AUTOMATING THE COMPOSITION OF POPULAR MUSIC: THE SEARCH FOR A HIT

T. Millea And J. Wakefield
University of Huddersfield, Queensgate, Huddersfield HD1 3DH, UK

ABSTRACT

The field of automated music composition has existed since the 1950s and spans a wide variety of techniques. Popular music is often thought of as being compositionally simpler than classical forms but, as far as is known no automated composer has ever had a hit record. A system is proposed which uses the decomposition of an input set of existing music to guide the search for new popular music within an evolutionary algorithm. A novel representation of music is proposed to reduce the search space and is expected to improve the quality of the results.

Keywords: popular music, automated composition, representation, evolutionary algorithm

1. INTRODUCTION

"*Music is the universal language of mankind*" (Henry Wadsworth Longfellow)

Evidence suggests that every known culture has made and listened to music since the evolution of modern humans in East Africa around 200,000 years ago (Wallin et al 1963). Today we find music in our daily lives on television and radio, on our telephones, digital music players and computers, played while we shop, eat or drink, to celebrate or dance to and to mark social gatherings. Music is also big business. According to the Recording Industry Association of America (RIAA), the global music market has a turnover of around forty billion U.S. Dollars per annum. That all commercially successful music is, as far as we know, human-composed in an industry of this size is testimony to difficulty of the task of programming computers to compose music that we humans like.

The term *popular music* is usually associated with commercial music, i.e. music recorded and bulk distributed for profit, which has its origins in the sale of duplicated sheet music in the mid-eighteen century. Today, commercial music covers a bewildering array of diverse musical genres including avant garde, country, folk, jazz, latin, rap, rhythm and blues, reggae, rock, World and some eclectic crossovers of these. Here the term *popular music* shall be taken to mean little more than the the sum of its parts: the adjective popular, in the sense of appealing to, or being liked by, a large number of people or a large proportion of a given populous, and the noun music:

"Music is an art form whose medium is sound. Common elements of music are pitch (which governs melody and harmony), rhythm (and its associated concepts tempo, meter, and articulation), dynamics, and the sonic qualities of timbre and texture. The word derives from Greek μουσική (mousike), (art) of the Muses". (Wikipedia 2009)

The aim of this project is to develop an automated means of composing popular music. The system will generate new music based upon the analysis of a given set of existing music. For example, given a set of songs of a particular genre, the system should generate new songs of the same genre and of comparable standard. The interfaces of the system's input and output will be electronic scores in the ubiquitous Musical Instrument Digital Interface (MIDI 2009) file format. Lyrics, music performance and production are therefore outside the scope of the current work.

Popular music makes an interesting subject for automated composition. It has the advantage of being compositionally simple compared to classical forms. However, the requirement of popularity brings with it its own research challenges.

2. A BRIEF HISTORY OF AUTOMATED MUSIC COMPOSITION

The first significant computer-composed piece of music was the Illiac Suite, a piece for string quartet, in 1957 (Hiller & Isaacson 1957). Since that time, the field of automated music composition, also known as

algorithmic composition, has become an established inter-disciplinary field. It contains a great diversity of techniques which may be broadly categorised as follows.

2.1. Rule-Based

Since the Renaissance period, the process of musical composition has been partially automated through the formulation and application of music treatises and by teaching music students the rules and constraints that music treatises typically encode. Perhaps the most famous early example is Fux's treatise on counterpoint (Fux 1725). Composition may be automated by regarding a treatise as a set of constraints to be satisfied. For a survey of such work see Pachet & Roy (2001). Closely related are approaches based upon Chomsky's theory of generative grammar (Chomsky 1956), in which music is regarded as a sequence of sentences generated according to a set of production rules. The production rules may be derived from existing music and/or embody music theory, e.g. Holtzman (1980), Roads & Wieneke (1979) and Steedman (1984).

2.2. Stochastic

Randomness is introduced to music composition systems to add originality and/or variation and to enable the composition of a large number of compositions from the same program with the same input. Statistical measures of features of existing music may also be used in the generation of new music to ensure it shares similar features. A common such use in this field is the use of Markov chains to stochastically generate the next musical event, e.g. a note or chord, based on the sequence of previous ones. Pachet's Continuator (Pachet 2003) is an excellent example that can convincingly generate short segments of music, or *continuations*, in the style of any human providing the input.

