Representing Musical Hierarchies with Constraints

Anthony Beurivé, Myriam Desainte-Catherine*

Abstract

In this paper, we study the impact of editing operations on the values of variables characterizing musical structures. Simple formal models of hierachical musical structures will be proposed for this purpose. Those models use constraints to distinguish between consistent and inconsistent musical structures. Consequently, the computed values have to satisfy these constraints.

1 Introduction

Today's software for composing music often use structured representations of musical pieces, like tracks or sequences. Several papers generalize or extend these representations by proposing formal hierachical models [2, 4]. In order to specify various musical structures, those models use temporal constraints in an explicit or implicit way. But they have never been used in a dynamic context where interactive editions occur. The problem with such editions is that they require efficient updates of the temporal attributes in a way that is consistent with the constraints.

This looks like a contradiction, because dealing with constraints in music is generally considered a very demanding task. Indeed, constraints have already been used for music composition. Satisfaction techniques, for example, have been used in automatic harmonization and counterpoint problems [8, 13, 3, 11], and proved to require a large amount of time and space. Yet, one cannot really consider these techniques in interactive editions of musical structures. Research focused on constrained interactive edition can be better found in multimedia [9, 12] and spatialization [10].

In this paper, we study the impact of editing operations on the values of variables characterizing musical structures. Simple formal models of hierachical musical structures will be proposed for this purpose. Those models use constraints to distinguish between consistent and inconsistent musical structures. Consequently, the computed values have to satisfy these constraints.

In most cases, though, we are faced with under-constrained structures, such that there may be still many solutions in the search for new values. To deal with this situation, we propose to use optimization techniques. A distance has to be minimized, which is parameterized by a function giving weights to variables. The weights are adjusted to fit a given editing operation and its awaited result.

Two absolute models and one relative model will be defined for musical structures. In the absolute models, the structures are characterized by variables representing their durations and absolute dates. In the relative model, the variables represent durations and relative offsets. It will be demonstrated that in the absolute models, an editing operation as simple as the assignment of a new starting date may involve the update of many other variables just to satisfy the constraints. This is generally not the case in the relative model.

In section 2, we present the notations and definitions used throughout the paper. In section 3, we survey the edition of musical structures. In sections 4, 5 and 6, the three models of musical structures are defined and studied. In section 7, we talk about existing implementations of this work. The conclusion appears in section 8.

2 Notations and Definitions

In this section, we introduce some notations and definitons about musical hierarchies with constraints.

*SCRIME, LaBRI, Université Bordeaux 1, myriam@labri.u-bordeaux.fr

2.1 Constraints and Hierarchy

Since a hierarchy is defined as a tree, we consider different kinds of constraints according to the variables they bind within a hierarchy.

- *Intrinsic constraints* concern variables of a given node of the hierarchy.
- *Hierarchical constraints* involve variables of a node and variables of its children.
- *User constraints* may concern variables of any nodes.

We consider the following subsets among hierarchical constraints.

- *Parent-child constraints* are constraints between the variables of a parent and the variables of its children; they provide a way to specify, for example, the temporal position of a child within its parent, independently from siblings.
- *Sibling constraints* are constraints between the variables of siblings; they provide a way to specify, for example, articulations between notes in a melody.

Constraints will be denoted by logical expressions. In those expressions, predicate symbols like $=$ and $\leq$ apply to arithmetic expressions involving variable symbols and numeric constants.

2.2 Musical Structures

A *musical structure* (or *musical hierarchy*) m is defined as a tuple of five elements

$$(A,\ (m_1,\ \ldots,\ m_k),\ C_i,\ C_h,\ C_u)$$

where

- A is a set of symbols naming local variables of m,
- $(m_1,\ \ldots,\ m_k)$ is a list of substructures where, for each $i \in [1, k]$, m_i is a musical structure,
- C_i is a set of intrinsic constraints,
- C_h is a set of hierarchical constraints,
- C_u is a set of user constraints binding variables of nodes in the subtree of m.

Local variables of different structures will be used in the same constraints. In order to admit the same symbol s to denote local variables in different structures m_1 and m_2, we use the notation $s(m_1)$ and $s(m_2)$.

A *basic musical structure* is a musical structure with an empty list () of substructures. It implies that its set of hierarchical constraints is empty, too.

