Rationalizing Musical Time: Syntactic and Symbolic-Numeric Approaches

Bernard Bel

Rationalizing musical time: syntactic and symbolic-numeric approaches*

Bernard Bel

> Music, like mathematics but unlike language, is not intelligible unless it is grammatical: its form is its content. As a product of "the unchanging human mind" and body in the context of different cultures, music reflects both man's biological structure and the patterns of interaction that have been institutionalized as systems of relationships in culture. (Blacking 1974)

This paper deals with various problems in quantifying musical time encountered both in the analysis of traditional drumming and in computer-generated musical pieces based on "sound-objects", hereby meaning code sequences controlling a real-time sound processor.

In section 1 it is suggested that *syntactic* approaches may be closer to the intuitions of musicians and musicologists than commonly advocated numeric approaches. Further, symbolic-numeric approaches lead to efficient and elegant solutions of constraint-satisfaction problems relative to *symbolic* and *physical* durations, as illustrated in sections 2 and 3 respectively.

1. A syntactic representation of musical accentuation

Many players of the *tabla*, a North Indian two-piece drum set, claim to follow a "rational" system of improvisation, the rules of which are generally not explicit and conveyed informally to students — much like a natural language. Therefore, a strong initial motivation of our formal study of the Lucknow tradition of tabla playing was the challenge of modelling a knowledge relying exclusively on oral transmission (Kippen & Bel 1989a). Indian musicians make use of onomatopeic syllables (*bol*s, from the verb *bolna*, to speak) to represent elementary sounds or finger movements that may precisely be transcribed on a computer (Kippen 1988:xvi-xxiii). The very first version of the *Bol Processor* (BP1) developed in 1982, was a customized word-processor allowing real-time transcription of drumming sequences thanks to a mapping of keyboard strokes to the vocabulary of tabla *bol*s. Analytical work was then undertaken with the aim of (1) making rules explicit for some compositional types, and (2) checking the consistency of musicians' assessments of correctness in both teaching and performance situations.

The following is an example of a compositional type known as *qa'ida*, the "theme and variations" form *par excellence* (Kippen 1988:xi). This example is borrowed from the Ajrara tradition. It should be read linearly from left to right, and each group represents a beat comprising six units.

* The Ratio Symposium (1992 décembre 14-16 : Den Haag, The Netherlands) In BARLOW, Clarence (ed.) The Ratio Book. Den Haag : Royal Conservatory - Institute of Sonology. 2001 [forthcoming], p. 86-101.

Theme:

dhin--dhagena	dha--dhagena	dhatigegenaka	dheenedheenagena
tagetirakita	dhin--dhagena	dhatigegenaka	teeneteenakena
tin--takena	ta--takena	tatikekenaka	teeneteenakena
tagetirakita	dhin--dhagena	dhatigegenaka	dheenedheenagena

A few variations:

dhin--dhagena	dha--dhagena	dhatigegenaka	dheenedheenagena
tagetirakita	dhin--dhagena	dhatigegenaka	teeneteenakena
dheenedheenagena	*teeneteenakena*	*tirakitatira*	*kitatirakita*
tagetirakita	dhin--dhagena	dhatigegenaka	teeneteenakena
tin--takena	ta--takena	tatikekenaka	teeneteenakena
taketirakita	tin--takena	tatikekenaka	teeneteenakena
dheenedheenagena	*teeneteenakena*	*tirakitatira*	*kitatirakita*
tagetirakita	dhin--dhagena	dhatigegenaka	dheenedheenagena

dhin--dhagena	dha--dhagena	dhatigegenaka	dheenedheenagena
tagetirakita	dhin--dhagena	dhatigegenaka	teeneteenakena
dhin--dhagena	*dha-dha-dha-*	*dhagenadheen--*	*dhagenadha--*
tagetirakita	dhin--dhagena	dhatigegenaka	teeneteenakena
tin--takena	ta--takena	tatikekenaka	teeneteenakena
taketirakita	tin--takena	tatikekenaka	teeneteenakena
dhin--dhagena	*dha-dha-dha-*	*dhagenadheen--*	*dhagenadha--*
tagetirakita	dhin--dhagena	dhatigegenaka	dheenedheenagena

dhin--dhagena	dha--dhagena	dhatigegenaka	dheenedheenagena
tagetirakita	dhin--dhagena	dhatigegenaka	teeneteenakena
dheenedheenagena	*dheenedha-dheene*	*dhatigegenaka*	*teeneteenakena*
tagetirakita	dhin--dhagena	dhatigegenaka	teeneteenakena
tin--takena	ta--takena	tatikekenaka	teeneteenakena
taketirakita	tin--takena	tatikekenaka	teeneteenakena
dheenedheenagena	*dheenedha-dheene*	*dhatigegenaka*	*teeneteenakena*
tagetirakita	dhin--dhagena	dhatigegenaka	dheenedheenagena

