COMPLEXITY DRIVEN RECOMBINATION OF MIDI LOOPS

George Sioros
University of Porto (FEUP) and INESC - Porto
Rua Dr. Roberto Frias, s/n 4200-465 Porto
Portugal
gsioros@gmail.com

Carlos Guedes
University of Porto (FEUP) and INESC - Porto
Rua Dr. Roberto Frias, s/n 4200-465 Porto
Portugal
cguedes@fe.up.pt

ABSTRACT

An algorithm and a software application for recombining in real time MIDI drum loops that makes use of a novel analysis of rhythmic patterns that sorts them in order of their complexity is presented. We measure rhythmic complexity by comparing each rhythmic pattern found in the loops to a metrical template characteristic of its time signature. The complexity measure is used to sort the MIDI loops prior to utilizing them in the recombination algorithm. This way, the user can effectively control the complexity and variation in the generated rhythm during performance.

1. INTRODUCTION

Devising different strategies for generating rhythm in real time is one of the goals of the project "Kinetic controller driven adaptive music composition systems". After proposing the use of Genetic Algorithms [1], and proposing a method for generate a metrical rhythm performance stochastically [2], we now propose a simple yet effective method for recombining MIDI drum loops of a certain style (such as those available in Apple's GarageBand). We developed a measure of rhythmic complexity in order to sort the loops prior to utilizing them in the recombination process. In this Max/MSP [3] application, the user can recombine in real time, with different degrees of complexity, a batch of MIDI drum loops in order to get non-excessively repetitive combinations of loops during a performance. The user can control the amount of variation in recombination during performance, as well as different degrees of complexity.

2. THE ALGORITHM

Recombinance is an effective technique to generate music according to a certain style [4]. The kin.recombinator application generates rhythmic patterns by recombining existing ones. The recombination process consists of playing back MIDI drum loop files by selecting portions of these files at regular intervals. An analysis of the files is performed prior to the recombination, in order to sort them according to their complexity and, in this way, better control the resulting rhythms.

The algorithm can be divided in two phases. In the first phase a set of MIDI drum loop files input by the user are analyzed and sorted according to how complex they are, from the simplest to the most complex. This complexity measure is based on a new method for measuring syncopation, by comparing the patterns against a template characteristic of their meter.

In the second phase, the patterns are played back and recombined. A new pattern is selected for playback on every beat. Playback is performed in a cyclic way; it restarts when reaching the end of the file. When a new a file is selected, playback always continues at the beat from where it was left in the previous file, always preserving the metrical position. The user controls the complexity of the resulted rhythm and the amount of variation by determining which patterns get selected for playback in an easy and intuitive way based on their order of complexity.

2.1 Calculation of the complexity scores and sorting of MIDI drum loops

Various approaches for measuring complexity in rhythmic patterns exist, such as pattern matching techniques [5], rhythmic syncopation measures [6] and analysis of the mathematical or geometrical properties of the patterns [6][7]. Here, we define a new one, which is based on the same principle as G. Toussaint's *metric complexity* [6], which is a comparison of a rhythmic pattern against a template characteristic of its meter. Unlike Toussaint's approach that uses the template as a way of calculating the metrical accents, we use the template as the fundamental tool for analyzing and defining relationships between the events comprising the pattern. Moreover, unlike most methods for measuring complexity, which use a binary representation of the patterns and ignore the amplitudes of the events that comprise the pattern, we take into account the relative amplitudes of the events in our calculation.

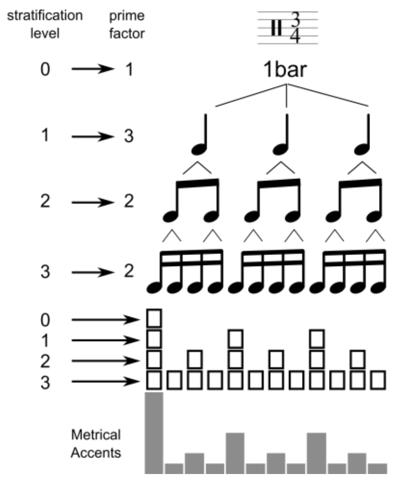

Figure 1. Stratification of a 3/4 meter to the 16th note level. Each pulse belongs to a stratification level and all lower ones.

