Co-evolving Melodies and Harmonization in Evolutionary Music Composition

Olav Olseng and Björn Gambäck(✉)

Department of Computer Science, Norwegian University of Science and Technology, Trondheim, Norway
olseng.o@gmail.com, gamback@ntnu.no

Abstract. The paper describes a novel multi-objective evolutionary algorithm implementation that generates short musical ideas consisting of a melody and abstract harmonization, developed in tandem. The system is capable of creating these ideas based on provided material or autonomously. Three automated fitness features were adapted to the model to evaluate the generated music during evolution, and a fourth was developed to ensure harmonic progression. Four rhythmical pattern matching features were also developed. 21 pieces of music, produced by the system under various configurations, were evaluated in a user study. The results indicate that the system is capable of composing musical ideas that are subjectively interesting and pleasant, but not consistently.

1 Introduction

The first usage of computers for algorithmic music composition is generally accepted to have been the Illiac suite [1] in the 1950s. Since then, a wide range of techniques have been applied to composition. Rule-based systems, stochastic methods, grammar-based methods, neural networks, and population-based methods have all been utilized. With the emergence of cheap computing power, processing units are readily available, but the inspiration needed to compose is not always there. Even for a person with the skillset to create compositions, the ability to constantly re-invent themselves and produce new and interesting music can be elusive. Computer aided compositional tools such as the Vox Populi system [2] are, and have been, used as a source of inspiration for composers. Systems have been designed to perform a wide range of compositional and musical tasks, such as generating melodies, harmonization, counterpoint, improvisation, arranging, and in some cases like Melomics and Iamus [3], the ability to create fully fledged compositions aimed at orchestral performances in concert halls.

McCormack [4] defined a set of five open questions for research in algorithmic composition, specifically in evolutionary music. His second open question is formulated as: *"To devise formalized fitness functions that are capable of measuring human aesthetic properties of phenotypes. These functions must be machine*

Thanks to the four reviewers for comments that helped to greatly improve the paper.

A. Liapis et al. (Eds.): EvoMUSART 2018, LNCS 10783, pp. 239–255, 2018.
https://doi.org/10.1007/978-3-319-77583-8_16

representable and practically computable." This is still an important issue [5]. Critique has been made towards artefacts generated by evolutionary systems, in terms of having little structure. Defining fitness features that attempt to stimulate the emergence of patterns is one goal of the present work. Furthermore, systems that are able to generate both melody and harmonziation most commonly do these two tasks sequentially, and by different modules in hybrid systems. Implementing an evolutionary system that has the ability to co-evolve melody and harmonization is novel and the main goal of the paper.

An overview of the state-of-art in algorithmic composition is presented in Sect. 2. Section 3 describes the implemented algorithm, including minor modifications to the Non-dominated Sorting Genetic Algorithm II [6], genotype representation, genetic operators and fitness functions. In Sect. 4, the various configurations for how the music was generated is presented. Section 5 describes a user survey to evaluate the generated music. Finally, Sect. 6 contains a discussion of the results, followed by conclusions and suggestions for future work.

2 Related Work

The use of **evolutionary algorithms** (EAs) within algorithmic composition, and computer aided composition, first appeared in the early 90s. Some of them were fully automated [7,8], but were solving simple thematic bridging problems, in essence formulated as a puzzle, or toy problem, where the fitness function just was a distance measure to a pre-defined solution. In the following years, systems were developed for various compositional tasks with a variety of automated fitness functions. Marques *et al.* [9] composed polyphonic pieces, Papadopoulus and Wiggins [10] did jazz improvisation, and Johnson *et al.* [11] composed melodies. All these implementations used a fitness function that was a weighted sum of features. Towsey *et al.* [12] provided a description of a set of features to analyse musical pieces of various styles, but no generative implementation was presented.

Fully automated fitness functions have been a point of much research and discussion [4,5,13]. It has even been suggested using no fitness function at all [14]. Freitas and Guimarães [15] proposed this as a means to circumvent the artistic constraints that fitness functions impose. As a consequence, much of the domain knowledge previously found in the fitness function is moved into the genetic operators, the representation, and the initial population to ensure that the produced material is acceptable. Another way of introducing originality and diversity to the system-produced artefacts is using multi-objective fitness functions. This has been applied to harmonization [16,17] and melodic composition [18]. All of the three aforementioned papers use two contradicting functions, one for consonance and one for dissonance. This results in a musical trade-off between the two functions. Compositions made by these systems are said to have a broader language of expression, producing subjectively interesting compositions.

Musical Interactive Genetic Algorithms (MIGAs) are EAs where the fitness function is replaced by a human evaluator who ranks the solutions in some fashion. The next generation of individuals is then produced based on the human

evaluation. Nelson [19] described a very simple MIGA for generating rhythmic patterns over short melodies, where the generated music just contained the presence or absence of sound. The most notable MIGA is Biles' GenJam [20], which improvises jazz solos. During the interactive fitness evaluation of the composed music, the user simply responds with a good or bad evaluation, and the system learns from this. A common problem for interactive genetic algorithms is user fatigue. Evaluating the candidate solutions for each generation can be extremely time consuming. Biles hence tried to eliminate the human evaluator in GenJam. First he unsuccesfully attempted to replace the human element with a trained neural network [21], but then successfully completely removed the fitness function [14]. The musical acceptability was ensured by programming the knowledge previously found in the fitness function into the genetic operators, as well as presenting a good initial population.