In this paper, we only consider musical structures which are trees: structures may have at most one parent, and cannot be part of their substructures. One possible extension concerns directed acyclic graph-like structures, where structures may have more than one parent.

Various kinds of musical structures will be defined in the following, each one representing a common musical object. They will be based on generic sets of variables, intrinsic constraints and hierarchical constraints.

2.3 Representation of time

We shall use the notion of a *time line* on which dates and durations are based. For our concern, a time line is defined as the set of real numbers, and a date or a duration is such a number. The time unit is chosen arbitrarily.

2.4 Models of Musical Structures

A model of musical structures is a set of musical structures, all based on a generic definition specifying common variables and constraints. In this paper, each time a model has to be defined, we give the generic definition of the structures it contains.

We consider two types of models of musical structures that are based on different time representations.

- The *Absolute Models* containing a single implicit time line on which every date is based.
- The *Relative Models* in which there exists an implicit time line for each hierarchical musical structure. Each substructure is associated to an occurrence date on the time line of its parent.

In the following of this paper, we shall compare absolute and relative models by means of variable updates resulting from an edition. We show that absolute models imply a larger amount of variable updates than relative models, due to the fact that they have to maintain absolute dates. This happens everytime a structure is moved or inserted.

On the contrary, in absolute models, it is possible to determine the temporal constraints between two musical structures just by comparing the values of their starting and ending dates, whatever their positions in the underlying hierarchy. This is not possible in the relative models, where musical structures are defined relatively to their parents, and not relatively to a single global time line. In those models, a date is only valid within a local time line. Thus, in order to use dates from different time lines, it is necessary to convert them by means of (sums of) occurrence dates of ancestors, up to the first common ancestor. This kind of operation occurs every time musical structures that are not siblings are synchronized.

2.5 Valuated Musical Structures

So far, only syntactical objects have been defined. Those objects have to be associated to a valuation giving values to all the temporal variables. We introduce the following definitions.

- The *set of variables* of a musical structure m is denoted by $V(m)$ and is the union of its set of local variables with the sets of variables of its children, recursively.
- The *constraint graph* of a musical structure m is denoted by $G(m)$ and is defined as the conjunction of its local constraint graph and the constraint graphs of its children, recursively.
- A *valuation* of a musical structure m is a mapping associating to each variable in $V(m)$ a value in its domain.
- A *consistent valuation* of a musical structure m is a valuation of $V(m)$ satisfying the constraint graph $G(m)$. The *set of consistent valuations* of a structure m is denoted by $\mathcal{V}(m)$. It may happen that the set of consistent valuations of a structure m is empty.
- The *distance* Δ between two valuations ν_1 and ν_2 of a set of variables V is defined as

  $$\Delta(\nu_1,\ \nu_2) = \sum_{x \in V} w(x) \times |\nu_1(x) - \nu_2(x)|$$

  where w is a mapping associating a weight to each variable in V.
- A *valuated musical structure* is a pair (m, ν) where m is a musical structure and ν is a valuation of m.
- A *consistently valuated musical structure* is a valuated musical structure (m, ν) where ν is a consistent valuation of m. Those structures will also be called *consistent structures*.

3 Edition of Musical Structures

This section introduces modifications of valuated musical structures that may occur during the edition of musical structures. These modifications may imply modifications of valuations and modifications of structures.

For that purpose, we introduce a set of hierarchy operations. Then, we specify which computations these operations imply. Obviously, we consider that the result of these operations should be consistent structures.

3.1 Hierarchy operations

We consider the following kinds of operations.

- structural operations:
  - building basic elements (leaves of the hierarchy)
  - building higher-level valuated musical structures containing other valuated musical structures.
- assignment of node variables.
- operations on constraints.

These operations may produce inconsistent structures. In those cases, the structures become consistent by the application of a resolution step, as defined in subsection 3.2.

Building Basic Elements

It is necessary to be able to build consistent structures with no substructures, like notes or empty structures. At creation time, these structures are considered leaf and root of their own hierarchy. Later on, they will become leaves or nodes of more important hierarchies.

Building Higher-level Musical Structures

Higher-level structures are build with existing consistent structures. Because of the use of hierachical constraints, a modification in the sequence of children may lead to modifications and inconsistencies in the constraint graph, leading to an inconsistent structure. These operations include:

- adding a structure as a child of another structure,
- removing a child from a structure,
- building a new structure from a sequence of children.

