Music Composition Based on Linguistic Approach

Horacio Alberto García Salas,[1] Alexander Gelbukh,[1] Hiram Calvo [1,2]

[1] Natural Language Laboratory, Computing Research Center,
National Polytechnic Institute, 07738, DF, Mexico
[2] Computational Linguistics Laboratory,
Nara Institute of Science and Technology,
Takayama, Ikoma, Nara 630-0192, Japan

itztzin@gmail.com, gelbukh@gelbukh.com,
hiramcalvo@gmail.com, calvo@is.naist.jp

Abstract. Music is a form of expression. Since machines have limited capabilities in this sense, our main goal is to model musical composition process, to allow machines to express themselves musically. Our model is based on a linguistic approach. It describes music as a language composed of sequences of symbols that form melodies, with lexical symbols being sounds and silences with their duration in time. We determine functions to describe the probability distribution of these sequences of musical notes and use them for automatic music generation.

Keywords: Affective computing, evolutionary systems, evolutionary matrix, generative music, generative grammars.

1 Introduction

Machine emotional intelligence is part of the objectives of affective computing research [20]. Music is one of the fine arts and represents a form of expression. A desirable feature for machines is that they could express musically since they do not yet have this ability [2]. The problem is how to teach machines to compose music.

Computers represent a musical instrument capable of generating a number of sounds. Development of computational models applied to humanistic branches as fine arts, especially music, has its results in generative music, music generated from algorithms. Different models have been applied in development of automatic music composers, for example, those ones based on neural networks [11], on genetic algorithms [21], on swarms [4], etc., resulting in a wide range of applications

Our work is to characterize music and find its patterns so it can be explained in terms of algorithms to model the process of musical composition. A notes' sequence has certain probability of appearing in a melody. There are certain sequences that occur more regularly that forms characteristic patterns for each musical composition. The likelihood for these patterns to appear is used by our algorithm to generate a musical composition automatically.

It is possible to develop computational tools to automate composition process using our model. The following are possible applications of such systems:

- Have a personal music composer.
- Create new music styles by finding different patterns styles and mixing them.
- Help people without musical knowledge to compose music. Providing tools to allow users edit generated compositions, resulting user's composition.
- Enable computers to have the capacity to carry out a process until now reserved for humans. Making this, machines will get human characteristics creating another way of human-machine communication.
- Offer another alternative for creation of music; as a consequence, other alternatives of music are possible to be listened.
- Have machinery for the generation of live music for restaurants, offices, shops, etc. with compositions created in real time by indefatigable musicians.
- Provide tools to allow children from a very young age to have direct contact with musical composition process, which stimulates their minds for better performance in human activities.

This paper is organized as follows. In Section 2 we describe different algorithms to develop the same task we do. In Section 3 we explain our system. In Section 4 we present some results and a discussion about how we can improve our model. Section 5 is the future work we endeavor to accomplish. Then we present some conclusions.

2 Related Work

The works [19] and [12] provide a comprehensive study of different methods that have been used to develop music composition systems based on: noise [5], knowledge, cellular automata, grammars [18], evolutionary methods, fractals, genetic algorithms [1], case based reasoning [14], agents [16] and neural networks [6, 11]. Some systems are called hybrid since they combine several of these techniques.

For example, Harmonet [11] is a system based on connectionist networks, which has been trained to produce coral style of J. S. Bach. It focuses on the essence of musical information, rather than restrictions on music structure. The authors of [6] believe that music composed by recurrent neural networks lacks structure, as they do not maintain memory of distant events, and developed a model based on LSTM (Long Short Term Memory) to represent the overall and local music structure, generating blues compositions.

The work [13] describes a system for automatic music genre recognition based on signal's audio content, focusing only on melodies of three music genres: classical, metal and dance. The work [3] presents a system to recognize through the contents of a music database, which includes audio files (MIDI), with the idea to make search based on music contours, *i.e.* in a relative changes representation in a melody frequencies, regardless of tone or time.

