Elevated Pitch: Automated Grammatical Evolution of Short Compositions

John Reddin[1], James McDermott[2], Anthony Brabazon[2], and Michael O'Neill[2]

[1] Trinity College Dublin reddinj@tcd.ie
[2] University College Dublin jamesmichaelmcdermott@gmail.com
anthony.brabazon@ucd.ie m.oneill@ucd.ie

Abstract. A system for automatic composition using grammatical evolution is presented. Music is created under the constraints of a generative grammar, and under the bias of an automatic fitness function and evolutionary selection. This combination of two methods is seen to be powerful and flexible. Human evaluation of automatically-evolved pieces shows that a more sophisticated grammar in combination with a naive fitness function gives better results than the reverse.

Key words: Grammatical evolution, music, automatic composition

1 Introduction

Automatic musical composition is a topic of both theoretical and practical interest, and evolutionary computation (EC) has been used for this task with some success. One approach which has received relatively little attention is EC guided by generative grammars — for example, grammatical evolution or GE [1]. There is a strong motivation for this approach, since generative grammars have been used extensively in music theory and analysis.

In this paper, we present a system for automatic grammatical composition and consider some key questions in this area. We introduce GE with an example of a grammar for generating music, and look at methods of implementing automatic fitness functions. This paper thus responds to McCormack's call [2] for research into automatic fitness functions. Since grammars can be open-ended and recursive, this paper also responds to Bentley's call [3] for open representations in creative domains. The chief contributions are a demonstration that GE is a method suitable for automatic composition, and a set of experiments with human evaluation of automatically-composed pieces.

2 Previous Work

There is a significant body of work in the area of generative composition. EC is a popular paradigm, whether using automatically-calculated fitness (e.g. [4]) or interactively (e.g. [5]). The ideas are not limited to academia [6].

However, a great deal of work remains to be done. Systems which use unstructured representations may be capable of producing all desired pieces, but at the cost of making them "needles in a haystack". A naive implementation of a linear-genome GA for composition might have each integer gene giving the pitch value of a directly-corresponding note. Such a GA could represent all possible melodies (for a fixed note duration), but the vast majority of these melodies will feel (at best) somewhat meandering.

An alternative is to impose structure on the search space using a formal grammar. Grammars are well-established as tools of musical analysis and creation [7–9]. There is evidence that listeners perceive in music the hierarchical structures characteristic of generative grammars [9]. The use of grammars addresses a key problem suffered by simple linear representations of music, which is that the compositional form is fixed. Representations for creative evolution may benefit from being open-ended [3]. Mapping via a grammar can lead to a piece of music of any number of voices, or any duration — as decided by evolution.

To our knowledge, Ortega and co-authors [10] are the only researchers to report work using GE to compose music. This paper was a useful proof-of-concept, though the key issues such as grammar design, fitness functions, and subjective evaluation were left unaddressed.

3 Introduction to Grammatical Evolution

GE is a type of genetic programming which uses generative grammars to define the possible forms of "programs" [1]. A grammar consists of multiple rules, each with a left-hand-side (LHS) and multiple possible productions. Derivation begins with a specified LHS. At each step, the first LHS in the derived string is replaced by one of its corresponding productions. The choices of productions are made according to successive values in the linear integer-valued genome. Two simplified fragments from the grammars described in Sect. 4.2 are given in Figs. 1 and 2.

```
<melody> ::= <chord_material><bass_material><melody_material>
<melody_material> ::= <bar><bar><bar><bar>
<bar> ::= <note_or_rest><note_or_rest><note_or_rest><note_or_rest>
<note_or_rest> ::= <quaver> | <quaver_rest> | <semiquaver><semiquaver>
<quaver> ::= <midi_pitch> 0.5
```

Fig. 1. A fragment of a simple musical grammar. 0.5 indicates the quaver's duration.

4 Experimental Setup

The overall aim of the experiments was to compare human judgement of pieces produced by contrasting GE setups. Four experiments are to be described: each consisted of two stages, as follows.

```
<melody_material> ::= <bar><bar_or_op><bar_or_op><bar_or_op>
<bar_or_op> ::= <bar> | <op>
<op> ::= COPYPREVBAR | SHUFFLEPREVBAR | TRANSPOSEPREVBAR <n>
       | SHUFFLEPREVBAR | REVERSEPREVBAR | INVERTPREVBAR
       | COPYPREVBARTRANSPOSEFINALNOTE <n>
```

Fig. 2. A fragment of a grammar with transformations. `<n>` indicates an integer-valued transposition offset.

Firstly, in each case, two sets of ten short pieces were produced according to two contrasting automatic evolution schemes. These schemes differed in each case by the grammar, the fitness function, or the contrast between evolution and mere random generation — as explained in Sect. 5.