Assignment Operations

The assignment of a value a to a variable t in a given consistent structure (m, ν) gives a valuated structure (m, ν') where

$$\forall x \in V(m),\ \nu'(x) = \begin{cases} \nu(x),\ x \neq t \\ a,\ x = t \end{cases}$$

Multiple assignment is defined as the assignment of a sequence of variables with a sequence of values. It is thus possible to assign several variables by a single operation.

Operations on Constraints

Operations on constraints consist in adding or removing constraints in a consistent structure. Thus, when such an operation is applied to a structure (m, ν), it gives a valuated structure (m', ν) where ν may not satisfy the constraint graph of m', leading to an inconsistent structure.

3.2 Computation Steps

The application of an operation on a consistent structure (m, ν) implies one or two steps of computation.

The first step consists in computing the raw result (m', ν') of the application. This step is called the *application step*.

The second step consists in computing a consistent structure (m', ν'') from the raw result. This step is called the *resolution step*. It only occurs if ν' is not consistent. The valuation ν'' is computed as a consistent valuation with minimal distance from ν'.

$$\nu'' \in \{\mu \in \mathcal{V}(m') \mid \forall \mu' \in \mathcal{V}(m'),\ \Delta(\mu, \nu') \leq \Delta(\mu', \nu')\}$$

This signifies that ν'' is one of the consistent valuations of m' which are closest to ν'. There may be one, several, or none of these valuations. If there is none, the structure m' has no consistent valuation at all, and the only acceptable solution might be to forget about (m', ν') and get back to (m, ν).

The way inconsistencies are handled depends on the operation that generated them. For an assignment operation, a general criterion in the search of ν'' is that it should admit the assigned values, unless there is no such consistent valuation. For other operations, there is no such a clear and general criterion. Several solutions exist, though, depending on each particular case whose description is beyond the scope of this paper.

Distance and Weight

As explained above, the distance between two valuations of a given structure plays a fundamental role in finding a consistent valuation during the resolution step. The weight function is the main parameter in the definition of distance. By modifying the weight function, it is possible to obtain different results from the resolution step.

Intuitively, if the weight of a note duration is greater than the weights of its starting and ending dates, the resolution has to change the position of the note rather than its duration. If it cannot change the position of the note due to some constraints, then it may change its duration. Moreover, if the duration of a note has to be changed as a consequence, for example, of an assignment operation, the weight function could be so that minimizing the distance leads to changing the ending date of the note rather than its starting date.

4 Absolute Model AM1

The absolute model AM1 is the most general one. It contains minimum of information and constraints. In this section, we first define the model AM1 and we provide simple examples of structures and editions. Then, we show that operations in AM1 do not provide commonly expected results in a musical context. This leads to the definition of the model AM2 in the next section.

4.1 Definition

In AM1, a generic musical structure m is defined by

$$(\{s,\ e,\ d\},\ (m_1,\ \ldots,\ m_k),\ C_i,\ C_h,\ C_u)$$

containing

- 3 variables s, e and d which domain is the time line; the variable s represents the starting date of m, e represents its ending date, and d represents its duration,
- a sequence of k substructures $m_1, \ldots, m_k$,
- a set of intrinsic constraints specifying the relations between the dates and the duration of m, and the fact that the duration is always positive,

$$C_i \quad = \quad \{s + d = e,\ 0 \leq d\},$$

- a set of hierarchical constraints specifying that the substructures occur during their parent

$$\begin{array}{rcl} C_h & = & \{s \leq s(m_i) \mid i \in [1,k]\} \cup \\ & & \{e(m_i) \leq e \mid i \in [1,k]\}. \end{array}$$

4.2 Example

Let m_1 and m_2 be the following musical structures.

$$\begin{array}{rcl} m_1 & = & (\{s, \ ,e \ ,d \ \}, (), \{s+d=e, \ 0 \leq d\}, \emptyset, \emptyset) \\ m_2 & = & (\{s, \ ,e \ ,d \ \}, (m_1), \{s+d=e, \ 0 \leq d\}, \{s \leq s(m_1), \ e(m_1) \leq e\}, \emptyset) \end{array}$$

The structure m_1 is a basic musical structure. The structure m_2 contains m_1 as a substructure. The hierarchical constraints in m_2 state that m_1 occurs during m_2.

Associating m_2 to the following valuation ν gives a valuated musical structure (m_2, ν), which happens to be a consistent structure.