Our observations based on several samples of variations (from performances and demonstrations by the late Ustad Afaq Husain Khan of Lucknow) suggested that variable lines (shown in italics) were constructed with "words", i.e. chunks of *bol*s of lengths three, four and six in permutations that we presumed to be context-free. This view was reinforced by the fact that no technical (i.e. fingering) difficulties were encountered when words were arranged in arbitrary order. Words occurring most frequently are listed in the following lexical rules:

A3	--> dhin--
A3	--> dha--
A3	--> dhagena
A4	--> tirakita
A3 A3	--> dhagenadhin--
A3 A3	--> dhagenadha--
A6	--> dha-dha-dha-
A6	--> dha-ta-dha-
A6	--> dheenedheenedheene
A6	--> dheenedha-dheene
A6	--> tagetirakita
A6	--> dheenedheenagena
A6	--> teeneteenakena
A6	--> dhatigegenaka

In view of their frequent occurrence in examples, words *dhagenadhin--* and *dhagenadha--* have been identified specifically. A grammar for defining all possible sequences in variable lines of six, twelve or twenty-four units is easy to construct (Kippen & Bel 1992). However, some of the pieces generated by the grammar display irregularities in their accentuation. For instance,

dhin--**ti**raki ta**dha**gena**dha**ti gegenaka**ti**ra kita**ti**rakita

impose a rhythm counter to the natural stresses of the beat and half-beat, and is therefore virtually impossible to recite or perform at speeds normally employed by musicians (mm = 108-120, i.e. up to twelve bols per second). In a four-beat string comprising twenty-four units, primary accents fall on beats and half-beats: 1, 4, 7, 10, 13, 16, 19, and 22. A cursory analysis of variations created by musicians showed that in addition to these divisions they employed hemiolic rhythmic patterns beginning on units 1, 7, and 13. This produces a series of secondary stresses on units 5, 9, 11, 15, 17, 21. The following is a list of possible starting positions for the blocks defined above:

A3: 1, 4, 7, 10, 13, 16, 19, 22
A4: 1, 5, 7, 9, 11, 13, 15, 17, 21
A6: 1, 4, 7, 10, 13, 16, 19
tagetirakita: 1, 4, 5, 7, 9, 10, 11, 13, 15, 16, 17, 19

The exceptional status of "**ta**ge**ti**rakita" is due to the fact that it is accentuated in two different ways. Therefore it is labelled with a new variable: C6.

We developed a way to define derivations of B24, B12, and B6 in a systematic way that took into account acceptable starting positions. The grammar was right-linear, therefore instruction "LIN" appears in the first line:

```
LIN
B24   --> A3 B21          ...(A3 in starting position: 24 - (3+21) +1 = 1)
B24   --> A4 B20
B24   --> C6 B18
B24   --> A6 B18
B21   --> A3 B18          ...(A3 in starting position:  24 - (3+18) +1 = 4)
B21   --> A4 B17          ...(cancelled: A4 in starting position 4)
B21   --> A6 B15
B21   --> C6 B15
B20   --> A3 B17          ...(cancelled: A3 in starting position 5)
B20   --> A4 B16
B20   --> A6 B14          ...(cancelled: A6 in starting position 5)
B20   --> C6 B14
etc...
```

This grammar may produce a string "A4 A4 A4 A4 A4 A4" whose only derivation is an unbroken series of *tirakita*s that musicians would certainly assess as incorrect. It was found that more than two consecutive A4's were not accepted. The solution to this problem lies in introducing left contexts in all rules producing A4. Rather then enlisting all acceptable left contexts (as standard Chomsky grammars would force us to do) we found it practical to introduce negative contexts.

The resulting grammar is

```
B24              --> A3 B21
#A4 #A4 B24      --> #A4 #A4 A4 B20
B24              --> C6 B18
B24              --> A6 B18
B21              --> A3 B18
B21              --> A6 B15
B21              --> C6 B15
#A4 #A4 B20      --> #A4 #A4 A4 B16
B20              --> C6 B14
…
etc...
```

Here, for instance, rule

```
#A4 #A4 B24    --> #A4 #A4 A4 B20
```

means *"B24" may be rewritten "A4 B20" if not preceded by "A4 A4"*.

A full description of the grammar of this *qa'ida* is discussed in (Kippen & Bel 1992, appendix 3) a variant of it is available in the Bol Processor BP2 shareware package. A detailed comment of syntactic extensions of the formal language model applied to musical sequences, which we call *BP grammars*, may be found in (Bel & Kippen 1992). Readers may also refer to (Bel 1992) to understand the parsing algorithm that is used for assessing the compatibility of arbitrary sentences with a given BP grammar.

The *qa'ida* example makes it clear that quantization of rhythm and metre, although generally described as a typical numeric problem (e.g., Vuza 1988), may also yield syntactic descriptions with the advantage of reflecting productive and analytical processes based on *permutations* and *substitutions*. Learning these processes from examples is an important part of the basic training in traditional drum improvisation/composition (Kippen & Bel 1989b).

