AN AUTONOMOUS STYLE-DRIVEN MUSIC COMPOSER

LUCIANO VIEIRA LIMA
Departamento de Engenharia Elétrica
Universidade Federal de Uberlândia (UFU)
Uberlândia – MG - Brasil
dreams@nanet.com.br

JOÃO JOSÉ NETO
Departamento de Engenharia de Computação e Sistemas Digitais
Escola Politécnica da Universidade de São Paulo (EPUSP)
São Paulo – SP – Brasil
jjneto@pcs.usp.br

Abstract

This paper resulted from our search for better knowledge of the human's mind creative processes, not only in music but also in other fields. Music is one of the most complex, abstract and intuitive activities executed by human beings. We believe that having success in this domain will help understanding and modeling phenomena involving creativity and temporal reasoning. Our approach has been musical composition based on existent musical styles, or modeling several composers' knowledge, creativity, styles and works. Our system's model composes music following the style of any composer, focusing mono- or polyphonic solo instruments. In a fashion similar to that used by many composers of pop-music, our work searches for solutions for this problem using a note-by-note approach, with no help of traditional rules or personal knowledge of musical theory. This article also discusses whether or not this task can be accomplished wit no user interaction and without backtracking procedures. Our composition system should start from musical inputs; form that all needed composing knowledge has to be extracted. This work illustrates mankind's need to leave to future generations not only results of work but also the creative process that originated them.

1-Introduction

Many works on this subject have been analyzed and discussed that use harmony rules, stochastic grammars and similarity methods in their creation process. For style-driven composition, reference composers' works are searched for elementary repetitive forms that identify their style. Such a system, presented by David Cope (Cope 1988, 89 e 92), uses the technique he calls EMI. In his work, Cope locates the smallest repetitive forms, grains (Smoliar 1991), in music, and associates them to verbs, objects, nouns, adjectives etc. The spaces that pass it action sensation are classified as verbs, and the variations that follow it, as objects, e.g., a music fragment that seem to be receiving some action is classified as a subject. Cope reorganizes subjects and verbs, forming new musical sentences, following given grammatical rules. By changing weights, Cope interacts dynamically with the composition process, tuning the quality of the output with respect to their closeness to the rules of the given grammar. For that purpose, trial and error, as well as backtracking techniques are thoroughly used in the generation of the compositions.

2-Goals

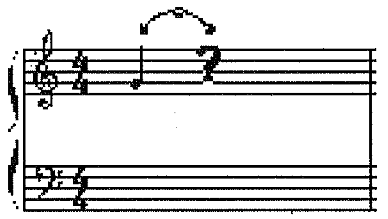
In contrast to similar systems published in the literature (Dodge & Jesse 1985; Hudak 1996, Todd 1989, Cope 1992), our system achieves several important goals:

1. it is able to generate new compositions with no interaction with the user
2. it does not require explicit rules of harmony and musical composition
3. it learns and generates music with no help of previous analysis
4. It is free of trial-and-error (Gogins 1991) procedures so all generated notes are correct.
5. Feedback of system-generated music do not change system's information base on style.

3-The model

Informally, a musical composition may be classified as pop music and classical music. Each piece of classical music in general shows its own well-defined, personal and explicit structure and semantics, in contrast to most pop music compositions.

The model presented in the present paper doesn't intend to capture and to generate new structures. Instead, given a structure, it models and simulates how each author links and generates musical notes in a musical sentence or theme. Once defined the desired structure, rhythm and musical harmony, our system starts to define the building grains of the composer's style from which it will build new musical compositions. In order to build each of these grains, the system starts from some initial note, and determines the following ones by answering successively the same question: which is the next note to be generated? (See figure)



This article presents for this question a proposal of answer that allow producing, at a low cost, quite satisfactory results.

4-Feelings

Great maestros have perhaps better musical knowledge than famous composers of the past do. However, most of them admit not to be able to compose comparably, for technical knowledge on composition techniques and harmonization are not enough to build good composers.