2.3. Chaotic

Chaos is the branch of mathematics that explains how complex, otherwise unpredictable systems are the deterministic result of the interaction of relatively simple starting conditions. Fractals, which are self-replicating patterns at ever higher scales, visibly demonstrate the chaotic principle. Chaotic music is generated through the interaction of a set of simple musical rules at ever higher scales, e.g. Bidlack (1990), Bolognesi (1983) and Harley (1994).

2.4. Artificially Intelligent

Artificial intelligence (AI) is the branch of computer science concerned with simulating successful, external human behaviour, as distinct from cognitive science which seeks to model the internal processes. The AI approach to music composition therefore does not seek to emulate human musical creativity but rather produce systems which generate music that could have been composed by a human. Many AI problems can be framed in terms of representation and search. Theoretically, a data type representing music defines a search space which only need be searched in order to find all existing and future music. Much effort in this approach is devoted to the choice and design of the representation, reducing the size of the search space and devising means of recognising good solutions when they are found. The field of AI also includes learning systems such as neural networks which may be trained on existing music then used to constrain or guide automated generation.

In addition to the above broad classifications, a distinction is made between fully automated systems and those, such as composing tools or David Cope's Musical Creativity work (Cope 2005), which involve some human guidance, interaction or selection.

In practice, many automated composing systems fall under more than one of these classifications as is the case with this project.

3. APPROACH: THE SEARCH FOR A HIT

In his book Six Steps to Songwriting Success, Jason Blume states that the most successful popular music structure is verse-chorus-verse-chorus-bridge-chorus (Blume 1999). An expert system could use such domain knowledge when generating new music. However, other genres of music may have different structures and different common elements. In order to cope with the widest variety of popular music genres, the proposed system should automatically extract this knowledge from its input. Besides, a musically-agnostic approach may extract common features that a human may miss while avoiding some hard-coded restrictions which do not apply to a particular input set. Evolutionary algorithms, e.g. genetic algorithms (Holland 1962, Holland 1975), are search mechanisms that, to varying degrees, model and acquire the search benefits of the natural evolution phenomena of mutation, recombination and survival of the fittest. A population of trial solutions to the given problem gradually adapt to their environment, as defined by some measure of fitness. The search may terminate upon finding an optimal solution, or upon some other

termination criteria, or it may continue indefinitely as in dynamic searches where the fitness function changes over time. For a comparison of the features of the most common evolutionary algorithms see Back & Schwefel (1993).

The following is the conceptual outline of an evolutionary algorithm for composing music.

create an initial random population of songs

REPEAT

FOR size of the number of offspring songs DO pick two songs

COMBINE them to form a new song

MUTATE the new song evaluate the FITNESS of the new song

SELECT a new population of songs

UNTIL we have a song with hit potential

Initially random songs are combined and mutated to form new offspring songs. The fitness of each song is calculated with respect to an analysis with the given set of existing songs. The fittest songs are the most likely to survive and pass-on their musical qualities. The search is simplest, fastest and most likely to succeed with the smallest possible search space. The dimensions and size of the search space is defined by the representation. This therefore means that choice of an appropriate representation is crucial to finding good solutions in an acceptable time frame.

The key research issues to address are therefore the musical representation, the combination and mutation operators of the evolutionary algorithm and a means of determining fitness.

3.2. Representation

The ideal representation corresponds to a search space that contains only hit songs that have not yet been written. The search in this case would never fail. The worse case search space is infinitely large and contains not one hit. Between these extremes there are many possible representations corresponding to search spaces which contain huge redundancy, e.g. admitting sequences of notes that lack musical meaning or structure, and noise which affects the search but is not critical to the composition, e.g. including the particular instruments used as in MIDI.

Low level MIDI note on and off events are first converted to a sequence of notes for each instrument.
The datatype for Note retains the musical data, i.e. pitch, volume, duration and instrument. Each Note also has an *offset* field which is the time to the onset of the next note and may be zero. This intermediate representation conveniently handles arbitrary polyphony as well as inter-note rests. This representation is amenable for the inference of the lead instrument, usually the melody or instrumental solo line, and corresponding harmonisation. Lead plus harmony is the amount of detail considered sufficient for musicians to recreate the given song, or for a non-musician to recognise it. Removing much of the noise and redundancy of the original MIDI files helps to vastly reduce the search space. As the next step, much of the absolute information, such as pitches and durations are replaced with pitch intervals and quantised to relative durations. This is the first order representation.

