# ALGORITHMIC COMPOSITION AS A CONSTRAINT SATISFACTION PROBLEM

*Rubén Hinojosa Chapel*
Music Technology Group
Pompeu Fabra University
Ocata 1, 08003 Barcelona, Spain
(2002) Revised: November 2005
contacto@hinojosachapel.com

## Abstract

Throughout the history, musicians have always used music composition rules, which actually are in most cases prohibitions or constraints. The elements of music with which composers work, as well as the possible combinations of those elements, make up a huge finite set of values. The composer's function is to select, from this big set, a subset of values, which will be arranged into a certain temporal organization, constrained by the music rules, and eventually create an artistic work. From a mathematical point of view, this process is closely related to one field of Artificial Intelligence: *Constraint Programming*. In this paper the author makes a reflection on the relationship between Music Composition and Constraint Programming. As a concluding idea, the author suggests that Constraint Programming could be a promising technique for the development of computer systems oriented to algorithmic composition. This text follows after the lectures given by professor Dr. Héctor Geffner, in the doctoral course *Problem Solving in Artificial Intelligence* (2001-2002).

## 1 Introduction

*"Schönberg – someone has said– is an inventor of theorems, who always seems to be demonstrating it on the blackboard, with geometric formulae... What it is true is that music can never turn to forces of unconscious, as poetry or painting does; it is an art – though the term scare me– terribly Euclidian. And a Bach's fugue, as well as a motete by Victoria, always can be translated on the blackboard by means of geometric formulae."* [1]

The high grade of abstraction and formalism that music has on its theoretical aspects, has often lead to be compared to mathematics. Not without reason, one of the most important music disciplines, Harmony, is used to be raised to the category of a science. It has been stated, for instance, that musicians have invented the notion of coordinates before mathematicians did it. In fact, a music score is actually a coordinate system, where frequencies values are represented onto the ordinate axis, and the time flow is represented onto the axis of abscissas.

In his creative labour, the music composer works with the so-called elements of music, such as: melody, harmony, timbre, articulation, dynamics, form, etc. We could compare this process to the one followed by a cook who creates a new recipe. He has at hand a lot of ingredients that can be used, but he will only use a little portion. He will select a small portion from the set of ingredients that he can buy, for instance, in one or several markets. After that, he will merge, prepare, cook, etc., these ingredients according to the rules and proportions studied and developed for centuries of culinary art. For instance, in general, the amount of salt that he can use has a very low superior boundary, because the food could become salty if he surpass that limit.

Now think about a music composer who wants to create a new piece. He has at hand a huge set of values (of frequencies or pitches, durations or rhythms, intensities or dynamics, timbres or instruments, etc.) that he can combine horizontally as well as vertically, in lots, so many different ways. The combinatory explosion is enormous, which is confirmed by the high number of known music pieces. So, what criterion should be followed in order to select a subset of values that can be handled and feasible of being logically organized? When I say "being logically organized" I think about on what it is used to be called "musical logic". But what define a musical logic?

Ask a person without musical knowledge to seat at the piano and try to play some keys for a while. Repeat later the same experiment with someone who owns music knowledge. Why do we think about anything but music when we listen to the first "player"? Why does the second playing will unmistakable sound, or will be recognized, as music? Answer: the first one lacks a system of rules, laws and constraints, while the second one owns such a system. The study and application of the rules, laws and constraints of music for centuries, has lead toward the creation of many sorted subsets of music elements and values, which we normally recognize as pieces of music.