There is a number of works based on evolutionary ideas for music composition. For example, [18] used generative context-free grammars for modeling the melody, through genetic algorithms making grammar evolve to improve the melody and produce a composition. GenJam [1] is a system based on a genetic algorithm that models a novice jazz musician learning to improvise. Musical phrases are generated at random and user feedbacks the system, generating new compositions improving through several generations. In [21] a genetic algorithm with coevolution, learning and rules is used in a music composer system. In it, male individuals produce music and female critics evaluate it to mate with suitable males creating new melodies generations.

3 Music Composer

A melody is a structure made up of other structures built over time. These structures are notes sequences. How many times a musical note is used after another reflects patterns of notes' sequences in a melody. A personal characteristic of each author is the use of certain notes' patterns with more regularity. We focus on finding these patterns over monophonic music to characterize it probabilistically.

Our model is built based on a linguistic approach [8]. It describes music as a language composed of sequences of symbols, which lexical items are sounds and silences throughout time. Each melody is made of phrases of this language. Notes of a melody represent sounds or silences. Sequences of notes form phrases of sounds.

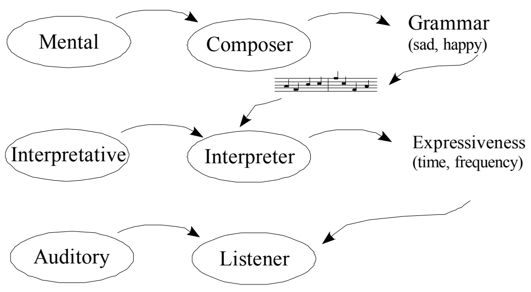


Fig. 1. Model of Music Process

Music process involves three main levels, mental, interpretative and auditory [15]. The process of musical composition is a mental process that involves the conception of an idea to be expressed in sounds and silences. The result of composition process is a musical composition and can be shaped in a score or in a sound file. In our model the language that represents the score is represented by a grammar. The performer turns the musical work into sound, adding his personal traits of expressiveness. The sound reaches the audience who gives meaning to the music according to how is perceived. Our model focuses on the mental level, see Fig. 1.

To model the process of musical composition we rely on the concept of *evolutionary systems* [7], which states that systems evolve as a result of constant change caused by the flow of matter, energy and information [9]. Evolutionary systems interrelate with their environment finding rules to describe phenomena, they use functions that allow them to learn and adapt to changes that come before them. These rules can be expressed in the form of grammars. A generative grammar G (V_n, V_t, S, R), where V_n is the set of non-terminal symbols, V_t is the set of terminal symbols or alphabet, which are the musical notes, the initial symbol S and a set of rules R.

Each genre, style and musical author has its own rules of composition. Not all rules are described in music theory. So to make automatic music composition we use an evolutionary system to find the rules that determine the form of each melody in unsupervised manner. The scheme of our model is shown in Fig. 2.

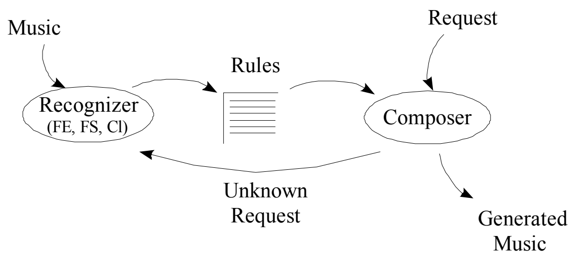


Fig. 2. Scheme of our model

A characteristic of our model is the ability to learn from examples of music m_i. From each example probabilistic grammars G_i are generated to describe patterns that characterize musical expressiveness of each melody. These learned rules are used to generate melody m_{i+1} automatically. The function R called recognizer generates production rules of grammar G from each musical melody, thereby creating an image of reality in musical terms.

$$R(m_i) = G$$

It is possible to construct a function C(G). C is called a musical composer and uses G, a generative grammar to produce a novel melody *m*.

$$C(G) = m$$

In this paper we are dealing with Composer and Recognizer Functions. To hear the music composition it must exist a function I called musical interpreter or performer that generates the sound of melody *m*. I recognize the lexical-semantic symbols of G that make the expressiveness of melody *m*.

$$I(m) = \textit{sound}$$

3.1 R Function Recognizer: Music Learning Module

Our model is modified according to each new melody. For every melody a musical language is generated that represents it. This is equivalent to generate a different automaton or a new compiler. Each example makes the model to restructure and to adapt to changes getting more musical knowledge.

