Transforming musical rhythms: meter and syncopation

George Sioros[1] and Carlos Guedes[2]

[1] Faculdade de Engenharia da Universidade do Porto
gsioros@gmail.com

[2] NYU Abu Dhabi
cag204@nyu.edu

Abstract. Syncopation is a rhythmic phenomenon found in various musical cultures. In this paper, we present a formalized model of syncopation. It consists of a limited set of simple rhythmic transformations that take the form of displacement of musical events. The transformations are based on fundamental features of the musical meter and syncopation, as seen from a cognitive and a musical perspective. Starting from a binary pattern the model generates a large number of output patterns by applying a series of the transformations. The patterns are then organized in tree structures that can be used both for analysis and generation of syncopation.

Keywords: syncopation, transformations, meter, rhythm, generation, analysis

1 Introduction

Rhythmic syncopation is essential for several and diverse styles of western music, as well as for certain non-western music. It often produces rhythmic complexity and tension [1, 2 p. 310, 3]. Definitions describe syncopation as a contradiction to the prevailing regularities of the rhythm commonly expressed in the musical meter [4]. Musicological definitions describe the concept of syncopation [4, 5] in more detail while musicians focus on the technique. More formalized definitions approach it as a matter of magnitude [6–10] and define metrics for measuring the amount of syncopation. Other studies are limited to specific cases [11] or music styles [6, 12].

Recently we presented an algorithm that allows for the manipulation of syncopation in binary patterns [13]. Here, we extend the algorithm to a set of formalized generic transformations that can analyze, generate and manipulate the syncopation in binary patterns. The transformations are based on the cognitive aspects of the phenomenon related to metrical expectations [2].

Using the transformations, one can generate a multitude of rhythms starting from a single pattern in a formalized and systematic way. The generated patterns are naturally organized in tree structures. Patterns belonging in the same tree originate from the same root, i.e. they originate from the same non-syncopating pattern. The tree structure can be useful in developing models for clustering rhythms together or in defining measures of rhythmic similarity and distance.

In section 2, we provide with a brief description of the main syncopation definitions and measures related to our model. In section 3, we describe a way of automatically constructing a metrical template and how it must be adapted for modeling syncopation. In section 4 we describe the transformations for binary patterns and we present the concept a syncopation tree as a structure for organizing rhythmic patterns. Finally, in section 5, we present the main conclusions of this study.

2 Definitions of Syncopation

Musical meter as a cognitive mechanism expresses our expectations about when music events will occur [14]. Meter is evoked by the regularities in the music and has the form of alternating strong and weak pulses. Strong pulses coincide with regularly occurring events [8]. Syncopation has been described as the feeling of surprise that arises when a rhythmic event that was anticipated at a particular moment does not actually occur [2].

Longuet-Higgins and Lee presented a model that identifies the syncopation in the pairs of notes and the rests or tied notes that follow them [15]. Accordingly, a rest or tied note in a strong metrical position preceded by an event in a weaker metrical position constitutes syncopation. David Huron, in his study of the American popular music [6], used a similar definition using the term "Lacuna". Behind both definitions lies the same expectation principle: an event in a weak metrical position is bound to an event in the following strong metrical position and when the expected strong note does not occur the weak one is left hanging [2 p. 295].

Longuet-Higgins and Lee assigned metrical weights to the metrical positions of the bar in order to quantify the metrical strength of each pulse. The weights correspond directly to the metrical level that each position initiates (see section 3 for a more detailed description). In Fig. 1, the metrical levels of a 4/4 meter are shown. If one would number the 5 levels starting with 0 for the slowest, the metrical weights would be the negative of those indexes (i.e. from -4 to 0). Longuet-Higgins and Lee defined syncopation as a note onset in a position with a low weight that is not followed by an onset in the following pulse with a higher weight.