2. Symbolic representation of discrete sound-object structures

2.1 Bol Processor BP2: the environment

"BP2" is a new version of Bol Processor operating in a MIDI environment for design-based (stipulatory) or improvisational *rule-based composition* (Laske 1989:51,53). It is available as a shareware package for Macintosh™ computers.

Several operational modes are available in BP2, from one that leaves all decisions to the machine (stochastic improvisation) to one that allows a composer to take stepwise decisions. Since BP2 does not yet contain a parsing module, it is oriented towards productive, rather than analytical, work.

The interaction of modules is summarized in the block diagram of Fig.1.

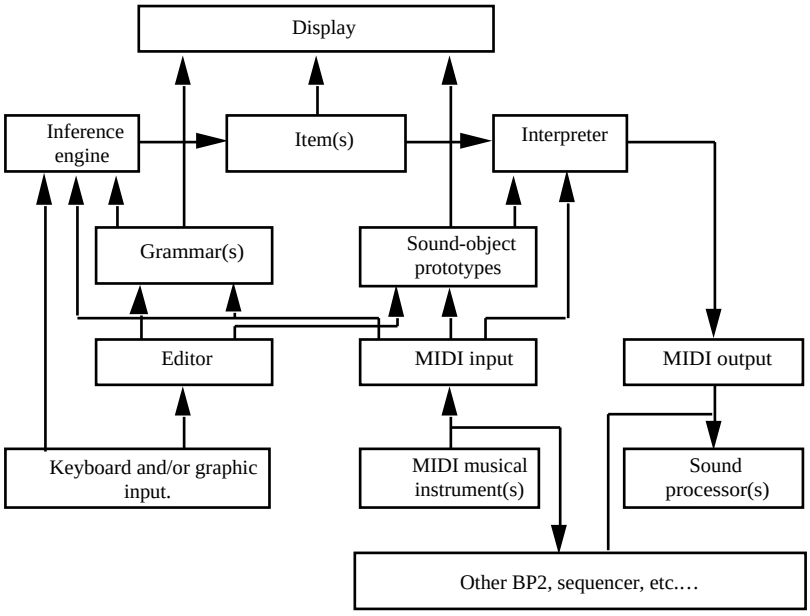

Fig.1 A block diagram of Bol Processor BP2

Three fields are used for storing a grammar, items generated by the grammar (on the basis of decisions taken by the inference engine) and *sound-object prototypes*. These prototypes are arbitrary sequences of messages loaded from a MIDI musical instrument and edited manually. Terminal symbols in the grammar are the labels of *sound-objects* replacing the onomatopeic syllables (*bol*s) used by BP1.

The interpreter works in three stages:

1) The item generated by the grammar is interpreted as a *polymetric expression* (see §2.7 infra). The output is a *complete polymetric expression* yielding a bidimensional array of terminal symbols called the *phase diagram* (see §2.3).

2) The expression is interpreted as a sound-object structure, using information about the *structure of time* (see §2.2) and object prototype definitions. The main output is an extension of the phase table containing the performance parameters of objects in the structure: their start/clip dates, time-scale ratios, etc.

3) MIDI messages contained in sound-objects are dispatched in real time to control the sound processor.

The block diagram indicates that an external control can be exerted on the inference engine, grammars, and the interpretation module. Specific MIDI messages may be assigned to changes of rule weights, tempo, the nature of time (striated/smooth) and many other parameters. Messages may also be used for synchronizing events during the performance, or even assigning computation time limits. These features are used in improvisational rule-based composition.

Several BP2's may be linked together and with other devices such as MIDI sequencers. Messages on the different MIDI channels may be used for making machines communicate or control several sound processors. Therefore it must be kept in mind that "sound-objects" do not necessarily produce sounds. Depending on the implementation they may contain any kind of control/synchronization message as well.

2.2 Symbolic time

Let us assume that "a", "b", "c", "e", "f", "g" and "-" are labels of arbitrary sound-objects (similar to "dhin", "dha", etc. in §1). Label "-" is reserved to silences which are viewed as particular objects. Fig.2 represents a structure of two sequences which, in first approximation, might be notated S_1 = "a b c a" and S_2 = "e - f g".