We are working with melodies, monophonic music, modeling frequencies and times of notes, the two more important variables of expressivity in music. Each of these variables forms a sequence along the melody. We construct a probability function for each sequence using a matrix. This matrix can be transformed into a probabilistic grammar. In 3.3 Matrix and Grammar we explain and algorithm to make this transformation.

We are going to explain how the frequency matrix works. Time's matrix works the same way. For example, Fig. 3 is a frequencies sequence of a melody. Where V_t={b, d#, e, f#, g, a, b_2, d_2, e_2, g_2} are the terminal symbols or alphabet of this melody. Each of these symbols of the alphabet corresponds to each note in a chromatic scale: A, A#, B, C, C#, D, D#, E, F, F#, G, G#.

El cóndor pasa (Peruvian song)

b e $d_{\#}$ e $f_{\#}$ g $f_{\#}$ g a b_2 d_2 b_2 e_2 d_2 b_2 a g e g e
b e $d_{\#}$ e $f_{\#}$ g $f_{\#}$ g a b_2 d_2 b_2 e_2 d_2 b_2 a g e g e
b_2 e_2 d_2 e_2 d_2 e_2 g_2 e_2 d_2 e_2 d_2 b_2 g e_2 d_2 e_2 d_2
e_2 g_2 e_2 d_2 e_2 d_2 b_2 a g e g e

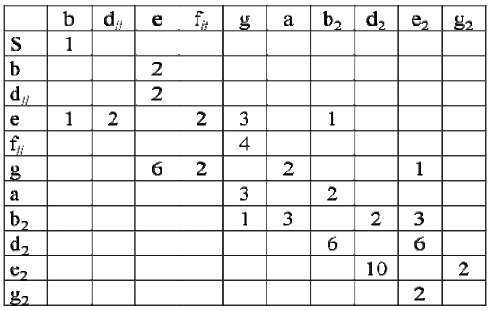
Fig. 3. Example of a monophonic melody

Let *Notes*[*n*] to be an array in which are stored the numbers corresponding to melody notes. Where *n* is the index which refers to each array element. Let $M_{i,j}$ be a matrix with *i* rows and *j* columns. Fig. 4 shows the learning algorithm we use to generate frequency distribution matrix of Fig. 5.

```
for each i ∈ Notes[n], j ∈ Notes[n+1] do
    $M_{i,j}$ = $M_{i,j}$ + 1
```

Fig. 4. Learning algorithm

We use a matrix of 60 rows and 60 columns representing 5 musical octaves to store frequency's sequences. 5 musical octaves are 60 chromatic notes. Frequencies matrix's tags of rows and columns are the notes (a, a#, b, c, c#, d, d#, e, f, f# g, g#, a_2, a_2#,…,g_5, g_5#). A matrix of 7 rows and 7 columns is used to store time's sequences corresponding to whole note (semibreve), half note (minim), quarter note (crotchet), eighth note (quaver), sixteenth note (semiquaver), thirty-second note (demisemiquaver) and sixty-fourth note (hemidemisemiquaver).



	b	$d_{\#}$	e	$f_{\#}$	g	a	b_2	d_2	e_2	g_2
S	1									
b			2							
$d_{\#}$			2							
e	1	2		2	3		1			
$f_{\#}$					4					
g			6	2		2			1	
a					3		2			
b_2					1	3		2	3	
d_2							6		6	
e_2								10		2
g_2									2	

Fig. 5. Frequency distribution matrix

Each number stored in frequency matrix represents how many times a row note was followed by a column note. An S row should be added to store the first note of each melody. S represents the axiom or initial symbol. Fig. 5 shows frequency distribution matrix after applying the learning algorithm to melody of Fig. 3. Matrix of Fig. 5 only nonzero contains columns and rows.