Thus, even by using well-known composition rules, and by well knowing and recognizing a given composer's style, the choice of how to use all this knowledge is always defined intuitively, taking into account cultural, social and emotional aspects of each one's life. So, each composer is permanently subject to dynamic unexpected events in his life, which may severely affect his or hers musical style (Lima 1998).

As an illustration, we can imagine a composer that fights his relatives has cut his account in the bank, has his dog run over, etc. Some time later, he composes a melody.

Many of these adversities will certainly have influence in the composition. These unexpected random facts are very hard to model.

As another example, we have a real fact occurred when the pop composer Eric Clapton lost his son in an accident. After a while, Clapton composed music in honor to his son, which registers that sad moment, and the deep suffering it induced. How to instruct the computer about the feeling of a son's loss? Anyone that has a son, or that lets himself to act as someone that has or that had one, realizes that it is very difficult to explain that in words, and far harder to model it.

Even for a human being, which never had a child, it is difficult to fully understand such a feeling. Thus, even for the composer himself, there are moments of inspiration, motivation and creativity whose replication is virtually impossible. From such unique facts, emerge several phases in a composer's career, although such phases are, also, strongly associated to the composer's technical evolution.

Yet regarding feelings, another fact also very hard to model is that the same person reacts differently to a single given stimulus in different circumstances. So, it is a complicated task to generate a system of automatic composition using computers embedding the ability to handle feelings, in an autonomous way, in the automatically generated melodies. So, how to model and to reflect in music great passions, sadness, joy, a disillusion? How to model feeling?

By now, this problem seems unsolved. In our system, we intend to reproduce or to simulate, in the generated music, part of the feeling expressed by composers in their music, no matter which these feelings are. This is a rather simpler task to perform. So, the choice of which note should follow a given one in a certain musical structure, is a problem for which solutions may be developed which are based solely on the contents of the music used as base, with no help of formal concepts and technical training. Being impossible to explain and to model the feelings of a composer, how to design a system that composes melodies reproducing them? We can only assure that undoubtedly some particular composer's feelings and emotions have been recorded in his music at the moment of its composition. Therefore, a system that composes based on the author's work only is expected to capture some of the moments and the original author's feelings from the original work and reproduce them, although in some imperfect form, in the generated melodies.

5-Our system

The system breaks music into 3 components: Rhythm, Harmonization and Melody.

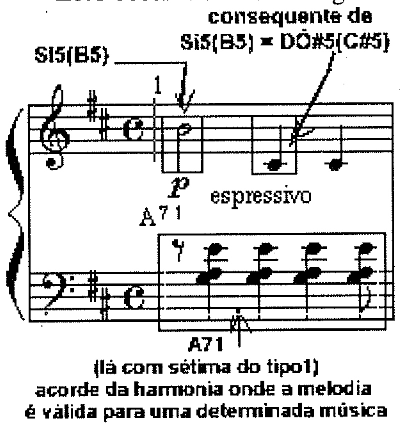
Similarly to what occurs in many style-driven composition systems, our system tries to generate the smallest grain, by reproducing the activity of the composer executes when originating such grains, according to the method described below.

The system described in this paper has been implemented in two versions, both in functional languages: the first was built in LISP and the second, in language CLEAN, to be executed in IBM-PC-compatible computers.

5.1-The concept of consequent note

A consequent note is any note the system is allowed to generate after an already composed one. Our approach to determine possible consequent notes of a given one it that the note has been used in similar circumstance in the compositions used as base. The key is defining the circumstances under which such similarity holds.

Let's observe the following illustration:



Converting the score's compass 1 to internal format gives:
1. (B5 1:2,C#5 1:4, C#5 1:4)
[At 7 1]; where 1:2 = half note and 1:4 = quarter note

Decomposing into the 3 components gives:
Rhythm = (1:2 1:4 1:4)
Harmony = (AT 7 1)
Melody = (B5 C#5 C#5)

Based on this information, and on the rest of the music, the system should be able to extract necessary data to generate new compositions. In the example notes and musical chords were given traditional names, but any name is allowed, since the system does not interpret them. Thus, note B5 could have been called, for example, note 1, chord A 7 1 could have been called chord 1 and so forth. At all occurrences of the same note or chord appear the same name is to be used.