The user provides rhythmic patterns in the form of a collection of MIDI files. The MIDI files are read and are quantized according to a fixed quantization grid. This way each measure in a pattern is subdivided into pulses according to the time signature and the quantization grid. The quantization grid value we use is the 32nd note. Deviations from that grid are considered to be micro-timing deviations and they are not treated in the current analysis. An amplitude value is assigned to each pulse, according to the MIDI velocities found at that time position after the quantization.

A metrical template that defines metrical hierarchy is constructed by stratifying the meter found in the MIDI files. This template consists of the metrical levels which comprise the meter. A “metrical accent” value is calculated and assigned to each pulse according to the metrical level it belongs to. Each rhythmic pattern found in the MIDI files is compared against the metrical template yielding a separate score for each pulse in the pattern. The result is further filtered by the aforementioned values of the metrical accents. The calculated scores can be thought of as a measure of how much each pulse contradicts the metrical structure described by the template and, in this sense, how much each pulse contributes to the syncopation of the pattern. Finally, a complexity score is assigned to each MIDI file taking into account, in addition to the syncopation, the density of MIDI events in each file.

2.1.1 Constructing rhythmic patterns from MIDI files

The user provides the MIDI drum loops as a set of MIDI files with the same bar-length and time signature. The MIDI note-on events are extracted from each file and their positions are quantized to a 32nd note grid. The lists of amplitudes for each pulse in the meter (i.e., their velocity value) make the rhythmic patterns to be constructed. It is common in MIDI drum loops that each different MIDI note number corresponds to a different timbre, e.g. one note number for a kick drum sound and another for a hit cymbal. We construct a different rhythmic pattern for each MIDI note number according to the MIDI velocity and time position of any note-on events corresponding to that note number. The final complexity score calculated for the MIDI file is obtained by averaging the scores of each rhythmic pattern constructed from the file.

Two more ways of translating MIDI note-on events into rhythmic patterns are provided: one for single-timbre MIDI instruments, where all notes correspond to the same timbre with different pitch, and one for a general MIDI drum library. For single-timbre instruments, MIDI note numbers are ignored and a single rhythmic pattern is constructed by taking the maximum MIDI velocity in each pulse. For the General MIDI drum library case, different groups of MIDI note numbers correspond to different patterns, e.g. all hi-hat notes are grouped together to construct one rhythmic pattern while other notes like bells are treated each one separately.

2.1.2 Constructing the metrical template

A metrical structure as the one described by F. Lerdahl and R. Jackendoff [8] can be constructed for each meter found in the MIDI files. It must be noted that in order for the metrical structure to be meaningfully in relation to the rhythmic patterns extracted from a MIDI file, we must assume that the meter of the template is in fact the meter of the patterns; that is, the time signatures found in the MIDI files are the actual time signatures of the drum loops contained in the files.

The meter is stratified into metrical levels in a hierarchical manner so that each pulse belongs to a specific metrical level and all lower ones. In order to stratify the meter the number of pulses is decomposed into prime factors (see Figure 1). Each prime factor describes how each stratification level is subdivided. Different permutations of the prime factors describe different metrical hierarchies. The stratification process is described in detail in [9]. A simplified version can also be found in [6]. A metrical accent value is assigned to each pulse based on the stratification level *i* it belongs to, following an exponential equation:

$$M_i = 0.5^i \tag{1}$$

Since the number of pulses in a certain meter is determined by the quantization grid, the lowest stratification level will always correspond to that grid. For the sake of sim-

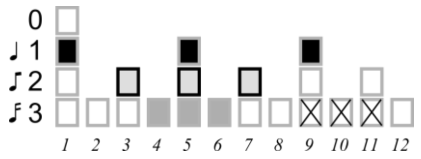

Figure 2. Pulse number 5 belongs to metrical levels 1, 2 and 3. Pulse 10 belongs only to level 3.

plicity, in the examples presented here we omitted the 32nd note metrical level.

