniment) causes that chromosome to be awarded with +15 fitness. On the other hand, a feature such as the presence of a no chord notes (in an individual for bass accompaniment) will result in the fitness being reduced by 20. A table of weights are provided by McIntyre for each of the different stages of generation accompaniment. That is, different features of an individual will be judged differently depending on which type of accompaniment is being generated.

Liu and Ting seem to claim that their project has many improvements over past genetic music generators [1]. The points they bring up are valid. Making use of human feedback in genetic algorithms is time-consuming and inefficient. Using music theory rules for a fitness function can allow for a faster breeding process that still closely approximates human composition [1]. However, they make these claims in such a way that makes them seem uninformed. In their introduction, they talk about a number of past music generators that made use of genetic algorithms, briefly describing each one. At the end of their walk through history they talk about how they use music theory as the basis for their fitness function and they say it in such way that makes it seem like they were one of the first to design their fitness function in such a manner [1]. By taking a look at the first project they cited in their introduction, it becomes evident that was not the case.

III. Early Work

The first work cited by Liu and Ting is Bach in a Box: the evolution of four part Baroque harmony using the genetic algorithm by Ryan A. McIntyre [2]. Published in 1994, this article is about McIntyres attempt to use artificial intelligence to compose Baroque-style music with the use of a genetic algorithm. He spends most of his Background section defining what made music from the Baroque period unique. He explains how music from that era was made up of multiple parts or voices that each have their own unique tune but also form chords when combined. Using a simple major-key soprano melody as input, McIntyres program is able to accurately generate the alto, tenor and bass harmonies to create a basic Baroque-style composition. McIntyre also states that he is quite pleased with his results [2]. What makes his article noteworthy in this context, however, is how he formulated his fitness function.

McIntyre uses rules based on music theory to make his fitness function. Specifically he uses a set of seven functions that grade each individual in such a way that the sum of the best scores they could give equals 68 times the length of the individual. These functions are all based on concepts of music theory such as harmonic resolution, chord repetition, or a lack

thereof depending on the context, and contrary motion. There are two functions, one based on chord formation and one based on the space between tunes, the take higher precedence over the other five. What this means is that an individual must score high enough on these two rubrics before it can be judged further [2]. What this tells us is that, whether falsely interpreted or not, Liu and Tings supposed claim that they pioneered the use of music theory in fitness functions is false. While this may have never been the intention of the authors, the ambiguous wording in the antepenultimate paragraph of their article. Regardless, we would now like to talk about our project as it compares to Liu and Tings.

IV. How ours Relates

Our fitness function, more formally, is computed for each individual $x_i \in \{x_1, x_2, ..., x_n\}$ where n is the population size, using a vector of recent input, $\vec{y}$, and a vector $\vec{z}$. Each element of these vectors corresponds to a certain note on a piano keyboard, and the value of each element represents how often that note occurs in a signal. Thus, each vector can be thought of as a histogram of notes. The fitness of an individual $\vec{v}$ is given by:

$$Fit(\vec{v}) = \vec{v} \cdot \vec{y} - \vec{v} \cdot \vec{z}$$
$$= \sum_{i=1}^{m} \vec{v}_i \vec{y}_i - \vec{v}_i \vec{z}_i$$

where m is the number of elements (notes) in the vectors. This can be thought of as the "likeness" of an individual's histogram to the input histogram minus the "likeness" of it to the histogram for bad choices of notes.

The primary differences between our genetic music algorithm and the GA proposed in [1] are data structures used to represent musical information and the user interface. While the GA by Liu and Ting computes a musical accompaniment given an entire section of music as an input, our artificial intelligence is designed to be operated in real time during a performance. In addition, our genetic algorithm models the performance by maintaining a FIFO queue of recently input notes and maintains a histogram of this queue. The queue is the individuals chromosome. This histogram is used directly as the input to the genetic algorithms fitness function; that is, each individual is ranked based on the frequency of occurrence of notes its proposed queue. The representation implemented by Liu and Ting relies on storing musical sections as a whole in data structures [1] (our implementation always stores a number of the most recent notes, regardless of musical section). The differences between these data structures relate to differences in the motivations of the fitness functions. Our fitness function increases as an individuals histogram is more similar to the histogram of the input and decreases proportionally to its similarity with a precomputed histogram of bad notes. In contrast, the fitness function of Liu and Ting, is comprised of rules inspired by music theory. Their AIs chromosomes are judged by their compliance to these rules [1]. In addition to the different data structures, the dissimilarities also extend to the interface. As our genetic algorithm is meant to provide accompaniment while the user is playing music, all fitness functions are computed only with the knowledge of the recent portion of a performance. The Music Theory fitness function of Liu and Ting, however, is not limited to only a recent performance history but may consider large musical sections as the output of their genetic algorithm is meant to be computed for an entire pre-recorded performance.

V. Conclusion

There is still much work to be done in applying Artificial Intelligence to artistic endeavors. As shown by Liu and Ting [1] and McIntyre [2], an important area of discussion in applying genetic algorithms to music is defining an objective fitness function to judge phenomena, like music, that are experienced subjectively. The application of Music Theory is an important step in bridging this gap. Our genetic algorithm is loosely determined by some ideas taken from Music Theory but relies on a fitness function that is designed to consider recent performance only in an attempt to provide an AI that is useful for live performance accompaniment.

References

[1] Chien-Hung Liu and Chuan-Kang Ting, "Polyphonic accompaniment using genetic algorithm with music theory," 2012 IEEE Congress on Evolutionary Computation, Brisbane, QLD, 2012, pp. 1-7.

[2] A. McIntyre, "Bach in a box: the evolution of four part Baroque harmony using the genetic algorithm," Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence, Orlando, FL, 1994, pp. 852-857 vol.2.