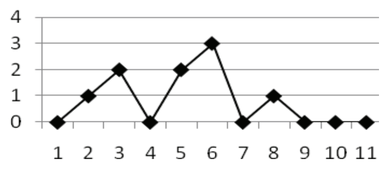


Fig. 6. Frequency distribution of *e* note

Rows of matrix of Fig. 5 represent frequency distribution of each note. In Fig. 6 we show as example the frequency distribution of *e* row. How many times a note is followed by another note can be used to calculate its probability distribution.

3.2 C Function Composer: Music Generator Module

C (Composer) function generates a note sequence based on probabilities determined from frequency distribution matrix. From each note is possible to go only to certain notes according to frequency distribution for each note. The most probable notes form characteristic musical patterns.

To determine the probability that a note follows another note, we need to determine the cumulative sum of each matrix row of Fig. 5. Let $M_{i,j}$ to be a matrix, with *i* rows and *j* columns. We calculate the cumulative sum for each *i* row such that $M_{i,j} \neq 0$. The partial *i* row sum is stored in each non-zero cell. We add a column T where the total cumulative sum for each row *i* is stored.

```
  for each $i \in M$ do
    for each $j \in M$ do
      $T_i = T_i + M_{i,j}$
$M_{i,j} = T_i$    for each $M_{i,j} \neq 0$
```

Fig. 7. Cumulative frequency distribution algorithm

With each new melody m_i matrix $M_{i,j}$ is modified. This means that world's representation of our model has changed. It has more music knowledge. Fig. 8 is cumulative frequency distribution matrix after applying cumulative frequency distribution algorithm Fig. 7 to frequency distribution matrix Fig. 5.

For music generation is necessary to decide next note of the melody. To take this decision a human composer bases in his musical knowledge. In our model this decision is made based on the cumulative frequency distribution matrix using the note generator algorithm Fig. 9.

	b	$d_{\#}$	e	$f_{\#}$	g	a	b_2	d_2	e_2	g_2	T
S	1										1
b			2								2
$d_{\#}$			2								2
e	1	3		5	8		9				9
$f_{\#}$					4						4
g			6	8		10			11		11
a					3		5				5
b_2					1	4		6	9		9
d_2							6		12		12
e_2								10		12	12
g_2									2		2

Fig. 8. Cumulative frequency distribution matrix

For example let us generate a melody based on matrix of Fig. 8. Music generation begins choosing the first composition note. We choose one of possible beginning notes, that is, notes which are first notes of melodies examples. S row of matrix of Fig. 8 contains all beginning notes. In our example, applying note generator algorithm Fig. 9 there is only one possible note to choose. This first note represents an *i* row of $M_{i,j}$ which we use to determine the next note. The same happens with second note. Only *e* note can be chosen from the first note *b*.

```
while(not end)
  p=random(T_i)
  while(M_i,j < p)
    j++
  new_note = j
  i=j
```

Fig. 9. Note generator algorithm

The first two notes of this new melody are $m_{i+1} = \{b,\ e\}$. Applying the note generator algorithm to determine the third note: We take the value of column $T_e = 9$. A p random number between zero and 9 is generated, $p = 6$. To find next note we compare p random number with each non-zero value of the E row until one greater than or equal to this number is found. Then column g is this next note since $M_{e,g} = 8$ is greater than p=6. The column j=g is where it is stored this number that indicates the following composition note and the i following row to be processed. The third note of new melody m_{i+1} is g. So $m_{i+1} = \{b,\ e,\ g,\ldots\}$. Then to determine the fourth note we must apply the note generator algorithm to $i = g$ row.

Since each non-zero value of *i* represents notes that were used to follow the *i* note, then can use them to generate patterns found in melodies examples. Generated music reflects these patterns learned from music examples.

While the system generates a musical composition with each note it modifies itself, increasing the likelihood for that note to be generated again. This is another way the system evolves. Besides we added a forgetting mechanism to ensure that the values do not overflow, which causes the notes played the least, lesser probability to be played again even they are not forgotten.

