On the Relation of Quality and Quantity in the Context of Musical Composition

*Alexandre Reche e Silva
Universidade Federal do Rio Grande do Norte (UFRN)
alereche@gmail.com

Abstract: *With this paper we aim to highlight the connection between quality and quantity, from a musical point of view. For this, we heuristically sketch a typology of musical qualities. Every quality offers a gamut of gradations. Each degree inside this range can be indexed as a value, making a range of quantities available. The changes of a musical quantity over time is represented as a list of values. This list can be manipulated through a variety of mathematical operations. Such approach can be applied to any musical quality (thus, encouraging students to face the elements assembled in a composition from the start). Some of these operations are presented here as functionalities of J-Syncker, an assistant software for the generation of pre-compositional material.*

Keywords: *Music Composition. Musical Qualities. Musical Quantities. Lists. J-Syncker.*

I. Introduction

Listening reveals nuances inside its own realm. Basically, we perceive a sound by means of its intensity (strong or weak), duration (long or short), pitch (high or low) and timbre (smooth or rough). Furthermore, each one of this basic specialties (or qualities) signals variations to the attentive listener.

In a composition, musical qualities also describe changes in their values over time. We represent these values (or quantities) using the concept of list. For instance, we can use a list to depict the quantities associated to dynamics or melodic contour (respectively, variations of intensities and pitches).

Values of the same list can be used to quantify different musical qualities. On the one hand, this helps achieve coherence. On the other, it communicates dynamism to the craft of composition.

Employing operations on lists can be strategically used for musical ends, producing a wealth of pre-compositional material. It is also a way to summarize composition techniques, suggesting another manner of formalizing knowledge to the field of music composition. Once quantitative variation described by musical qualities is numerically represented, it can be subjected to a variety of mathematical resources. Still, this approach can be applied to any musical quality.

*This paper reports part of my postdoctoral research (2015-16) sponsored by CAPES (Brazil) and Fulbright (USA) commissions. I would like also to acknowledge Heather Dea Jennings and Agamenon de Clemente de Morais Júnior for their kind attention to review the paper.

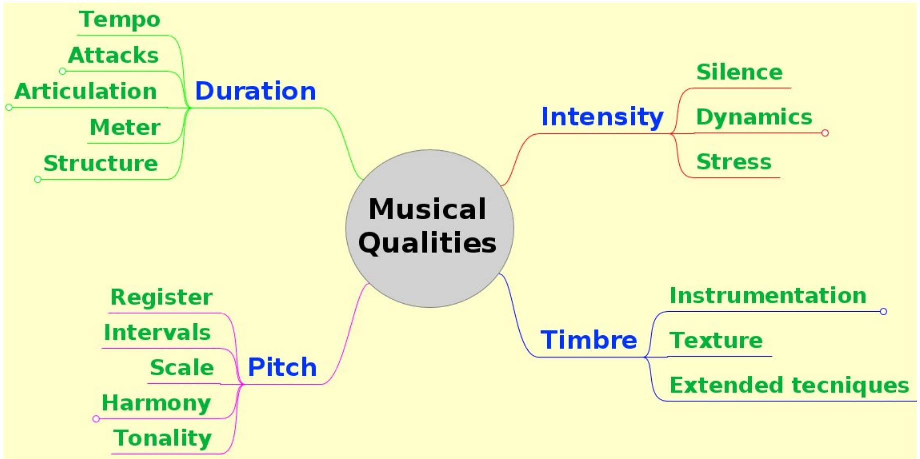

Figure 1: *An heuristic mapping of Musical Qualities.*

It is highly desirable for such an approach to be assisted by a computer application, through which a composer can listen to and choose from output results. In the proceedings of this paper we will talk about a specific application, written as a proof of concept. We will describe some of its functionalities in the context of referred operations on lists. For now, let us go back to the musical qualities.

II. Musical Qualities

In Figure 1 we map[1] a first generation of musical qualities: intensity, timbre, pitch, and duration. Intensity relates with the volume of musical sounds. Timbre is referred as the tone color (or that which differentiates one sound source from another). Pitch is related to sound frequencies. Duration refers to the length of events[2].

It is possible to subdivide these first branches, reaching new subtleties in the auditory sense. Within intensity we may locate silence (absence of intensity); dynamics (changes in intensity) and stress (a way to differentiate some events from others, bringing them to the foreground of listening). Within timbre, instrumentation deals with the possibilities of timbristic combinations; texture deals with the number of voices (or instruments) sounding in a given moment. Within pitch we locate register (a range of notes possible to be played in a instrument); intervals; scale; harmony and tonality. Within duration we have tempo (speed of execution); the size of attacks; articulations (the nuances in the lengths of notes); meter (a way to group beats in measures); and structure (rhythm of sections[3]). Also in this domain is the total duration of a given work.

Furthermore, we witness a combination of them sprouting new entities. Rhythm is in itself a complex combination of properties related to duration along with stress. Melody could be seen as how pitches change as function of durations and intensities. Orchestration is the variation of instrumentation and texture along with structure; the same with modulation, i. e., sectional articulation of tonalities.

[1]The software used to this end was Freeplane [11].

[2]Extra musical elements are beyond the scope of this work, such as lyrics, written descriptions, other medias (choreography, video, etc.), soggetto cavato, mood markings and so on.

