A cooperative approach to style-oriented music composition

D. Espí, P. J. Ponce de León, C. Pérez-Sancho, D. Rizo, J. M. Iñesta, F. Moreno-Seco, and A. Pertusa

Departamento de Lenguajes y Sistemas Informáticos
University of Alicante, Spain
inesta@dlsi.ua.es

Abstract. Evolutionary methods have been largely used in algorithmic music composition due to their ability to explore an immense space of possibilities. The main problem of genetic related composition algorithms has always been the implementation of the selection process. In this work, a pattern recognition-based system helped by a number of music analysis rules is designed for that task. The fitness value provided by this kind of supervisor (the music "critic") models the affect for a certain music genre after a training phase. The early stages of this work have been encouraging since they have responded to the a priori expectations and more work has to be carried out in the future to explore the creative capabilities of the proposed system.

1 Introduction

There are three main types of algorithmic composition approaches; those that follow prespecified rules to generate new melodies, those that learn a set of patterns from a collection of melodies, and the systems that use evolution to produce output more perfectly matched to some aesthetic criteria [1].

Compositions from rule-based systems are unlikely to be surprising, and connectionist music composition systems like neural networks can come up with new compositions, but they will not be particularly novel in an interesting way [1]. Genetic algorithms have a valuable property: they have the capability to explore an immense space of possibilities and, through the mutation operator, they are able to model something hardly computable like "inspiration" (something very vague that can be viewed as "finding something of artistic value by chance").

Evolutionary methods have been largely used in algorithmic composition. Genetic algorithms are a technique first proposed by Holland [2], and it is based upon the evolution by natural selection proposed by Darwin. Its main constituent parts are a representation for chromosomes (the candidate solutions), an initial population of chromosomes, a set of operators to generate new candidates, an evaluation function and a selection method. Some genetic algorithms applied to automated composition have been discussed by Wiggings in [3].

Anyway, the results of automatic composition are subjective. There is not a definitive evaluation system; even the impressions of some persons listening to

the same music can be different. This is why the one of the main problems of genetic related composition algorithms has always been the implementation of the selection process (the music "critic") .

The aid of human skills for that permits to model the musician's taste and even web-based human assessment of the individuals have been described in the literature [4–6], but this appealing scenario is not practical in the long term, when hundreds or thousands of melodies have to be evaluated and rated. When human critics are used, these evolutionary systems can produce pleasing and sometimes surprising music, but usually after many tiresome generations of feedback [7].

Some works [3] use music-theoretical knowledge as fitness function, like Morioni et al [8], who use a fitness criteria that takes into account melodic fitness, harmonic fitness and voice range fitness. Other previous works try to induce musical structure from a corpus to get new melodies [9]. An extension of this procedure was done by Spector and Alpern [10], who applied a hybrid rule-based and neural network critic trained with a corpus of well-known works, and Baluja et al. [11] who also used a similar combination of neural and genetic techniques. Neural techniques have been also used by Machado et al .[12] to identify the author of a given piece and using the results for creating critics.

If we want a connectionist algorithm to learn about the goodness of a given melody, a corpus must be tagged (scored) before. This is usually done by a human, so a subjective factor is introduced. In the present work, a combination of genre classifiers help to assess how probable is that a given generated melody will belong to a certain genre. In order to train the classifiers, each melody is tagged with its corresponding genre, which is an easier and less subjective task than rating their artistic value.

The fitness value provided by pattern recognition-based supervisor models the affect for a certain music genre after a training phase and it is also helped by musicology rules. This way, a style-guided composition scheme is achieved in which "style" can mean music genre, a given author, or even particular tastes, depending on the training examples[1]. Thus, the same overall scheme can be used to generate very different melodies just changing the training data.

The individuals are melodies represented using a tree structure, and mutation and crossover techniques are defined on this data structure.

2 Methodology

The proposed composition system is a hybrid structure composed basically of a sequence generator implemented by a genetic algorithm and an automatic supervisor in charge of providing the fitness function, implemented through a pattern recognition-based system that is also assisted by a number of basic music analysis rules. The overall scheme of the setup is displayed in fig. 1.

[1] Music style is a vague concept that can be interpreted in a number of ways, all of them subjective. Some authors consider the music style as the set of musics that provide a given emotion (aggressive, calm, romantic, etc.) while others focus on music genre as the set of authors or pieces sharing some common characteristics.