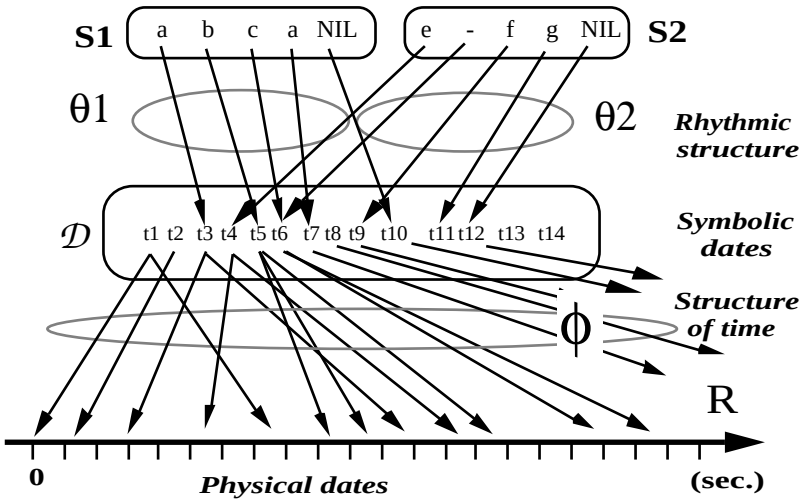

Fig.2 A representation of sequences S1 and S2

In Fig.2 a set of strictly ordered *symbolic dates* $\mathcal{D}$ = {t1,t2,…} is introduced along with θ_i , an injective mapping of each S_i into $\mathcal{D}$. By convention, each θ_i is a monotonous increasing function: sequentiality implies that all objects appearing in a sequence are ordered in increasing symbolic dates. Each mapping θ_i may in turn be viewed as a restriction to S_i of a general mapping θ which we call the *rhythmic structure* of the sound-object structure. "NIL" symbols are used to mark the ends of sequences.

The mapping of sequences to the set of symbolic dates is mainly information about the ordering of any pair of events belonging to either sequence. Here, for instance, S_1 and S_2 will be partly overlapping.

The set of symbolic dates $\mathcal{D}$ is then mapped to physical time, i.e. the set of real numbers R. We call this mapping Φ the *structure of time.* (The same was called *structure temporelle* in Xenakis 1963:190-191,200) In the example shown above, Φ is a multivocal mapping, which means, for instance, that each sound-object "a" and "e" at symbolic date t_3 would be performed twice. In general only strictly increasing (univocal) mappings are envisaged, so that:

$$\forall i,j \in N, \quad t_i < t_j <=> \Phi(t_i) < \Phi(t_j) \ .$$

In this case, if we consider $Dist(t_i,t_j) = |\Phi(t_j) - \Phi(t_i)|$ (the absolute value of the difference), *Dist* is a distance on $\mathcal{D}$. Besides, since

$$\forall i,j,k \in N, \quad Dist(t_i,t_j) + Dist(t_j,t_k) \geq Dist(t_i,t_k)$$

($\mathcal{D}$,Dist) is also a *metric* space. ($\mathcal{D}$,Dist) is also *Euclidian* (*metronomic time*) if the additional property holds:

$$\forall i,j,k,l \in N, \quad j - i = l - k \ => \Phi(t_j) - \Phi(t_i) = \Phi(t_l) - \Phi(t_k)$$

The composition of the two mappings ($\Phi \ . \theta$) is the *in-time structure* of the musical item, i.e. the mapping that permits its actual performance.

As suggested by the terminology, structure of time and in-time structures are two concepts borrowed from Xenakis (1963). We find these concepts essential as they deal with sets of physical dates not necessarily structured as a Euclidian space.

2.3 Phase diagram

Both sequences of this example may be represented together in a single array (the *phase diagram*), the columns of which are labelled and ordered on symbolic dates. (See Fig.3)

t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12	t13	t14
_	_	a	_	b	c	a	_	_	NIL	_	_	_	_
_	_	_	e	_	-	_	_	f	_	g	NIL	_	_

Fig.3 A phase diagram

The phase diagram contains *empty sound-objects* "_" indicating the prolongation of the preceding sound-object, if any. These should not be confused with silences "-".

Using the information displayed in the diagram, S_1 and S_2 are properly notated:

S_1 = a _ b c a _ _ $\qquad\qquad$ S_2 = e _ -_ _ f _ g

Here, we call *symbolic duration* of a sound-object the relative position of the next non-empty sound-object or "NIL" marker. For example, in S_2 the symbolic durations of objects "e", "-", "f" and "g" are two, three, two and one respectively. In S_1, there are two consecutive occurrences of "a" with respective durations two and three.

If for example "a", "b", etc. would represent notes in conventional music notation, assuming that "b" is a quarter note would imply that "e" is a half-note and "-" a dotted half-note rest.

2.4 Smooth vs. striated time

Pierre Boulez (1963:107) introduced the notions of *smooth time* ("temps lisse") and *striated time* ("temps strié") to characterize two typical situations in music performance. Striated time is *filled with (regular or irregular) pulse*, whereas smooth time *does not imply any counting*: A particular case of striated time is of course a metronomic pulse. Examples of smooth time are common outside Baroque music, e.g. melodic introductions in Indian *raga* music.

In computer-generated music, these notions are bound to the structure of time (the Φ mapping): in striated time, Φ is known in advance, whereas in smooth time it is determined at the time of performance. Therefore, a *striated structure of time* is a set of physical dates defining *reference streaks* on which sound-objects should be positioned (see §3) whereas a *smooth structure of time* is a set of dates determined by the sound-objects themselves.