[3]The idea of structure as a rhythm of sections was an insight shared by Professor Ilya Levinson (Columbia College Chicago) in a personal communication with the author.

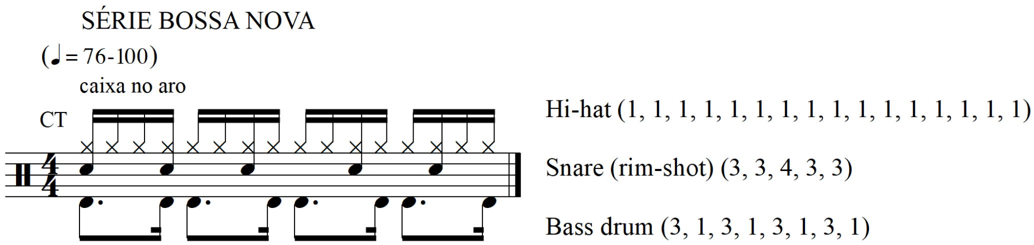

Figure 2: *An instance of Bossa Nova rhythm to drums [13, p. 60].*

Musically speaking, it is possible to notice that listening reveals nuances inside its own realm. We may identify sub-senses tracking a variety of stimuli. Music can invite a listening expansion in which different qualities may be perceived residing even within the auditive experience.

III. Musical Quantities

In a composition, musical qualities change their values over time. Pitches go up and down. Durations become longer and shorter. Intensities get strong and soft. The same may happen with timbre, texture, tonality and so on. These changes define ranges of possibilities even inside a very quality. It seems reasonable to relate a scale to each musical quality, i. e., a gamut of varieties provided by every domain [4].

Let us use a list to collect the values involved in a musical quality, changing over time. We employ the term list here as a sequence of quantities derived from a given musical domain. Next, we describe some attributes of a list. Take for instance a list of some composers's birthdays (6,1,19,12,2,15). The size (also referred to as length) gives the number of elements in the list. In this case, the size is 6. A list has an order of elements. Thus, an element has a position in a list. The birthday 12 is in the forth position. Also, an element may have a specif type, i. e., numeric, alphabetic, alphanumeric, etc. Our list has only numeric values. Let total be the sum of all numeric elements. In our case, total is 55.

Now we will use a list to represent the quantities associated to a musical quality. We will represent attack durations, for instance. Consider the information displayed in Figure 2.

Let us take only the snare part. The list (3,3,4,3,3) was generated taking into account the durations of each attack. The basic unity in this case is the 16th note. So, "1" is represented by a 16th note, "2", by a 8th note, "3", as a dotted 8th note, "4", by a quarter note and so on. (It is also possible to obtain different notations by tying music figures in different combinations.)

Let us consider some of the attributes of this list from the point of view of the rhythm. The size of the list (or the number of elements) gives the number of attacks. The one designated to the snare has 5 elements, thus, 5 attacks. The list sum gives the total duration of the rhythm (which is the duration of all attacks together). It is equal to 16 units. As we already know, the value of an element gives the duration of a single attack. The first attack takes 3 units, the second 3, the third 3, and so on. So, we can say that this specific rhythm of 16 units were partitioned into 5 attacks. This information will be represent as 16P5. However, notice that the sequence of 3, 3, 4, 3 and 3 units is unique. Just imagine that (4,3,2,3,4) is also another member of a 16P5. We will cover more about partitioning in the next section.

[4] This approach lies in the heart of the system of musical composition devised by Schillinger [15]

Next, we will use the same list to quantify different musical qualities. Besides rhythm itself, we will provide an example for structure and intervals.

In terms of structure, we perform a one-to-one correspondence between two lists, the second being a list of sections:

$$(3,3,4,3,3) \rightarrow (A,B,C,B,A) = (3A,3B,4C,3B,3A)$$
=(A,A,A,**B**,**B**,**B**,C,C,C,C,**B**,**B**,**B**,A,A,A)

Thus, an example of one list controlling the number of repetitions of the members of another list. Here, we can say that the first was used as a list of "coefficients of recurrence" [15].

In terms of intervals, we apply the values of the list as intervals (in semitones) to generate a scale of non neighbor pitches.

3 3 4 3 3
$F\sharp ACEGB\flat$

Likewise, we could keep applying the same list over other domains. Thus, rhythm may be represented as a list of durations. A structure, as a list of sections. A scale, as a list of pitches (or a list of pitch intervals).

Up to this point we have described how to represent quantities related to a musical quality. In the next section we will employ operations to produce changes in the quantities of such qualities. This will be very useful to aid the process of obtaining variety out of the same musical seed.

IV. Operations on lists

In the previous section we described a way to represent two important attributes of a list. They are sum and size. We represented them with the expression [sum][Part][size]. So, 16P5 means that a total of 16 were partitioned in five elements (also referred to as parts, terms, summands or addends). We use this device only with integers. For instance, we depict in the Figure 3 an example of partitions of the integer 8.

Notice how the representation [sum][Part][size] changes according to the place in the partitions generations. In this case it goes from 8P1 to 8P8, revealing 8 generations.

The number of partitions grows very quickly as bigger integers are used. Thus, we apply a restriction criteria, depicting only results with greater values placed on the left[5].