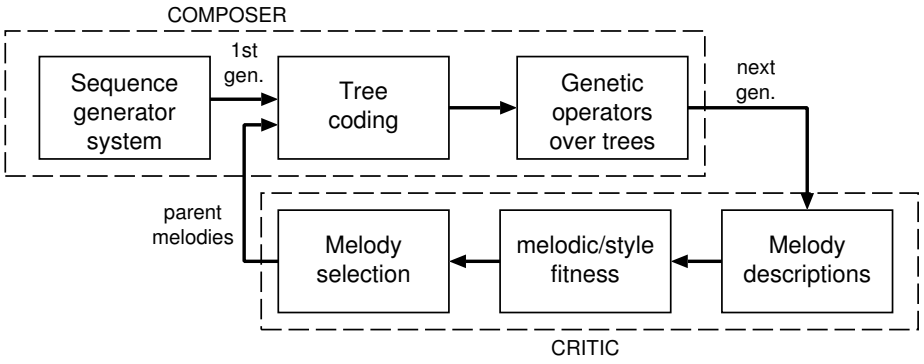

Fig. 1. Overall scheme of the system architecture.

The individuals are melodies of fixed duration. We define melody as a monophonic sequence of notes. The length of a music piece is usually expressed in *measures*. The measure is the basic unity of music structure and it can be defined as a group of beats, caused by a regular recurrence of accented beats. We will work in the context of the 4/4 meter, in which a measure lasts 4 beats. For the structural reasons described below it is interesting to establish the duration as a power of two, so we have considered 16 measures ($16 \times 4 = 64$ beats). Each individual will be represented as a melody tree [13].

The first generation of melodies will be provided by a sequence generator, based on stochastic methods and probabilities inferred from real musical data. Then, the melodies will evolve applying the genetic operators to the trees, with some constraints related to maintain both the melody length and their viability.

A feature extractor will analyze the melodies (the trees) in order to compute suitable descriptors that will be the input of the evaluation system. The descriptors are statistical features based both on the global distribution of notes, silences, intervals, etc. (shallow statistical descriptors) [14], and on the frequencies of appearance of note subsequences (a sort of n-grams) [15].

Once the different individuals have been analyzed, the feature vectors are input into the evaluation system in order to compute their fitness. This fitness is composed of three parts. Two of them are related to the melody "style" and the other evaluates the correctness of the music structure. After fitness computation, the best individuals get into the next generation step, as usual.

2.1 Melody representation structure

The genotype of each melody is defined as a tree. Tree structures have been proposed [13, 16] as an alternative to string representations for melodies.

A melody has two main dimensions: time and pitch. Basically, the first is determined by note onset times and durations, and the second by the fundamental frequencies of the notes. Usually (see [17] for example), strings have been used

in order to code a monophonic sequence of notes. For representing a melody as a string, symbols from a pitch description alphabet, Σ_p, and from a duration alphabet, Σ_d, are combined in $s \in \Sigma^*$, $s = \sigma_1\sigma_2...\sigma_{|s|}$.

Different kinds of properties can be used for the symbols to represent the sequence of pitches and durations in a melody. They can be either *absolute*, if the property depends only on the represented note (like absolute pitch: C, C#, D, ... or duration figure, like whole, half, quarter, etc.) or *relative*, if it is defined in terms of relations to other notes (like pitch intervals or duration ratios).

In string representations, note durations are coded with explicit symbols, so the use of different codings can lead to very different results. The main advantage of using trees is that they are able to implicitly represent this dimension, making use of the proportional nature of the different duration figures, in the sense that note durations are multiples of basic time units, mainly in a binary (sometimes ternary) structure: a *whole* note lasts twice a *half* note, whose length is the double of a *quarter* note, etc. (see Fig. 2). This way, trees are less sensitive to the used codes, since there are less degrees of freedom for coding.

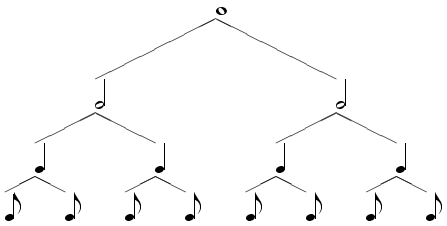

Fig. 2. Duration hierarchy for different note figures. From top to bottom: whole (4 beats), half (2 beats), quarter (1 beat), and eighth (1/2 beat) notes.