Recently, focus has been directed to the encoding between genotype and phenotype. Given a complex search space, random mutation are unlikely to make similar musical figures appear in the artefacts. The Melomics system [3] contributed an "evo-devo" approach, whereby the system's genotype has an indirect encoding, which during the development phase expands the genome into a more advanced piece of music, enabling Melomics to deliberately develop themes, global structure and alter motifs. Due to the inherent strengths and weaknesses of different approaches, all previously mentioned techniques have at some point been hybridized. Two such **hybrid systems** are HARMONET [22] and its further development MELONET [23]. HARMONET has a 3-layered architecture, enabling it to compose 4-part chorales. The first layer is a neural network that extracts harmonic information from a melody. This information is then fed to a rule-based layer that creates a chord progression for the melody. The third layer arranges a 4-part harmonization for the melody. MELONET adds a fourth layer, another neural network that creates melodic variations for the voices, to make the produced artefact more interesting to the listener.

Evolutionary algorithms lend themselves very well to hybridization. The system introduced by Bell [24] uses an interactive evolutionary algorithm to evolve Markov chains that write melodies with accompanying text. Each genome is developed into three first-order chains, one for melody, one for rhythm and one for harmonization. Thywissen [25] designed an algorithm where generative grammars are evolved, while Khalifa *et al.* [26] used grammars as a part of the fitness function. Phon *et al.* [27] programmed a genetic algorithm for harmonization, with some of the genetic operators following explicit rules, instead of random mutations. NEUROGEN [28] used ANNs trained on a corpus as a fitness function to emulate the musical style found input data, while Chen and Miikulainen [29] evolved recurrent neural networks for melody generation. MetaCompose [30] incorporates both melody generation and harmonization. It composes autonomously and not based on any musical material. Chord progressions are generated by a random walk through a directed graph; a melody is then created for the chords through a multi-objective evolutionary algorithm, supported by a probability driven module for rhythms and accompaniment.

3 Multiple-Objective EA Implementation

Evolutionary algorithms (EAs) are inspired by the process of biological evolution, and try to optimize a solution with regards to some objective or environment. The algorithm is in essence a parallel heuristic search, performed by maintaining and developing several potential solutions at once. The algorithm combines and mutates these solutions in an effort to converge towards a global minimum or maximum, an optimal solution, with regards to a fitness function. In the standard EA the fitness function assigns a single numerical value to the phenotypes; however, in Multiple-Objective Evolutionary Algorithms (MOEAs), the phenotypes are evaluated by more than one fitness function, called objectives, and thus have more than one fitness value. This enables the algorithm to work towards multiple, and sometimes conflicting, goals. The result is a trade-off between the objectives, unless some solution exists where all objectives can be maximized.

The non-dominated sorting genetic algorithm (NSGA) is a multiple-objective evolutionary algorithm that uses Pareto dominance and Pareto optimality, from game theory, to rank the individuals in a population. A solution is Pareto dominated if there exists another solution where one fitness value is better, and the remaining features are not worse. A solution is Pareto optimal if it is not Pareto dominated by any other solution. The present system is an implementation of NSGA-II [6], which improves the run-time of NSGA from $O(MN^3)$ to $O(MN^2)$, where M is the number of objectives and N the population size. NSGA-II also supports elitism to improve fitness faster and a mechanism called crowding distance, which preserves diversity when exploring the state space.

NSGA-II differs from the standard EA in how it compares fitness values between individuals. In the standard EA, where there is only one fitness value, the individual with the higher fitness value could be considered better. In NSGA-II, a rank is assigned each individual in a population based on Pareto optimality. The ranking is computed by finding the non-dominated individuals and removing them from the population. Some individuals that were previously dominated then become non-dominated. This is done iteratively until there are no more individuals left in the population. The rank assigned to an individual is the iteration it was removed from the population. The non-dominated solutions removed in the first iteration obtain rank 1, the solutions removed in the second iteration obtain rank 2, and so on. When doing parent selection, tournament selection is used and the ranks are compared. If two individuals with the same rank are to be compared, the tie is resolved by comparing the crowding distance. This is in essence a measure of how close an individual is to its closest neighbours in the fitness landscape. The bigger the crowding distance, the more unique the solution is within a rank. By selecting individuals with a high crowding distance, genetic diversity is maintained in the population. Newly generated individuals are merged with the current population before ranking is performed. The population can then be trimmed with regards to a maximum population size. This is how elitism is supported, as individuals from previous generations will survive if they have a high enough rank to remain in the population during adult selection.

During preliminary tests of the algorithm, the performance of the ranking scheme collapsed when four objectives were used. 100% of the population would be ranked to the first front, meaning that they were all Pareto optimal, but still far from the true Pareto front. When this happens, there is an extreme selection pressure, and exploration of the search is hampered. Deb [31] describes this as a phenomenon that occurs when there are many fitness functions. This problem also occurred even if genetically identical phenotypes were removed from the population before ranking. Many remaining phenotypes would still have the same fitness value, despite being genetically very different. This implementation therefore employs a scheme that eliminates fitness duplicates to lessen the genetic drift, as described by Aguirre and Tanaka [32], to improve the performance when there are epistatic interactions in genomes. Since notes in the melody are evaluated in relation to consecutive and preceding notes, as well as the harmonization, employing this scheme improves the convergence of the algorithm by limiting the genetic drift. For parent selection, a binary tournament selection with an 80% chance to keep the highest ranked individual was adopted.

Since melody and harmony are evolved simultaneously, some global constraints are put on the domain. In particular, a diatonic key must be set to ensure that the harmonic domain is consistent between individuals of the population. If this is not done, crossover operators are likely to change the harmonic and melodic context to the point of reverting progress.