A typical GE setup was used, with a population size of 200 and 300 generations. Fixed-point crossover had a probability of 0.9 and the mutation probability was 0.01. Selection was roulette-wheel, and replacement was generational with 1-individual elitism. The chromosome size was 500 and two occurrences of the GE wrapping operator were allowed per individual.

Secondly, in each case, the ten pairs of pieces were presented to multiple subjects for listening and evaluation. The order within each pair was randomised. The number of volunteers for the four experiments was between 13 and 22. Subjects were required to state a preference for one track or the other.

4.1 Fitness Functions for Musical Composition

The question of automated fitness functions is a central one in evolutionary art and music [2], and one which will likely never be fully resolved. Perhaps the best that can be hoped for is to produce a fitness function which penalises obviously undesirable qualities on a limited domain: as Dahlstedt [4] says, automatic fitness functions can be used to weed out bad results, relying on the expressive power of the representation to produce good ones.

With this in mind, we immediately define one simple fitness function for short pieces of music by fixing the desired scale (say, C major) in advance, and counting the notes in the piece which are not in this scale. By minimising this number through evolution, it is possible to achieve pieces which conform to the simple requirement that the piece be strictly in the desired key. However, this fitness function is clearly insufficient in itself. There are many possibilities for more sophisticated measurement of the overall "quality" of a piece of music.

Towsey and co-authors defined [11] a set of 21 melodic measures, each of which calculates the value of a statistic over a melody. For example, *pitch range* subtracts the lowest from the highest pitch, and *note density* calculates the proportion of notes to rests. These measures were calculated over a corpus of "good" melodies to give an allowable range for each measure.

This approach is adopted here for our second fitness function. We chose the 6 measures given by Towsey et al. with the lowest standard deviations across the set of melodies they used, omitting two inapplicable ones (*rhythmic variety* and *rhythmic range*), and replacing them with two more measures with very low standard deviations. The 6 measures to be used are summarised in Table 1.

Table 1. Melodic measures.

Measure	Desired value	Standard deviation
Pitch variety	0.27	0.11
Dissonance	0.01	0.02
Pitch contour	0.49	0.06
Melodic direction stability	0.40	0.11
Note density	0.49	0.15
Pitch range	0.50	0.10

Fitness was formulated based on these measures in a method similar to that of Dahlstedt [4]. Given a desired value v_d, and a standard deviation s, values $v \in (v_d - s, v_d + s)$ within the desired range were awarded a component-fitness value of 0. Values below the range were awarded a component-fitness value of $\sqrt{(v_d - s) - v}$, whereas values above the range were awarded a component-fitness value of $\sqrt{v - (v_d + s)}$. The component-fitness values were then averaged across all six measures (giving a result in $[0, 1]$), and the result was averaged with that returned by the simple fitness function described above.

4.2 Musical Grammars

Two types of grammars were used in experiments. An *unstructured* grammar (as in Fig. 1) simply states that a piece consists of N voices, a voice consists of M bars, a bar consists of 4 beats, and each beat consists of either a quarter-note or two eighth-notes, and each note consists of a note proper, or a rest. Such a grammar imposes very little structure on the search space — in fact, it is equivalent in a sense to the linear-genome GA mentioned earlier.

A *structured* grammar (as in Fig. 2) was designed which takes advantage of some aspects of musical knowledge available to all composers. For example, when composing a melody, it is common to repeat a sequence (perhaps with alteration), to shuffle previous notes, to invert or reverse a sequence or to transpose a sequence. These transformations were implemented in the grammar so that when the mapping process chose to express a transformation, it operated on the material of the previous bar and the result lasted exactly one bar. This is a simple implementation to be expanded in future work.

5 Results

Some example tracks, together with the experimental software, are available for download from `http://www.skynet.ie/~jmmcd/elevated_pitch.html`.

Experiment 1 compared random generation and evolution. A 16-bar unstructured grammar was used, with a simple fitness function. Fig. 3(a) shows the results. For each user, the number of preferences of unevolved and evolved tracks were summed: the boxplots represent the distribution of these sums. The evolved tracks were preferred much more often.

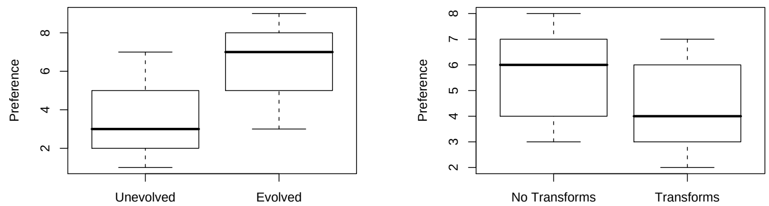

Fig. 3. Results from experiments 1 and 2. The evolved pieces were preferred to the unevolved ones, but the addition of transformations to the grammar was not successful.