This family holds coherence once it is performed systematically within the bounds of its parent integer. Such coherence is very useful when applied to musical purposes. For instance, its members can be used in succession but also simultaneously, producing rhythmic polyphony.

Next, we are going to use a list to quantify rhythmic durations. Let us reuse the pattern executed by the snare in Figure 2, now represented as the list (3,3,4,3,3) – a member of 16P5. In Figure 4 we depict some operations on it, producing variants as results. (Notice also how the partitions change by means of such operations.)

We will describe each of these operations, as they are numbered in clockwise direction.

1. Reverse returned the original list. This is due to the list elements being palindromic, i. e., the same thing can be read back and forth. Such a non-retrogradable rhythm was referred as "the charm of impossibilities" by Messiaen [10];

[5] When permuted results are taking into account, the term Composition is used instead of Partition.

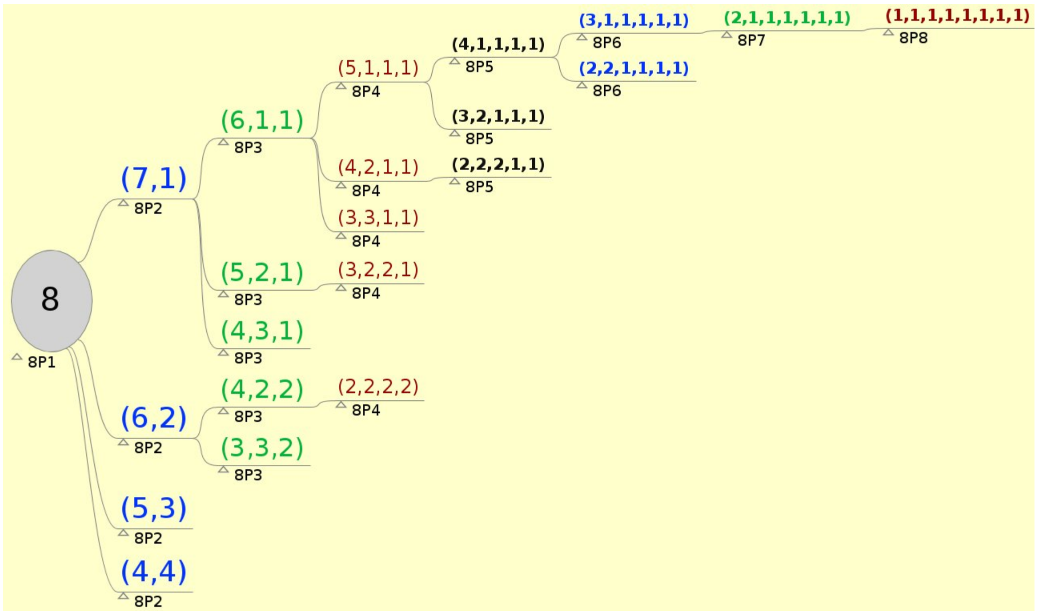

Figure 3: *Partitions of the integer 8.*

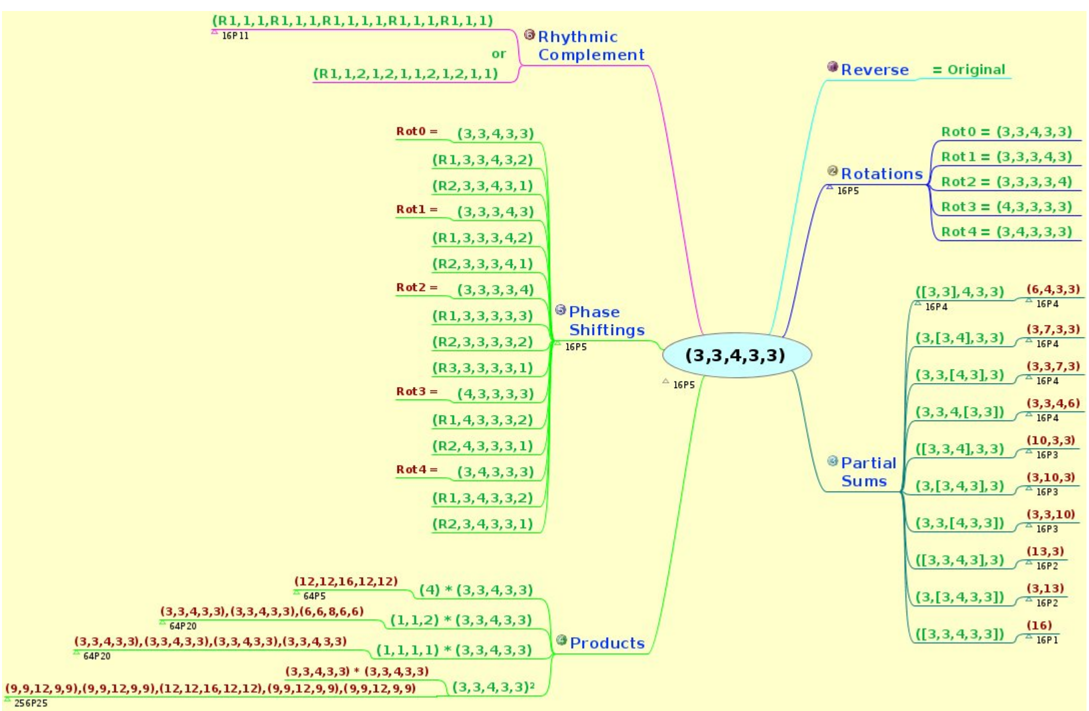

Figure 4: *Operations on a list of rhythmic durations (3,3,4,3,3).*

2. Rotations can be obtained imagining the list tied in its extremities and rotated clockwise or anticlockwise (the latter being notated with negative indexes, i. e., Rot_{-2}). The number of possibilities equals the number of elements (considering that the rotation zero is the original list). This operation is referred to as circular permutation [15, p. 116];