3.3 Matrix and Grammar

There are different ways to obtain a generative grammar G. A particular unsupervised case is an evolutionary matrix [10]. The algorithms described in Figs. 4, 7 and 9 of functions R and C represent an evolutionary matrix. An evolutionary matrix is a way of knowledge representation. From frequency distribution matrix and the T total column Fig. 8 is possible to generate a probabilistic generative grammar.

```
for each i ∈ M do
  for each j ∈ M do
    if $M_{i,j} \neq 0$
      $M_{i,j} = M_{i,j} / T_i$
```

Fig. 10. Probability algorithm

To apply the algorithm, we need to determine probability matrix from frequency distribution matrix of Fig. 8. Probability matrix is calculated with the probability algorithm of Fig. 10.

	b	$d_\#$	e	$f_\#$	g	a	b_2	d_2	e_2	g_2
S	1									
b			1							
$d_\#$			1							
e	1/9	2/9		2/9	3/9		1/9			
$f_\#$					1					
g			6/11	2/11		2/11			1/11	
a					3/5		2/5			
b_2					1/9	3/9		2/9	3/9	
d_2							6/12		6/12	
e_2								10/12		2/12
g_2									1	

Fig. 11. Probability matrix

Exist a grammar G{V_n, V_t, S, R} such that G can be generated from M, where M is the probability matrix Fig. 11. V_n is the set of no-terminals symbols, V_t is the set of all terminal symbols or alphabet; in this particular case the alphabet represents melody's notes. S is the axiom or initial symbol and R is the set of rules generated. To transform matrix of Fig. 11 into grammar of Fig. 13 we use the following algorithm:

- We substitute each tag row of M with a no-terminal symbol of grammar G, Fig. 12.
- Each column tag must be substituted by its note and its non-terminal symbol, Fig. 12.
- For each *i* row and each *j* column such that $M_{i,j} \neq 0$, *j* column represents a terminal symbol and a X_n no-terminal symbol with probability $p = M_{i,j} / T_i$ to occur. Then rules are of the form $V_n \rightarrow V_t V_n(p)$.

In this way the grammar is G{V_n,V_t, S, R}. V_n={S, X_1, X_2 X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}} is the set of no-terminals symbols. V_t={b, $d_\#$, e, $f_\#$, g, a, b_2, d_2, e_2, g_2} is the set of all terminal symbols or alphabet. S is the axiom or initial symbol. Rules R are listed in Fig. 13.

	bX_1	$d_{\#}X_2$	eX_3	$f_{\#}X_4$	gX_5	aX_6	b_2X_7	d_2X_8	e_2X_9	g_2X_{10}
S	1									
X_1			1							
X_2			1							
X_3	1/9	2/9		2/9	3/9		1/9			
X_4					1					
X_5			6/11	2/11		2/11			1/11	
X_6					3/5		2/5			
X_7					1/9	3/9		2/9	3/9	
X_8							6/12		6/12	
X_9								10/12		2/12
X_{10}									1	

Fig. 12. Transition matrix

$S \rightarrow b\ X_1(1)$
$X_1 \rightarrow e\ X_2(1)$
$X_2 \rightarrow e\ X_3(1)$
$X_3 \rightarrow b\ X_1(1/9)\ |\ d_{\#}\ X_2(2/9)|\ f_{\#}\ X_4(2/9)\ |\ g\ X_5(3/9)\ |\ b_2\ X_7(1/9)$
$X_4 \rightarrow g\ X_5(1)$
$X_5 \rightarrow e\ X_3(6/11)\ |\ f_{\#}\ X_4(2/11)\ |\ a\ X_6(2/11)\ |\ e_2\ X_9(1/11)$
$X_6 \rightarrow g\ X_5(3/5)\ |\ b_2\ X_7(2/5)$
$X_7 \rightarrow g\ X_5(1/9)\ |\ a\ X_6(3/9)\ |\ d_2\ X_8(2/9)\ |\ e_2\ X_9(3/9)$
$X_8 \rightarrow b_2\ X_7(6/12)\ |\ g_2\ X_{10}(6/12)$
$X_9 \rightarrow d_2\ X_8(10/12)\ |\ g_2\ X_{10}(2/12)$
$X_{10} \rightarrow e_2\ X_9(1)$

Fig. 13. Probabilistic generative grammar

4 Results and discussion

Examples of music generated by our system can be found at www.olincuicatl.com.

To evaluate whether our algorithm is generating music or not, we decided to conduct a Turing-like test. 26 participants of the test had to tell us if they like music generated by our model, knowing anything about that but it was automatically music generated.