Besides all the subjective factors which take part on the creation of a musical composition, among others: inspiration, intuition or experience, the musician selects objective "parameters" with which he designs, step by step, how will his piece of music be. Normally he will decide if he is going to use a polyphonic or homophonic texture, which will the structure be, for how many instruments, what kind of harmony, etc. In many cases he must have to keep himself inside a, more or less, strict set of constraints. For instance, he must not write a note for an instrument that cannot be played. Each instrument has a register or range of note values that can be played on it. In general, the study of music theory involves the study of numerous rules and restrictions. Let's see an example of a real musical rule, taken from a harmony book [2]:

***Linking of the tonic chord with the dominant and subdominant ones***

*Before trying the most simple links of chords, it is necessary to take into account certain general* ***rules****:*

1- *The common note to the chords we are trying to link (c-e-g  g-b-d) should remain on the same voice (...)*
2- *All the voices but the bass, should be conducted by joined grades as possible, avoiding the frequently use of walking by disjoint grades or melodic jumps.*
3- *The soprano and the bass voices should walk, with preference, by contrary movement.*
4- *The distance between the bass and the tenor voices can surpass sometimes the octave; but for obtaining a well balanced sonority it should be considered the octave as the maximum admissible distance between the soprano and the alto voices, and mainly, between the alto and the tenor voices.*

We have seen that music composers, on their creation work, handle lots of variables, each of them has its own domain of values. These values are restricted by a set of rules, which should be simultaneously satisfied. But this situation I have just described is actually what it is known as a *Constraint Satisfaction Problem*, a kind of problem studied as a part of one of the Artificial Intelligent disciplines: *Constraint Programming.*

## 2 Constraint Programming

*Constraint Programming* (CP) is the study of computer systems based on restriccions. The main idea of CP lies on solving problems by means of constraint statements (requirements) related to the area of such problems, and therefore, finding the solutions which satisfy all the restrictions.

*"Constraint Programming represents one of the closest approaches computer science has yet made to the Holy Grail of programming: the user states the problem, the computer solves it."* (E. Freuder, cited in [3])

The representation of a problem by means of constraints is usually very flexible, because they can be added, deleted or modified. The programming process has two general steps:

1) generation of one representation of the original problem as a CSP (*Constraint Satisfaction Problem*), and
2) search of a solution to this CSP.

A CSP can be defined as:

- a finite set of *variables* X = $\{x_1,...,x_n\}$,
- for each variable $x_i$, a finite set $D_i$ of possible values (its *domain*),
- a finite set of *constraints* which restricts the values that can simultaneously be assigned to the variables over its respective domains.

A *solution to the CSP* is the assignment of a value to each variable, such that all the constraints are satisfied at the same time. A CSP is *consistent* if it has one or more solutions. When solving a CSP, we would wish to find:

- any solution, no matter which is,
- every possible solution,
- an optimal solution, or at least a good solution, according to a given objective function defined in terms of some or all variables.

The solutions to a CSP can be found by systematically searching through possible assignments of values for each variable. Searching methods are divided into two wide groups:

1) those which pass through the partial solution space (or partial values assignment),

2) and those which completely explore the space of values assignment in a stochastic way.

The task of constraints programming is to reformulate the initial problem as a CSP, and to solve it by means of general methods or related to a certain particular domain. General methods are usually related with techniques to reduce the search space, and with specific search methods. The fundamental idea is to convert a given CSP into an equivalent one; that is to say, into another CSP which owns the same solutions set, but easier to solve. This process is known as *Constraints Propagation*. Algorithms for constraints propagation reduce the search space and, therefore, restrict the combinatory explosion [4].

On the other hand, specific domain methods are usually provided as algorithms for specific purposes or specialized packs, often known as *Constraint Solver*s. Some examples are:

- a program that solves systems of linear equations
- a package for linear programming
- an implementation of the unification algorithm, a cornerstone of automated theorem proving [4].

We have seen that musicians have always been solving constraint satisfaction problems for creating their musical work, though without giving it that name. As a scientific discipline, Program Restriction has actually formalized and developed the theoretical-mathematical-algorithmic basis of one of the several intellectual processes that take place in the human mind, making possible its simulation (programming) by means of computer systems of Artificial Intelligence, and the resolution of real-life problems in an automatic way.