3.1 Genotypes and Phenotype

The genotype consists of two separate parts. One for the melody and the other for the harmonization. Representing the music this way allows the algorithm to optimize either one separately during run-time, or both simultaneously.

A flexible representation of the **melody genotype** is implemented to allow the exploration of a large melodic space, enabling the algorithm to represent a variety of musical expressions and to develop as many different musical ideas as possible. Some implementations, such as GenJam [20], use a database of melodic figures to map the resulting genotype into a melody. Such a database is not part of this architecture. Instead, efforts were made to put as little musical knowledge as possible into the representation, by allowing occurrences of what can be considered as "errors" in certain musical paradigms. The model of Jeong and Ahn [18] is very flexible, and is used as a baseline for the implemented model. A measure of music is divided into fractions, and each fraction represents a time-step, or a beat. Each step is occupied by an integer value that represents one of three things: either a pitch, a rest, or a hold. A rest means the absence of anything played, and a pitch value means that a new note of the corresponding pitch starts, analogous to striking a key on the piano. A hold means that the previous note, a pause or a pitch, is held from the previous step. The minimal time-step used in this model is a $\frac{1}{16}$ note, which results in twice as many time-steps per measure compared to the representation used by Jeong and Ahn. The representation also supports repeating pitches, and leaps of any size, within the defined range of allowed pitches. In order to produce melodies that are easy

0	4	7	-1

Fig. 1. Example harmony gene, in this case a C major triad chord.

to sing, the range of pitches employed is 65–85, corresponding to the integer-pitch mapping defined in MIDI [33]. Hence, the representable pitches are in the range from F4 to C#6, and the total number of different representable melody genotypes n, of length x measures, is $n = 22^{16x}$.

Harmonization is defined by chords, where a new chord appears at the first beat of every measure. Each chord contains at least three notes, with an optional fourth note that is a flavour note allowing chords to be built with a more musically advanced expression, compared to what is possible with only three pitches. In the **harmony genotype**, each chord is represented as a vector of four bytes, with each byte holding a value in the range $[-1, 11]$. The integers ≥ 0 represent a pitch, and -1 represents absence of a pitch (in case only three notes are used). An example of the genotype can be found in Fig. 1. In this representation, a 0 translates into a musical pitch of C, and following a chromatic scale, a 2 will represent an D, and so on. The total number of different representable harmony genotypes n, of length x measures, is therefore $n = (12 \cdot 11^3)^x$.

Both genotypes employ direct representations, so no explicit developmental method or phenotype is required. However, for optimization reasons a phenotype representation is employed to aid in the evaluation of the generated music. The developmental method iterates through the genotype and creates a variety of structs. These contain information about pitch and rest positions, durations and intervals, and mostly serve as indices utilized by the fitness functions.

3.2 Genetic Operators

As the system has two different data structures in the genotype (for melody and harmony), two different sets of mutation operators are needed. In addition, the algorithm incorporates a set of crossover operators in which both harmonization and melody are crossed over together, since they are heavily interdependent. Note that the application of these operators intentionally may produce genotypes that not necessarily make musical sense: the operators must be able to create unfit offspring in order provide alternatives to the fitness function.

One, and only one, of five possible **melodic mutation operators** is always applied during offspring creation, while restricted so that a pause cannot follow a pause or a held pause. The five melodic operators are: *Note Mode* generates a new value at a randomly selected time-step in the melody genome, either by copying the pitch value from the previous note or, if not applicable, choosing it at random. *Random Pitch* generates a random pitch value at a random position in the genotype. *Pitch Modulation* changes a randomly selected pitch by x semitones, where $x \in [-4, 4]$ and $x \neq 0$. If modulation puts a pitch outside of the valid defined range, the modulation is performed in the other direction, by changing

the sign of the modulation. *Note Position* selects a random pitch or rest in the genotype, and moves it by a random amount of time-steps x, where $x \in [-16, 16]$ and $x \neq 0$. *Duplication* causes the largest changes in the genome. It selects a random half measure, with a starting bound restricted to the beginning or middle of a measure, and overwrites another half measure in the genome with the same values, so that repetition of patterns occur by intention, and not by chance.

Similarly, there are three **harmonic mutation operators:** *Chord Change* changes the entire chord with regards to the musical key. It selects a random root pitch for the chord, within the scale of the provided key, and derives a triad native to the scale of the key. The operator might also add a flavour note, with a 50% chance, which will always be the appropriate seventh in regards to the root pitch and the scale. *Chord Pitch* normally modulates a random pitch in a random chord by a semitone up or down, but if the fourth pitch of the selected chord is chosen, a random pitch is inserted if the fourth pitch is absent, while if a pitch is present, it has a 50% chance to be set to the silent value, otherwise it will be modulated like a normal pitch. *Chord Swap* swaps the position of two randomly selected chords in the genotype.

The algorithm employs two different **crossover operators** that are applied during offspring creation by a 50% chance. In *single point crossover*, two genomes are spliced together with the splicing points selected at random, and restricted to being at the start of a measure. The same splicing point is used for both genotypes. *Single measure crossover* copies a measure from one genome to the other and is in essence a dual point crossover operator, with restrictions so that the splicing points are always placed at the start and end of the same measure.

3.3 Fitness Objectives

To evaluate the phenotypes, the implementation uses four different fitness objectives: melodic local and global objectives, harmonization and harmonic progression objectives. Each objective has a different purpose and evaluates a different aspect of the generated music.

