MUSAS: A Program for Analysis and Synthesis of Musical Spaces

Gilberto Carvalho, agmc@dedalus.lcc.ufmg.br
Hudson Lacerda, hfml@dedalus.lcc.ufmg.br
Escola de Música da Universidade Federal de Minas Gerais

Abstract

The main purpose of this paper is to describe the *MUSAS* program, which is a environment that aims to conjugate the Forte's theory of the pitch-class sets with newly created musical spaces. In the introduction are the main forces, under our point of view, that drove us to project and to implement such a king of program. Following, there is a section on the fundamentals of the program theoretical background (pitch-class sets and space generation). In third place, we talk about the implementation and, to finish, there is a conclusion where we explain some critical considerations on the project up to its current state.

1 Introduction

The *MUSAS* (Musical Space Analyzer and Synthesizer) program is a tool for musical analysis and composition that works with two main approaches: the creation of musical spaces and their analysis by the pitch-class set theory (from now on called PCT). Besides the applications mentioned above, the program intends to have didactic ends on teaching post-tonal theory as well as in other disciplines where its use would also be possible. Nowadays, there are several programs that accomplish similar tasks [1].

A crucial subject that is directly related to the development of a music composition program is the use of discrete spaces in contemporary music, specifically in music using computers. Since the end of the forties on the aesthetics today known as

---

[1] Some of these programs are: PCN (Jamary Oliveira, UFBa), SOLOMON (Larry J. Solomon, Soft Stuff Computer, P.O. Box 3385, Tucson, AZ 85722) and MOD12 (Thomas R. Demske, tdemske@msn.com).

eletroacoustic music has provided to the composer a new vision and a new perception of the musical space that is based, mainly, on timbre, that is, on a continuous variation of the components of the well-known physical phenomenon called timbre. In spite of the contribution of then new means to approach musical spaces, in the early days, those approaches deviated the musicians' attention from extremely interesting possibilities of organization to be developed on the new machines. Such reaction is comprehensible due to the internal and external crises of the main system of organization of the musical space known at that time as the serial system. Nevertheless, it is noticed today that the intersection of these two extreme possibilities of treatment of the space (discrete and/or in a continuous way) presents a richness that will be difficult to be reached if one used only one kind of approach. Besides, by its conjoint treatment, it becomes possible for us to take advantages of a quantitative control combined with a qualitative control of the musical space.

Such reflections drove us to design a program that could accomplish analyses as well as syntheses of discrete spaces with musical applicability. One of the properties of the program is to allow the manipulation of complex spaces by the PCT tools, which also facilitates (besides the applications specifically musical) new perspectives of a theoretical point of view.

This paper is divided in four parts: this introduction; a section about PCT, space synthesis and the application of the first on the new generated spaces; a section on the *MUSAS* implementation and a conclusion.

2 Theoretical background

For the understanding of the program it is necessary to know PCT's (also known as post-tonal theory[2]) basic notions and concepts. With this in mind, we will briefly review this theory's foundations.

An important concept in PCT is that of *equivalence classes*. These can be defined as, being considered for all $a \in A$:

$$[a] = \{x \in A \mid xRa\},$$

[2] References on post-tonal theory are (Forte 1973) and (Straus 1990).

where $[a]$ it is the set of the equivalent elements to the element a (equivalence class), A is the universe set, x represents each element of the universe set and R represents the *equivalence relationship*[3].

The equivalence relationship employed for pitches can be defined in the following way:

$$x \equiv y \leftrightarrow x \bmod M = y \bmod M$$

where x and y are pitches and M is the module.

For instance, considering the module 12, the sets of pitches {60, 64, 63} and {84, 88, 87} are equivalent and represented both by the set {0, 4, 3}. Notice that the latter is now a set of pitch-classes representatives (from now on the expression "pitch-class sets" will be referred, to this kind representation).

The concept of equivalence classes can also be applied to pitch-class sets. Pitch-class sets that are equivalent under transposition and/or inversion belong to the same class.

The transposition of a pitch-class set to certain interval consists of the addition of a fixed value to each element of the set:

$$T(a,i) \leftarrow a + i$$

On the other hand, the inversion is a process composed of two stages: calculation of the classes of inverse pitches and its transposition to an interval. The inverse class can be defined as:

$$a' \leftarrow M - a$$

where a' is the inverse class of the pitch class a and M is the module. Thus, the operation of inversion can be defined this way:

$$I(a,i) \leftarrow T((M - a),i)$$

For instance, in module 12, the set A = {11, 7, 8} will always belongs to the class of B = {0, 4, 3}, because it is possible to accomplish the mapping of these two sets through the inversion operation:

$$I(A,11) = B$$

[3] The notation employed is as follow: [ ] $\rightarrow$ equivalence class and { } $\rightarrow$ set.