David Temperley explored the uses of syncopation in rock [11], using a definition that bears strong resemblance to the Longuet-Higgins and Lee definition and Huron's expectation principle. In his study, he defined the term as the displacement of events from metrically strong positions to preceding weaker ones.

The above definitions suggest that a formalized way of describing the underlying meter is needed before we are able to manipulate the syncopation in a rhythmic pattern. We need, therefore, to construct a metrical template that corresponds to the time signature. In the next section we describe such a template, and an automatic way of constructing it. It will serve as the basis for all the manipulations that follow.

3 Construction of a Metrical Template

Complex music, even when it is not repetitive, it often evokes the sensation of a regular pulse in listeners that becomes evident when they tap in synchrony with the music. A single music can evoke simultaneously more than one such pulse sensations with different durations [16]. When those pulses are overlaid they form a hierarchical structure like the one in Fig. 1 where slower layers have pulse durations that are integer multiple of all faster ones [17, 18]. The various pulses have different metrical strength values that represent the alternating strong and weak beats commonly found in a musical meter. Similarly to the metrical structure used by Longuet-Higgins and Lee in their syncopation definition [15], each pulse initiates a metrical level according to the time signature and its metrical strength is proportional to that level, e.g. the quarter notes or the sixteenth notes found in a 4/4 bar. The pulses constitute a metrical grid which quantizes the time positions of the onsets of the events in a rhythmic pattern resulting into a binary representation of the rhythm.

The metrical template is constructed automatically for each time signature and tempo by successively subdividing the duration of the bar into faster metrical levels. For example, the 4/4 meter can be subdivided first into two half notes, then each half note into two quarter notes, each quarter note into two eight notes and so on, until the fastest metrical subdivision is reached. The metrical levels are indexed by numbers (hereafter metrical indexes) starting with the number 0 for the slower one. The metrical strength of a pulse is then proportional to the metrical index of the corresponding level. The stronger a pulse is, the slower the metrical level it belongs to (lower index), so that weak pulses belong to faster metrical levels (higher indexes). In Fig. 1, an example of such a metrical template is given. A detailed description of an automatic way of generating a metrical template for any given time signature can be found in [19].

The lower threshold for the duration of a metrical subdivision has been estimated in several studies to be roughly around 100ms [16, 20, 21 p. 29]. Therefore, the fastest metrical subdivision that is included in the metrical template must respect this boundary and therefore depends on the tempo.

An upper threshold for the duration of the metrical levels needs also to be determined. The need for such a threshold becomes apparent when one tries to de-

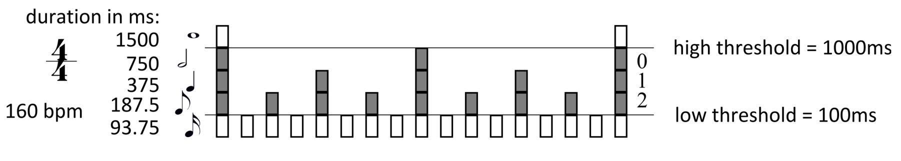

Fig. 1. Example of a metrical template for a 4/4 meter. The meter is successively subdivided generating the 5 metrical levels. The metrical strength (*rectangles*) of each metrical position corresponds to the metrical levels it belongs to. The very fast metrical levels (below 100ms duration) and very slow ones (above 1s duration) are disregarded (*white rectangles*). On the right the index of each level is shown (0 – 2)

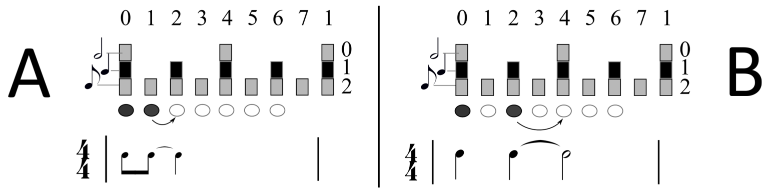

Fig. 2. Syncopation at slow metrical levels. A: a pattern syncopating at the eighth note metrical level. B: The same pattern at half speed. Above the two patterns the corresponding metrical template is shown. The black rectangles represent the beat level.

syncopate certain rhythmic patterns. For example, in the patterns of Fig. 2, one will follow different approaches for pattern *A* and *B*. While in pattern *A*, the tied eighth note should clearly be moved to the following quarter note, in pattern *B*, the tied quarter note falls on the beat and therefore is not considered syncopation.