Note by note generation, using the concept of consequent note is not as simple as it can seem. It first seems that the problem consists only in consulting the notes that the composer used in its music and those that follow it. So, any note that the author has used as consequent to the previous one it should be a correct note in a future composition. That is not true, and it can cause serious trouble in the automatic process of composition, as illustrated in the following example:

Suppose that, in a given melody, the author used do2, sol1 and la1 as consequent to note do1. If the author uses solely mi1 as a consequent of la1, and only la2 as consequent of mi1, and do1, sol2 and fa1 as consequent for sol1.

With these informations, the following information is collected by the system:
Consequent of DO1 = (DO2, SOL1, LA1)
Consequent of LA1 = (MI1)
Consequent of MI1 = (LA1)
Consequent of SOL1 = (DO1, SOL2, FA1)

This database supports an algorithm that randomly generates the first note, and automatically determines a consequent for that note, and so forth. This way, a possible composition generated automatically by such algorithm could be: (SOL1 - DO1 - LA1 - MI1 - LA1 - MI1 -... - repeating indefinitely the sequence LA1 - MI1).

Here the algorithm fails, because, when generating a note that has an only recurrent sequence of consequents, the system starts to repeatedly generate that sequence of musical notes. In the algorithm it should exist, something else, allowing the generation of the next note, whatever would be the data resultants of the simple collection of the consequent notes

that the author used in his original compositions. So, only the notes of its melody should not analyze the original music. Rhythm and harmony decisively affect the choice of consequent note, and should also be considered. Even when the original music is monophonic, harmony is decisively embedded in the composed melody.

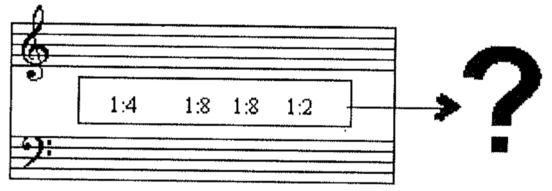
Avoiding recurrent sequences of notes is another point that distinguishes our system from many similar works. Our solution is based on composition by compasses. As each compass carries together information on rhythm, the closure of its metric and the solution for its harmony, each compass may be considered a basic self-contained element inside the final composition.

5.2-The algorithm

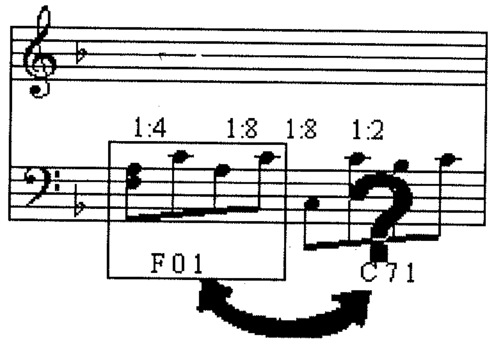
For implementing the composition routines, we must first create information bases and histograms, that allow translating all events in a music, linking them tightly in order to preserve the maximum of information that enables producing new music, with a style as close as possible to the music used as base of the adopted model.

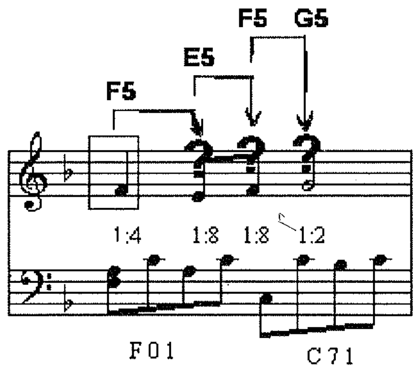
Three basic histograms are created with that purpose:

· **RHYTHM HISTOGRAM** - it indicates, for each compass, which rhythm can follow a given one. That information is calculated with base on the probability that, in similar conditions, the original composer would choose each of these compasses as a consequent.