2.5 Out-time objects

Sound-objects have strictly positive symbolic durations. In some cases it is useful to dispose of "flat" objects with null durations which we call *out-time objects*. These may be defined from sound-objects whose actions are executed "simultaneously" or in a very short sequence. In the BP2 environment, a typical application of out-time objects is the exchange of parameters or synchronization messages.

Given a sound-object labelled "a", the corresponding out-time object is notated "<<a>>". Using this convention, a string like

<<a>> b

represents a structure in which out-time object "<<a>>" starts at the same symbolic date as sound-object "b".

2.6 Tempo markers

Sound-object sequence "a b c d e f" may be notated "/1 a b c d", where "/1" is an *explicit tempo marker*. If the same sequence is to be interpreted five times faster we may notate it "/5 a b c d e f". This notation was already used in Bol Processor BP1 to indicate *bol density*: North Indian drummers and dancers designate by *dogun, tigun*, etc., *bol* densities of two,three, etc, *bol*s per *matra* (beat).

Using explicit tempo markers makes it possible to modify tempo within a single sequence. For instance, in sequence

/2 a b c d e f /3 g h i j k l m n o

"a" … "f" will be interpreted at bol density 2 (two sound-objects per beat), then "g" to "o" will be interpreted at bol density 3. (This may also be viewed as a tempo acceleration of 3/2.) The same expression may also be notated:

/6 a_ _ b_ _ c_ _ d_ _ e_ _ f_ _ g_ h_ i_ j_ k_ l_ m_ n_ o_

Silences may be notated with hyphens or integer numbers. The following notations are strictly equivalent:

/2 a b - - c d /5 e - - - - f g h
/2 a b - _ c d /5 e - _ _ _ f g h
/2 a b 2 c d /5 e 4 f g h

Rational numbers may also be used to indicate fractional silences, e.g.

/1 a b /2 c d e f 4/3 g h

in which "a" and "b" are interpreted at *bol* density one, "c", "d", "e", "f", "g" and "h" at *bol* density two, while sequences "cdef" and "gh" are separated with a silence of duration 4/3. Since the "4/3" silence occurs at *bol* density two, its actual symbolic duration is 4/3 x 1/2 = 2/3. Here, BP2 will expand the representation as follows:

/6 a _ _ _ _ _ b _ _ _ _ _ c _ _ d _ _ e _ _ f _ _ - - - - g _ _ h _ _

where the "4/3" silence appears as "- - - -" (or "- _ _ _", or "4" equivalently).

2.7 Polymetric expressions

Suppose that we wish to superimpose two sequences A and B defined by rules:

A —> a b c
A —> d e f g
B —> h i

in which "a", "b", … "i" are labels of sound-objects. Alternate definitions of "A" indicate that it may contain either three or four objects. To start with, we do not know how to interpret the exact superimposition of two sequences: combining "abc" and "hi" may for example yield the following phase diagrams:

```
a b c     a b c     a _ b _ c _     a _ b c       etc...
h i _     _ h i     h _ _ i _ _     h i _ _
(1)         (2)         (3)             (4)
```

Prolongational symbols "_" could even be replaced with silences "-". However, since silences do not explicitly appear in the grammar, we may postulate that creating them is not a valid choice. Further we discard interpretations (1) and (4) in which equal symbolic durations are not maintained within the same string "h i". Finally, it is reasonable to expect a synchronization of both the start and clip points of the synchronized sequences. Therefore there is no reason to start "a" before "h" as suggested by interpretation (2).

Finally, the most intuitively appealing interpretation (in the absence of any additional information) is the one shown in (3). A notation of superimpositions is now introduced:

{A,B} (equivalently, {B,A}) is the superimposition of sequences “A” and “B”. We call “{A,B}” a *polymetric expression* whose *arguments* are “A” and “B”.

Using this notation, a grammar yielding all acceptable superimpositions of “A” and “B” would be:

S —> /1 {A1,B1}
S —> /1 {A2,B2}
A1 —> a _ b _ c _ B1 —> h _ _ i _ _
A2 —> d e f g B2 —> h _ i _

Once a string like “/1 {d e f g, h _ i _}” has been produced, it is necessary to check that it contains equally many terminal symbols in both arguments, failing to which the phase diagram cannot be constructed.

Evidently it is cumbersome to be forced to give two possible versions of “B”, the more so because they point to identical ratios of symbolic durations: “B2” is similar to “B1” in every respect. Ideally, the following grammar should be used:

S —> /1 {A,B}
A —> a b c
A —> d e f g
B —> h i

expecting that there will be a method for interpreting an *incomplete polymetric expression* like

/1 {a b c, h i}

as

/2 {a _ b _ c _ , h _ _ i _ _ }

i.e. a *complete polymetric expression*. Note that the tempo marker now indicates *bol* density two because durations have been stretched. Thus, for instance, the symbolic duration of “b” remains one beat. A compact representation of this complete expression makes use of explicit tempo markers in each argument, i.e.:

{/3 a b c , /2 h i}

showing the classical “three-in-two” polyrhythm, along with the information that tempo should be divided by a *time scale factor* of three so that actual durations will be the ones we expect.