The evidence about the possibility of  expressing music theories as a CSP, has lead some researchers toward the development of computer composition / harmonizing systems based on this formalism [5]. This theory has also been used for supporting automatic systems for musical analysis. In [6] the authors describe an interactive tool that uses constraint propagation inside an expert system for music composition, where traditional counterpoint is modeled as a CSP.

Another developed project has been implemented using a constraint programming language called Oz, and its name is COMPOzE. We would like to show, with this system, how we could apply the Constraint Satisfaction Problem theory, in practice, to algorithmic composition.

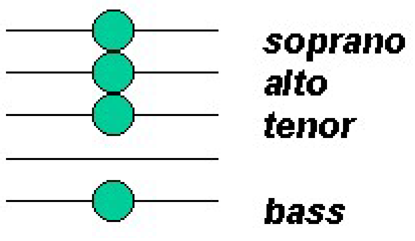
## 3 COMPOzE: Music Composition Through Constraint Programming

COMPOzE, developed by Martin Henz *et al.* ([7, 8]), is an interactive system based on Constraint Programming for the creation of simple four voices music pieces, giving a musical development scheme. This scheme (or plan) describes the harmonic flow and some features of the desired composition. The system shows a graphical user interface that lets the selection of musical rules through direct manipulation. The composition process, as well as the solutions, are represented graphically. The user can also listen to the solutions and compare it. Eventually COMPOzE generates a MIDI file.

The musical plan is made of a harmonic progression, for instance: **T**onic, **S**ubdominant, **D**ominant and **T**onic. It is also made of some parameters such as: tonality, tempo, rhythm, etc., and some features of the chords (major, minor, accidental notes, etc.). The chord sequence to be generated should obey the harmonic progression, as well as the programmed composition rules (constraints). These rules include restrictions for the notes (tessitura), for the chords (crossing prohibition of voices, bass distance), for the sequences (jump compensation, bass rhythms), etc. The following are real examples of these musical rules:

- **Crossing Prohibition:** The voices within one chord may not cross, in a sense that a lower voice may not play a higher note than a higher voice. For example, the bass may not play a higher note than the tenor within a chord.

- **Jump Law:** A jump of a voice from a chord to its neighbor that exceeds a given distance must be soothed in the following chord by a jump of one or two steps in the opposite direction.



**Chord**

In accordance with the initial hypothesis, the most suitable and natural framework for formalizing the composition of music is given by the constraint satisfaction theory. When representing this problem as a CSP, it would be expressed in the following way:

- In general we have $n$ x $v$ variables, where $n$ is the number of chords in the sequence, and $v$ the number of notes in each chord. In this case we have four voices by chord, namely: **B**ass, **T**enor, **A**lto and **S**oprano. We'll name the variables as follows: $B_i$, $T_i$, $A_i$, $S_i$, where $i \in \{1,\ldots, n\}$.
- The domain for these variables is given by a range of playable pitches.
- The harmonic functions and the composition rules can be formulated as constraints ruling between one or several variables. For example, the crossing prohibition can be expressed by the following constraint:
  $\forall i \in \{1,\ldots, n\}, B_i \leq T_i \leq A_i \leq S_i$

In a more formal way, the Constraint Satisfaction Problem could be expressed in the following terms:

**Variables and domains**

- 4 x $n$ variables $B_i$, $T_i$, $A_i$, $S_i$, $i \in \{1,\ldots, n\}$,
- with domain of integer values $\{0,\ldots,60\}$.

**Constraints:**

- Harmonic functions: $B_i$ mod 12 $\in \{0, 4, 7\}$
- Tessitura: $\forall i\ \ 0 \leq B_i \leq 20$
- Bass distance: $\forall i\ \ B_i + BassDist \leq T_i, A_i, S_i$
- Jump compensation:

$$\forall i\ \ S_i - S_{i+1} \geq jump \rightarrow S_{i+2} - S_{i+1} \in \{1, 2\}$$
$$S_{i+1} - S_i \geq jump \rightarrow S_{i+1} - S_{i+2} \in \{1, 2\}$$

The goal of Constraint Propagation is to progressively restrict the set of possible values the variables can get, applying the restrictions until eventually only one value for each variable is found. The set of possible values is kept inside a structure called *constraint store*. For instance, the fact that the base pitch of the first chord must be taken from the first 25 pitches of the scale is expressed by the constraint: $B_1 \in \{0,\ldots,24\}$ in the constraint store. More complex constraints are expressed by propagators, which observe the constraint store and amplify it if possible.