CHORDS HISTOGRAM - This information allows the system to compose harmony for the generated music, by indicating which chord type can follow a preceding one, with base on the rhythm of the current compass, its antecedent and its consequent, and observing, also, the probability that, in similar conditions, the original composer would choose some chord as a consequent for the previously generated one.





· **NOTES HISTOGRAMS** - This histogram allows the system to compose the melody, based on the rhythm and harmony of the current, preceding and succeeding compasses, statistically indicating with support of similar situations observed in the author's original works, which notes should follow a given one.

The composing algorithm

After building histograms, the composing algorithm is naturally divided in three different strongly interconnected stages: the Rhythm stage, the Harmony stage and the Melody stage. A distinguished features in this system is that, in the note-by-note generation process, consequent notes are determined by the rhythm and especially by the harmony in each compass.

5.2.1-Composing rhythm

Choose the type of musical style (rhythm) or an author's music.

1. Determine the number of compasses for the new music.
2. Randomly choose a type for the first compass or choose a starting compass for the music.
3. Randomly choose a type for consequent compasses, according to the obtained compass histogram. Randomness is assured by the manner histograms are built by the system.
4. Repeat step 4 until all compasses determined in item 2 are built.

Example for three compasses:

1. (1:2, 1:2)
2. (1:2) (1:2)
3. (1:4, 1:4, 1:4, 1:4)

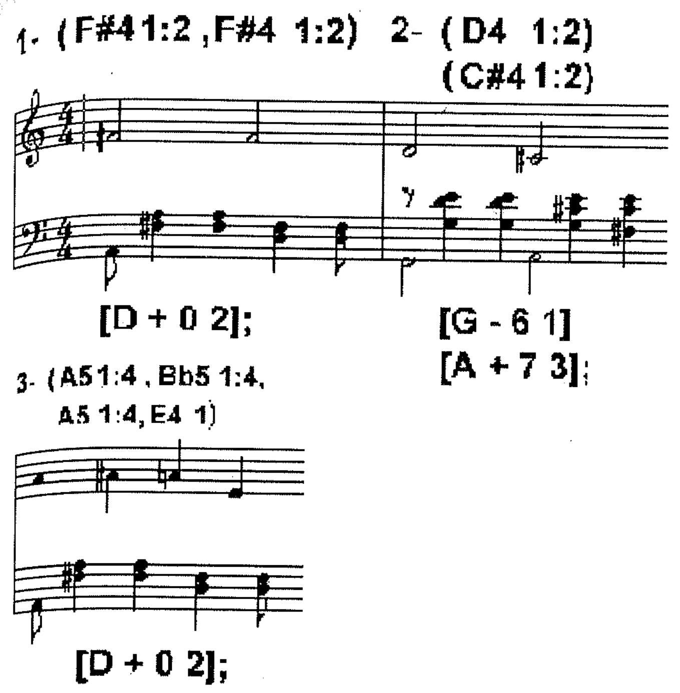
So, compass 1 will have two notes whose duration is of a minim (1:2). The compass will have just one chord. It indicates that compass 2 will have, also, two notes with duration of a minim each, but this compass will have two chords. Finally, the third compass indicates that the compass will have four notes with duration of a quarter note (1:4) and a single chord.

5.2.2-Composing harmony

1. Randomly choose the first chord of the current compass.
2. Randomly choose a consequent chord for the current compass. In the case the chord fills the whole compass, choose randomly a chord for the consequent compass.
3. Repeat item 2 until all chords in all compasses are formed.

Example for three compasses (continued):

1. (1:2, 1:2) [D + 0 2]
2. (1:2) [G - 6 1]   (1:2) [A + 7 3]
3. (1:4, 1:4, 1:4, 1:4) [D + 0 2]



That means compass one has one chord [D + 0 2] driving the generation of two notes, the second compass has two chords [G - 6 1] and [A + 7 3], driving the generation of one note, and the third compass has one chord [D + 0 2], which drives the generation of four notes.