Interpreting polymetric expressions is the task of a fast algorithm implemented in Bol Processor BP2 (Bel 1991-1992). Since the algorithm makes use of arithmetic operators such as LCM (lowest common multiple) altogether with rewrite procedures it may be classified a *symbolic-numeric* method. Thanks to recursivity it is possible to interpret nested expressions such as

{i {a b, c d e}, j k}

yielding

{/6 i {/6 a b, /9 c d e}, /4 j k}

(in which the *time scale factor* is six).

If some arguments contain explicit tempo markers indicating a compulsory bol density, the algorithm will try to satisfy all constraints so that, in the end, arguments of polymetric expressions have identical symbolic durations. In some cases there is no solution; therefore it is preferable (and always possible) to avoid writing explicit tempo markers in sequences or in polymetric structures, as we will now show.

Polymetric representation of a sequence

Introducing a string of silences as the first argument of a polymetric expression is a good method for suppressing explicit tempo markers in a sequence. For instance,

a b c _ /3 d _ e

may be notated

a b c _ /3 {- - -, d _ e}

which is equivalent to:

a b c _ /3 {3, d _ e}
a b c _ {3/3, d _ e}
a b c _ {1, d _ e}

The advantage of the last notation is that the same expression may be used at different tempos. For instance, when it needs to be performed four times faster we just write

/4 a b c _ {1, d _ e}

rather than

/4 a b c _ /12 d _ e

which forces us to recalculate the second tempo marker. *Polymetric representation, therefore, make it possible to build very complex musical structures by way of simple rewriting rules (formal grammars), given that the computation of symbolic durations and the matching of superimposed sequences is ultimately taken care of by a unique and efficient polymetric interpretation algorithm.*

Other features relative to polymetric expressions (along with typical examples in conventional music notation) may be found in (Bel 1991-1992).

3. The time setting of sound-objects

Informally, *instantiating* a sound-object means dispatching to the sound processor all messages defined in its prototype. A naive interpretation of sequences of sound-objects would be to arrange all corresponding time intervals in a strictly sequential way. Duthen and Stroppa (1990) have suggested a more general approach starting from the assumption that any sound-object may possess one or several time points playing a particular role, e.g. a climax. These points are called *time pivots*. Further they suggest to construct sound structures using a set of synchronization rules. Their approach is attractive but it is hard to implement if the formalism of synchronization rules remains too general. Therefore we retained a simplified version of Stroppa's idea, assigning each object one single pivot.

Let us for instance consider a complete polymetric structure {S1,S2,S3} derived as

{a _ b c d _ e , a _ f _ g h _ , j i _ a _ i _ }

yielding the phase diagram shown Fig.4.

```
a    _    b    c    d    _    e    NIL
a    _    f    _    g    h    _    NIL
j    i    _    a    _    i    _    NIL
```

Fig.4 A phase diagram of {a _ b c d _ e , a _ f _ g h _ , j i _ a _ i _ }

The definition of each sound-object contains the relative location of its pivot and metrical properties allowing the calculation of its "time-scale ratio" — informally, a factor adjusting the duration of the sound-object to the current speed of performance.

The following is a graphic representation of a possible instance of this polymetric structure, as displayed by BP2:

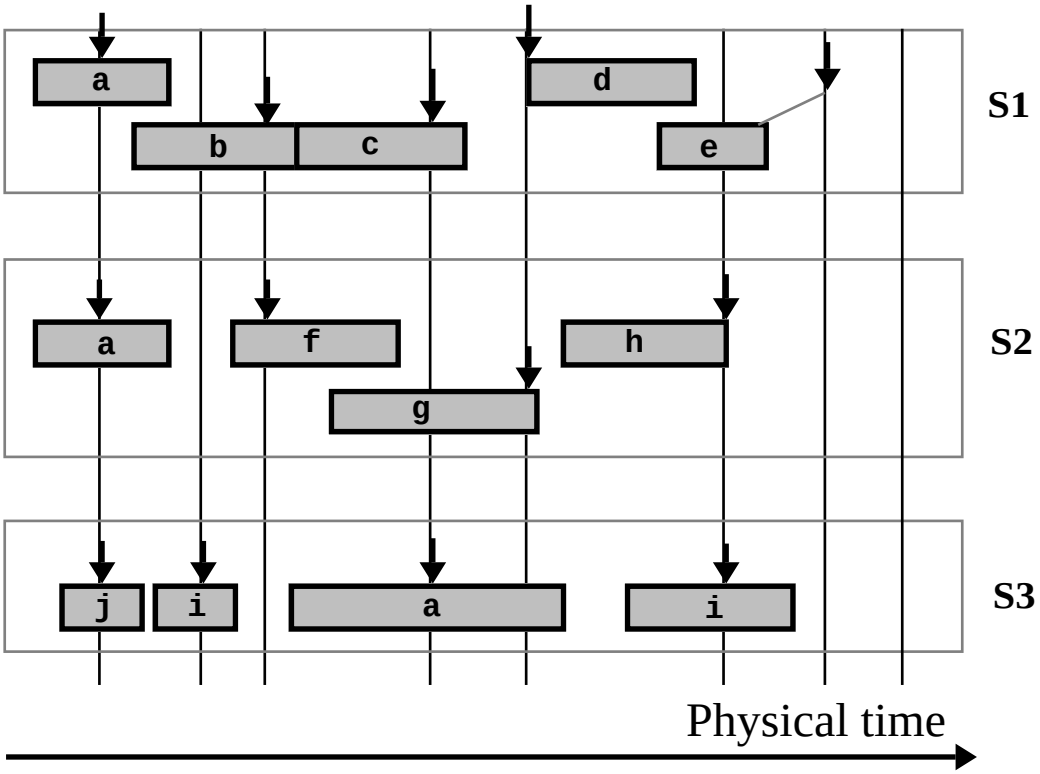

Fig.5 A structure of sound-objects

Here, the structure of time is an irregular pulsation represented with vertical lines (time *streaks*). The time-span interval of each sound-object is shown as a rectangle with arbitrary vertical width and position. These positions have been chosen to separate objects on the graphic. It is clear, for example, that "c", "f", "g" and "a" have overlapping time-span intervals between the third and fourth streaks. Lengths of rectangles represent the physical durations of sound-objects. Out-time objects, if any, would appear as vertical segments.

Vertical arrows indicate time pivots. As shown with object "e", the pivot is not necessarily a time point within the time-span interval of the sound-object.

This graphic represents the *default positioning* of objects with their pivots located exactly on time streaks. Although it is reasonable that instances of "c", "f" and "a" are overlapping between the third and fourth streaks since they belong to distinct sequences performed simultaneously, it may not be acceptable that "f" overlaps "g" in a single sequence S2; the same with "d" and "e" in sequence S1. It may also not be acceptable that the time-span intervals of "j" and "i" are disjoint in sequence S3 while no silence is shown in the symbolic representation.

How could one deal with a constraint such as <<*the end of sound-object "f" may not overlap another sound-object in the same sequence*>> ? If object "g" is relocatable then it may be delayed (shifted to the right) until the constraint is satisfied. We call this a *local drift* of the object. Yet the end of "g" will also overlap the beginning of "h". Assume that this too is not acceptable and "h" is not relocatable. We should then look for another solution, for example truncate the beginning of "h". If this and other solutions are

not acceptable then we may try to shift “f” to the left or to truncate its end. In the first case it might also become necessary to shift or truncate “a”…

So far we suggested constraint propagation within one single sequence. In the time-setting algorithm the three sequences are taken in order S1, S2, S3. Suppose that the default positioning of objects in S1 satisfies all constraints but no solution has been found to avoid the overlapping of “f” and “g” in S2. A new option is to envisage a *global drift* to the right of all objects following “f” in S2. The global drift is notated Δ on Fig.6. All time streaks following the third one are delayed (see dotted vertical lines).

