Evolutionary music composition system with statistically modeled criteria

Zdzisław Kowalczuk[1], Marek Tatara[1], and Adam Bąk[1]

Department of Robotics and Decision Systems,
Faculty of Electronics, Telecommunications and Informatics
Gdansk University of Technology,
ul. Narutowicza 11/12, 80-233 Gdańsk, Poland,
kova@pg.gda.pl

Abstract. The paper concerns an original evolutionary music composition system. On the basis of available solutions, we have selected a finite set of music features which appear to have a key impact on the quality of composed musical phrases. Evaluation criteria have been divided into rule-based and statistical sub-sets. Elements of the cost function are modeled using a Gaussian distribution defined by the expected value and variance obtained from an analysis of recognized music pieces. An evolutionary algorithm, considering a reference sequence of chords as an input, is created, implemented and tested. The results of a sampling survey (poll) proves that the melodies generated by the system arouse the interest of a listener.

Keywords: evolutionary optimization, evaluation criteria, music features, chords, music composition, recognized music patterns

1 Introduction

Music is an important element of man's everyday live since prehistoric times. Despite a wide variety of developed music pieces, artists continuously amaze their listeners with new concepts. With so many works developed to date, the question arises of how many unique works can be still invented. Composition process can be seen as a form of a strive for the best solution; however, the main problem lies in the appropriate definition of a fitness function. Assuming that a searched space is finite, a computer system can be used to seek for a solution inside it. If the intuition and experience of a composer is translated into an adequate fitness function, there will be a chance to obtain a new and interesting music compositions.

While developing a fitness function one should keep in mind that composing a music work is a complex process, which can be based not only on know-how and experience, but also on invention and creativity. In the recent years many attempts are undertaken to employ a computer to compose music. Review of the state of the art is provided in [1], where among the utilized methods, genetic algorithms, Markov chains, grammatical methods and artificial neural networks are enumerated. In this project, we decided to implement genetic algorithms with two types of fitness function.

As mentioned above, the key to successful composing is a proper definition of the fitness function. Among the existing programs, *GenJam* developed by Biles [2] provides a solution, resulting in melodies which are rated by a supervisor, who helps the system to learn. Liu and Ting [3] define 42 rules founded on music theory and calculate the times that the specific rules occur in selected songs. On this basis, and taking into account the listeners' reviews, the authors develop a final fitness function. Another approach presented in [4] is based on the Zipf-Mandelbrot law, where the assumption is made that pleasant music is characterized by certain values of selected features. An alternative approach, emphasizing the importance of the initial population rather than a properly designed fitness function, is reported in [5].

In this paper we describe a project called Music Composer, developed for two kinds of fitness function: statistical and rule-based. A set of statistical features, which are evaluated during the operation of the algorithm, are normalized as in [6]. For the rule-based fitness function, a part of the features is inspired by [3, 7].

2 Numerical Representation of a Music Score

Due to numerous possible combinations in composing music, the issue of its numerical representation arises. Since we consider the problem of evolutionary composition of melody lines based on a given sequence of chords, the numerical representation of melodies must embrace the possibility of representing different rhythmic values. Moreover, the genetic operations used in genetic algorithms (GA) should be adapted to this representation. In order to fulfill the above conditions, the following assumptions for a generated melody are applied:

- rhythmic values of notes are multiples of a sixteenth note
- time signature is fixed and set to $\frac{4}{4}$
- tempo is set by the user
- resulting music line is monophonic.

Due to the constant rhythmic values of particular notes, these assumptions facilitate implementation. Moreover, there is no need to check and verify the sum of rhythmic values inside each bar, because the GA itself does not affect the overall length of a chromosome; thus, the sum of rhythmic values remain unchanged, which is important in performing crossover or mutation.

The structure implemented in our solution is the one described in [3], where a single chromosome consists of a sequence of signed integers, and its length corresponds to the duration of a melody: each bar is represented by 16 values, and each of them corresponds to a rhythmic value of a sixteenth note. The set of possible allele values is shown in Table 1.

An exemplary genotype is presented as a sequence of integers in Table 2 and as a corresponding music score in Fig. 1.