Experiment 2 compared (8-bar) unstructured and structured grammars using the simple fitness function. Fig. 3(b) shows the results. There was a preference for the outputs of the unstructured grammar — a surprising result. The hypothesis was formed that the short length of the pieces in this experiment, together with the possibility that the melody would rest in the first 2 or 4 bars, was hiding the shortcomings of the unstructured grammar. This led to the design of new grammars similar to those used in the previous experiment, but producing 12-bar pieces and preventing the melody from remaining silent in the initial bars.

Experiment 3 compared these 12-bar unstructured and structured grammars, again with the simpler fitness function. The results are shown in Fig. 4(a). Users preferred the structured grammar, with results significant at the $p < 0.05$ level.

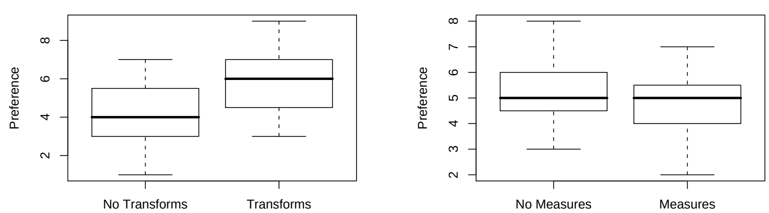

Fig. 4. Results from experiments 3 and 4. This time, transformations made a big improvement, but additions to the fitness function made things worse.

Finally, experiment 4 compared the simple fitness function and the more sophisticated one using melodic measures. A 12-bar unstructured grammar was used. The results (in Fig. 4(b)) turned out to be worse for the fitness function using melodic measures, contrary to expectation. One possible explanation is that here automatic evolution, in attempting to satisfy multiple objectives, failed to eliminate all non-scale notes from the pieces.

6 Conclusions

Through human evaluation of automatically-composed pieces, GE has been shown to be capable of producing results which are better than random generation, and it has been shown to be sufficiently flexible to be extended in multiple directions, through constraints (in the grammar) and biases (in the fitness function), separately or together. This flexibility is a great advantage of GE.

The strongest results appeared in the third experiment: subjects were able to recognise that structure had been imposed on melodies and several subjects reported that they enjoyed this experiment more than others. One subject commented "[...] after the first few plays, I feel like I began to detect how the transformations were occurring." This is a positive result.

Our work in using GE in creative domains is ongoing. Among many possibilities we mention four for potential future work: better automated fitness functions; the comparison of the grammatical approach with state-of-the-art implementations of other methods, such as tree-genome GP and N-gram models; more complex grammatical models; and interactive GE for musical composition.

Acknowledgments John Reddin is funded under Science Foundation Ireland's UREKA scheme. James McDermott receives Irish Research Council for Science, Engineering and Technology post-doctoral funding.

References

1. O'Neill, M., Ryan, C.: Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language. Kluwer Academic Publishers (2003)
2. McCormack, J.: Open problems in evolutionary music and art. In Rothlauf, F., et al., eds.: Applications of Evolutionary Computing, Springer-Verlag (2005)
3. Bentley, P.J.: Exploring component-based representations - the secret of creativity by evolution? In Parmee, I.C., ed.: Proceedings of ACDM 2000, University of Plymouth (2000) 161–172
4. Dahlstedt, P.: Autonomous evolution of complete piano pieces and performances. In: Proceedings of Music AL Workshop. (2007)
5. Nelson, G.L.: Sonomorphs: An application of genetic algorithms to the growth and development of musical organisms. URL `http://timara.con.oberlin.edu/~gnelson/gnelson.htm` (1993)
6. Eno, B.: A Year with Swollen Appendices. Faber and Faber (1996)
7. Lerdahl, F., Jackendoff, R.: A Generative Theory of Tonal Music. MIT Press, Cambridge, MA (1983)
8. Kippen, J., Bel, B.: Modelling music with grammars: Formal language representation in the BOL processor. In Pople, A.M..A., ed.: Computer Representations and Models in Music. Academic Press, London (1992) 207–238
9. Marsden, A.: Generative structural representation of tonal music. Journal of New Music Research **34**(4) (2005) 409–428
10. de la Puente, A.O., Alfonso, R.S., Moreno, M.A.: Automatic composition of music by means of grammatical evolution. In: Proceedings of APL 2002, Madrid. (2002)
11. Towsey, M., Brown, A., Wright, S., Diederich, J.: Towards melodic extension using genetic algorithms. Educational Technology & Society **4**(2) (2001)