3. Partial sums performs a series of adjacent additions. The brackets within a list (depicted in Figure 4) contain the elements to be added. They are displaced to the right in each turn of sums until they contain the last element. These additions start with two elements then three, four and so on until the brackets contain all elements to be added, thus returning the total sum. In each turn the resulted lists have fewer elements. Such an operation generates "familiar" rhythms, since the results are tight connected to the original seed list. A similar operation is referred to as "Summation Series" by Schillinger [15, p. 119]. This operation also can be seen as an inverse operation of partition. (Notice that other rounds of partial sums could be performed with the results obtained in the example depicted here, giving rise to new generations. Some results of such recursion are redundant however. Exclusion of identical lists would be needed in these cases.);

4. Products. A list can be multiplied by an integer, becoming expanded by this factor. If multiplied by 2, for example, it will get 2 times expanded, i. e., each of its elements will increase by a factor of 2. (Which is the numeric representation of the augmentation technique, used as a contrapuntal device. We will cover this correspondent relationship later in this paper.) Furthermore, our example list can be multiplied by another list. (We choose to use partitions of 4 in this example: (1,1,2), which is a form of 4P3 and (1,1,1,1), a form of 4P4). In this case we apply the distributive property, so that each value of the first list multiplies all the elements of the second list. Notice that this operation may not be commutative for all cases, i. e., $(1,1) * (2,3,4)$ is different to $(2,3,4) * (1,1)$. Finally, a list can be raised to an exponent. The squared list is depicted, which is the product of it by itself;

5. Phase shifting. Each entrance of the pattern is shifted one rhythmic unity to the right. In this case the basic unit is the 16th note. (Notice that some results coincide with rotations.) See for instance an interesting video[6] with a graphic representation of Steve Reich's Clapping Music. In this work Steve Reich uses a shifted result (Clap 2) against the original (Clap 1), causing a kind of musical interference of attacks. For other compositional purposes the shifted result can be also used as a new durational pattern;

6. Rhythmic complement. A second rhythm sounds in the empty spaces of the first. It can be seen as a boolean negation as the durations are represented as a series of 1 and 0. 1 for an attack unit and 0 for the unities between them. Then, each 1 becomes a 0 and vice versa. In Table 1 we show the steps of such operation.

Line 1 depicts the elements of the list $(3,3,4,3,3)$ as a grouping inside the 16 durations. Line 2 holds its representation using 1 for attack and 0 for non attack. Line 3 is the negation of line 2, i. e., the result of the binary NOT operation. Line 4 converts back from 1 and 0 notation (using R to symbolize rests in the place of zeros). Line 5 depicts a optional form, "tying" rests with its previous durations[7].

In Table 2 we suggest some of the operations on lists that can be strategically used for musical ends.

[6]"Steve Reich – Clapping Music (Scrolling)". Gerubach (Youtube channel). Available at <`https://www.youtube.com/watch?v=lzkOFJMI5i8`>. Access 10/01/2016.

[7]This operation can be correlated to bit shifts, a kind of bitwise operation.

Table 1: *Rhythmic complement operation steps*

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	3			3			4				3			3		
2	1	0	0	1	0	0	1	0	0	0	1	0	0	1	0	0
3	0	1	1	0	1	1	0	1	1	1	0	1	1	0	1	1
4	R1	1	1	R1	1	1	R1	1	1	1	R1	1	1	R1	1	1
5	R1	1	2		1	2		1	1	2		1	2		1	1

It could even be desirable to search for a set of "primitive" (low level) operations, such as subtract, that can be combined in order to generate other (high level) operations, such as derivate (which return the difference of the values of adjacent numerical elements). List operations also can be used nested. For instance, consider the nesting operations depicted below:

$$reverse(rotate_n(product(\mathrm{f};list)))$$

Table 2: *A sample of operations on lists*

accumulate	GCD–LCM	permute
append	intersect	power
(bitwise operations)	max	reverse
combine	min	riffle
derivate	mod	rotate
divide	mode	sort
factor	multiply	subtract
find	partition	sum

Table 3: *Some correspondences between composition techniques and list operations*

Augmentation	product(n;*list*)
Rotation	rotate(index;*list*)
Expansion	product(n;*list*)
[Pitch] Inversion (chromatic)	complement(12;list)
Retrograde	reverse(*list*)
Rhythmic complement	NOT(*list*)
[Phase] Shifting	bitShifts(index;*list*)
Transposition	add(n;list)

Where f is the factor that multiplies the list elements. Product's result are passed to Rotate, where n is the index of rotation. Finally, the result is retrograded by Reverse.

Working with operations on lists is also a way to summarize musical composition techniques. For example, take the correspondences depicted in Table 3.