Table 1. Projection of music values on a genotype

Numerical value	Meaning
<0,127>	MIDI note value (from C-1 to G9)
-1	rest (pause)
-2	prolongation of a preceding note or rest

Fig. 1. Music score corresponding to the exemplary genotype

2.1 Description of Genetic Algorithm Implementation

Two types of genetic operators are implemented: single point crossover and mutation with musical meaning. The former one is simple mechanism of exchanging equally long parts from the same locus between two chromosomes. The latter one consists of six different types of mutations:

- **interval mutation** - changes the interval between two consecutive notes to another within one octave
- **a single note transposition** - changes the pitch of a selected note by a random interval within one octave
- **prolongation of a note** - changes the rhythmic length of two selected notes or rests (pauses)
- **mutation of rests to notes ratio** - changes a selected note to a rest with the same rhythmic value, or, if a rest is chosen, it is changed to a note with the same rhythmic value
- **rhythmic mutation** - adds a random number of the value "-2" at a selected locus of the chromosome
- **mutation of long-to-short notes ratio** - prolongs a note if classified as short, and divides into two notes if classified as long (a short note means a rhythmic value lower than a quarter note, and a long note means a rhythmic value greater than a quarter note).

The effects of the described mutations are numerically presented in Table 3 and in Fig. 2 as a corresponding music score. The first bar (and the corresponding first row of the table) contains an original track on which mutations are performed. Next six bars (and the rows of the table) present the effects of the introduced mutation operators.

Note that the last three mutations have been added after an analysis of the obtained results. Mutation of the rests-to-notes ratio has been implemented because the resulting melodies consisted only of pitches (rests were not present); thus causing this mutation have resulted in injection of rests to some output melodies. Mutations of the rhythmic and long-to-short notes ratio have been

Table 2. Exemplary genotype composed of three different pitches and a rest

Note	D5	C5	Rest		F5				C5							
Value	74	72	-1	-2	77	-2	-2	-2	74	-2	-2	-2	-2	-2	-2	-2

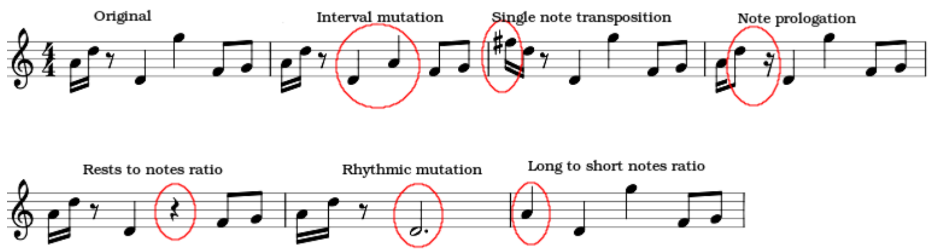

Fig. 2. Music score with the effects of mutation performed on an exemplary melody

Table 3. Influence of the implemented mutations on an exemplary genotype

Original	69	74	-1	-2	62	-2	-2	-2	79	-2	-2	-2	65	-2	67	-2
Interval mutation	69	74	-1	-2	62	-2	-2	-2	69	-2	-2	-2	65	-2	67	-2
Single note transposition	78	74	-1	-2	62	-2	-2	-2	79	-2	-2	-2	65	-2	67	-2
Single note prolongation	69	74	-2	-1	62	-2	-2	-2	79	-2	-2	-2	65	-2	67	-2
Rests to notes ratio	69	74	-1	-2	62	-2	-2	-2	-1	-2	-2	-2	65	-2	67	-2
Rhythmic mutation	69	74	-1	-2	62	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2	-2
Long to short notes ratio	69	-2	-2	-2	62	-2	-2	-2	79	-2	-2	-2	65	-2	67	-2

introduced because the resulting melodies were constructed of sixteenth notes only; thus the introduction of such a rhythmic variety appeared to be necessary.

The binary tournament selection is implemented, where each parent is chosen as a fitter one from a pair of randomly picked individuals. Table 4 shows the parameterization of GA used in the conducted experiments. The newly generated population completely replaces the previous population at the end of each epoch (no elitism). If the number of chords is lower than the number of bars, the chord sequence is automatically repeated to fit the number of bars.

Table 4. Settings of the genetic algorithm

Population count	512
Number of bars	12
Mutation probability	0.2
Crossover probability	0.8
Epochs count	500
Scale	G major
Chords progression	G C D C

3 Fitness Function

Many authors take into account a wide range of quality criteria for automatic process of composing music, there is a need for pre-selection of them.