We compiled 10 melodies, 5 of them generated by our model and another 5 by human composers and we asked 26 human subjects to rank melodies according to whether they liked them or not, with numbers between 1 and 10 being number 1 the most they liked. None of subjects knew about the order of music compositions. These 10 melodies were presented as in table 1.

Table 1. Order of melodies as they were presented to subjects

ID	Melody	Author	ID	Melody	Author
A	Zanya	(generated)	F	Dali	Astrix
B	Fell	Nathan Fake	G	Ritual Cibernetico	(generated)
C	Alucin	(generated)	H	Feelin' Electro	Rob Mooney
D	Idiot	James Holden	I	Infinito	(generated)
E	Ciclos	(generated)	J	Lost Town	Kraftwerk

Test results were encouraging: since automatically generated melodies were ranked at 3rd and 4th place above human compositions of very famous bands. Table 2 shows the ranking of melodies as a result of the Turing-like test we conducted.

Table 2. Order of melodies obtained after the like Turing test

ID	Ranking	Melody	Author
B	1	Fell	Nathan Fake
D	2	Idiot	James Holden
C	3	Alucín	(generated)
A	4	Zanya	(generated)
F	5	Dali	Astrix
H	6	Feelin' Electro	Rob Mooney
J	7	Lost Town	Kraftwerk
E	8	Ciclos	(generated)
G	9	Ritual Cibernético	(generated)
I	10	Infinito	(generated)

We have obtained novelty results comparable with those obtained by other developments [21, 1], modeling frequency and time of a melody with simple algorithms.

To the ears of musicians compositions generated by our system sound similar to the used examples. However we are developing other algorithms in order to shape the musical structure [17]. We consider if a larger corpus is used the results will considerably improve.

It is necessary to develop more sophisticated forgetting functions to improve the method.

5 Conclusions and Future Work

We have developed a model for music composition process. A way to represent music based on an evolving matrix [10] a paradigm for knowledge representation.

We developed an algorithm to transform a matrix where we represent music into a grammar, what it is a linguistic representation of music.

Generative music presents new forms that not always match with traditional rules of music. This feature is perhaps one of the attractions of these new forms of music which breaks with preset patterns.

Transition patterns are measured statistically to determine the probability of moving from one musical note to another. This process can be model with a grammar, automata, matrix, etc. We propose a model, regardless of the modeling tool, for characterize music composition process.

In our future work we will characterize different types of music, from sad to happy, from classic to electronic in order to determine functions for generating any kind of music.

We are currently developing systems to improve using matrices of 3, 4 or *n* dimensions, which may reflect the many variables involved in a musical work. To model more music variables will be reflected in music expression.

We plan to make matrices evolve into some other matrices to produce music morphing. Also, we are interested in develop a polyphonic model. Finally, it is necessary to develop better forgetting functions.

Acknowledgements. The work was done under partial support of Mexican Government (CONACYT 50206-H, IPN-SIP 20100773, IPN-COFAA, IPN-PIFI, SNI, CONACYT scholarship for Sabbatical stay to the second author) and the Japanese Government (JSPS). The third author is a JSPS fellow.