The **Melodic Local Objective** is based on one of the fitness functions described by Wu *et al.* [34]. It is concerned with the tonality of the melody, the relation of pitches to the given key, and the interrelation of pitches within a measure. The objective is designed with the intention that melodies generated by the algorithm sound harmonic and pleasant. In this objective, a pitch can be defined as: a *chord pitch* (and non-harmonic) if it occurs within the chord of the given measure, regardless of whether the pitch occurs in the provided key or not; a *scale pitch* (also non-harmonic) if it occurs within the given scale, or key; any other pitch is a *non-scale pitch*. Further, a pitch that is not classified as a chord pitch is categorized if certain conditions are met. A *passing tone* is a pitch that is initiated by a step from a chord tone, and resolved by a step in the same direction to another chord tone. A *neighbour tone* is a pitch that is initiated by a step from a chord tone, and resolved by a step to the same chord tone. The sum of passing tones and neighbour tones are labeled *ornament tones*.

The objective scores assigned to each phenotype are based on nine conditions. The first six give a score of +1, if the condition is met: Non-harmonic Pitches < Chord Pitches, Passing Tones ≤ Scale Pitches, Neighbour Tones ≤ Scale Pitches, Non-scale Pitches ≤ Ornament Pitches, Ornament Pitches ≤ Scale Pitches, and if the first pitch is a chord pitch. The last three conditions instead deduct points: −1 for each unresolved non-scale pitch within the measure, $(-x-7) : x > 7$, and $-13 : x = 13$, where x_i is the semitone distance of each interval in the measure. The maximum score obtainable by this objective per measure is thus 6. In the special case that no pitches are present in a measure, a score of 0 is assigned. Hence empty measures could initially score higher than (penalized) random measures; however, empty measures will be penalized by their pitch frequency in the Melodic Global Objective below. The total score of a phenotype is the sum of the scores assigned to each measure.

The **Melodic Global Objective** is based on the high-level melodic features found in Towsey *et al.* [12], developed for compositional and analytical purposes of monophonic melodies. The objective analyses an entire melody, and gives a more global evaluation of the melody, based on 18 different statistical features, as normalized values (see Appendix A). Features 1–7 and 9–13 are taken directly from [12]. All of the features from the paper were initially implemented. The pattern features, except pitch and timing repetitions, were discarded through qualitative evaluation. Towsey *et al.* also excluded these features, as they had very large standard deviations. The features that were developed to replace them are features 15–18, and are all based on pattern matching. Other features such as dissonant intervals are covered by the melodic local objective described above. Each feature is scored by proximity, given as $1 - |x - y| : x, y \in [0, 1]$ where x is the feature value of the phenotype and y is a target value provided to the algorithm. The final score assigned to the phenotype by this fitness objective is the sum of all feature scores, resulting in a maximum score of 18.

The representation for the harmonization, often referred to as chords, supports every possible chord that can be created with four pitches. The **Harmonization Objective** attempts to stabilize the chords vertically, based on the melody and the musical key within each measure, by building triads that belong to the key. It also allows the presence of a fourth pitch in the chord, but does not encourage it. The Harmonization Objective is largely based on the simplicity fitness objective of Freitas and Guimarães [16]. The difference in feature values and features in this system is due to the absence of a corresponding dissonance function. The final score for a phenotype is the summed score of eight conditions for each chord (shown in Appendix B), with a maximum achievable score of 0.

The **Harmonic Progression Objective** can be considered as a horizontal, or global, objective for the harmonies generated by the algorithm. While the harmonization objective attempts to establish triads in regards to the melody within a measure, this objective rewards, or punishes, chords based on their relation to chords in the other measures of the phenotype. It establishes harmonic variety by punishing excessive repetitions of any chord, as well as enforcing a global variety of distinct chord roots (the conditions and scores are given in

Appendix C). It is a necessary function, which will greatly influence the music; however, the weights currently assigned can be seen as a starting gambit and are mainly grounded in the authors' intuitions. Some simple rules for how certain chords should be resolved by the following chord are also implemented. The maximum achievable score for this feature is 0.

4 Experiment Design

The algorithm (implemented in Java) was run with maximum population size = 1000 and maximum generation count of 3000 using 17 different configurations to generate a total of 21 artefacts. The various configurations are determined by the set of target values for the Global Fitness Objective, the supplied genetic material for deriving the initial population, whether randomly generated individuals were used for initialization or not, and variations of genotype lengths. These parameter values were found to be fitting during preliminary test runs of the algorithm, in terms of convergence to a set of near optimal solutions. In several cases a single optimal solution was evolved that obtained the maximum possible score in all fitness objectives, a perfect score. The transformation of the phenotypes into something that would be audibly presentable to the human evaluators was done by translating the generated melodies and chord progressions into MIDI. Using Cubase 5, a digital audio workstation, each melody and chord progression was played by a sampled piano at 120 beats per minute. No performance data was added. Regarding the genetic operators, there is a defined probability, in this case 50%, that exactly one crossover operator will be applied, and there will always be exactly on mutation operator applied. The probabilities of selecting a specific crossover operator were 66% for Single Point Crossover vs 34% for Single Measure Crossover, while the mutation probabilities were defined as 15% each for Note Mode, Random Pitch, Pitch Modulation, Chord Change and Chord Pitch; 10% for Chord Swap; and 7.5% for Note Position and Duplication.

With each run of the algorithm, a single produced artefact was chosen for evaluation in the survey. In the case that a single individual was non-dominated, it was selected for the survey. If the algorithm did not converge on a single solution after 3000 generations, but rather a Pareto front with several rank 1 phenotypes in it, the phenotype was selected purely on achieved fitness scores. A consequence of this methodology is that one or more features in the Melodic Global Objective might not be identical to the target value, for two possible reasons. Either the solution converged on a local maxima in the fitness landscape, or a target value for at least one feature in the objective contradicts a rule or condition in one of the other fitness functions.