A propagator inspects the store with respect to a fixed set of variables. When values are ruled out from the domain of one of these variables, it may add more information on others to the store, i.e., it may amplify the store by adding constraints to it. As an example consider the crossing prohibition. It can be expressed for the first chord by installing the following three propagators:

```
B1 =<: T1 T1 =<: A1 A1 =<: S1
```

To explain how they can amplify the constraint store, let us assume that

$A_1 \in \{30,\ldots,45\}$ and $S_1 \in \{25,\ldots,60\}$

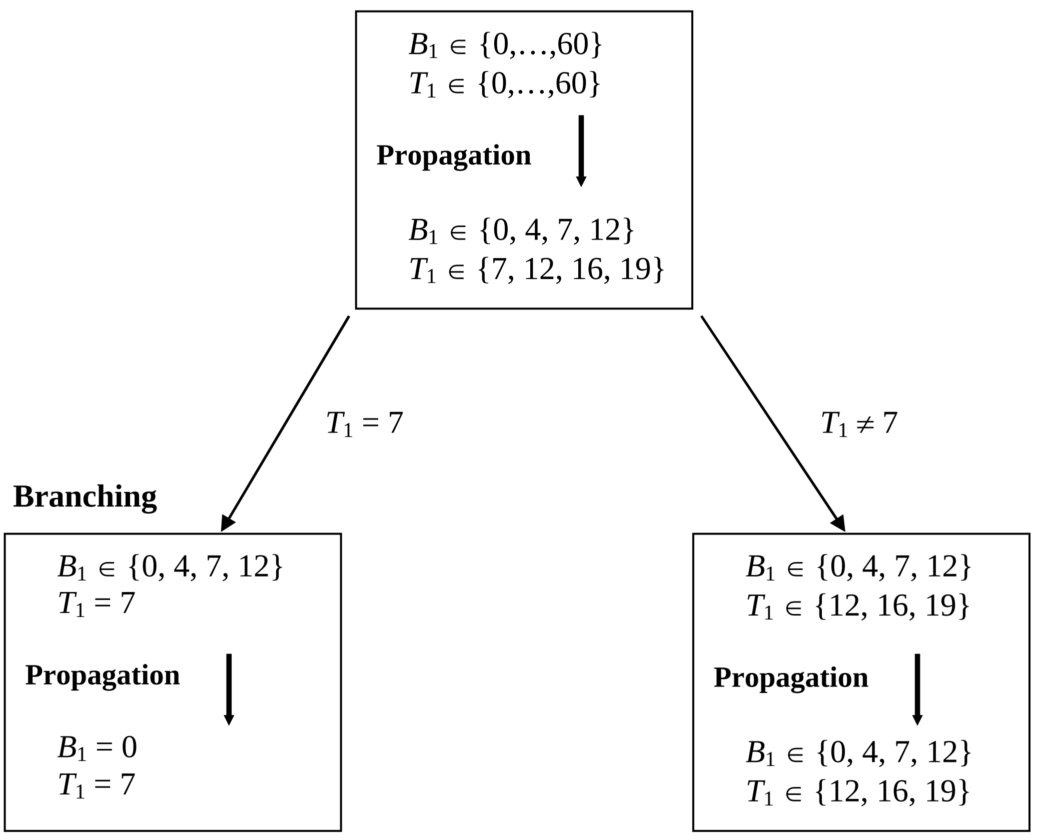
Then the third propagator will exclude the values 25,...,29 from the domain of $S_1$, reducing its domain to $\{30,\ldots,60\}$. Inversely, if we later know that $S_1 \in \{30,\ldots,40\}$, then $A_1$ will be restricted to the new domain $A_1 \in \{30,\ldots,40\}$. Note that this propagator remains active, waiting for more information on either $A_1$ or $S_1$. It only ceases to exist when it becomes clear that it will never amplify the store again. Let's see the following graphics:

**Initial constraints:**
$B_1 \in \{0,\ldots,60\}$
$T_1 \in \{0,\ldots,60\}$
$BassDist = 6$

$\downarrow$ **Propagation:**
$B_1$, $\mathrm{T}_1$ mod 12 $\in \{0, 4, 7\}$
$0 \leq B_i, \mathrm{T}_i \leq 20$
$B_1 + BassDist \leq T_1$

**Result of the propagation:**
$B_1 \in \{0, 4, 7, 12\}$
$T_1 \in \{7, 12, 16, 19\}$
$BassDist = 6$

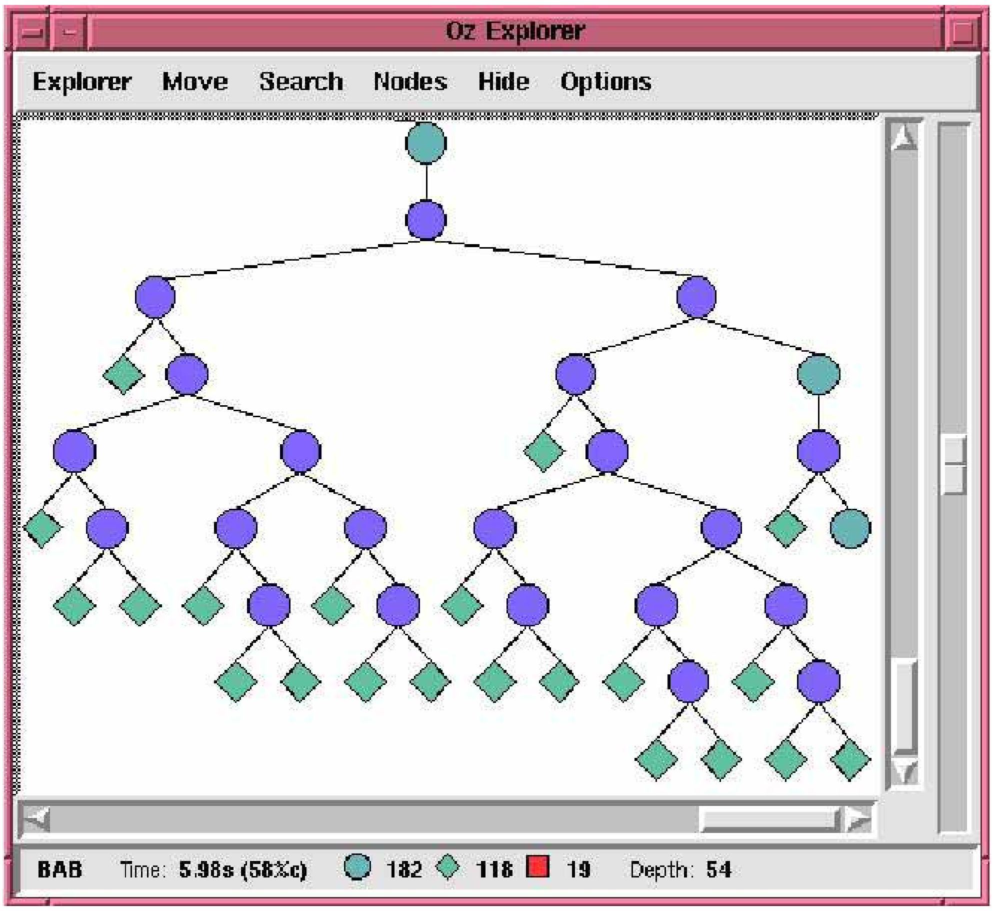
## SEARCHING



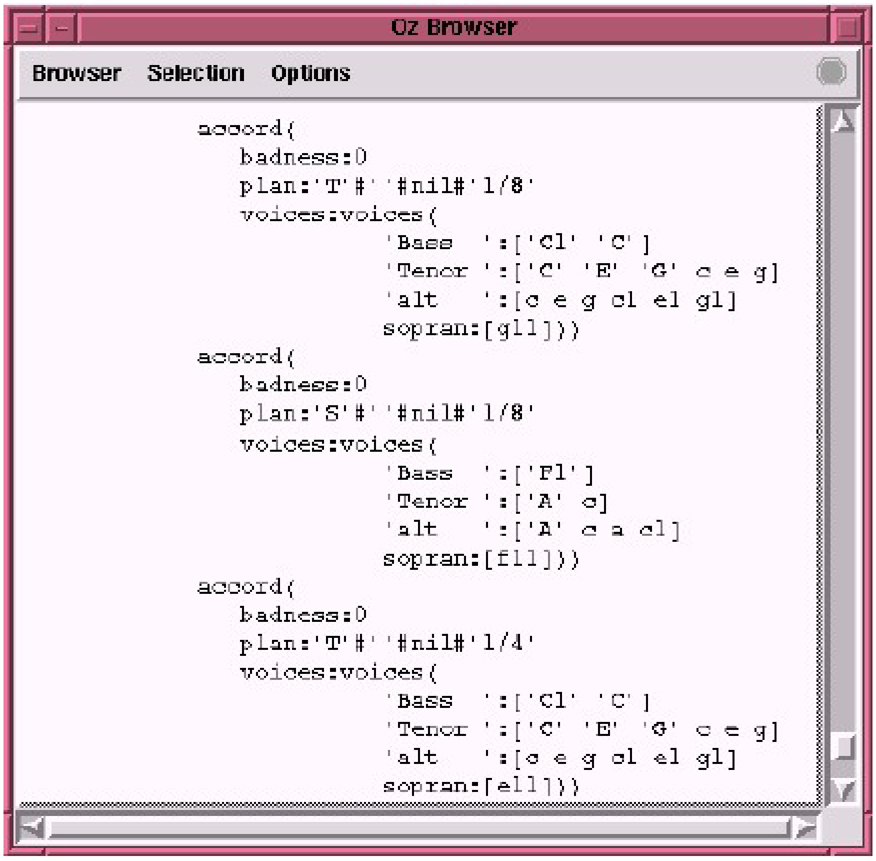
$B_1 \in \{0,\ldots,60\}$
$T_1 \in \{0,\ldots,60\}$
Propagation
$B_1 \in \{0, 4, 7, 12\}$
$T_1 \in \{7, 12, 16, 19\}$
$T_1 = 7$
$T_1 \neq 7$
Branching
$B_1 \in \{0, 4, 7, 12\}$
$T_1 = 7$
Propagation
$B_1 = 0$
$T_1 = 7$
$B_1 \in \{0, 4, 7, 12\}$
$T_1 \in \{12, 16, 19\}$
Propagation
$B_1 \in \{0, 4, 7, 12\}$
$T_1 \in \{12, 16, 19\}$

COMPOzE takes as input a musical plan, and gives as result one or more compositions, which are structured according to that plan. At the same time, they follow the defined user criteria. The system permits the user to decide, for each musical rule, if it should be ignored (`off`), strictly obeyed (`hard`), or preferable obeyed (`soft`), by means of a value (`weight`) between 0 and 100.

The implementation uses the *branch-and-bound* technique for minimizing the number of broken soft rules. If there are several of these rules, they are assessed by their weights. The Oz explorer shows the search tree while the user can, interactively, listen to the generated solutions by mouse click over the graphically represented solution nodes.