Members in the right column represent the techniques used in music composition. Members in the left represent operations to perform analogous tasks. Notice that some of them need extra arguments, such as numbers and indexes (beyond the list itself). In the cases of `NOT` and `bitShifts` operations, the numeric list passed as argument should be first converted to a list of ones and zeros (see Table 1, steps 1 and 2).

It is convenient to open the use of the same operations on different musical qualities. In this way, much of the same set of operations can be applied over durations, intensity, pitch and timbre related lists.

Thus, we presented an attempt to gather an expanding collection of operations on lists. These lists are representations of values changing in time related to one or more musical qualities. Our goal is the generation (and variation) of pre-compositional material. Furthermore, to lay foundation for compositional pedagogy. Such an effort also casts a glimpse towards a creative theory of music composition.

So far we have been giving a systematic approach to Musical Composition, formalizing knowledge of its field by means of mathematical modeling. In extension, this also leads towards a next step in formalization. Namely, it is highly desirable that such an approach be assisted by a computer application (through which a composer can listen to and choose from computed results). In the next section, we will explore that possibility.

V. Synckers

My research project is entitled "Generation of pre-compositional material based on a computational interpretation of the Schillinger System of Musical Composition (SSMC)" [15]. Implementations based on the SSMC are feasible, for it also makes use of a mathematical approach[8] . A computer application was originally conceived as the main result of this research. It is devised to facilitate understanding of a variety of devices shown in SSMC. The application aims to benefit students and scholars who can make use of it according to their demands and musical purposes.

Oposmodus, Symbolic Composer and OpenMusic are among some of the softwares that could fit in the category we propose. Their disadvantages are due they are proprietary and do not offer versions to the Linux[9] operating system. From the academic context, we name similar works in implementing SSMC techniques as [12],[8], [5]and [4].

Two solutions are being developed in parallel. One application, called J-Syncker uses some of the devices presented in SSMC, making them available through a user friendly interface. J-Syncker is available to download at <`http://j-syncker.weebly.com/`>. Pd-Syncker is another solution developed by our team. Its development runs in parallel with J-Syncker. It is programmed in Pure Data. The advantage of this language is to be dedicated to multimedia processing (audio, video, Internet and MIDI devices). Pd-Syncker is available to download at <`http://pd-syncker.weebly.com/`>. Both solutions have been used as tools in Compositions Workshop courses given at School of Music of UFRN.

For the sake of space, next we will focus only to J-Syncker, showing how it works and how to perform some of the operations listed previously.

i. J-Syncker

The application is called J-Syncker for three reasons. First, the program was written in the Java programming language. It is common to see programs written in this language starting with the letter "J", for example, JFugue, Jedit, JabRef, etc. Second, the program deals primarily with the

[8]The Mathematical Basis of the Arts (MATHBART) is his published *magnum opus* [14]. It turns out that the SSMC is the musical branch of MATHBART.

[9]The Linux platform have being preferred for it is open source, free software, shared worldwide and benefit people who do not have affordable access to proprietary technology – offering an alternative to barriers imposed by such paradigm.