Each measure is initially represented by a subtree. Each note or rest will be a leaf node. The left to right ordering of the leaves keeps the timing of the notes in the melody. Each leaf level determines the duration of the note it represents, as displayed in Fig. 2: the root (level 1) represents the duration of the whole measure (a *whole* note), both nodes at level 2 represent the duration of a *half* note. In general, nodes at level i represent a $1/2^{i-1}$ of a measure.

The leaf nodes contain a label value coding any pitch representation symbol, $\sigma_i \in \Sigma_p$. For our purposes, relative codes are preferable because this way, the same melodic line can be represented invariant to transpositions (displacements in pitch of the note sequence). The problem is that relative codings are dangerous when using for composition, because they can lead the melody out of the useful range of pitches. A representation relative to the tonic note has been used instead. This provides the advantages of relative codings without their problems. Once given the tonality, the whole melody is pitched.

An example of this scheme is presented in Fig. 3 for a melody in the tonality of C major. In the resulting tree, the left child of the root has been splitted into two subtrees to reach the level 3, that corresponds to the first note (a quarter

note, duration of a $1/2^2$ of the measure, pitch 'a' that is 1 semitone below the tonic). In order to represent the durations (both are 1/8 of the measure) of the rest and note G, a new subtree is needed for the right child in level 3, providing two new leaves for representing the rest ('-' is the code for rests) and the note ('G', 7 semitones over the tonic). The half note (C, is the tonic) onsets at the third beat of the measure, so it is represented in level 2, according to its duration.

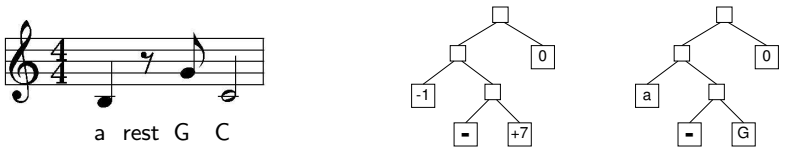

Fig. 3. Simple example of tree construction using intervals for the pitches in the leaves. The central tree displays the interval values relative to the tonic and the right one, the actual character representation of those values. If a 'C_3' is given as tonic, an absolute reference is obtained for the melody.

One note can last more than its usual duration (when it is tied to others or dotted) or can be syncopated (it does not begin in its natural position, according to its duration and the music meter). In those cases, more than one leaf is needed for representing it. Only the first part of the note is coded by its pitch while the other parts of the note are represented by leaves labeled as 'continuation'.

The internal representation of the tree in the chromosome is a parenthesized character string corresponding to its pre-order representation. In order to avoid two characters representing intervals (for the '+' and '−' signs), the interval values have been mapped into ASCII printable characters. If the interval is positive, uppercase letters were used (A for 1 semitone, B for 2, etc.), while negative intervals were mapped into the corresponding lowercases. Since only 26 letters are available, only 24 semitone intervals have been considered (2 octaves), keeping the characters "y" and "z" (upper and lowercase) for coding intervals larger than 2 and 3 octaves, respectively. If the interval is zero, a "0" is utilized. Silences are coded by "-" and continuations by "!".

This way, the tree in Fig. 3 will be represented by the chromosome `((a((-G)))0)` Note that this tree represents a single measure, while the whole melody is composed of 16 measures. For it, the adjacent measures are grouped together in pairs in the immediate upper level of the tree, and then adjacent pairs are grouped again, an so on until reaching the root of the tree, where the 16 measures are grouped. See Fig. 4 for an illustration of this structure.

Thus, two different kinds of internal nodes appear in the melody tree:

Measure nodes: Those in the upper 5 levels (being level 1 the root) of the tree, representing one or more measures.

Note nodes: Those in levels lower or equal than 6. The duration of the music fragment represented by them is always shorter than a measure.

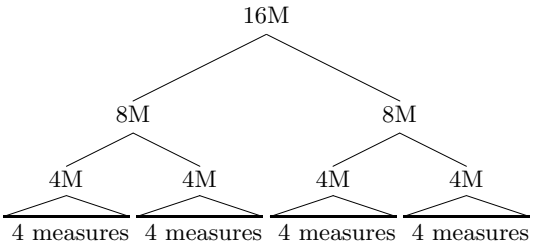

Fig. 4. Structure of a tree coding the 16 measures of a whole melody.