Starting Dates	Ending Dates	Durations
$\nu(s(m_1)) = 10$	$\nu(e(m_1)) = 35$	$\nu(d(m_1)) = 25$
$\nu(s(m_2)) = 5$	$\nu(e(m_2)) = 40$	$\nu(d(m_2)) = 35$

4.3 Edition

Let us consider an assignment operation changing the starting date of m_2. This assignment operation is interpreted here as a temporal move of m_2. In such an operation, the duration of m_2 should not change. We show the consequences on m_1.

The new valuation ν' after the application step is the following. The bold value represents the assigned value.

Starting Dates	Ending Dates	Durations
$\nu'(s(m_1)) = 10$	$\nu'(e(m_1)) = 35$	$\nu'(d(m_1)) = 25$
$\nu'(s(m_2)) = \mathbf{7}$	$\nu'(e(m_2)) = 40$	$\nu'(d(m_2)) = 35$

This valuation is not consistent. The unsatisfied constraint is the intrinsic constraint

$$s(m_2) + d(m_2) = e(m_2).$$

Consequently, a resolution step is required. One of the consistent valuations of this structure has to be chosen. In order to reach the awaited result, the weight function has to be properly computed. Let us indicate some criterions used in this computation.

According to the general criterion given in subsection 3.2, the weight of $s(m_2)$ should be stronger than the weight of the other variables.

$$\forall v \in V(m_2) \backslash \{s(m_2)\}, \ w(v) < w(s(m_2))$$

There still exist several valuations satisfying the general criterion, so that it is necessary to use additional criterions. Moving m_2 implies that its ending date is decayed the same amount than its starting date, while its duration is unchanged. Consequently, the weight of $d(m_2)$ should be stronger than the weight of $e(m_2)$.

$$w(e(m_2)) < w(d(m_2))$$

We can choose arbitrary values for the weights, since they satisfy the previous criterions. Thus, let us substitute the values of ν' in the formula of the distance between ν' and ν''.

$$\begin{array}{rcl} \Delta(\nu', \nu'') & = & 1 \times |10 - \nu''(s(m_1))| + \\ & & 1 \times |35 - \nu''(e(m_1))| + \\ & & 1 \times |25 - \nu''(d(m_1))| + \\ & & 3 \times |7 - \nu''(s(m_2))| + \\ & & 1 \times |40 - \nu''(e(m_2))| + \\ & & 2 \times |35 - \nu''(d(m_2))| \end{array}$$

Minimizing this expression gives the unique following consistent valuation, whose distance with ν' is 2.

Starting Dates			Ending Dates			Durations		
$\nu''(s(m_1))$	$=$	10	$\nu''(e(m_1))$	$=$	35	$\nu''(d(m_1))$	$=$	25
$\nu''(s(m_2))$	$=$	**7**	$\nu''(e(m_2))$	$=$	**42**	$\nu''(d(m_2))$	$=$	35

It is clear that this distance is minimal, since only the fifth term is different from zero and it carries the smallest weight.

5 Absolute Model AM2

In music, the possibility to move a substructure together with its parent is an interesting operation. For example, in the case of a melody, it is important that the notes move together with the melody as the result of the resolution step.

In the previous example, it appears that the structure m_1 does not move with m_2, though it is one of its components. The variables of m_2 would only change if one of the constraints binding them was unsatisfied. For example, if $s(m_2)$ was given a value of 12 which would unsatisfy the hierarchical constraint

$$s(m_2) \leq s(m_1)$$

then it would necessitate to change $s(m_1)$ and maybe $d(m_1)$ or $e(m_1)$ (giving appropriate weights for such changes to occur).

We propose now another model called AM2 which is an extension of AM1 and such that moving a structure implies moving its substructures. For that purpose, it is necessary to define new variables and new constraints.

In a structure, each substructure is associated to a new variable, called its occurrence date in its parent, which represents the difference between its starting date and the starting date of its parent. This variable is local to the parent, not to the child.

This new variable is given a weight which will be taken into account during the resolution step. If this weight is properly adjusted, it can imply that the child is moved (i.e. its starting and ending dates are changed) while its parent is moved.