2.1.3 Comparing the rhythmic patterns to the template

The comparison of a rhythmic pattern to a metrical template is essentially a process of spotting the pulses in the rhythmic pattern that contradict the prevailing meter described by the template. In that sense, these pulses are mainly responsible for any syncopation present in the rhythmic pattern and the result of the comparison is a measure of that syncopation. Eliminating the events that occur regularly on the beat in some metrical level helps distinguishing the syncopating pulses. Events that occur regularly on the beat do not generate syncopation. The remaining events would be isolated ones, that mostly occur in low metrical levels, i.e. in "off-beat" positions, and that contributes to the complexity more actively. For example, a cymbal hit on every quarter note beat with more or less the same amplitude contributes less than a snare that happens only at specific off-beat positions and contributes more to a more complex rhythm.

In order to define how each pulse contributes to a steady beat, its relation to the rest of the pulses must be examined. We examine the relations between the pulses of the pattern in light of the metrical template, taking advantage of its hierarchical character. Each metrical level is examined separately, so that each pulse in a pattern is assigned a separate score for each metrical level it belongs to. A low score in a metrical level signifies that the pulse contributes to a steady beat in that level and therefore does not contradict the meter, e.g. a quarter note surrounded by equally loud quarter notes has a low score in the quarter note metrical level, while a loud quarter note with no neighbors will have a higher score. After examining all metrical levels, the minimum score for each pulse is kept.

The score is calculated as the average difference of the amplitude of the pulse under consideration from the amplitudes of the neighbor pulses in each metrical level. In the example of Figure 2, pulse number 5 gets three scores, one for each metrical level it belongs to, namely that of the quarter note (1), the eighth note (2) and the 16th note (3). Each score is the average of the two differences of amplitudes between pulse 5 and i) pulses 1 and 9 for metrical level 1, ii) pulses 3 and 7 for level 2 and, finally, iii) pulses 4 and 6 for level 3. On the other hand, pulse 10 gets only one score (metrical level 3) arising from the two differences from pulses 8 and 11. Negative differences are always set equal to zero.

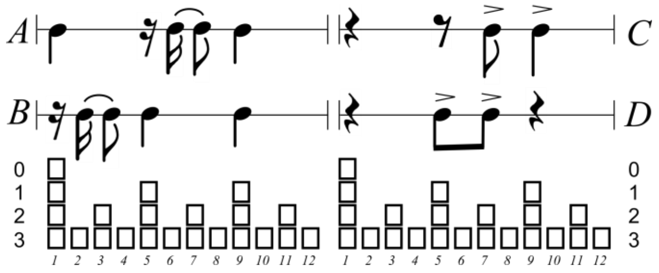

Figure 3. Left: The contradiction of the rhythmic pattern to the meter is greater in pattern *B* than in *A* since pulse 1 belongs to a higher metrical level than pulse 5. Right: A loud event before an accent (C) enforces the accent while a loud event after an accent (D) weakens the accent.

One important feature of the metrical structure of the template is the alternation of metrical levels, in other words, the highest metrical level of a pulse is always different from the ones of its immediate neighbors. As a consequence, pulses in low metrical levels are always surrounded by pulses in higher levels and in most cases their neighbors also belong to different metrical levels (see pulse 10 in Figure 2). An isolated event in such a pulse produces a strong syncopation feel. This syncopation is stronger when the difference of the metrical levels of the pulses is larger. Compare, for example, any of the pulses 4, 6, 8 or 10 to pulses 2 or 12. In the absence of an event in pulse 1, pulses 2 and 12 create a stronger contradiction to the meter because pulse 1 belongs to the highest metrical level (see Figure 3, A and B). Having this in mind, we introduced a weighting factor in the amplitude differences calculated above. This factor is proportional to the difference of the highest metrical level of the two pulses that the amplitudes are taken from. This way the amplitude difference between pulse 2 and 1 has more weight than the one between 6 and 5. Similarly, the amplitude difference between pulse 5 and 1 has more weight than that between 5 and 9.

An important aspect of rhythmic patterns is the direction in which they are always performed. Time in music, as in everything else, flows in only one direction. Pulses succeed one another in a specific order. The relation of a pulse to its previous pulse is not equivalent to that to the following pulse. Two equally loud events one after another create the impression of an accent on the second event rather than on the first. Let us return to our example template of Figure 2.