However, the two cases are identical with respect to the definitions presented in section 2. The difference between the two is their relation to what is considered to be the beat level (or tactus), that is the most salient metrical level. The metrical salience of each level depends predominantly by its duration, with a peak salience in the region between 500ms – 1s [16, 21 chap. 2]. As the example of Fig. 2 illustrates, syncopation involving slower metrical levels than the beat is not felt as strong.

We chose the level that falls in the range between 500ms and 1s as the slowest metrical level represented in our structure. For example, in the case of the metrical template of Fig. 1, a 4/4 meter at 160bpm (q.n. = 375ms), only 3 metrical levels survive with corresponding durations of 750ms (0), 375ms (1) and 187.5ms (2).

4 Displacing Event Onsets

According to the Longuet-Higgins and Lee [15] or Huron’s [2 p. 295] definition of syncopation, a syncopating event is an event in a weak metrical position that is not followed by an event in the next strong metrical position. If this event was shifted to the strong position the syncopation would be eliminated. With this observation in

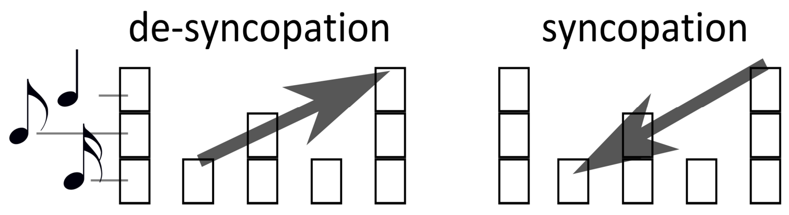

Fig. 3. The de-syncopation transformation shift events forward to slower metrical levels. The syncopation transformation shifts events backwards to faster metrical levels.

mind, the event at the weak position can be thought of as belonging to the strong position but been anticipated at an earlier position. Therefore, one can imagine the inverse shift as a way of generating syncopation at a strong metrical position, i.e. by shifting an event found at a strong position to an earlier, weaker pulse (Fig. 3). The transformations described in section 4.1 are such simple shifts of onsets in a binary pattern. The following formalization of the transformations consists of defining the conditions under which such shifts remove or generate syncopation in a given binary rhythmic pattern. Applying the formalized transformation on a binary pattern gives rise to the syncopation tree, a network of interconnected patterns, described in 4.2. In section 4.3 we provide with recursive algorithms for automatically generate specific branches.

4.1 The Transformations

The transformations take a binary pattern and manipulate the syncopation by shifting the onsets between the pulses according to the metrical template. In order to eliminate the syncopation, one needs to shift forward the corresponding event to a stronger metrical position (Fig. 3, left). In a pattern with several syncopating events, each such shift results in a new pattern with decreased syncopation. After all events have been shifted to non-syncopating positions, we call the resulting pattern a root. The de-syncopation process moves events from fast metrical levels to slower ones.

Syncopation is generated by anticipating events in weaker metrical positions (Fig. 3, right). Each shift of an event results in a new pattern with increased syncopation. The syncopation process, opposite to the de-syncopation process, “pushes” events to the faster metrical levels. When all events are moved to the fastest metrical positions the resulting syncopated pattern cannot be further syncopated.

A detailed description of the transformations follows.

Syncopation. Onsets in pulses that belong to strong metrical positions (slow metrical levels, low metrical indexes) are shifted to preceding pulses belonging to weaker metrical positions (faster metrical levels, higher level indexes).