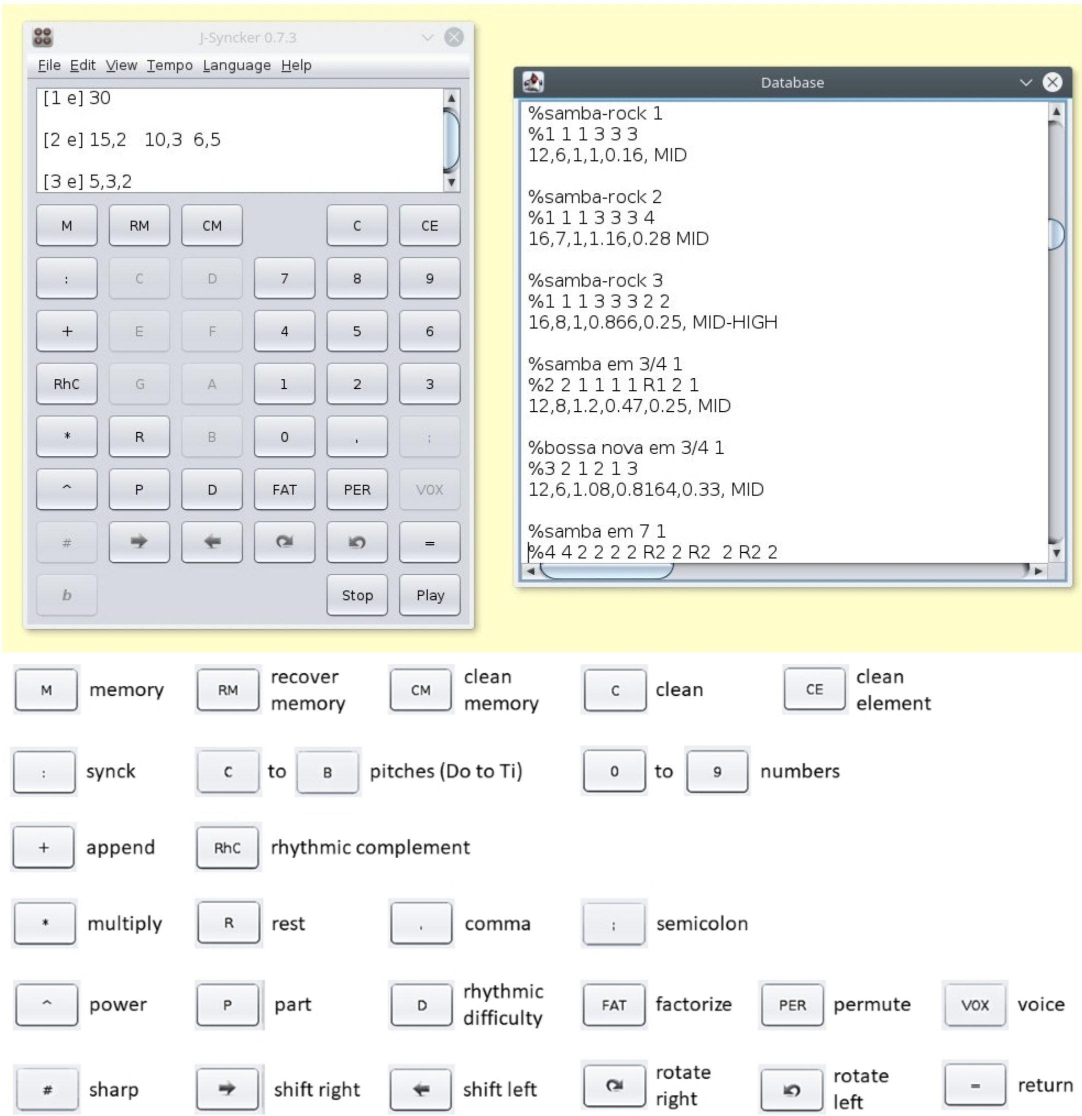

Figure 5: *J-Syncker's GUI, elements description and Database window.*

concept of synchronization (see [17]). Syncker was the short word chosen for synchronizer. Third, and in a more veiled way, the name of the application suggests a rhyme with Joseph Schillinger – or just J. Schillinger. [18] As it is written in Java, J-Syncker is multiplatform. It is free and open source.

J-Syncker has a user-friendly interface. Its design is inspired by the idea of the scientific calculator. This format is somewhat familiar and demonstrates features available on the GUI. Additionally, this kind of interface suggests the mathematical character underlying a compositional process (something that, from the outset, is not always apparent in music composition).

Next we will describe some of the functionalities from J-Syncker's GUI, depicted in Figure 5.

To begin with, it is possible to enter any durations list, typing directly in the format of a **comma separated list**. This means that one comma separates each two operands (integers, in this case). Use R button (or type "r") to enter a rest, followed by its duration.

To **clean** elements or the entire display, click the buttons CE and C respectively (or backspace and delete keys respectively).

To select, use the mouse to click and drag (double click or Ctrl+A selects all). Copying (Ctrl+C), cutting (Ctrl+X) and pasting (Ctrl+V) are also allowed.

To choose from **presets lists** of rhythmic durations, go to View in menu bar then choose Database (or simply Shift+D). Such lists are based on rhythms found in [13], [15] and [3]. (This database is used to evaluate rhythmic complexity. The complexity of a given numerical list representing rhythmic durations may be evaluated by pressing the "D" button. For more on this see [2])

Click the button (or type the capital letter) "M" and a number (from 0 to 9) to store data in **memory**. The same procedure can be used to recover and clean memories, using RM button (or Alt+R) and the CM button (or Alt+C) respectively.

To **append** one list to another, enter the first, click the plus button (or type the "+" character) then click the equal button (or hit Enter key).

The partition operation is performed by entering an integer and pressing P button (or typing letter "p") then hitting Enter key. The results are shown indexed from 1 to $n-1$ elements. One can use the form nPm, where n is an integer and m the number of parts (also hitting Enter key for the result be evaluated). In this case only partitions with m parts will be depicted. The resultant format is similar to the one depicted in J-Syncker's display in Figure. Actually, this one generates the output of the **factorization** (or multiplicative partition). To achieve that result, click the FAT button (or type letter "f") instead of P. Likewise, use the format nFm to obtain only multiplicative partitions of n with m parts. (The output showed in the display of J-Syncker in Figure 5 was obtained after entering 30F + Enter. To show only results with two values 30F2 + Enter would suffice.)

PER button **permutes** the elements of a list previously entered.

The two buttons with circular arrows (or left and right arrow keys) rotate a list in both directions. The two buttons with straight arrows (or shift+Left/Right arrow keys) shift a list in both directions. The basic unit is the step of shifting. (This operation resembles the correlated bitwise operation.)

The asterisk button performs the **product** operation. Its factors can be integers and/or lists of integers. Press the equal button to evaluate the result.

The caret button (or typing the caret character) is used to perform **exponentiations**. A list of integers can be passed as the operand in this operation. (Pressing equal button is also needed.)

The so-called **rhythmic complement** is performed by hitting the RhC button (or pressing the minus character). Performing it again over the result gives back the original.

Click **Play** button (or hit space bar) to listen the results of the various edits and operations in loop. (The same procedure can be used to **pause** and resume playing.) **Stop** button (or Esc key) stops the loop.

Other functionalities are also found[10] in the J-Syncker menu bar. The drop down menus are described as follows, according to Figure 6, in clockwise direction.