The root node represents 16 measures. The nodes in the second level represent 8 measures each, those in the third level, 4 measures each. Two measures is the duration represented by the nodes in the fourth level and the individual measures are represented by the nodes in the fifth level. The rest of levels represent the internal structure of each measure.

2.2 Genetic operators applied

The usual genetic parameters have been utilized (mutation, crossover, population size, number of generations, etc.), but some of them are specific of our representation scheme, as explained next.

Melody population initialization. The initial population of melodies is generated using a simple sequence generation method. Since we are interested in style-oriented composition, we have used an algorithm that generates sequences in which the different intervals are conditioned by the probabilities of having them according to their appearance frequencies in a training set, composed of real melodies from a given style.

From the melodies in the training set, the probability of an interval σ_i is calculated as $P(\sigma_i) = \frac{N(\sigma_i)}{\sum_j N(\sigma_j)}$. Where $N(.)$ is the number of times that an interval appears in the set.

Evolution. For the evolution of the population, an adaptive approach has been utilized. The evolution is divided into four parts. In the first one, the individuals are randomly selected for crossover, with a probability P_c of being selected. In the second part, a roulette weighted with the fitness function for each individual is utilized. The same scheme is used again for the third part, but using a non linear distribution of the weights in order to give more chances to the best individuals. In the last part, only the best fitted individuals are selected for crossover.

Crossover function Two classes of crossover can happen depending on the tree node selected for interchange the genetic material:

Nodes at or above level 5: whole measures are interchanged. This forces the other individual to select a node at the same level in order to interchange the same amount of measures, thus maintaining their melody length.

Nodes at or below level 6: part of a measure is interchanged. In this case, the interchange can take place at different levels, thus changing the relative length of the notes, but always inside a measure (except if the first node is a continuation event, that extends the duration of the last note or rest of the previous measure; always a rest if it is the first node in the new individual).

Mutation Mutation is applied both to leaf and internal nodes with a probability P_m. This value is adjusted at each generation inversely proportional to the variance of the fitness function for the population. This way, diversity is ensured. Three kinds of equally probable mutations are utilized:

1. **Interval:** the label of a leaf node is changed randomly, without using any information of the current label. It can be changed by any other label from the pitch alphabet, $\Sigma_p \cup \{\text{-}, !\}$.
2. **Duration increment:** the selected leaf node is moved one level upwards, deleting its uncle node and his possible offspring. Thus, its duration is doubled taking the place of the next notes.
3. **Duration reduction:** A leaf node is converted into an internal node with two offsprings, each of its half duration. The left one will take the father's label and the right one will be random.

2.3 Melody description models

One of the goals of this work is to study the ability of pattern classifiers to classify algorithmically generated music into a given style. This scheme has been proved to be useful for real music of the most relevant composers in classical and jazz music, using models of melody statistical descriptors [14, 15]. Those schemes are applied here. For that, the evolving melodies have to be described using the statistical features that will be the input to those recognition systems.

The first model is the shallow description scheme [14]. A number of statistical descriptors are computed related to melodic, harmonic and rhythmic properties of a melody like, for example, average note pitch, average note durations, pitch interval range, etc. A total of 28 descriptors are available. See [14] for details.

These descriptors create a cloud of points in $\mathbb{R}^{28}$ representing the different melodies in the training set for a given style. Under the hypothesis that new melodies of that style should be close to those already existing, the distance from the point representing the evolving melody to the cloud is given as a measure of style for it. The selected approach to compute that distance is to select the k nearest neighbors to the test melody using the GNAT (Geometric Near-neighbor Access Tree) algorithm [18], returning the distance d to the centroid of those k points.

The other approach to style modelling is that of n-words events [15]. The former approach is based under a hypothesis that is very weak in terms of the huge

number of different melodies that can provide similar statistics with different note combinations. We could state that proximity to the style cloud provides a necessary condition but not a sufficient condition for a melody to look style-like.