To compare pitch-class sets, it is necessary that their elements are organized in certain way. There are three kinds of organization: the *ascending ordering* inside of the module; the *normal form* (the intervals are packed recursively to the left) and the *prime form* (the intervals are packed recursively to the left after what the set is transposed in order to begin with zero)[4].

Having approached the pitch-class sets focusing on their elements, we will now approach the interval relationships between them. A definition of *interval* becomes necessary.

An interval between two pitch-classes *a* and *b* are defined as the absolute value of the difference between them ($|a-b|$). The classes of interval equivalence are defined for the relationship:

$$i \equiv M - i$$

where *i* is the interval between the pitch-classes and *M* is the module. As we can see, for each module, there will be $\left\lfloor \frac{M}{2} \right\rfloor + 1$ classes of interval equivalence.

The interval content of a pitch-class set can be summarized by an *interval vector*. This vector has $\left\lfloor \frac{M}{2} \right\rfloor$ positions, because the intervals of the class [0] are not considered (intervals between pitches of the same class). Each position of the interval vector contains the number of intervals occurrences of the corresponding class present in the set. The first position indicates the number of intervals of the class [1], the second position, the number of intervals of the class [2], and and so on.

Pitch-class sets can still maintain several other relationships: *inclusion* (involving subsets and supersets), *complement* (considering the universe set), *union*, *intersection*, *difference*, and so on. Besides these, there are operations involving ordered sets (i.e., twelve-tone row).

Another possibility offered by the program is the synthesis of musical spaces. These can be of three kinds: straight, curved or

---

[4] Because in a set class there are two normal forms beginning with zero (one in the original form and another inverted), the prime form will be that one more packed to the left.

irregular [5].

Space		Characteristics
straight	$\rightarrow$	maintains in all its extension a module and a fixed unit of analysis.
curve	$\rightarrow$	has a module and/or unit of analysis varying regularly in all its extension.
irregular	$\rightarrow$	statistical distribution of frequencies. It can have a module, but it doesn't have a defined unit of analysis.

For the synthesis of straight spaces *MUSAS* uses an elaboration of a Pierre Barbaud’s relationship mentioned in (Brown 1982) [6]:

$$S_{i,j} = M^{j} . M^{\frac{i}{N}}$$

where S is a two-dimensional matrix that stores the calculated space, i is the index of each element of each instance j of the space, M is the module [7] and N is the number of divisions.

There are two possibilities for the calculation of curved spaces. It is possible to calculate a space between two reference points, which derives conceptually from the usual calculation of straight spaces. For this possibility is used a elaboration of the (Moore 1990)'s relationship:

$$A_i = v1 + (v2 - v1)\frac{1 - e^{(\frac{i}{N-1})\alpha}}{1 - e^{\alpha}}$$

where A is an array that stores the calculated values, i it is the index of each element, $v1$ and $v2$ are, respectively, the initial and

[5] The classification presented here is based on (Boulez 1963).

[6] The Brown's expression is:

$$F = 2^{k} . 2^{\frac{n}{12}},$$

where F is frequency, k is the module instance, n is the index of the element inside the module. The constants 2 and 12 are, respectively, the module and its number of internal divisions.

[7] In this case, the module is a proportion between frequencies. For instance, a module equal to 1.5 corresponds to an interval of 7 half-tones (perfect fifth).

final values of the interval, N it is the resulting number of elements of the interpolation and α it is the interpolating coefficient[8]. In the implementation is avoided the division by zero that could happen in the linear interpolation.

The other possibility concerns the calculation of a composed curved space, which can be calculated with a derived relationship of the previous one:

$$M_{i,j} = v_j + (v_{j+1} - v_j)\frac{1 - e^{\frac{ix_j}{N-1}}}{1 - e^{x_j}}$$

where M is two-dimensional matrix, i is the index of each element of each space, j is the index of each space, v is a breakpoints array (initial and final points for each portion of the composed space), N is the number of elements of each space and x is an array of interpolating coefficients, different for each space. Of course is also open the possibility of doing a number of different internal divisions for each space, being, in this case, the spaces with less elements completed with zeros in the matrix M.

Finally, with concerning the irregular spaces, these are supplied directly by the user or calculated automatically through different distributions. After defined they can be treated with the same tools of the other spaces.