The 17 different experimental configurations covered two versions of randomly generated individuals used in generating the initial population (0 resp. 10 individuals), three different counts of total measures in the developed phenotypes (8, 12 and 16), and four different target value configurations for the 18 features in the Melodic Global Objective fitness function. Further, the provided data differ in two ways: a single piece of music was, or was not, provided to the

algorithm; and either a melody or a chord progression, and a varying amount of randomly generated phenotypes could be supplied. A total of six different pieces of material was provided: three pieces of simple music was used, and split into melody and harmonization. The music pieces consisted of eight bars of the verse from "She Loves You" by The Beatles, the twelve measure verse of Britney Spears' "...Baby One More Time", and eight measures from the theme of the children's march "Over the Hills and Far Away" by Percy Aldridge Grainger, that are frequently used as a first lesson for piano students. Of the four target value configurations for the global melodic fitness function features, one set of target values was defined by the authors, as it qualitatively produced satisfying phenotypes during development of the fitness functions. The other three configurations were obtained by analysing the melodies of the three music pieces[1].

5 Results

A quantitative online study was performed to evaluate the generated artefacts. The main goal was to evaluate how different configurations, and achieved fitness values, impact how the musical ideas are perceived by a human audience. Participants of the survey were presented one artefact generated by the system at a time and asked to evaluate and score it on three criteria: *pleasantness, interestingness and randomness*. The assignable values in each criterion were integers on a scale from 1 to 5, where 1 was defined as the lowest possible value and 5 the highest. The terms were not explicitly described to the participant, but antonyms were provided. Respectively *unpleasant, boring and structured*. The artefacts themselves were also referred to as musical ideas, as they are not fully produced or arranged pieces of music.

93 participants contributed to the survey, with 35 of them evaluating all 21 artefacts. A total of 884 evaluations were made. The average number of evaluations per artefact was 42. The participants were initially asked to describe their relationship to music (e.g., consumer, musician, hobby musician, composer and producer). Pieces in the survey were all presented in a random order, to eliminate possible biases on the artefacts presented last in the survey due to fatigue. Randomizing the order of presentation also helps getting a uniform distribution of data, if many participants do not complete the survey.

The aggregated average values (and standard deviations) for all artefacts were: Pleasant: 3.256 ± 1.079. Interesting: 3.048 ± 1.140. Random: 2.550 ± 1.152.

Grouping the results from the survey based on their fitness configuration class, pleasantness does not show big variations. All fitness configuration scored slightly above 3 in terms of pleasantness, which is the absolute neutral in terms of the scale used for scoring. In terms of interestingness, the configurations based on "hits" outperformed the other two, which are respectively made up of arbitrary numbers and of a very simple child song theme, which has very few notes, a very small pitch range, and very simple rhythms. The child song also scored the lowest of all configurations in terms of randomness.

[1] For some examples of the produced music, see http://bit.ly/2DzbFzf.

If the results are grouped by their seed material class, some interesting numbers appear. In terms of previous research [15], the genetic material provided for initialization has a profound impact on the output of the algorithm. In this grouping, the class that is provided melodies for generating the initial population does indeed score highest in both pleasantness and interestingness, and the lowest in randomness. This class consists of 6 artefacts, where all of them converged to a single solution. Investigating the class grouping by random seeds showed the lowest amount of differentiation in scores between the configurations. This is interesting as it implies that the algorithm is able to reach similar solutions regardless of noise in the starting material.

11 of the 21 runs of the algorithm converged on a single non-dominated phenotype. Five of these converged to a single fitness perfect solution. All of these artefacts were created by seeding a melody and hence had a perfect score in the global melody objective. Looking at pattern feature correlations, features 15–18 in the global melody objective are the pattern matching features developed for this work. While feature 18, the positional rest measure repetitions feature, shows no sign of correlation, the other three do. Feature 15, 16 and 17, have a statistically significant correlation to the interestingness score at a 99.5% confidence interval. Feature 15 and 17 have a statistically significant correlation to randomness at a 95% confidence interval. Feature 15 and 16, Whole Measure Distinctness and Half Measure Distinctness, respectively, correlate positively with interestingness, which means the more distinct patterns appear, the more interesting the pieces are perceived. It is doubtful that this correlation is of a linear nature, but within the range of values for the fitness features obtained by the artefacts it might be. All the feature scores were below 0.5. Feature 17, positional rhythmic measure repetitions, has a negative correlation to both interestingness and randomness, which is confusing. When feature 17 gets close to its maximum value, it means that a very rigid structure is observed, where the exact rhythmical patterns are repeated at given positions later in the piece.