The Oz explorer shows the search tree



```
accord(
   badness:0
   plan:'T'#' '#nil#'1/8'
   voices:voices(
               'Bass  ':['Cl' 'C']
               'Tenor ':['C' 'E' 'G' c e g]
               'alt   ':[c e g cl el gl]
               sopran:[gl1]))
accord(
   badness:0
   plan:'S'#' '#nil#'1/8'
   voices:voices(
               'Bass  ':['Fl']
               'Tenor ':['A' c]
               'alt   ':['A' c a cl]
               sopran:[fl1]))
accord(
   badness:0
   plan:'T'#' '#nil#'1/4'
   voices:voices(
               'Bass  ':['Cl' 'C']
               'Tenor ':['C' 'E' 'G' c e g]
               'alt   ':[c e g cl el gl]
               sopran:[el1]))
```
A fragment of COMPOzE code inside the Oz browser

## 4 CONCLUSIONS

Musicians have used throughout centuries of music practice, composition rules which actually are in most cases restrictions or constraints. They allow navigating and reducing an enormous search space formed by a huge finite set of musical values. This kind of problem is quite similar to those solved daily by everybody in many areas of the human activity. They have been studied and formalized by an Artificial Intelligence discipline: Constraint Programming.

The idea of music composition modeling as a constraint satisfaction problem emerges in a natural way, when we intend to develop algorithmic music systems with the tools Artificial Intelligence offers nowadays. On one hand, that vision is supported on the natural relationship between music composition and constraint programming, and on the other hand, on the different research projects carried out in the past years, not only for music composition, but also for harmonization or analysis.

In this paper we have discussed the existing relationship between an eminently intellectual artistic activity, music composition, and its possible scientific formalization with theoretical and practical objectives. With one of the developed projects, the COMOzE system, we have exemplified how this formalization could be carried out, as well as the automatic solution to the proposed problem; that is, the automatic creation of a simple piece of music.

Some interesting subjects, such as the discussion of philosophical topics around the necessity (or not) of creating automatic systems for music composition, are beyond the scope of this text. Nevertheless, we think conveniently to say that the study of this kind of problem can be as important, to Artificial Intelligence research, as the automatic solution of games like the 15 puzzle, the Rubik cube, or the *n* queens problem.

## 5 REFERENCES

1- Carpentier, Alejo. *Crónicas*. Tomo I, Editorial Arte y Literatura. La Habana, Cuba, 1975.

2- Scholz, Hans. *Compendio de Armonía*. Editorial Labor S.A., Barcelona, 1951.

3- Barták, Roman. *Constraint Programming: In Pursuit of the Holy Grail*. http://kti.ms.mff.cuni.cz/~bartak/downloads/WDS99.pdf

4- R. Apt, Krzysztof. *Constraint Programming*. ERCIM News No.39 - October 1999. http://www.ercim.org/publication/Ercim_News/enw39/apt.html

5- Pachet, F. and Roy, P. 2001. "Musical harmonization with constraints: A survey". *Constraints*, 6(1):7-19. 2001. http://www.csl.sony.fr/downloads/papers/2000/pachet-constraints2000.pdf

6- Ovans, Russell and Rod Davison. "An interactive Constraint-Based Expert Assistant for Music Composition". *Proceedings of the Ninth Canadian Conference on Artificial Inteligence*, pp. 76-81. http://citeseer.nj.nec.com/ovans92interactive.html

7- Henz, Martin, Stefan Lauer, and Detlev Zimmermann. "COMPOzE --- Intention-based Music Composition through Constraint Programming". *Proceedings of the 8th IEEE International Conference on Tools with Artificial Intelligence*, Nov16--19 1996, IEEE Computer Society Press. ftp://ftp.ps.uni-sb.de/pub/papers/ProgrammingSysLab/COMPOzE96.ps.gz

8- Henz, Martin. *COMPOzE Intention-based Music Composition through Constraint Programming*. http://www.comp.nus.edu.sg/~henz/talks/tai96/