5.2.3-Composing melody

1. Randomly choose the first note for the first chord of the first compass
2. If that is the last note in the chord, choose randomly its consequent transition note from the current chord to its consequent chord, otherwise, choose a consequent note for the chord randomly.
3. Fill all notes in all compasses repeating item 2 for all chords in the music.

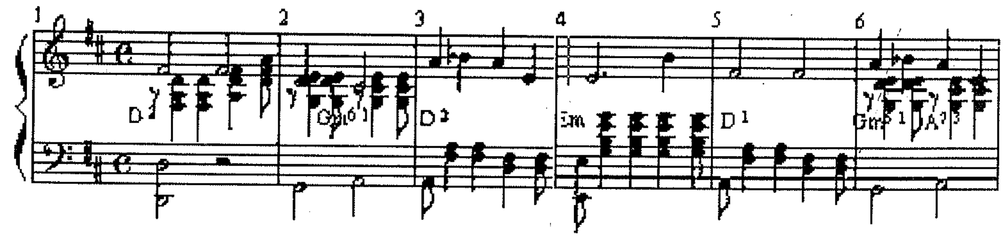
Example for three compasses (final):

1. (F#4 1:2, F#4 1:2) [D + 0 2]
2. (D4 1:2) [G - 6 1] (C#4 1:2) [A + 7 3]
3. (A5 1:4,Bb5 1:4,A5 1:4,E4 1:4)[D + 0 2]

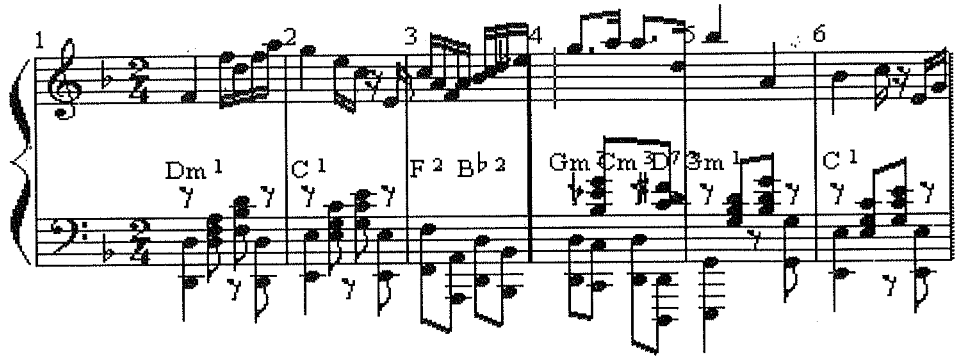
Translating into a conventional score (CPN) leads to the score above.

5.3-Fragments of music, composed by the system based on classic authors:

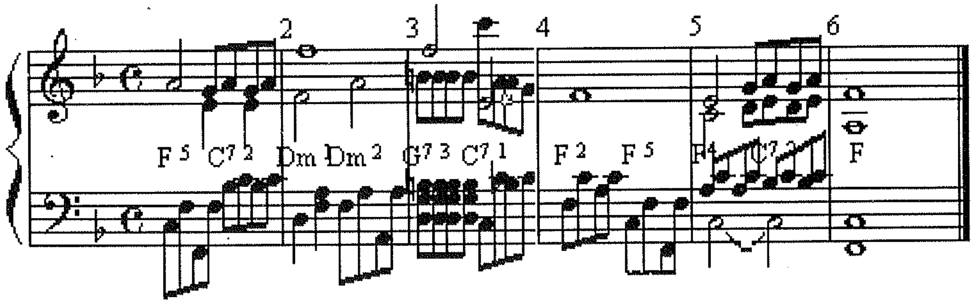
1. Six compasses based on None but a Lonely Heart by P. Tchaikowsky



2 - Six compasses, based on the Symphony of the Cantata 156 by J. S. Bach



3 - Six compasses based on Tristesse by F. Chopin.



Note that our system uses neither trial and error nor backtracking procedures; it can learn and generate music just analyzing existent music; Its base of information remains unchanged after being feedback with generated music; it uses no explicit formal harmony or composition rules.