From the File menu, it is possible to save results as MIDI or text formats. MIDI files can be opened in other applications for editing. The text file format were conceived to report the musical arguments used in some operations. (This functionality is still in need of bug fixes.) Besides MIDI, it is possible to open some results automatically in the music notation software MuseScore [16],

[10]GUI buttons not described are related to operations not covered here. Synchronization of lists of durations, pitches, meter, voices and pitch contour deserve a specific report, thus lying beyond the scope of this paper. As an introduction to the subject see [17].

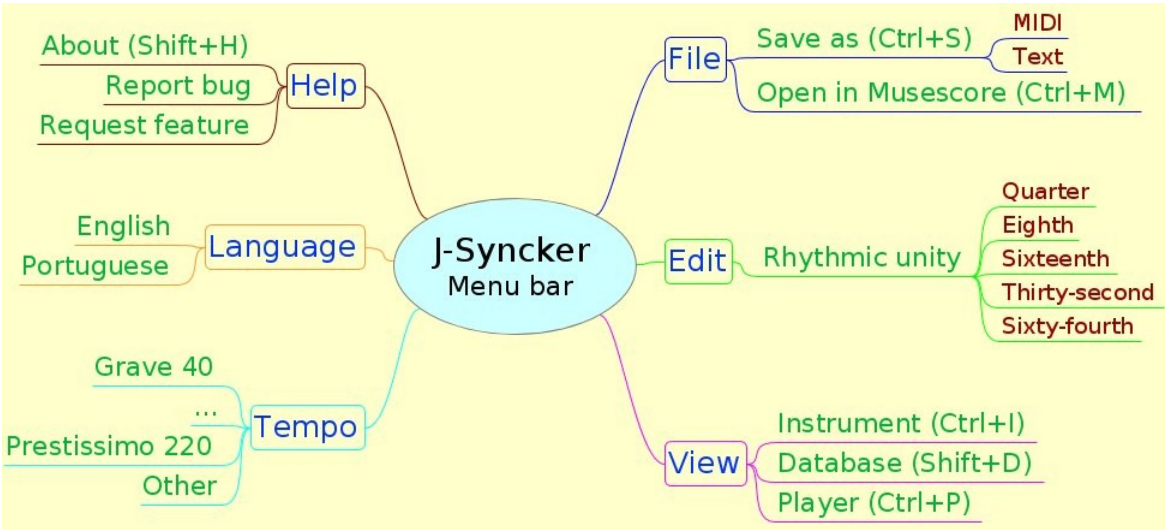

Figure 6: *J-Syncker's menu bar elements.*

once it is installed in the machine. (Currently, this is only possible in the Linux platform, due to MuseScore version updates.)

From the Edit menu, it is possible to set the basic rhythmic unit. That is, a number one in a list could be represented by one of the following rhythmic notes: quarter, eighth, sixteenth, thirty-second or sixty-fourth.

From the View menu, it is possible to change the MIDI program name, as a away to set a new timbre from a synthetic bank of instruments (or just Ctrl+I). The Database of rhythm attributes used in complexity measurement is made available also from this menu (or Shift+D). The player functionality was intended to be implemented in future versions.

From the Tempo menu, it is possible to set the speed of playback. One can choose from preset options (from *Grave to Prestissmo*) or enter a number through the "Other" option at the end of the list of tempos.

From the Language menu, it is possible to translate GUI elements between Portuguese and English.

From the Help menu, it is possible to retrieve information about the software (Shift+H), report a bug or request a feature.

In the next sections, we will bring back some of the previous discussion topics for further comments and lastly point to perspectives and future works.

VI. Discussion

The term "musical quality" was used here as an alternative to music element and music parameter. Element is too generic, meaning a constitutive part of a collection, function or system. Parameter has been used in a variety of definitions (even in the sense of quantity). Etymologically, it came from Greek, being an articulation of two words, *para* (beside) and *metron* (measure). Thus, a reference to a given meter. [7] It is defined in multiple ways such as "a rule or limit", "an arbitrary constant", "an independent variable" and "a set of physical properties". [9] The latter definition is the closer to its use in Music Theory. With that in mind, the term was put aside in order to avoid ambiguity and lack of clarity. Another reason to choose "musical quality" was to combine, compare or cooperate with the term "musical quantity". The aim was to bring forth the relation of these two terms, regarding manifestations of a musical stimulus, i. e., quality and quantity.

In Music Theory literature, numbers are commonly used to represent pitch related qualities (such as [6] and [20], among others). Still other authors expanded to other quality branches (such as [15], [10] and [1]).

Once quantitative variations described by musical qualities are numerically represented, they can be subjected to a variety of mathematical resources. The values plotted from a numeric list can also be seen as contours, bringing the underlying variation to an imagery media, helping prominently visual subjects.

It is important to highlight that this approach can be applied to any musical quality (durations, pitches, timbres, intensities and their combinations). It encourages students to face early on the elements assembled in a music composition. Such a comprehensive overview certainly brings benefits to "multi-quality" writing. (It may contribute to overcome the habit of presenting a score with only pitches and durations written.)

Typical techniques from the field of Music Composition [19] can be "re-presented" by means of operations on lists. This can be useful in aiding the process of obtaining variety out of the same musical seed.

Once established, such a paradigm makes clear the desire for an interface between Music and Mathematics. It can be used theoretically first in the form of a model. Later, it can be used technologically, in the form of a computer application.