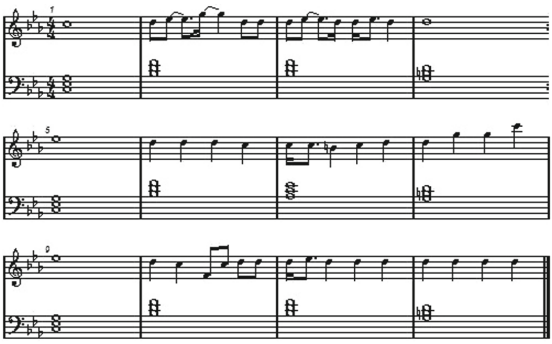

Fig. 2. Example of generated music

The sentiments collected from the optional comments in the survey can provide some interesting opinions. Enjoyment and appreciation of music is a very subjective experience. A self-labeled "Consumer" left a comment on the last question: *"Not that much good music"*, and had consistently given scores below 2 for pleasantness and interestingness. Another participant commented on the piece shown in Fig. 2: *"Some weird melody beats... Some times very standard, but other times suddenly weird."*, and scored it 4 in randomness, but rated it at 5 in interestingness and 4 in pleasing. By looking at the sheet music, you can see that it has some unusual rhythmical phrasings in measure two and three, which are also not repeated later in the piece. This sentiment is somewhat re-iterated by another participant who wrote: *"First 10 seconds sound like an intro, but after that is a bit more expected"*. Pieces that were generated without melodic seeds, sported rhythmical phrases that were critiqued at times. This was, however, not always the case. A comment on another piece stated: *"Very good melody line. Nice rhythm on that."* A possible cause of this is that the features that measure rhythms do not reward good rhythmical phrases, nor punish bad ones.

6 Discussion

In terms of novelty, systems within algorithmic composition that make both harmonization and melody is not unprecedented. Evolutionary algorithms that generate either melodies or harmonization are fairly common, but developing both in tandem is not common. Developing a multiple-objective evolutionary algorithm (MOEA) for co-evolving melody and harmonization provided quite a few obstacles. Simply making the algorithm converge on a set of solutions that were acceptable to humans proved challenging. The state space of possible solutions for genotypes of eight measures is of size 10^{205} in magnitude, and 10^{410} for 16 measures. This is caused by developing both melody and harmonization at the same time, as the state space grows multiplicatively. For eight measures of melody exclusively, the state space is 34 orders of magnitude smaller than when combined with a harmonization. Also the use of $\frac{1}{16}$ notes increases the state space tremendously. The size of the state space could be dramatically reduced by only allowing notes of $\frac{1}{8}$ or bigger. However, supporting such a large state space is necessary to allow a wide range of musical expressions. Allowing the algorithm to create such a huge variety of phenotypes can aid in evaluation of fitness functions on a general basis, in relation to human perception of music. If the algorithm is capable of representing what could be considered as "bad music", it is possible to validate fitness functions and features if they guide the search towards what is considered "good" solutions, or "good music."

Having to apply the step of the algorithm that removes fitness duplicates is something that could be considered a weakness of the implementation, as it confines the algorithm to explore a relatively small part of the state space on any given run. If several equal best solutions in a maxima exist, only one will be kept. Removing genetically identical phenotypes, instead of fitness identical, might be more desirable. It would also allow the algorithm to escape local maxima more

frequently. However, this was not computationally feasible while working with four fitness objectives. Not eliminating fitness duplicates also has the benefit of providing options to anyone who is using the algorithm. Having a set of slightly different solutions, that are considered equally good, could provide any user of the implementation with more ideas, more inspiration. While the algorithm did generate pieces of music that were appreciated by participants of the study, it was not able to do so consistently. The results proved to be slightly more pleasant and interesting when melodic material was provided for initialization, but no evidence was found as to conclude why. An hypothesis is that provided material provides a scaffolding for the algorithm to work with, especially in terms of phrasing. However, the system does not evaluate phrasing explicitly in any of its fitness functions, which leaves room for improvement.

Integrating four fitness functions on a new model spurred quite a few unforeseen interactions between the fitness functions and the model. The most apparent problem with co-evolving melody and harmonization was the lack of harmonic context needed by the individual fitness objectives. Pitches that are outside of the musical key are not punished in the melodic harmonic objective, as long as the same pitch occurs in the chord of the measure. The same is also true for the harmonization objective, where a non-scale pitch in a chord is not punished if it occurs in the melody. The result of this interaction was that both the melody and harmonization disregarded the musical key it was supposed to compose in, and generated very atonal pieces.

7 Conclusion

A novel approach to generating musical material with a multi-objective evolutionary algorithm was implemented, along with a set of 43 fitness measures split into four objectives. The implementation was then used to produce 21 pieces of music with different configurations that were evaluated in a user study. The system is not able to consistently write music that is perceived by everyone to be aesthetically good. The best results were achieved when some melodic material was supplied during the initialization phase. Features that evaluate rhythmical phrasing is lacking in the fitness functions, and the performance of the system could possibly improve by introducing new features that guide the search of the algorithm towards good rhythmical phrasing. While the implementation shows that it is feasible to develop both harmonization and melodies in parallel with an evolutionary algorithm, the benefits of doing so are not prominent besides the ability for both harmonization and melody to adapt to each other during generation. Due to the large search space and relatively high amount of fitness objectives, computational viability was hard to achieve. To do this, only short musical ideas were developed, and a scheme for removing duplicates in the population was implemented, which might not be desirable. Overall the algorithm will probably never write any musical masterpiece, but evaluations made by participants of the study indicate that some of the generated pieces were subjectively enjoyable. This suggests that the implementation could be useful in terms of

computer assisted composition. In the future, efforts could be made to develop fitness features that concern melodic phrases, especially rhythms. Toussaint [35] suggests ways of generating "good" musical rhythms, and Toussaint *et al.* [36] give a variety of features and concepts that could be adopted.