A piece of popular music typically contains much repetition and variations, e.g. the chorusses and verses both usually repeat their respective melodies. The music may be therefore represented *in less space* as a sequence of higher level patterns and their variations. Consider a sequence of n notes in which a pattern of p notes which occurs f times. Like grammatical generation in reverse, we can replace each occurrence of the pattern with an identifier, c.f a non-terminal symbol. The new symbol *covers* f x p notes or f x p / n of the total. However, the new symbol must occur f times in place of the original pattern and require a reference to the original pattern which is of length p. Therefore the pattern whose substitution saves the greatest space is the one that maximises (f x p) / (f + p). This corresponds to the intuitive notion of repetitiveness.

Once the most repetitive pattern has been substituted, the second most repetitive pattern is identified and substituted and so on until the entire sequence is covered and replaced by a new sequence of symbols. This is the 2nd order representation. The same algorithm is applied to the new sequence, to find patterns of patterns, in order to generate the third order representation, and so on until there are no further repeated

patterns. The resulting set of sequences and lookup tables corresponds to a top down hierarchical description of the original piece with the production rules and symbols in order of repetition required to recreate it. Work is currently underway to implement this algorithm.

Thus we have a higher-order, recursive data type capable of capturing repetition and variation of musical patterns at every scale. The lowest order represents patterns of individual musical events, e.g notes, while the highest order will represent the top-level structure of the song. Only the relative pitches and quantised durations of melody and its harmonisation are retained in order to further reduce the search space of the evolutionary algorithm and simplify the musical analysis of the input.

3.3. Combination

The purpose of combination or crossover in an evolutionary algorithm is to mix elements of two or more trial solutions in the hope of that some offspring are fitter than their parents. Here we wish to combine the elements of two songs in a musically meaningful way. There are a number of possible means of combination. Firstly, as in genetic algorithms, sequences from each parent may be simply spliced together at a randomly chosen point in the corresponding level. Secondly, some of the nonterminal symbols in one order of one song may be redirected to refer to patterns in corresponding order below in the other song. A simple example at the highest level would be the replacement of the chorus of song A by the chorus from song B. At the lowest level it could be the replacement of a repeated bar or line in one song by one from the other. This type of combination will be termed *crossindirection* and may be *partial* as per these examples or *total* where the offspring randomly acquires entire sequence at each level from the corresponding levels of each of the parents. Finally, elements or entire sequences may jump levels, e.g. a second order pattern in one parent may be promoted to a third order pattern in the offspring. In each case, where the corresponding number of symbols differs between the parents there will an added conforming step to ensure no reference is made to a nonexisting pattern.

3.4. Mutation

The basic unit of repetition in music is the bar. It is the unit from which all higher orders of patterns are built. If we enforce a same-length-multiple restriction of patterns at any given order, we gain the convenience of being able to represent one pattern as a vector offset of another. Where there are no common symbols in corresponding positions between two patterns, those patterns can be said to be independent. In all other cases, one can be said to be a variation of the other. A variation may also be regarded as a mutation of its corresponding difference vector applied to patterns other than those from which they were derived in order to achieve new and musically meaningful variations. Mutations will therefore be derived automatically from the input set.

The severity of a mutation is ideally controlled in relation to the overall success over a number of mutations, where a successful mutation is defined as one that results in increased fitness. The underlying principle is that in a random population, far away from the optimal solution, there is an expectation that half of all mutations are successful. However, as the optimal is approached, there are fewer opportunities for improvement, so the average success rate falls until, at the optimal, all mutations are unsuccessful. The gradual reduction in mutation severity as the population evolves is analogous to the reduction of energy in simulated annealing (Kirkpatrick & Sorkin 1998). Related to the velocity of convergence towards the optimal, this control becomes a powerful negative feedback mechanism in the search.

In order to control mutation severity, we require a means to measure the distance between the two same-length patterns from which it was derived. This work adopts the Kolmogorov variational distance metric using the Temporal Elements Displayed as Squares (TEDAS) system as described by Toussaint (2004). The distance between two rhythmic patterns is calculated graphically. The x-axis is time. Note onset times are marked on the x-axis and a square is inserted between each pair. The two patterns of squares are overlaid and the variational distance is the total difference in their areas. The method was derived from one originally intended to measure the distance between two melodies where the vertical axis represented pitch or pitch class, i.e. A-G# without octave information. Here we modify the original such that the vertical element of the difference is calculated from the harmonic distance between corresponding notes based on the harmonic series, i.e. in ascending order: octave, perfect fifth, perfect fourth, major third, and minor third and so on in declining pitch intervals.