The n-word based models are based on text categorization methods. The technique encodes note sequences as character strings, therefore converting a melody in a text to be categorized. Such a sequence of n consecutive notes is called a n-word. This way, much restrictive constraints are imposed on adjacent notes. All possible n-words are extracted from a melody, except those containing a silence lasting four or more beats. This method generates n-words by encoding pitch intervals and relative duration information. For each n-note window, intervals and duration ratios are obtained, respectively, by the equations:

$$I_i = Pitch_{i+1} - Pitch_i \qquad (i = 1, \ldots, n-1) \tag{1}$$

$$R_i = \frac{Onset_{i+2} - Onset_{i+1}}{Onset_{i+1} - Onset_i} \qquad (i = 1, \ldots, n-1) \tag{2}$$

and each n-word is defined as a string of symbols:

$$[\ I_1\ \ R_1\ \ \ldots\ \ I_{n-2}\ \ R_{n-2}\ \ I_{n-1}\ \ R_{n-1}\ ]$$

where the intervals and duration ratios have been mapped into alphanumeric characters (see [15] for details). In order to compute R_{n-1}, the duration of the last note in the sequence substitutes for $Onset_{i+2} - Onset_{i+1}$ in Eq. 2.

The *distance* to the style will be given by the class-conditional probability of a melody, given by the probability distribution of n-words for each genre, as learned from a training set. Two different distributions were utilized: a Multivariate Bernoulli model, which reflects how likely the words appear in a melody for a given style, and a Multinomial model, which estimates the frequency of presence of the words for the style. For those n-words not appearing in the training set a discount model [19] is applied in order to not return a zero probability.

These measures of style are complemented by a melodic analysis of the individual, as explained below.

2.4 Fitness function

The fitness will be a combination of stylistic aesthetics and melodic correctness, expressed in mathematical form as a linear combination of those factors:

$$F = w_s S + w_n N + w_m M$$

where, S is the style measure using the distance d to the cloud in the shallow descriptor space, normalized through $S = \frac{1}{d+1}$. N is the second stylistic evaluation, as the probability of the melody to belong to the distribution established by the style training set, using the n-words scheme. N is a probability so it is normalized by definition. Finally, M is the result of the melodic analysis of the melody, defined in [0,1] as explained below. The weights are selected holding $w_s + w_n + w_m = 1$. This way, $F \in [0, 1]$. The weights w_i are free parameters of

the system so the influence of the three aspects can be tuned in order to test and modify its performance.

For computing M, a number of features that are described below are analyzed and converted into numerical rates. M is computed as the average of them:

$$M = \frac{\text{NT} + \text{CS} + \text{RP} + \text{JE} + \text{MP} + \text{RT} + \text{JR}}{7} \quad .$$

All those features are normalized, so M is normalized too. The features are:

- (NT) The relevant notes of the melody not in the chord should be tensions[2]. A note will be *relevant* if it holds any of the following conditions:
  1. It is a note of the chord[3].
  2. It is a long note (longer than one beat).
  3. It is a short note followed by a long rest or jump[4].
  4. It is a short note in a strong beat that is followed by a note in a weak beat that is the descending second[5].

  This feature is computed as $\text{NT} = N_{tn}/N_n$, where N_{tn} is the number of relevant notes that are tensions and are not in the chord, and N_n is the number of relevant notes that are not in the chord.
- (CS) Conclusive sense: This aspect of the melody is rated according to the last note of the melody, this way:

$$\text{CS} = \begin{cases} 1 & \text{if it ends with the tonic or root note.} \\ 0.3 & \text{if it ends with a fifth of the key.} \\ 0 & \text{otherwise} \end{cases}$$

- (RP) The best range of pitches is of a tenth interval (16 semitones).

$$\text{RP} = \begin{cases} 1 & \text{if } 16 = \text{RS} \\ 1/|16 - \text{RS}| & \text{otherwise} \\ 0 & \text{if RS} > 32 \end{cases}$$

  where RS is the interval range for the melody in semitones.
- (JE) Ratio of scales[6] ending with jump, $\text{JE} = \frac{E}{S}$, where S is the total number of scales, and E those ending with a jump.
- (MP) Melodic profile. The more abrupt the worse. It will be measured as the ratio of jumps to the total number of intervals: $\text{MP} = 1 - \frac{N_j}{N_i}$, where N_j is the number of jumps and N_i the total number of intervals in the melody.

[2] A note is a tension if it is not in the chord, and is one tone above or below any note of the triad chord, without making a tritone neither with the third nor with the seventh of the chord.

[3] The chord will be considered as the triad (root note, third, and fifth), plus the seventh note of a given tonality. For example: {C,E,G,B♭} for C major.