Our system also differ from other systems based on note-by-note composition, for applying composition concepts on musical compasses and for using harmony (chords) in the decision process while choosing consequent notes.

One may ask: if our system adds nothing to the rhythm, chord and melody histograms for a given author, how would new music be generated?

Creating new music in a given style should not be interpreted as creating new music in a new style. Our system was not designed to make any improvements or to set modifications to an author's original style. While other systems accept musical intervals of a sequence, such as - la - do - mi - as well as mi - do - la , if a given author just used the first sequence in his compositions, accepting the other sequence may alter the style we are trying to follow.

As an analogy, one may analyze a piece of text whose author always uses sentences in direct speech. When doing automatic composition, using reversed speech in the generated phrases would depart away from the original style.

Vocabulary is to be preserved, as well as the general form of the sentences, which must follow original writer's phrases in their grammatical structure and style.

Similarly, the system presented in this article creates new melodies with no change to the original histograms, in contrast to other style-driven systems.

Our method for determining consequents may also be compared to a picture-generating system: such a system would build human pictures in which we would find more than one nose, mouths and ears would be wrongly placed, etc. This really happens, since no grammar is supplied to define the structure of the work to be generated. Instead, as it is implemented by now, our system was designed to generate those parts - noses, mouths, necks, etc, from given models extracted from existent work. In this case, the system tries to generate its output following as close as possible the style of the artist whose work is being used as model.

6-Future work

Presently a research is in progress, in order to improve our already developed tool by eliminating one of its fragilities, the absence of structure in the implemented generation algorithms. As it has been conceived to generate basic elements of a music, the external consequence of using it to generate a whole music allows one to easily identify the mentioned fragility in the resulting output, which, although being very pleasant in its melody, metrics and harmony, undoubtedly do lack global structure.

The way algorithms were developed, it is assumed that melody histograms used by the system to generate music are extracted from significant samples of original pieces, and that they are representative of music fragments of the same type.

Current work is being developed to extend the generating program in order to convert the existing one into a module of a larger system, which will have an external control section designated to handle structure.

In order to accomplish that goal, our project will take place in several phases; each one giving rise to a product representing some significant progress when compared to the previous one:

6.1-Phase 0 - in the current stage, we have a generator that is able to compose homogeneous fragments of music, starting from samples of original fragments assumed to be of the same nature. No connection is allowed among generated fragments, and there are no tests on inadequate classification of the fragments used as model. The practical output of the currently available implementation is in general pleasant but structure music.

6.2-Phase 1 - in progress - a natural progress to the previous phase consists of allowing the user to provide samples of several kinds as models. Users may then request to the system that composes a sequence of fragments following some user-specified order. The system then generates and links the composed fragments through appropriate passage sequences. The result is a sequence of properly linked fragments that follow a structure manually imposed by the user. There will be no tests on the imposed structure or on the criteria used to classify the given samples.

6.3-Phase 2 - to be initiated next - in this phase, a first automatic procedure will be included for generating the structure of the generated music. Users should manually replace the system's previous fixed structure with some structuring rule. The system will then apply that rule to generate all needed different structures of each kind, following the supplied rule.

Usual sentence derivation from a grammar will automatically generate the desired structure for the generated music. The user may impose restrictions, e.g. on the maximum number of fragments, of fragments of a given type, of repetitions of certain sequences of types. The expected result is the automatic generation of music with sound similar in quality to the previous ones, and showing the structure created by the system under user's specification.

6.4-Phase 3 - to be started in short term - the next stage refers to obtaining structuring rules automatically. The underlying grammar of the structure imposed to automatically generated compositions will be itself deduced by the system. A musician must help in this activity by supplying structural analyses for a significant set of works to be used as models, identifying and classifying the fragments in each work. Then, an algorithm is executed to make a grammatical inference that generalize the information extracted from the given samples, in order to infer a formation rule to be used as a grammar describing the desired structure. This phase will result in a complete synthesis program, for music will then be obtained automatically both in contents and in formal structure. User's participation will be restricted to inserting information on the structure of the samples used as models.