Consider the relation between pulse 7 and pulses 5 or 9. A loud event on pulse 7 affects differently a loud event on pulse 5 from one on pulse 10 (see Figure 3, C and D). In order to take into account this fact in our evaluation of how much an event contradicts the meter, the weights discussed in the amplitude differences calculations need to be modified, so that a smaller weight is to be given to the amplitude difference with the previous pulse than with the following one.

The various pulses in a rhythmic pattern have a different potential in contradicting the prevailing meter, or syncopating, depending on which metrical levels they belong to. This syncopation potential can be thought of as the opposite of the metrical accent, which essentially is the potential of a pulse to contribute to a steady beat. A loud event in a pulse that belongs to a high metrical level (high metrical accent), does not have a lot of "chance" of contradicting the meter, even if it is isolated without any events in its vicinity. The scores calculated above represent how much a pulse contradicts the meter with respect to its relation to its neighbors but do not take into account this syncopation potential. We therefore multiply the scores previously calculated by a factor proportional to the inverse of the metrical accent of equation (1). This way a pulse that belongs to metrical level 0 could never contradict the meter, irrelevant of if an event exists in this or any other pulses of the meter. Of course, the absence of an event in a high level pulse creates the possibility for an event in some other pulse, probably of a low metrical level, to produce a strong syncopation, but this is reflected on the syncopation potential of the low metrical level pulse.

The last step taken in the calculation of our syncopation measure is to sum the scores of all pulses in the pattern and normalize the result. Normalization is performed by dividing the result by the maximum possible sum for the meter and bar-length. This maximum is calculated by comparing against the metrical template a pattern in which all pulses of the lowest metrical level have maximum amplitudes and all other pulses have zero amplitude.

As it was described in the previous paragraphs, the weights used in the amplitude differences are calculated according to the metrical levels of the pulses and to whether the difference is taken from the previous or the following pulse. We set the exact weights empirically, by experimenting with various combinations. For differences between pulses of the same metrical level the weight was set to be half of that between pulses which belong to the two extreme metrical levels. The difference from the previous pulse was set to the 80% of the weight of that from the next pulse.

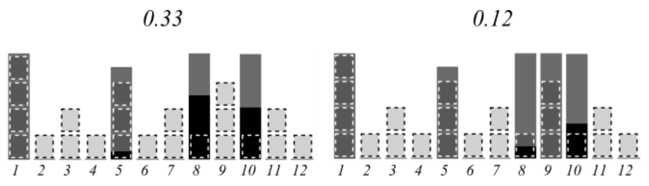

Figure 4. Two patterns (grey bars) compared to the same metrical template (dashed squares). The black bars represent the score of each separate pulse. The total syncopation score is shown above each pattern.

2.1.4 Examples of measuring syncopation

In this section an illustrative example of the method for measuring syncopation is given. A short evaluation of the method follows, by measuring the syncopation of six clave and bell rhythms of the African, Brazilian and Cuban music. A similar comparison of other syncopation and complexity measures based on these patterns can be found in [6].

The two patterns of Figure 4 are compared against the same metrical template, namely that of a 3/4 meter. The only difference between the two patterns is found in pulse 9, a pulse that belongs to a high metrical level. In pattern *A* pulse 9 is silent while in pattern *B* it has maximum amplitude. Although this difference does not cause any changes in the score of that pulse, it affects drastically the scores of the other pulses in the patterns. The two immediate neighbors, pulse 8 and 10, both have maximum amplitudes. When no event exists in pulse 9, both pulses have high scores since no events exist in their vicinity, with that of pulse 8 being a little higher. In pattern *B*, the amplitude of pulse 9 causes both scores of pulses 8 and 10 to drop. This drop is larger for pulse 8, so that, pulse 8 now has a smaller score than pulse 10. This inversion in the relation of the scores of the two pulses is the result of the amplitude weights used. In the absence of an event in pulse 9, the previous pulse creates a stronger syncopation. In the presence of high amplitude in pulse 9, the following pulse weakens the accent, while the previous one tends to enforce it. Although the difference is small, it can be of importance when sorting drum loops of the same music style with little differences.