Two approaches to the construction of matching (fitness) functions, based on rules and statistics, are applied here. The former one utilizes the fundamentals of music composition theory, where the technical correctness of a melody is usually measured by checking the level of deviation from predefined musical rules. It requires knowledge in the field of music theory. In the statistical approach, the probability theory is used to describe dependencies between notes in a melody. Frequently, there is an assumption made that a musically pleasant melody is characterized by certain values of selected statistical features [4]. In this approach no theoretical knowledge is required. However, to accurately tune the fitness function, an analysis of recognized compositions, taking into account particular statistics, can be helpful. In our solution, both approaches have been implemented and compared.

Since the process is based on a composer's creativity, the degree of compliance to a specific feature describing a musical work may differ from one musical piece to another. Thus, we propose to model the level of compliance with each feature r_i, being an element of the fitness function, as a Gaussian distribution:

$$f_{r_i}(x) = w_i \cdot \exp\left(\frac{-(r_i(x) - \mu_i)^2}{2\sigma_i^2}\right) \tag{1}$$

where x is the analyzed music composition, w_i is weight of the feature r_i, σ_i is standard deviation of the feature, μ_i is the mean value of this feature, and $r_i(x)$ is a measure of the feature r_i for composition x. Note that (1) takes values from the range $< 0, w_i >$. In order to create a final cost function, the distributions for all features must be summed:

$$f(x) = \sum_{i=1}^{N} w_i \cdot \exp\left(\frac{-(r_i(x) - \mu_i)^2}{2\sigma_i^2}\right) \tag{2}$$

Note that (2) consists of only statistical features and therefore it is implemented only in the statistic approach. The rule-based fitness function is modeled as a weighted sum of elements, each corresponding to the number of occurrences of a specific feature.

The features included in the fitness function take values between 0 and 1, and the weights w_i are (for now) equal to 1 for all features. Musical meanings of numerical values for particular features are described in the further part of this paper, distinguishing the rule-based and statistical features.

3.1 Rule-based Features

Considering a set of features within the rule-based approach, the inspiration was partially taken from [3, 7]. Musical rules regarding the presence of specific intervals desired in the outcome melody, and regarding the relation between pitch and chord, and between pitch and scale or one-line octave, are adapted. Moreover, in order to maintain the possibility for producing longer notes, rules rewarding the presence of particular notes (in the sense of their rhythmic value) are also introduced. Similarly, to maintain rests, another rule is included. The rules are listed in Table 5.

Table 5. Selected rule-based features, taken into account in the fitness function

#	**Feature**
1.	Perfect consonances
2.	Imperfect consonances
3.	Dissonances
4.	Minor and major seconds
5.	Intervals smaller than octave
6.	Whole note rhythmic values
7.	Half note rhythmic values
8.	Quarter note rhythmic values
9.	Eighth note rhythmic values
10.	Sixteenth note rhythmic values
11.	Sum of rhythmic values in one-line octave
12.	Sum of rhythmic values in scale
13.	Sum of rhythmic values in current chord
14.	Sum of rests' rhythmic values
15.	Pitches in strong beat

3.2 Statistical Features

The second selected fitness function is based on statistical measures of describing important notes relationships in the composed or produced melody. Again, a set of features in this cost function has been pre-selected and modified on the basis of literature review, as shown in Table 10.

3.3 Tuning of the Fitness Function

Since the score for each statistical feature in GA is modeled using a Gaussian distribution, the mean value and standard deviation for each of them is required to obtain a complete description of the fitness function. These values can be tuned through a time-consuming observation of input-output pairs (where by an input we mean the set of mean values and standard deviations, and by an output - composed melodies), when iteratively changing the parameters. Such approach requires, however, an expertise from a GA designer in the field of music theory, in order to see whether the melodies are technically correct and pleasant for the listener, or not. Another approach is based on the analysis of recognized music works, where the mean value and standard deviation are calculated in a statistic way. The melodies taken from the following works have been analyzed:

- "Swan theme from Swan Lake" by P.I. Tchaikovsky
- "Canon in D" by J. Pachelbel
- "Prelude from Cello Suite No. 1" by J.S. Bach
- "Allegro from Eine Kleine Nachtmusik" by W.A. Mozart
- "Entr'acte from Carmen" by G. Bizet
- "Danse des petits cygnes from Swan Lake" by P.I. Tchaikovsky
- "Air on the G String from Orchestral Suite No. 3" by J.S. Bach
- "Can Can" by J. Offenbach
- "Humoresque" by A. Dvorak
- "Entertainer" by S. Joplin
- "Ave Maria" by F. Schubert
- "Arioso from Ich steh mit einem Fuß im Grabe" by J.S. Bach.