6.5-Phase 4 - this phase is foreseen to begin in some years. It is much more complex, because it should include heavier artificial intelligence resources; in order to operate with no human-supplied information concerning the structure of the pieces used as models. Structural analysis should be made by programs with enough built-in knowledge on musical analysis to allow automatic identification of the musical elements present in the score. This phase will produce software that will be able to perform musical analysis and synthesis, without direct participation of the user.

7-References

COPE, DAVID. **Experiments Musical in Intelligence (EMI): Non-linear Linguistic-Based Composition** Interface, vol, 18, p. 117-139, 1989.

COPE, DAVID. **Computer Modeling of Musical Intelligence in EMI**. Computer Music Journal, vol 16, no. 2, Summer, p. 69-83, 1992.

COPE, DAVID. **Music & Lisp**. OH Expert, March, 1988, p.26-33.

DODGE, CHARLES; JESSE, THOMAS A.. **Composition With Computers - Computer Music - Synthesis, Compositions and Performance.** NY, Schirmer, London, Coller MacMilan, 381 pg., p.265-321, 1985.

GOGINS, MICHAEL. Interated Functions Systems Music. Computer Music Journal, vol. 15, no. 1, Spring, 1991, p.40-48

HUDAK, PAUL; MAKUCEVICH, TONE; DDE, SYANGA; WONG,BO.**Haskore Music Notation–An Algebra of Music** -. Journal of Functional Language, vol 6, May, 1996 p.465-483

LIMA, LUCIANO VIEIRA. **Um Sistema de Composição Musical Dirigido por Estilo.** Tese de Doutorado, Universidade de São Paulo, São Paulo - Brasil, 1998.

SMOLIAR, STEPHEN W. Book Review and Response - **Models of Musical Communication and Cognition**. Interface, vol. 18, 1990, p. 361-371

TODD, PETERSOM. **Connectionist Approach to Algorithmic Composition**. Computer Music Journal, vol. 13, no.4, Winter,1989, p.27-43.

SOUND FUNCTORS APPLICATIONS

JÔNATAS MANZOLLI[2], ADOLFO MAIA Jr.[1,2]
Mathematics Department[1]
Interdisciplinary Nucleus for Sound Communications (NICS)[2]
University of Campinas (UNICAMP) - Brazil
Jonatas@nics.unicam.br, adolfo@nics.unicamp.br

ABSTRACT

The concept of mathematical structures called Functors can be useful to develop a large number of compositional algorithms. We introduce here concepts such as Categories and Functors as generalised sound construction tools. We define two applications: a functor between plane curves and sound events based on the MIDI protocol and a functor between the class of continuous curves $\mathbf{C(x)}$ *defined in a finite interval* $\mathbf{I} \subset \Re$ *and the class* Ψ *of Fourier Spectra* $\mathbf{A}(\nu)$.

INTRODUCTION

Music composition is plenty of procedures that can be related to mathematical symmetries, it is presented in (Hofstadter 1989), a study relating music, design and mathematical structures. On the other hand, music symmetries can be better understood and handled if we search for underlined or hidden mathematical structure that generates them. In this work we show that the intuitive use of symmetry and associations in music can be studied and explored through the mathematical concepts of Category and Functor.

This approach is interesting, not only because it can be used to classify sound material, but mainly for it furnishes new relations pointing out to an enormous variety of sound generation methods. In short, we claim that Functors, in the same way it was firstly devised in mathematics (Maclane & Birkohoff 1953, 1979; MacLane 1971), can be generalised as universal tools to construct mathematical models for music composition and sound synthesis.

There has been a series of approaches applying mathematics to build sound structures such the use of 1/f noise fractal distribution (Voss & Clark 1978; Bolognesi 1983), non-linear dynamical systems and iterated function systems (Pressing 1988; Scipio 1990; Gogins 1991). There is a study about these systems in (Manzolli, 1993a).

Our group has been worked with applied mathematics to produce sound design machines focusing new methods for sound synthesis using non-linear dynamics (Manzolli 1993b), mathematical models for algorithm composition such as Markov Chains and