In the absence of an event in pulse 9, a small contribution in the total syncopation of pattern *A* arises in pulse 5. Pulse 5 and 9 are related through the quarter note metrical level. The contribution is small for two reasons. On one hand, because of the high amplitude of pulse 1 which belongs to a higher metrical level and therefore gets a higher weight than pulse 9. On the other hand, pulse 5 belongs to a pulse with a high metrical accent, so that its syncopation

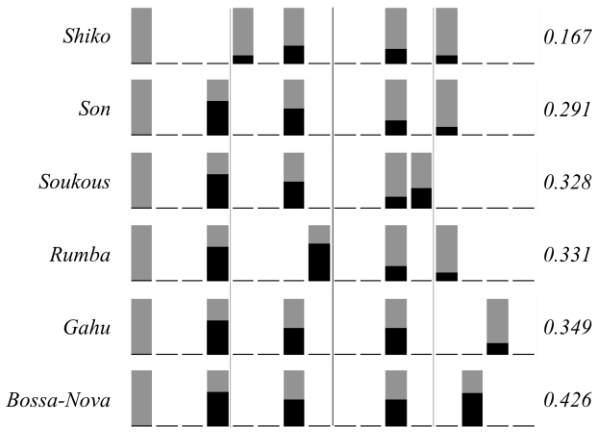

Figure 5. The syncopation of the six fundamental 4/4 clave and bell patterns is measured.

potential is low.

In order to evaluate the syncopation measure described above, we measured six clave and bell rhythmic patterns, namely the Shiko, Son, Soukous, Rumba, Gahu and Bossa-Nova. These patterns are all five-note patterns with 4/4 time signature and are some of the most frequently used in the African, Cuban and Brazilian music.

In Figure 5 the six patterns are presented together with their syncopation scores. Dynamic accents were not considered; all pulses have either maximum amplitude or are completely silent. The black bars represent the relative contribution of each pulse to the total syncopation score. The scores obtained by the calculations seem to agree with our experience that Shiko is the easiest pattern, Rumba is of medium complexity and Gahu and Bossa-Nova are amongst the most difficult to perform. The order from simple to complex is also in agreement with the cognitive complexity measure proposed by J. Pressing (see [5] and [6]).

2.1.5 Calculating the complexity of a MIDI file

The complexity of a drum loop can be thought of as a vector in a two dimensional space, where one dimension is the density of the events and the other is the syncopation. The length of the vector is the complexity score. The syncopation measure is already normalized and can be directly used as one of the coordinates of the vector. The density of events is calculated as the sum of all the MIDI velocities found in the MIDI file. This sum represents an effective density, since it does not correspond to the number of events in the pattern. This number needs to be normalized before it can be used as a coordinate in our complexity plane. We normalize the density by dividing with the largest density value in the collection of MIDI files provided by the user. The total complexity of a file is then calculated as:

$$Complexity = \sqrt{density^2 + syncopation^2} \qquad (2)$$

This is the value used to sort the MIDI files, from the most simple to the most complex.

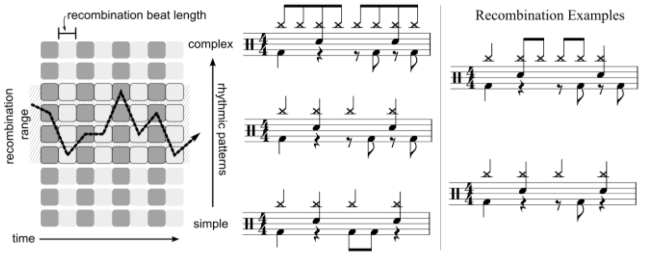

Figure 6. The rhythmic patterns, sorted from simple to complex, are selected for playback at regular intervals. An example of three patterns (middle) and their recombination (right) for two complexity values, simple (bottom) and complex (top) are shown.

2.2 Recombining the rhythmic patterns

The sorted MIDI files form a two dimensional space, where the vertical dimension represents their order of complexity and the horizontal represents their evolution in time (see Figure 6). All files share the same time signature and have the same bar-length and, therefore, are perfectly aligned.

A global transport controls the current playback position which is common to all files. The tempo of each file is ignored and the playback follows the transport's tempo, controlled in real time by the user. Playback is performed in a loop. At every beat, a new file is randomly selected and playback continues in this file at the current transport's position (see Figure 6), preserving always the metrical position. The duration of the recombination beat is defined according to the time signature, e.g. in a 4/4 meter, the beat would correspond to the quarter notes.