Mutations are therefore musical variations automatically derived from the input. Their severity is calculated using a adapted Kolmogorov TEDAS system and mutations are picked at random from those of the appropriate severity in order to control the convergence of the evolutionary algorithmic towards fit solutions.

3.5. Fitness

The fitness of an offspring song will be calculated as the similarity between it and the given input songs. Straightforward metrics such duration, tempo, tonal range, etc. will be combined with Markovian analyses which reward trial songs sharing patterns at corresponding scales with the input set. Due to the stochastic nature of the search, it is expected that the evolving population of songs will acquire fitness-neutral musical features not present in the input set leading to original compositions. A number of fitness functions will be investigated with a view to meeting the overriding aim of the project.

4. CONCLUSIONS

This paper has briefly looked at the field of automated music composition and identified popular music as a candidate for research. It has also outlined in more detail a proposed approach using an evolutionary algorithm to evolve initially random pieces of music guided by an analysis of a given set of existing pieces of music. A novel musical representation has been adopted in order to significantly reduce the size of the search space and facilitate analysis and comparison. Besides simplifying the MIDI file input to its bare musical essence, i.e. lead and harmonisation, a recursive pattern-factoring algorithm has been devised in order to capture and represent the inherent repetition prevalent in popular music. Implementation work is currently underway and progress to date appears promising

REFERENCES

BACK T. and SCHWEFEL H. (1993), *An Overview of Evolutionary Algorithms for Parameter Optimization*. Evolutionary computation 1:1-23.

BIDLACK R. (1990), *Music From Chaos: Nonlinear Dynamical Systems as Generators of Musical Materials*. Music)--University of California, San Diego.

BLUME J. (1999), *6 Steps to Songwriting Success.* ISBN 0823084221.

BOLOGNESI T. (1983), *Automatic Composition: Experiments With Self-Similar Music*. *C*omputer Music Journal 25-36.

CHOMSKY N. (1956), *Three Models of Language*. IRE Transactions in Information Theory 2:113-124.
FUX J.J. and NAVARRE J.P. (1725) *Gradus Ad Parnassum,* P. Mardaga.

HARLEY J. (1994), *Algorithms Adapted From Chaos Theory: Compositional Consideration*s. Proceedings of the International Computer Music Conference 209-209.

HILLER L and ISAACSON (1957), *Illiac Suite: For String Quartet*. New Music Edition Corporation

Holland, J. H. (1962). *Outline for a logical theory of adaptive systems*. Journal of the Association for Computing Machinery, 3, 297–314.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: The University of Michigan Press.
HOLTZMAN S.R. (1980), *A Generative Grammar Definition Language for Music*. Journal of New Music Research 9:1-48.

KIRKPATRICK S. and SORKIN G. (1998), *Simulated Annealing*. The handbook of brain theory and neural networks book contents. Pages: 876 - 879. ISBN:0-262-51102-9

MIDI Manufacturers Association (accessed 18.09.2009), http://www.midi.org/

PACHET F. (2003), *The Continuator: Musical Interaction With Style.* Journal of New Music Research 32:333-341.

PACHET F. and ROY P. (2001), *Musical Harmonization With Constraints: A Survey*. Constraints 6:7-19.

ROADS C. and WIENEKE P. (1979), *Grammars as Representations for Music.* Computer Music Journal 48-55.

STEEDMAN M.J. (1984), *A Generative Grammar for Jazz Chord Sequences*. Music Perception 2:52-77.

TOUSSAINT G. (2004), A Comparison of Rhythmic Similarity Measures. *Proc. 5th International* Conference on Music Information Retrieval 242–245.

TURING A.M. (1950), *Computing Machinery and Intelligence*. Mind 59:433-460.

WALLIN N.L., BROWN S. and MERKER B. editors (1963), *The Origins of Music*, ISBN 0262731436.

WIKIPEDIA (accessed 18.09.2009) http://en.wikipedia.org/wiki/Music