The type of specification conducted here helps guide developers in the designing phases of a software project. J-Syncker is one example. Thus, a process for dealing with the articulation of music composition and mathematical modeling may constitute steps such as, formalizing, pseudocoding, coding and testing.

Listen and choose from computed results. This allows focus on the appreciation of computed results. The speed allows the composer to select from a large quantity of outputs.

That being said, one thing still needs to be considered: technologies come and go. As we have seen, they fall out of fashion, are discontinued or even became proprietary. Above all, what is worthy of attention is to focus on the paradigm. It is the paradigm that has a bigger chance to survive.

VII. Future Work

This paper was intended for students and scholars, as well as people interested in the intersection of Music Composition and Mathematics. By doing this, it also takes a step towards increasing the collaboration among such individuals.

In a broad perspective, we intend to continue developing pedagogical content in order to design a knowledge base (be it in a form of book, site or the like). Additionally, courses (and even curricula) based on this content is a desirable consequence. It will be possible also to adapt this approach to teaching existing courses (such as Scorewriting, Arranging, Analysis and so on.)

More specifically, the J-Syncker project has some known bugs to be fixed. It succeeded as a proof of concept and is being used as a central tool in musical courses at the School of Music at UFRN.

Pd-Syncker project needs the porting from Pd-Extended (discontinued) to Pd-Vanilla to be finished. The website needs to be translated to English. Coding externals in the C language also will bring benefits from the point of view of robustness.

A LibSyncker project has being designed to implement numerous operations on lists in a systematic way. We intend to update the lineage of both Synckers. This will demand a solution to work in a clever and broad way, to work with lists with various musical qualities, further advancing the concept of music synchronization.

References

[1] BABBITT, M. (1962). Twelve-tone rhythmic structure and the electronic medium. **Perspectives of New Music**, pp. 49-79.

[2] BEZERRA, G. S. (1962). **Elaboração de um aferidor de complexidade rítmica utilizando redes neurais artificias.** Monografia (Graduação em Ciência da Computação) - Universidade Federal do Rio Grande do Norte, Universidade Federal do Rio Grande do Norte, Natal.

[3] BOBBITT, R. (1959). The physical basis of intervallic quality and its application to the problem of dissonance. **Journal of Music Theory**, vol. 3, no. 2, pp. 173-207.

[4] DEGAZIO, B. G. (1988). Fractal Geometry and the Schillinger System. **Proceedings of Diffusion!**, Jean-Francois Denis, ed., Montreal, Canada.

[5] DEGAZIO, B. G. (2004). The Transformation Engine. **Proceedings of ICMC**.

[6] FORTE, A. (1973). **The structure of atonal music.** Yale University Press.

[7] HARPER, D. (2016). **Online etymology dictionary**. Available at <`http://etymonline.com/`>. Access 10/18/2016.

[8] JONES, B. (2011). **The Composer's algorithmic assistant: Based on the Schillinger System of Musical Composition.** Phd Thesis. Computer Science and Information Systems. Pace University.

[9] MERRIAN-WEBSTER (2016). **Online dictionary.** Available at <`http://www.merriam-webster.com`>. Access 10/18/2016.

[10] MESSIAEN, O. (1956). **The technique of my musical language**, trans. John Satterfield. Paris: Alphonse Leduc, vol. 46.

[11] POLIVAEV, D. (2016). **Freeplane:** free mind mapping and knowledge management software. 1.5.16. Written in Java. Opensource. Multiplatform. 2016. Available at <`http://freeplane.sourceforge.net/`>. Access 10/18/2016.

[12] RANKIN, M. (2012). **A computer model for the schillinger system of musical composition**. B.Sc. Thesis in Computer Science. The Department of Computer Science. Australian National University.

[13] ROCCA, E.(c1986). **Ritmos brasileiros e seus instrumentos de percussão.** Rio de Janeiro: Escola Brasileira de Música.

[14] SCHILLINGER, J. (1948). **The mathematical basis of the arts.** New York: Philosophical Library.

[15] SCHILLINGER, J. (2004). **The Schillinger system of musical composition.** Harwich Port, Massachusetts: Clock & Rose.

[16] SCHWEER, W. et al (2016). **MuseScore.** 2.0.3. Written in C++, Qt. Opensource. Multiplatform, 2016. Available at <`http://www.musescore.org/`>. Access 10/18/2016.

[17] SILVA, A. R. (2004). Estendendo o conceito de sincronização presente na teoria do ritmo do Sistema Schillinger de Composição Musical. In: Congresso da ANPPOM, 20., Florianópolis. **A Pesquisa em Música no Século 21: trajetórias e perspectivas.** Florianópolis: [s. n.], pp. 61-68.

[18] SILVA, A. R., BEZERRA, G., and GAGLIANO, G. A. (2014). **J-Syncker**, versão beta (0.7.3), aplicativo para geração de material pré-composicional baseado em uma interpretação computacional do "Sistema Schillinger de Composição Musical", 2014. Available at <http://j-syncker.weebly.com/>. Access 2016/12/02.

[19] SOLOMON, L. (2002). **Variation techniques for composers and improvisors.** Website, 2002. Available at <http://solomonsmusic.net/vartech.htm>. Access 2016/07/26.

[20] STRAUS, J. N. (1990). **Introduction to post-tonal theory.** Englewood Cliffs, NJ: Prentice Hall.