References

1. Biles, J. A. (2001) GenJam: Evolution of a jazz improviser. Source. Creative evolutionary systems. Section: Evolutionary music. Pages: 165–187. Morgan Kaufmann Publishers Inc. San Francisco, CA, USA.
2. Birchfield, D. (2003). Generative model for the creation of musical emotion, meaning and form. Source International Multimedia Conference. Proceedings of the 2003 ACM SIGMM workshop on Experiential telepresence. Berkeley, California SESSION: Playing experience. Pages: 99–104. ACM New York, NY, USA.
3. Blackburm, S. and DeRoure, D. (1998). A tool for content based navigation of music. Source International Multimedia Conference. Proceedings of the sixth ACM international conference on Multimedia. Bristol, United Kingdom. pp. 361–368.
4. Blackwell, T. (2007). Swarming and Music. Evolutionary Computer Music. Springer London. pp. 194–217. Subject Collection: Informática. In SpringerLink since 12 October 2007.
5. Bulmer, M. (2000). Music From Fractal Noise. University of Queensland. Proceedings of the Mathematics 2000 Festival, Melbourne, 10–13 January 2000.
6. Eck, D. Schmidhuber, J. (2002). A First Look at Music Composition using LSTM Recurrent Neural Networks. Source Technical Report: IDSIA-07-02. Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale.
7. Galindo Soria, F. (1991). Sistemas Evolutivos: Nuevo Paradigma de la Informática. Memorias XVII Conferencia Latinoamericana de Informática, Caracas Venezuela, July 1991.
8. Galindo Soria, F. (1994). Enfoque Lingüístico. Instituto Politécnico Nacional UPIICSA ESCOM.
9. Galindo Soria, F. (1997). Teoría y Práctica de los Sistemas Evolutivos. Mexico. Editor Jesús Manuel Olivares Ceja.
10. Galindo Soria, F. (1998). Matrices Evolutivas. La Revista Científica, ESIME del IPN, #8 de 1998, pp. 17–22. Cuarta Conferencia de Ingeniería Eléctrica CIE/98, CINVESTAV-IPN, Cd. de México, September 1998.
11. Hild, H. Feulner, J. Menzel, W. Harmonet. Harmonet: A Neural Net for Harmonizing Chorales in the Style of J.S.Bach. in in Neural Information Processing 4 (NIPS 4), pp. 267–274, R.P. Lippmann, J.E. Moody, D.S. Touretzky (eds.), Morgan. Kaufmann.Universität Karlsruhe, Germany.
12. Järveläinen, H. (2000). Algorithmic Musical Composition. April 7, 2000. Tik-111.080 Seminar on content creation Art@Science. Helsinki University of Technology Laboratory of Acoustics and Audio Signal Processing.

13. Kosina, K. (2002). Music Genre Recognition.. Diplomarbeit. Eingereicht am Fachhochschul-Studiengang. Mediente Chnik und Design in Hagenberg. June 2002.
14. Maarten, G. J. A. López. M. R. (2006). A Case Based Approach to Expressivity-Aware Tempo Transformation. Source Machine Learning Volume 65, Issue 2–3 (December 2006). pp. 411–437. Kluwer Academic Publishers Hingham, MA, USA.
15. Miranda E. R., Jesus L. A., Barros B. (2006). Music Knowledge Analysis: Towards an Efficient Representation for Composition. Springer Berlin/Heidelberg, Vol. 4177. Current Topics in Artificial Intelligence. Selected Papers from the 11th Conference of the Spanish Association for Artificial Intelligence (CAEPIA 2005). pp. 331–341. Subject Collection: Informática. In SpringerLink since 13 October 2006.
16. Minsky, M. (1981). Music, Mind, and Meaning. Computer Music Journal, Fall 1981, Vol. 5, Number 3.
17. Namunu, M. Changsheng, X. Mohan, S K. Shao, X. (2004). Content-based music structure analysis with applications to music semantics understanding. Source International Multimedia Conference. Proceedings of the 12th annual ACM international conference on Multimedia. Technical session 3: Audio Processing. pp. 112–119. ACM New York, NY, USA.
18. Ortega, A. P. Sánchez, A. R. Alfonseca M. M. (2002). Automatic composition of music by means of Grammatical Evolution. ACM SIGAPL APL. Volume 32, issue 4 (June 2002) pp. 148–155. ACM New York, NY, USA.
19. Papadopoulos, G., Wiggins, G. AI Methods for Algorithmic Composition: A Survey, a Critical View and Future Prospects. AISB Symposium on Musical Creativity 1999, pp. 110–117. School of Artificial Intelligence, Division of Informatics, University of Edinburgh.
20. Picard, R. W., Vyzas, E., Healey, J. (2001). Toward Machine Emotional Intelligence: Analysis of Affective Physiological State. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 23, No. 10, October 2001.
21. Todd, P.M. & Werner, G. M (1999). Frankensteinian Methods for Evolutionary Music Composition. Musical networks. p. 385. Editors: Niall Griffith, Peter M. Todd. MIT Press, Cambridge, MA, USA.