The parameters resulting from the analysis of these music works, as well as the ones obtained through subjective tuning are gathered in Table 6.

Table 6. Expected values and standard deviation for the set of selected normalized statistical features ('-' indicates a 'no-result' case, ignored in fitness functions

	Music work analysis		**Subjective tuning**	
Feature	**Expected value**	**Standard deviation**	**Expected value**	**Standard deviation**
Mean pitch	0.564	0.065	0.5	0.3
Pitch deviation	0.053	0.013	0.1	0.2
Off-scale pitches	-	-	0	0.3
Chord pitches	-	-	0.5	0.3
Dissonances	0.029	0.045	0.1	0.3
Minor and major seconds	0.553	0.078	0.5	0.3
Intervals larger than octave	0.007	0.010	0	0.3
Mean rhythmic value	0.282	0.120	0.4	0.3
Rhythm deviation	0.156	0.058	0.2	0.3
Strong beat	0.788	0.218	0.9	0.3
Rests to notes ratio	0.105	0.092	-	-

The rule-based features have been tuned via properly adjusted weights based on a subjective analysis of the input-output pairs, and the resulted weights are listed in Table 7. Note that shorter notes receive a proportionally lower reward for each occurrence due to the fact that shorter notes in a bar results in a larger number of intervals, each rewarded accordingly. Such a tuning procedure has resulted in the presence of each type of notes in the output melodies. Similar relationship holds between the rules concerning the type of intervals (perfect consonances, imperfect consonances, dissonances) and the ones concerning the intervals being in a scale or in a current chord; which (in some cases) has led to rewarding the same interval by two rules.

Table 7. Weights of rule-based features, obtained from subjective tuning

Feature	**Weight**
Perfect consonances	1
Imperfect consonances	3
Dissonances	3
Minor and major seconds	4
Intervals smaller than octave	2
Whole note rhythmic values	80
Half note rhythmic values	40
Quarter note rhythmic values	20
Eighth note rhythmic values	5
Sixteenth note rhythmic values	0
Sum of rhythmic values in one-line octave	3
Sum of rhythmic values in scale	1
Sum of rhythmic values in current chord	1
Sum of rests' rhythmic values	5
Pitches in strong beat	5

4 Evaluation of the System

For the final evaluation, 9 melodies, generated with different settings, were selected. There were two criteria: fitness function and configuration. Among the fitness functions we distinguished the rule-based, statistic and those based on various music work analysis. The three configurations are presented in Table 8.

Table 8. Musical configurations used to create melodies for the poll

#	**Name**	**Tempo**	**Scale**	**Chord progression**	**Bars count**
1.	Slow	64	G major	G C D C	4
2.	Moderate	96	a minor	a d a e	6
3.	Fast	128	E major	E A H E	8

As a result, combining the configurations using the three fitness functions, 9 melodies were obtained.

The poll was conducted by 39 persons via the Internet. Participants, after listening to one of the melodies, were asked to rate it in the scale from 1 (unpleasant) to 5 (pleasant), and to try to determine its 3D psychological estimation (in terms of valence, arousal and dominance, in short VAD) from a certain Self-Assessment Manikin model [8]. Moreover, every participant had the possibility to comment each composition. The order of displaying different compositions to each participant, was fixed randomly. The answers to the question concerning the pleasantness of the analyzed melodies are presented in Fig. 3.