5.1 Definition

In AM2, a generic musical structure is defined by

$$(\{s,\ e,\ d,\ s_1,\ \ldots,\ s_k\},\ (m_1,\ \ldots,\ m_k),\ C_i,\ C_h,\ C_u)$$

with

- $k+3$ variables, the first three ones as in AM1, the other ones being the occurrence dates of the substructures,
- a sequence of k substructures $m_1, \ldots, m_k$,
- a set of intrinsic constraints

$$C_i \quad = \quad \{s + d = e,\ 0 \leq d\},$$

- a set of hierarchical constraints including the constraints defining the occurrence dates of the substructures,

$$\begin{aligned} C_h \quad = \quad & \{s \leq s(m_i) \mid i \in [1,k]\} \ \cup \\ & \{e(m_i) \leq e \mid i \in [1,k]\} \ \cup \\ & \{s_i = s(m_i) - s \mid i \in [1,k]\}. \end{aligned}$$

5.2 Example

Let us consider again the structures m_1 and m_2, this time in AM2.

$$
\begin{aligned}
m_1 &= (\{s,\ e\ ,d\}, (), \{s+d=e,\ 0 \leq d\}, \emptyset, \emptyset) \\
m_2 &= (\{s,\ e\ ,d,\ s_1\}, (m_1), \{s+d=e,\ 0 \leq d\}, \\
&\quad \{s \leq s(m_1),\ e(m_1) \leq e,\ s_1 = s(m_1) - s\}, \emptyset)
\end{aligned}
$$

Associating m_2 to the following valuation ν gives a consistent structure (m_2, ν).

Starting Dates	Ending Dates	Durations
$\nu(s(m_1)) = 10$	$\nu(e(m_1)) = 35$	$\nu(d(m_1)) = 25$
$\nu(s(m_2)) = 5$	$\nu(e(m_2)) = 40$	$\nu(d(m_2)) = 35$
$\nu(s_1(m_2)) = 5$		

5.3 Edition

Let us consider again an assignment operation changing the starting date of m_2. The new valuation ν' after the application step is the following.

Starting Dates	Ending Dates	Durations
$\nu'(s(m_1)) = 10$	$\nu'(e(m_1)) = 35$	$\nu'(d(m_1)) = 25$
$\nu'(s(m_2)) = \mathbf{7}$	$\nu'(e(m_2)) = 40$	$\nu'(d(m_2)) = 35$
$\nu'(s_1(m_2)) = 5$		

This valuation is not consistent. The unsatisfied constraints are the intrinsic constraint

$$s(m_2) + d(m_2) = e(m_2)$$

and the hierarchical constraint

$$s_1(m_2) = s(m_1) - s(m_2).$$

This time, the variable $s_1(m_2)$ can be associated to a strong weight such that its value in ν' is maintained in ν''. Maintaining this value implies changing some variables of m_1. Moving m_1 can be accomplished by giving $d(m_1)$ a stronger weight than $s(m_1)$ and $e(m_1)$. The following consistent valuation thus minimizes the resulting distance.

Starting Dates	Ending Dates	Durations
$\nu''(s(m_1)) = \mathbf{12}$	$\nu''(e(m_1)) = \mathbf{37}$	$\nu''(d(m_1)) = 25$
$\nu''(s(m_2)) = \mathbf{7}$	$\nu''(e(m_2)) = \mathbf{42}$	$\nu''(d(m_2)) = 35$
$\nu''(s_1(m_2)) = 5$		

This example shows a possible large amount of updated variables (bold values) as a result of the resolution step.

6 Relative Model RM

The absolute model AM2 is close to a relative model because of the use of the offsets of children into their parents. Let us now compare it with a model called RM including only relative information.

In RM, there exists an implicit time line for each hierarchical musical structure. Each substructure is associated to an occurrence date on the time line of its parent. Thus, musical structures are defined relatively to their parents, and not relatively to a single global time line. A date is only valid within the local time line. In order to use dates from different time lines, it is necessary to convert them by means of (sums of) occurrence dates of ancestors, up to the first common ancestor.