A strong pulse might be preceded by more than one weak pulses, e.g. a quarter note is preceded by an eighth note but also by a sixteenth note (Fig. 4). We define as the type of syncopation the value of the difference of the metrical levels of the two pulses: the pulse that the onset belongs originally and the one that it is shifted to[1].

Each syncopation shift is described by a pair of numbers: the pulse that originally carries the onset and the type of shift that the onset undergoes. They can be thought of as the “coordinates” of the transformation. The pulse index directly corresponds to the “horizontal” coordinate. The type value corresponds to the vertical length of the arrow that represents the shift (Fig. 4). In general, the larger the type of the transformation the fastest is the metrical level that the onset is shifted to. Encoding the shifts into

[1] Alternatively we could have encoded the transformation as the pair of pulses, the initial pulse of the onset and the pulse it is shifted to. This way of encoding correctly describes the particular transformation. However, as it will become apparent in the following, using the level differences is a more general and more flexible representation.

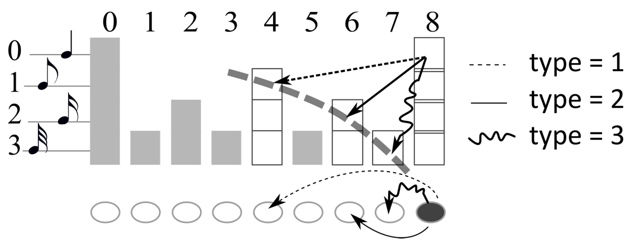

Fig. 4. An example of the types of syncopation transformations available for an onset at a specific pulse (pulse 8). White square represent the preceding pulses of faster metrical levels that are available for syncopating pulse 8. The rest of the pulses have been greyed out. The corresponding de-syncopation transformation is obtained by reversing the direction of the arrows.

types of metrical level differences provides with the freedom to associate them either to metrical positions, to metrical levels or to specific onsets. The number of syncopation types that is available for each pulse depends on how many faster metrical levels exist in the template.

An onset cannot be shifted if one or more onsets block its way. This rule ensures that the order that the events are performed is preserved. In the example of Fig. 4, we could not apply the transformation (8, 1) if an onset was found in any of the pulses 5, 6 or 7. The transformation is forbidden.

De-syncopation. The de-syncopation transformation is a direct consequence of its operational definition followed in this article [2 p. 295, 15]. This definition attributes the feeling to the event being anticipated. However, the syncopation is only felt at the moment of the following silent pulse and is retrospectively attributed to the event being heard. The following alternative phrasing of the definition puts the focus on to the silent pulse where the syncopation is actually felt instead of the onset that initiates it: "*syncopation is found in the silent pulses that belong in strong metrical positions (slower levels, lower indexes) and that are preceded by onsets in one of the immediately preceding weaker metrical positions*". Thus, the two transformations are essentially one the reverse of the other. The de-syncopation is applied to the silent pulse by shifting the preceding onset. In Fig. 4, onsets that might be found in pulses 4, 6, or 7, can be de-syncopated by reversing the direction of the arrows. In Fig. 5, de-syncopating pulse 8 is done by shifting the onset of pulse 6 to pulse 8 (solid arrow), yielding the corresponding type and completing the pair (8, 2).

The de-syncopation can be thought of as the analysis process yielding the type of transformation that was previously applied in order to syncopate. Because of the nature of the transformation, it is essential to de-syncopate first the faster metrical levels and then the slower ones. For example, if an event is found in pulse 5 in Fig. 5, in order to de-syncopate pulse 8, we must first de-syncopate pulse 6. That is equivalent to reversing the order of the syncopation transformations as well as the direction. The