Melody 9 have obtained mainly positive comments, and thus has achieved the best mean rating (equal to 3.28). The music score for this melody is presented

Table 9. Numeration of the melodies rated by poll respondents

Track number	Fitness function variant	Configuration
1.	Statistical based on recognized works	Slow
2.	Subjective statistical	
3.	Subjective rule-based	
4.	Statistical based on recognized works	Moderate
5.	Subjective statistical	
6.	Subjective rule-based	
7.	Statistical based on recognized works	Fast
8.	Subjective statistical	
9.	Subjective rule-based	

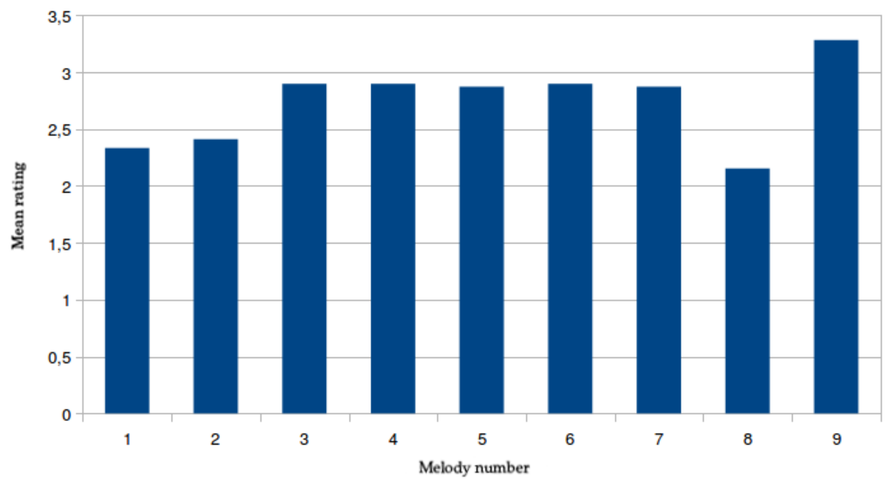

Fig. 3. Mean rating obtained from poll for the selected melodies

in Fig. 4. Similarly, for the melody with a lowest rating the score is presented in Fig. 5, for which the respondents pleaded the lack of any melody outline. Such comments pointed out important issue, which should be taken into account while developing fitness function - the concept of musical phrases, which are short musical fragments, which are repeated throughout the musical work with some variations. They can be encountered in the recognized music works.

The attempt to measure emotions invoked by the melodies turned out to be beyond the scope of this work. Respondents found it difficult to determine proper parameters of his/her VAD emotional state, which resulted in high variances in a VAD plot. Thus, the results seemed to be more of a random guess than a truly emotionally-based response. Certainly, such test can be suitable for more complex polyphonic compositions.

Fig. 4. Highest rated melody generated using Music Composer

Fig. 5. Lowest rated melody generated using Music Composer

5 Conclusions

In this paper we have described an innovative program for composing melodies using genetic algorithms. The developed solution implements a crossover operator and six musically pertinent mutations. The user chooses between rule-based and statistical fitness functions, and tune their parameters. Three sets of tuned parameters are possible to import: subjective statistics, subjective rule-based settings, and 'objective' statistics based on selected music works. For the purpose of program evaluation, nine melodies have been picked out and rated in a poll, in which respondents rated these melodies and marked their emotions induced by the melodies, taking into account an emotional VAD model (valence, arousal and dominance).

The rating of the melodies varied from 2.15 to 3.28, which indicates that they may arouse the interest of a listener. In case of some melodies, respondents drew attention to the fact that there was a lack of musical connection between consecutive parts. Indeed, the melodies were evaluated in the GA as a whole by considering the applied principles of statistics, bypassing the important relationship between successive bars.

It is thus clear that the absence of such factors in the fitness function, results in omitting the concept of musical phrases - i.e. music parts repeated in original or modified form in subsequent bars or groups of bars. Therefore, inclusion of a feature considering musical phrases appears to be a milestone, necessary in further development of this project.

Difficulty in evaluating the listener VAD emotion can be caused by the fact that the program is currently focused on composing monophonic melody lines, disregarding various accents or possible harmonies. Moreover, the choice of an adopted instrument, which plays the melody, is also important, and may influence the emotion.

Further development of this project will focus on the introduction of features that will bring musical phrases into the tunes produced by the program. We plan to make the program for the production of sounds (pitches) occurring simultaneously. We also want to go beyond the monophonic texture (which will require new methods for evaluation of the features). Similarly, addition of accents can contribute to a higher quality of the outcome melodies.