6.1 Definition

In RM, a generic musical structure m is defined by

$$(\{d,\ s_1,\ \ldots,\ s_k\},\ (m_1,\ \ldots,\ m_k),\ C_i,\ C_h,\ C_u)$$

with

- $k+1$ variables, the first one being the duration of m, the other ones being the occurrence dates of the substructures, as in AM2,
- a sequence of k substructures $m_1, \ldots, m_k$,
- a set of intrinsic constraints

$$\begin{aligned} C_i &= \{0 \leq d\} \cup \\ &\quad \{0 \leq s_i \mid i \in [1,k]\}, \end{aligned}$$

- a set of hierarchical constraints including the constraints on the ending dates of the substructures,

$$C_h = \{s_i + d(m_i) \leq d \mid i \in [1,k]\}.$$

6.2 Example

Let us consider the following structures m_1 and m_2 in RM.

$$\begin{aligned} m_1 &= (\{d\}, (), \{0 \leq d\}, \emptyset, \emptyset) \\ m_2 &= (\{d,\ s_1\}, (m_1), \{0 \leq d,\ 0 \leq s_1\}, \{s_1 + d(m_1) \leq d\}, \emptyset) \end{aligned}$$

This example represents the same structure as in the example in subsection 5.2, but the structure m_2 is not associated to an absolute starting date. In order to have exactly the same structure, it is necessary to define a musical structure p encapsulating m_2.

$$p = (\{s_1\},\ (m_2),\ \emptyset,\ \emptyset,\ \emptyset)$$

Let us point out that p does not contain any duration variable. Such a structure represents a local time line with infinite duration. It allows to encapsulate other structures without introducing any constraint.

Associating p to the following consistent valuation ν gives a consistent structure (p, ν).

Durations	Offsets
$\nu(d(m_1)) = 25$	$\nu(s_1(m_2)) = 5$
$\nu(d(m_2)) = 35$	$\nu(s_1(p)) = 5$

6.3 Edition

Let us consider again an assignment operation changing the starting date of m_2. The new valuation ν' after the application step is the following.

Durations	Offsets
$\nu'(d(m_1)) = 25$	$\nu'(s_1(m_2)) = 5$
$\nu'(d(m_2)) = 35$	$\nu'(s_1(p)) = \mathbf{7}$

This valuation is consistent. There is no unsatisfied constraint. Thus, no resolution step is required.

7 Implementations

Both models AM2 and RM have been implemented. The model AM2 first appeared in a software called BOXES [6, 5], while RM appeared afterwards in a new version of BOXES, still under development. In both implementations, the resolution step uses the simplex algorithm provided by the Cassowary library [1, 7]. In BOXES, the user can build musical structures which are directed acyclic graphs. Each structure is represented as a box which may contain other boxes. Basic structures are empty structures or sounds. Intrinsic and hierarchical constraints are build and managed automatically by the software. The user can add arbitrary constraints between boxes to represent his own musical structures. For example, he can build sequences of sounds and represent special articulations between sounds by the mean of user constraints.

The model AM2 is well adapted in the context of such a software. It is easy to implement because the management of user constraints is more simple than in RM: it does not require to convert dates. Unfortunately, as expected, AM2 loses efficiency in the case of large structures. This is the reason why RM was chosen afterwards.

A screenshot of BOXES is given in figure 1. On this screenshot, sounds are displayed in boxes as amplitude and frequency curves.

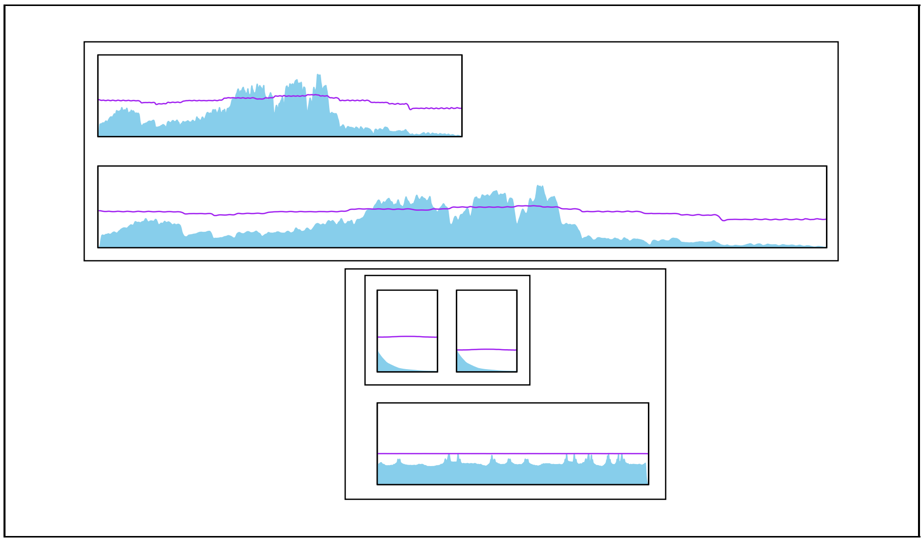

Figure 1: A Screenshot of BOXES.