Appendix A: Melodic Global Objective, feature descriptions

1. *Pitch Variety:* The ratio of distinct pitches to notes.
2. *Pitch Range:* The semitone difference between the highest and the lowest pitch in the melody, divided by the maximum range possible in the model (here: 20).
3. *Step Movement:* The number of intervals that are of step distance, divided the total number of intervals.
4. *Non-Scale Pitch Quanta:* The number of notes outside the scale, divided by the total number of notes.
5. *Contour Stability:* The proportion of consecutive intervals that follow in the same direction. Two consecutive intervals incorporate three pitches and if all the three notes (ignoring rests) are of the same pitch, it is counted as moving in the same direction.
6. *Contour Direction:* The sum, measured in semitones, of all rising intervals divided by the (absolute) sum of all intervals.
7. *Pitch Frequency:* The ratio of notes with pitches, to total time-steps.
8. *Rest Frequency:* The ratio of notes that are rests, to total time-steps.
9. *Rest Density:* The proportion of silent time-steps.
10. *Rhythmic Variety:* The degree of 16 distinct note durations (16_{th} to whole note) used.
11. *Syncopation:* The proportion of syncopated notes (i.e., notes with duration $\geq$ one beat, that start off the beat defined by the time-signature).
12. *Repeated Pitches:* The ratio of intervals of 0 semitones (in consecutive pitches), to the total number of intervals.
13. *Repeated Timings:* The proportion of consecutive pitches (i.e., ignoring rests) with the same duration.
14. *On-Beat Pitch Coverage:* The ratio of beats that contain pitch notes.
15. *Distinct Whole Measure Patterns:* The ratio of distinct whole measure rhythmical patterns.
16. *Distinct Half-Measure Patterns:* The ratio of distinct half-measure rhythmical patterns.
17. *Positional Rhythmic Measure Repetitions:* Number of positional repetitions (a measure repeated either directly, two measures later, or four measures later; rests are ignored).
18. *Positional Rest Measure Repetitions:* same as 17, but for rests (pitches are ignored).

In 17 and 18, the last four measures are only checked against the previous four, and not between themselves.

Appendix B: Harmonization Objective, conditions and scores

−50 if the chord root is not a pitch within the key.
−40 if there is no major or minor third in the chord, in relation to the chord root.
−10 if no perfect fifth is present in the chord (a diminished fifth is not punished if it falls naturally within the scale, i.e., if it is a 2nd step root chord in a minor key or 7th step root chord in a major.)
−5 for a unison that is not of the root pitch; a triad unison is punished harder (−10).
−20 if any of the pitches is in a semitone distance to any other note in the chord (this condition is ignored in case the pitch is considered a meaningful seventh).
−10 for dissonant pitches (that do not belong in the triad chord specified by the chord root).
−30 for invalid pitches (pitches not found in the chord, or in the melody within the measure the chord is played; a major third is not considered dissonant, nor invalid, if it appears within a dominant chord in a minor key).
+10 for a meaningful seventh, i.e., one resolved by a step in the following chord.

Appendix C: Harmonic Progression Objective, conditions and scores

−30 if the first chord in is not the tonic chord.
−20 if a phenotype contains no chord with a 5th step root, a dominant.
−20 if the chord following a dominant chord is not a tonic chord.
−10 for an unresolved diminished chord and −20 for an unresolved X7 chord.
−20 if the same chord root appears in three consecutive positions.
If a chord is repeated in the position eight measures after, the phenotype is rewarded by $x - 8 : x > 8$, where x is the amount of measures in the phenotype.
If a phenotype does not have a predefined number of distinct chord roots, it is punished by $|y - x| \cdot -1 : x, y \in [1, 11]$, where x is the number of distinct chord roots in the phenotype and y the target value.

References

1. Hiller, L.A., Isaacson, L.M.: Musical composition with a high-speed digital computer. J. Audio Eng. Soc. **6**(3), 154–160 (1958)
2. Moroni, A., Manzolli, J., Zuben, F.V., Gudwin, R.: Vox Populi: an interactive evolutionary system for algorithmic composition. Leonardo Music J. **10**, 49–54 (2000)
3. Sánchez-Quintana, C., Moreno-Arcas, F., Albarracín-Molina, D., Fernández, J.D., Vico, F.: Melomics: a case-study of AI in Spain. AI Mag. **34**, 99–103 (2013)
4. McCormack, J.: Open problems in evolutionary music and art. In: Rothlauf, F., et al. (eds.) EvoWorkshops 2005. LNCS, vol. 3449, pp. 428–436. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-32003-6_43