References

1. Fernández J. D., Vico F.: AI methods in algorithmic composition: A comprehensive survey. Journal of Artificial Intelligence Research 48, 513–582 (2013).
2. Biles J.: GenJam: A genetic algorithm for generating jazz solos. ICMC Proceedings, 131–137 (1994).
3. Liu C., Ting C.: Evolutionary composition using music theory and charts. IEEE Symposium on Computational Intelligence for Creativity and Affective Computing (CICAC), 63–70 (2013).
4. Manaris B., Vaughan D., Wagner C., Romero J., Davis R. B.: Evolutionary music and the Zipf-Mandelbrot Law: Developing fitness functions for pleasant music. Applications of Evolutionary Computing, 522–534, Springer, Berlin-Heidelberg (2003).
5. Waschka R.: Avoiding the fitness "bottleneck": Using genetic algorithms to compose orchestral music. ICMC Proceedings, 201–203 (1999).
6. Towsey M., Brown A., Wright S., Diederich J.: Towards melodic extension using genetic algorithms. Educational Technology & Society 4(2) 54–65 (2001).
7. Wiggins G., Papadopoulos G., Phon-Amnuaisuk S., Tuson A.: Evolutionary methods for musical composition. International Journal of Computing Anticipatory Systems (1999).
8. Bradley M. M., Lang P. J. Measuring emotion: the Self-Assessment Manikin and the Semantic Differential, *Journal of behavior therapy and experimental psychiatry*, 25(1) pp. 49-59, 1994.

Table 10. Description of implemented statistical features and meaning of its values

Average pitch: $\frac{\text{Mean pitch MIDI value}}{\text{The highest possible difference in MIDI values (127)}}$		
Limiting values	0	Only lowest MIDI pitches are present
	1	Only highest MIDI pitches are present
Pitch deviation: $\frac{\text{Standard deviation of pitches' MIDI values}}{\text{The highest possible standard deviation}}$		
Limiting values	0	All notes have the same pitch
	1	There are highest and lowest possible notes
Off-scale pitches: $\frac{\text{Number of pitches from a scale}}{\text{Number of pitches out of a scale}}$		
Limiting values	0	All notes are from a given scale
	1	All notes are out of a given scale
Comments		Rests are ignored
Chord pitches: $\frac{\text{Sum of rhythmic values of pitches from a given chord}}{\text{Sum of all rhythmic values in a melody}}$		
Limiting values	0	All pitches are out of a given chord
	1	All pitches are from a given chord
Dissonances: $\frac{\text{Number of dissonances}}{\text{Number of all intervals}}$		
Limiting values	0	No dissonance is present
	1	Every interval is dissonance
Comments		Rests between notes and single notes are ignored. The following intervals are treated as a dissonance: tritone, major seventh, minor seventh, and all intetrvals larger than one octave.
Minor and major seconds: $\frac{\text{Number of second interval occurrence}}{\text{Number of all intervals}}$		
Limiting values	0	There is no second interval
	1	Each interval is either major or minor second
Comments		Rests and single notes are ignores; intervals are examined ignoring rests between two notes
Intervals larger than octave: $\frac{\text{Number of intervals larger than octave}}{\text{Number of all intervals}}$		
Limiting values	0	There are no intervals larger than octave
	1	Every interval is larger than octave
Comments		Rests and single notes are ignores; intervals are examined ignoring rests between two notes
Mean rhythmic value: $\frac{\text{Mean value of logarithm of (rhythmic value +1)}}{\text{Logarithm of (whole note rhythmic value + 1)}}$		
Limiting values	0	Impossible, but value close to 0 means that melody is constructed only from sixteenth note
	1	Melody is constructed only from whole notes
Comments		Whole note has rhythmic value 4, sixteenth note 0.25
Rhythm deviation: $\frac{\text{Standard deviation of logarithm of (rhythmic value +1)}}{\text{Logarithm of (whole note rhythmic value + 1)}}$		
Limiting values	0	Every rhythmic value is the same
	1	Melody has the longest and the shortest possible notes
Comments		
Rests to notes ratio: $\frac{\text{Sum of rests rhythmical values}}{\text{Sum of all rhythmic values}}$		
Limiting values	0	There are no rests in a melody
	1	Melody is made only of rests