8 Conclusion

In this paper, we defined various formal models for representing musical hierarchies with constraints. We compared, from an operational point of view, their suitability to a dynamic context of musical composition. This study is based on the experience we gained in developping software based on two different models: an absolute and a relative ones. Of course, the relative model is more efficient and more elegant than the absolute one.

To illustrate our purpose, we choosed to show very simple examples in order to be able to give the maximum of details about the resolution, without being too complicated. Of course, more complicated structures can be represented and edited the same way. In particular, user constraints can be used to bind structures that are not siblings and the resolution can provide very satisfying results depending on the distance that is used. The most difficult task is certainly to adjust the weights of all the variables in order to obtain awaited results of the resolution steps. We are currently working on a method for automatically computing weights in a generic way depending on the structure of the constraint graph.

Acknowledgement. This research was carried out in the context of the SCRIME[1] project which is funded by the DMDTS of the French Culture Ministry, the Aquitaine Regional Council, the General Council of the Gironde Department and IDDAC of the Gironde Department. SCRIME project is the result of a cooperation convention between the Conservatoire National de Région of Bordeaux, ENSEIRB (school of electronic and computer scientist engineers) and the University of Sciences of Bordeaux. It is composed of electroacoustic music composers and scientific researchers. It is managed by the LaBRI (laboratory of computer science of Bordeaux). Its main missions are research and creation, diffusion and pedagogy thus extending its influence.

References

[1] G. J. Badros and A. Borning. The Cassowary linear arithmetic constraint solving algorithm: Interface and implementation. Technical Report UW-CSE-98-06-04, University of Washington, `http://www.cs.washington.edu/research/constraints/cassowary`, 1998.

[2] M. Balaban and C. Samoun. Hierarchy, time and inheritance in music modelling. *Languages of Design*, 1(3), 1993.

[3] P. Ballesta. *Contraintes et objets : clefs de voûte d'un outil d'aide à la composition musicale ?* PhD thesis, Université du Maine, 1994.

[4] K. Barbar, M. Desainte-Catherine, and A. Miniussi. The semantics of a musical hierarchy. *Computer Music Journal*, 17(4):30–37, 1993.

[5] A. Beurivé. `http://dept-info.labri.u-bordeaux.fr/~beurive/BOXES`.

[6] A. Beurivé. Un logiciel de composition musicale combinant un modèle spectral, des structures hiérarchiques et des contraintes. In *Journées d'Informatique Musicale, JIM 2000*, Bordeaux, May 2000.

[7] A. Borning, B. Freeman-Benson, and M. Wilson. Constraint hierarchies. *Lisp and Symbolic Computation*, 5:223–270, 1992.

[8] K. Ebcioğlu. An efficient logic programming language and its application to music. In Jean-Louis Lassez, editor, *Proceedings of the Fourth International Conference on Logic Programming*, MIT Press Series in Logic Programming, pages 513–532. The MIT Press, 1987.

[9] M. Jourdan, C. Roisin, and L. Tardif. Constraints techniques for authoring multimedia documents. In *ECAI 98 Workshop on Constraints for artistic applications*, Brighton (UK), August 1998.

[10] F. Pachet and O. Delerue. MusicSpace: a constraint-based control system for music spatialization. In *Proceedings of the 1999 International Computer Music Conference*, pages 272–273, 1999.

[11] F. Pachet and P. Roy. Integrating constraint satisfaction techniques with complex object structures. In *15th Annual Conference of the British Computer Society Specialist Group on Expert Systems, ES'95*, pages 11–22, Cambridge, 1995. Best technical paper award.

[12] L. Tardif. *Kaomi : réalisation d'une boîte à outils pour la construction d'environnements d'édition de documents multimédias.* PhD thesis, Institut National Polytechnique de Grenoble, 2000.

[13] C. P. Tsang and M. Aitken. Harmonizing music as a discipline of constraint logic programming. In *Proceedings ot ghe 1991 International Computer Music Conference*, pages 61–64, 1991.

[1] Studio de Création et de Recherche en Informatique et Musique électroacoustique, www.scrime.u-bordeaux.fr