5. Galanter, P.: The problem with evolutionary art is. In: Di Chio, C., et al. (eds.) EvoApplications 2010. LNCS, vol. 6025, pp. 321–330. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12242-2_33
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
7. Horner, A., Goldberg, D.E.: Genetic algorithms and computer assisted music composition. In: Proceedings of International Conference on Genetic Algorithms, pp. 337–441 (1991)
8. Ricanek, K., Homaifar, A., Lebby, G.: Genetic algorithm composes music. In: Proceedings of Southeastern Symposium on System Theory, pp. 223–227 (1993)
9. Marques, V.M., Oliveira, V., Vieira, S., Rosa, A.C.: Music composition using genetic evolutionary algorithms. In: Proceedings of IEEE International Conference on Evolutionary Computation, pp. 714–719 (2000)
10. Papadopoulus, G., Wiggins, G.: A genetic algorithm for the generation of jazz melodies. In: Proceedings of Finnish Conference on Artificial Intelligence (STeP) (1998)
11. Johnson, M., Tauritz, D.R., Wilkerson, R.: Evolutionary computation applied to melody generation. In: Proceedings of Artificial Neural Networks in Engineering (ANNIE) Conference (2004)
12. Towsey, M.W., Brown, A.R., Wright, S.K., Diedereich, J.: Towards melodic extension using genetic algorithms. Educ. Technol. Soc. **4**(2), 54–65 (2001)
13. de Freitas, A.R.R., Guimarães, F.G., Barbosa, R.V.: Ideas in automatic evaluation methods for melodies in algorithmic composition. In: Proceedings of 9th Sound and Music Computing Conference, pp. 514–520. Copenhagen, Denmark (2012)
14. Biles, J.A.: Autonomous GenJam: eliminating the fitness bottleneck by eliminating fitness. In: Proceedings of 2001 Genetic and Evolutionary Computation Conference Workshop Program, San Francisco, California (2001)
15. Freitas, A., Guimarães, F.: Originality and diversity in the artificial evolution of melodies. In: Proceedings of 13th annual conference on Genetic and Evolutionary Computation, pp. 419–416 (2011)
16. Freitas, A., Guimarães, F.: Melody harmonization in evolutionary music using multiobjective genetic algorithms. In: Proceedings of Sound and Music Computing Conference (2011)
17. De Prisco, R., Zaccagnino, G., Zaccagnino, R.: EvoBassComposer: a multi-objective genetic algorithm for 4-voice compositions. In: Proceedings of 12th Annual Conference on Genetic and Evolutionary Computation, pp. 817–818. ACM (2010)
18. Jeong, J.H., Ahn, C.W.: Automatic evolutionary music composition based on multi-objective genetic algorithm. In: Handa, H., Ishibuchi, H., Ong, Y.-S., Tan, K.-C. (eds.) Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems - Volume 2. PALO, vol. 2, pp. 105–115. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-13356-0_9
19. Nelson, G.L.: Sonomorphs: an application of genetic algorithms to the growth and development of musical organisms. In: Proceedings of Biennial Symposium on Arts and Technology, pp. 155–169 (1994)
20. Biles, J.A.: GenJam: a genetic algorithm for generating jazz solos. In: Proceedings of International Computer Music Conference (1994)
21. Biles, J.A., Anderson, P., Loggi, L.: Neural network fitness function for a musical IGA. In: Proceedings of International Symposium on Intelligent Industrial Automation and Soft Computing (1996)

22. Hild, H., Feulner, J., Menzel, W.: HARMONET: A neural net for harmonizing chorales in the style of J.S. Bach. In: Advances in Neural Information Processing Systems, pp. 267–274 (1992)
23. Feulner, J., Hörnel, D.: Melonet: neural networks that learn harmony-based melodic variations. In: Proceedings of International Computer Music Conference, pp. 121–121. International Computer Music Association (1994)
24. Bell, C.: Algorithmic music composition using dynamic Markov chains and genetic algorithms. J. Comput. Sci. Coll. **27**(2), 99–107 (2011)
25. Thywissen, K.: Genotator: an environment for exploring the application of evolutionary techniques in computer-assisted composition. Organised Sound **4**(2), 127–133 (1999)
26. Khalifa, Y., Khan, B.K., Begovic, J., Wisdom, A., Wheeler, A.M.: Evolutionary music composer integrating formal grammar. In: Proceedings of 9th Annual Conference Companion on Genetic and Evolutionary Computation, pp. 2519–2526. ACM (2007)
27. Phon-Amnuaisuk, S., Tuson, A., Wiggins, G.: Evolving musical harmonisation. Artificial Neural Nets and Genetic Algorithms, pp. 229–234. Springer, Vienna (1999). https://doi.org/10.1007/978-3-7091-6384-9_39
28. Gibson, P.M., Byrne, J.A.: Neurogen, musical composition using genetic algorithms and cooperating neural networks. In: Proceedings of 2nd International Conference on Artificial Neural Networks, pp. 309–313. IET (1991)
29. Chen, C.C.J., Miikkulainen, R.: Creating melodies with evolving recurrent neural networks. In: Neural Networks. In: International Joint Conference on Proceedings, IJCNN 2001, vol. 3, pp. 2241–2246. IEEE (2001)
30. Scirea, M., Togelius, J., Eklund, P., Risi, S.: MetaCompose: a compositional evolutionary music composer. In: Johnson, C., Ciesielski, V., Correia, J., Machado, P. (eds.) EvoMUSART 2016. LNCS, vol. 9596, pp. 202–217. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-31008-4_14
31. Deb, K.: Multi-objective optimization. In: Burke, E., Kendall, G. (eds.) Search Methodologies, pp. 403–449. Springer, Boston (2013). https://doi.org/10.1007/978-1-4614-6940-7_15
32. Aguirre, H.E., Tanaka, K.: Selection, drift, recombination, and mutation in multiobjective evolutionary algorithms on scalable MNK-landscapes. In: Coello Coello, C.A., Hernández Aguirre, A., Zitzler, E. (eds.) EMO 2005. LNCS, vol. 3410, pp. 355–369. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31880-4_25
33. MIDI Manufacturers Association: The complete MIDI 1.0 detailed specification: incorporating all recommended practices. MIDI Manufacturers Association (1996)
34. Wu, C.L., Liu, C.H., Ting, C.K.: A novel genetic algorithm considering measures and phrases for generating melody. In: Proceedings of 2014 IEEE Congress on Evolutionary Computation, pp. 2101–2107. IEEE (2014)
35. Toussaint, G.T.: Generating "good" musical rhythms algorithmically. In: Proceedings of 8th International Conference on Arts and Humanities, pp. 774–791. Honolulu, Hawaii (2010)
36. Toussaint, G.T., Matthews, L., Campbell, M., Brown, N.: Measuring musical rhythm similarity: transformation versus feature-based methods. J. Interdisc. Music Stud. **6**, 23–53 (2012)