Instead of selecting a file out of the whole collection of the provided MIDI files, the selection process is restricted to a smaller collection of files. During the performance, the user controls the resulted rhythm by controlling in real time the range of files, from the simplest to the most complex that can be selected for playback. Increasing the range leads to more variation in the resulted rhythm, while moving the entire range vertically to more or less complex patterns controls the complexity of the rhythm.

The output of the recombination algorithm undoubtedly depends on the provided MIDI files. In order for the output to be coherent, all files should belong to the same music style. When the files have a similar structure, either a large scale structure consisting of several bars, or at the beat level, this structure will be reflected also in the outcome of the recombination. This comes about as a direct result from using a global transport to control playback.

3. MAX/MSP APPLICATION

The algorithm has been implemented as the *kin.recombinator* Max/MSP application. A collection of Max/MSP externals, java classes suitable to be loaded to the mxj Max/MSP object and Max/MSP abstractions were developed as parts of the application.

The user drags and drops a folder containing MIDI files into the Max/MSP application. The files are automatically sorted and the global Max/MSP transport controls playback. The user can graphically control the range of files being recombined at any one time with a range slider like the one in Figure 7.

The MIDI files are read and quantized to the 32nd note level by the java class *kinMIDIFileReader*. Rhythmic patterns are constructed in the form of lists of amplitudes and are passed together with the respective time signatures to a subpatch where the effective density and syncopation score are calculated by the *kin.OffBeatDetector* Max/MSP external.

The score is then stored in a collection object. After finishing calculating the scores for all the files, the files are sorted according to their scores.

The *kin.RecombineMIDIFiles* abstraction is performing the playback. It selects at every beat a new file for playback which is read by the *kin.MIDIFileReader*.

The *kin.recombinator* Max/MSP application, as well as a Max/MSP patch for testing out the syncopation measure can be downloaded at the group's web site:
http://smc.inescporto.pt/kinetic/

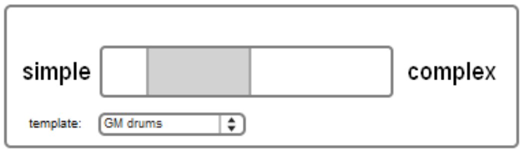

Figure 7. The kin.recombinator user interface. During the performance the complexity and amount of variation can be controlled graphically with a range slider.

4. ACKNOWLEDGMENTS

This research was done as part of the project "Kinetic controller driven adaptive music composition systems", (ref. UTAustin/CD/0052/2008), supported by the Portuguese Foundation for Science and Technology for the UT Austin| Portugal partnership in Digital Media.

5. REFERENCES

[1] G. Bernardes, C. Guedes and B. Pennycook: "Style emulation of drum patterns by means of evolutionary methods and statistical analysis." *Proceedings of the Sound and Music Computing Conference (SMC).* Barcelona, Spain, 2010

[2] G. Sioros and C. Guedes: "Automatic Rhythmic Performance in Max/MSP: the kin.rhythmicator", *Proceedings of the International Conference* on *New Interfaces for Musical Expression*, Oslo, Norway, 2011

[3] http://www.cycling74.com

[4] D. Cope: *Experiments in Musical Intelligence*, Middleton, A-R Editions, 1996

[5] J. Pressing: "Cognitive complexity and the structure of musical patterns", *Proceedings of the 4th Conference of the Australian Cognitive Science Society*, Newcastle, Australia, 1997

[6] G. T. Toussaint: "A mathematical analysis of African, Brazilian, and Cuban clave rhythms", *Proceedings of Bridges: Mathematical Connections in Art, Music, and Science*, Towson University, Baltimore, Maryland, July 27-29, 2002

[7] F. Gómez, A. Melvin, D. Rappaport, and G. Toussaint: "Mathematical measures of syncopation", *In BRIDGES: Mathematical Connections in Art, Music and Science*, p. 73–84, Jul 2005.

[8] F. Lerdahl & R. Jackendoff: *A Generative Theory of Tonal Music*, Cambridge, The MIT Press, 1996

[9] C. Barlow: "Two essays on theory", *Computer Music Journal*, 11, 44-60, 1987