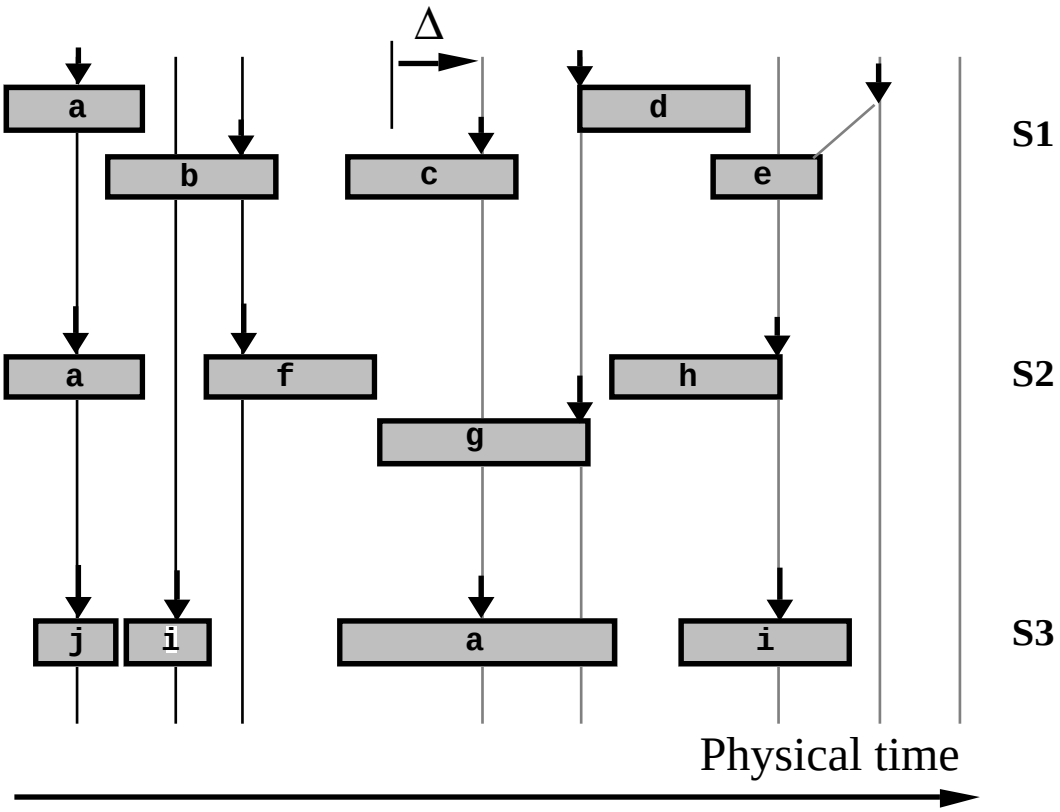

Fig.6 A solution using global drift

This solution is called “Break tempo” because its effect is similar to the the *organum* in conventional music notation. Although the global drift increases the delay between the third and fourth streaks, the physical durations of sound objects are not changed because their time-scale ratios have been calculated beforehand.

Now the positioning of objects in S2 is acceptable, but it might have become unacceptable in S1: there may be a property of “b” or “c” saying that their time-span intervals cannot be disjoint, so that “c” could be shifted to the left, etc. Evidently, whenever a global drift is decided the algorithm must start again from the first sequence.

The process of locating — i.e. *instantiating* — sound-objects, as illustrated in this example, is the task of the *time-setting algorithm* imbedded in BP2. If no global drift is created, the time complexity of the time-setting algorithm is $O(nmax.imax^3)$, where *nmax* is the number of sequences and *imax* the maximum length of a sequence. In the worst case, the time complexity is $O(nmax^2.imax^3)$. The algorithm is described in great detail in (Bel 1991-1992).

4. Conclusion

Work with Bol Processor BP1 and BP2 has been beneficial in finding a workable compromise between general formal language models, whose mathematical properties are well established although they often bear little musical relevance, and ad hoc representations fulfilling the requirements of only particular musical tasks.

Polymetric structure interpretation and constraint-based time-setting of sound objects contribute to compensate the rigidity of the timing of computer-generated musical pieces, as the synchronization and accurate timings of concurrent musical processes are handled by the computer on the basis of (possibly incomplete) information on structures and sound-objects.

References

Bel, Bernard, 1990

Acquisition et représentation de connaissances en musique. PhD dissertation, Université Aix-Marseille III.

1991

Two algorithms for the instanciation of structures of musical objects. Internal report GRTC 458. Groupe Représentation et Traitement des Connaissances, CNRS, Marseille.

1992

Symbolic and sonic representations of time-object structures. In (M. Balaban, K. Ebcioglu and O. Laske, eds.) *Understanding Music With AI*. Menlo Park: AAAI Press: 64-109.

Bel, Bernard & Jim Kippen, 1992

Bol Processor grammars. In (M. Balaban, K. Ebcioglu and O. Laske, eds.) *Understanding Music With AI*. Menlo Park: AAAI Press: 366-400.

Blacking, John, 1974

Ethnomusicology as a key subject in the social sciences. *In Memoriam Antonio Jorge Dias*, 3 (1974), Lisbon (Portugal): 71-93.

Boulez, Pierre, 1963

Penser la musique aujourd'hui. Paris: Gonthier.

Duthen, Jacques & Mario Stroppa, 1990

Une représentation de structures temporelles par synchronisation de pivots. In (B. Vecchione & B. Bel, eds.) *Le fait musical — Sciences, Technologies, Pratiques*. Marseille: Colloque CRSM-MIM "Musique et Assistance Informatique".

Kippen, Jim, 1988

The Tabla of Lucknow: A Cultural Analysis of a Musical Tradition. Cambridge (UK): Cambridge University Press.

Kippen, Jim & Bernard Bel, 1989a

Can a computer help resolve the problem of ethnographic description? *Anthropological Quarterly*, 62, 3: 131-144.

1989b

The identification and modelling of a percussion 'language', and the emergence of musical concepts in a machine-learning experimental set-up. *Computers and the Humanities*, 23,3: 199-214.

1992

Modelling music with grammars: formal language representation in the Bol Processor. In (A. Marsden and A. Pople, eds.) *Computer Representations and Models in Music*. London: Academic Press: 207-238.

Vuza, Dan Tudor, 1988

Sur le Rythme Périodique. In (M. Boroda, ed.) Quantitative Linguistics. *Musikometrika*, 37, 1: 83-126.

Xenakis, Iannis, 1963

Musiques formelles. Paris: La Revue Musicale.

Augmented translation, 1971: *Formalized music*. Bloomington: Indiana University Press.