[4] Defined as an interval larger that a sixth ($> \pm 9$ semitones).

[5] In a 4/4 metrics context, the first and third beats are the strong ones.

[6] Defined as sequences of 4 or more different and adjacent diatonic notes. All of them have to be in the same direction (ascendent or descendent).

– (RT) Ratio of tensions solved into notes of the chord: RT $= \frac{T_f}{N_t}$, where T_f is the number of tensions followed by a note of the chord and N_t the total number of notes.
– (JR) Ratio of jump returns. Jumps that are followed by an interval in the opposite direction are better rated. It is computed as JR $= \frac{J_r}{J_t}$, where J_r is the number of jump returns and J_t if the total number of jumps in the melody.

2.5 Convergence

The fitness function F is normalized. For achieving a value $F = 1$ is needed that all the partial functions, $S = N = M = 1$. For a value $M = 1$ it is necessary that the melody fits all the criteria imposed above. This is difficult, but not impossible. For $S = 1$ the distance to the centroid of the k nearest neighbors in the training set must be 0 (or ε for floating point operations). Considering the high dimensionality of the representation space this is highly improbable. But $N = 1$ is even harder, because (after a re-normalization strategy) it needs that all sequences of n notes in the individual appear in the training set for that style. If multinomial distributions are utilized, then all the sequences of n notes have to appear with the same frequencies than those in the training set.

For all these reasons it is expected that the fitness values F reach in practice a low value, as it is observed in Fig. 5. Due to the unpredictable value that F will reach for each experiment, a convergence criterion based in the number of generations, N_g, has been utilized.

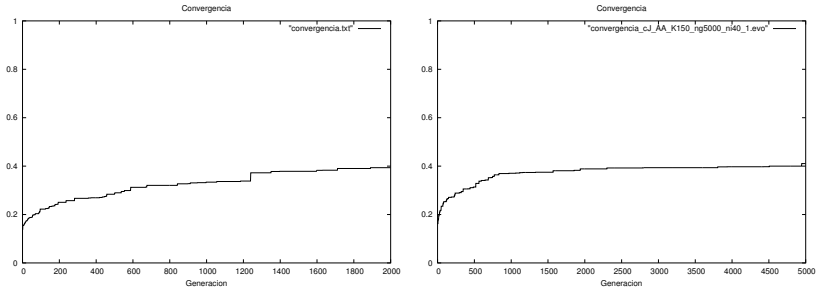

Fig. 5. Convergence for two experiments with (left) 2000 and (right) 5000 generations.

Note in Fig. 5 the usual behavior of genetic algorithms, where in the first generations the evolution is very rapid, but it becomes slower as the number of generations advances.

2.6 Experiments and results

Extensive tests have been carried out exploring the performance of the system under different parameter values. For style training we have used two corpora: 65 MIDI files of classical pieces and 85 of jazz standards, respectively. The total

length was around 15,000 bars (more than 10 hours of music). Classical works were taken from Mozart, Bach, Schubert, Chopin, Grieg, Vivaldi, Schumann, Brahms, Beethoven, Dvorak, Haendel, Paganini, Lehar, and Wagner. Jazz samples were tunes from a variety of well known jazz authors including Charlie Parker, Duke Ellington, Bill Evans, Miles Davis, Jobim, Monk, Jobim, etc.

There are lots of parameters to be tuned in such a system, but most of them have very little influence on its performance. Here is a list of those who have shown really influent:

- Style-conditioned probability table for the initial population.
- Number of neighbors k utilized for style distance calculation.
- Number of generations until stop evolution, N_g.
- Number of individuals of the population, N_i.
- Recombination probability, P_c.
- Mutation probability, P_m.
- Weights w_s, w_n, and w_m controlling the relative importance of shallow, n-words, and melodic fitness.
- Statistical distribution for the n-words: binomial (BN) or multinomial (MN).
- n for the length of n-words.

For each time the system has been run, the best individual has been taken and it has been rated (from 0 to 10) from a group of musicians in order to analyse the performance as a function of the different parameters. For example, MN distributions have performed better than BN. Other usual values among the best rated melodies were $N_g = 2500$, $N_i \in [30, 40]$, $P_c = 0.9$, $P_m = 0.3$, and $n \in \{2, 3\}$. But there is not a combination of them that overcomes the rest. The order of time for a $N_g = 2500$, $N_i = 100$ evolution was around 120 minutes on a Pentium Centrino at 1.8GHz, 1GB RAM.