It is important to emphasize that ambiguity can exist among the kinds of space described above. For instance, a curved space whose curvature is small, or an irregular space whose pitches are distributed by approximately the same intervals between them will be noticed as straight spaces. On the other hand, a musical space based on a straight space can give the illusion of an irregular space, if the used intervals are, in a proportional way, very different. The musical context can be decisive about the way a musical space is perceived.

The pitch-class sets theory, formalized by (Forte 1973), is based on the temperate conventional space module 12 (with the half-tone as

---

[8] The interpolating coefficient defines the curve (by its signal) as well as the curvature degree (by its absolute value) that will be used in the interpolation.

unit of analysis). However, the same theoretical tools can be applied –at least partially– to other spaces, temperate or not. In the straight spaces PCT can be applied integrally only guarding against to the properties owed to the interval symmetries (a module divided in an odd number of parts, for instance). In respect to the curved spaces, the main problem turns around the transposition and inversion operators, considering that they will give results distorted by the curvature of the considered space. Finally, concerning the irregular spaces, the applicability will depend on the definition degree of the space. For each one of these three types of no-conventional spaces, there are inside *MUSAS* tools adapted to their treatment and practical use, including purely instrumental music.

3 IMPLEMENTATION

The *MUSAS* program was conceived as a set of windows that was capable of accomplishing a wide range of analysis operations and synthesis of musical spaces, such as described in the previous section. In order to accomplish this we thought of four kinds of windows:

- Main window
- Data window
- Operations window
- Output window

The main window is a typical MID (Multiple Document Interface) window from which all the another are called from menus or shortcut keys. The data windows are text windows in which the user files the results of the operations accomplished in work sessions. They are children of the main window. The operations windows are windows in which the user accomplishes the operations needed to a project. There are five kinds of operations windows:

- Unitary operations (involving one set)
- Binary operations (involving two sets)
- Operations on orderly sets
- Automatic analysis operations
- Space synthesis operations

The operations windows are modeless dialogs with own icons and without "parent-like" relationship with the main window, what allows a great independence in their manipulation. They consist of a set of radiobuttons and/or checkboxes (by means the user selects which kind of operation he/she wants to accomplish) to the which are added edition fields that facilitate the input of the selected operation parameters. After the calculation the result is available in the output window (see below). Among the characteristics of the operations windows are the possibility of the audition of the set(s) and/or space(s) on which they act and the possibility of coexistence of several windows of the same kind of operation (for instance, the user can work simultaneously with two or more unitary operations windows).

The output window is a read- only text window that receives the results of the calculations accomplished by the operations windows. Those results that are important to a project should be copied to a data window and saved. This is necessary because the output window, acting theoretically as a infinite empty page, doesn't have capacity of data storage.

One of the needs of the program showed was the elaboration of a help file that could be as complete as possible, since the program, besides to have a complex operation aims also to didactic ends. Considering this last point, the help file was idealized in three parts:

- Theory: in this section PCT and the algorithms for space synthesis are presented in an interactive way (through links).
- *MUSAS*: in this section the operational characteristics of the program are presented accompanied of a tutorial seeking for the fast learning of *MUSAS*.
- Analysis of works: in this section musical works are analyzed employing the *MUSAS* tools.

It is worthy of note that all the parts of the help file are illustrated with sound musical examples (by means of wave files as resources in a help file DLL) and that, besides, they have links one another in several levels of depth.

The basic use of the MUSES is summarized by the configuration of a project (a desktop) consisting of the data windows as much as the operations windows (and their private data). Considering that there is the possibility of saving and loading a project, this process makes easy that a work (a musical analysis, for instance) be always retaked at the point where its last session was concluded.

4 Conclusion

We hope to have shown with this text the theoretical foundations and the main possibilities of the program that we are developing. The current state of the project counts three operation windows implemented (besides, naturally, the main windows and output window) and the help file structured and partially accomplished.

Possibilities for future versions include the implementation of new kinds of output that can supply the user with graphical results of the operations. Other possible amplifications of the program are operations on lists of sets and new forms of space generation.

As consequence of the implementation and practical use of the program, we can notice the need of new theoretical reflections on the applicability of PCT (mainly in the modeling and analysis of new kinds of spaces), and on the development of analytic processes that could be efficiently automated.

5 References

Boulez, P. (1977). *Penser la musique aujourd'hui*. Paris: Gonthier.

Brown, F. (1982). *La musique par ordinateur*. Paris: Presses Universitaires de France.

Forte, A. (1973). *The structure of atonal music*. New Haven: Yale University Press.

Moore, F. R. (1990) *Elements of computer music*. New Jersey: Prentice- Hall.

Straus, J. N. (1990) *Introduction to post- tonal theory*. Englewood Cliffs: Prentice Hall.