A number of samples, with the values taken for the free parameters has been set at the URL: http://grfia.dlsi.ua.es/cmWeb/demos/

3 Conclusions and future work

This work has presented the design of a hybrid cooperative system for evolutionary music composition. The fitness part of the system is implemented as a pattern recognition system that, based on training data, is able to guide the genetic algorithm in its composition work.

The early stages of testing the system have been encouraging since the performance has responded to the a priori expectations. More work has to be carried out in the future to explore the creative capabilities of the proposed system.

The use of a higher level structure on the sequence should improve the musicality of the results. Thus, structures like A-A-B-A are often used in jazz or blues. This way, for a 16 measure sequence, the system should compose 4 measures for A and for B, and then they will be fitted into the structure.

The use of other styles is being studied and corpora are being compiled. Also, the use of particular authors or personal tastes is being planned for testing the system capabilities.

4 Aknowledgements

This work was supported by the projects Generalitat Valenciana GV06/166 and Spanish CICyT TIC2003–08496–C04, partially supported by EU ERDF.

References

1. Todd, P., Werner, G.: Frankensteinian methods for evolutionary music composition. MIT press/Bradford books (1998)
2. Holland, J.: Adaptation in natural and artificial systems. Ann Arbor: The University of Michigan Press (1975)
3. Wiggins, G., Papadopoulos, G., Phon-Amnuaisuk, S., Tuson, A.: Evolutionary methods for musical composition. In: Proc. of CASYS'98 Workshop on Anticipation, Music and Cognition. (1998)
4. Putnam, J.B.: A grammar-based genetic programming technique applied to music generation. In: Evolutionary Programming V: Proc. of the 5th Annual Conf. on Evolutionary Programming, MIT Press (1996) 277–286
5. Tokui, N., Iba, H.: Music composition with interactive evolutionary computation. In: Proc. of 3rd Int. Conf. on Generative Art. (2000) 215–226
6. Fu, T., et al.: Evolutionary interactive music composition. In: Proc. of 8th Annual Conf. on Genetic and evolutionary computation, GECCO'06. (2006) 1863–1864
7. Todd, P., Miranda, E.R.: Putting some (artificial) life into models of musical creativity. In Deliege, I., Wiggins, G., eds.: Musical creativity: Current research in theory and practise. Psychology Press (2003)
8. Moroni, A., et al.: Vox populi: An interactive evolutionary system for algorithmic music composition. Leonardo Music Journal **10** (1994) 49–54
9. Cope, D.: Computers and musical style. A-R Editions, Inc. (1991)
10. Spector, L., Alpern, A.: Induction and recapitulation of deep musical structures. In: Proc. of the IJCAI-95 Workshop on Music and AI. (1995) 41–48
11. S. Baluja, D.P., Jochem, T.: Towards automated artificial evolution for computer generated images. Connection Science **6** (1994) 325–354
12. Machado, P., Romero, J., Santos, M.L., Cardoso, A., Manaris, B.: Adaptive critics for evolutionary artists. EvoMusart 2004, LNCS **3005** (2004) 437–446
13. Rizo, D., Iñesta, J.M., Moreno-Seco, F.: Tree-structured representation of musical information. LNCS **2652** (2003) 838—846
14. Ponce de León, P.J., Iñesta, J.M., Pérez-Sancho, C.: A shallow description framework for musical style recognition. LNCS **3138** (2004) 871–879
15. Pérez-Sancho, C., Iñesta, J.M., Calera-Rubio, J.: Style recognition through statistical event models. Journal of New Music Research **34** (2005) 331–339
16. Agón, C., Haddad, K., Assayag, G.: Representation and rendering of rhythmic structures. In: Proc. of 2nd Int. Conf. on Web Delivering of Music. Wedelmusic'02. (2002) 109–116
17. Doraisamy, S., Rüger, S.: Robust polyphonic music retrieval with n-grams. Journal of Intelligent Information Systems **21** (2003) 53–70
18. Brin, S.: Near neighbor search in large metric spaces. In: The VLDB Journal. (1995) 574–584
19. McCallum, A., Nigam, K.: A comparison of event models for naive bayes text classification. In: AAAI-98 W. on Learning for Text Categorization. (1998) 41–48