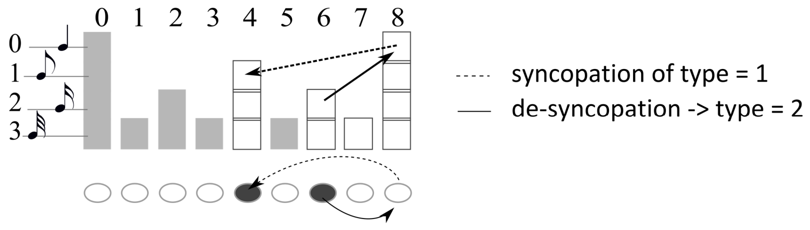

Fig. 5. If the order of onsets was not preserved during the transformations, the 1-1 correspondence of the two transformations would be lost.

result of the de-syncopation would be the corresponding transformations (6, 1) and (8, 1) in the reverse order.

The de-syncopation transformations reveal an important reason for preserving the order of onsets when syncopating and forbidding certain transformations. Imagine an onset originally existing in pulse 8 of Fig. 5 and apply the (8, 1) transformation (dashed arrow). The de-syncopation of pulse 8 would result in a different pattern than the one we started with. Moreover, it would return a different transformation type (2 as we already saw). Thus, the two transformations would not be reversible if two onsets could reverse their order. Moreover, this way the transformations are more general and can be expanded to include phenomenal accents, such as pitch or timbre changes, or other properties of the events that can make them distinct.

An important property of the transformations comes about from the alternating strong and weak pulses in the template. Since the beat level is the slowest metrical level included in the template, each transformation has local character; it displaces events within the duration of a single beat[2] (see Fig. 6), either forward to slower metrical levels or backwards to faster ones. Thus, any pattern longer than a beat can be considered a concatenation of shorter and independent single beat patterns.

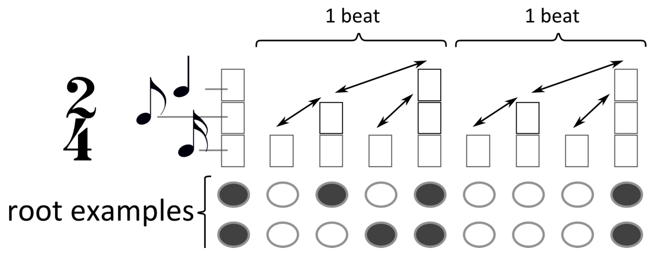

Fig. 6. The syncopation shifts can only move onsets in the duration of a single beat.

2 The duration of the beat is offset by a single pulse to what commonly is considered the duration of the beat (Fig. 6). The beat duration here ends ON the beat and includes all preceding pulses between the current and the previous beat.

4.2 The Syncopation Tree

The two transformations described above can be used to generate syncopation trees. Starting from a root pattern, one can apply all possible combinations of transformations to generate a large number of patterns. On the left side of Fig. 7, we present an example of how a branch is generated starting from the root pattern and applying a series of transformations until we reach the end of the branch where no onset can be syncopated further. However, one could start from any pattern and apply a series of transformations to reach another pattern of the same tree.

The root pattern of the example has 3 events and is considered to be a repeating loop (the first event is repeated at the end). The branch consists of a set of transformations of specific types indicated by the two numbers next to each pattern. The specific branch is generated by applying the transformations in a particular order. The pairs of numbers form two arrays: 1) one contains the indexes of the pulses in the particular order of the transformations and 2) the second contains the type of each transformation. The two arrays completely describe the branch. We refer to such coupled arrays as the branch arrays.

The order of the transformations can change to generate a different branch. In the right side of Fig. 7, we show the branches of the corresponding syncopation tree that include all possible permutations of the transformations in the left (solid lines, the thick solid line is the branch shown in detail at the left). Each dot in the right part of the figure represents a specific pattern and each line a transformation. It must be noted that not all permutations are possible. Here, the transformation (8, 1) must always be applied before the (6, 1) otherwise there will be no onset in pulse 6 to be shifted. Each pattern is connected to the root pattern in a number of steps that is independent of the exact branch that one might follow. This is a general property of all syncopation trees and not of this particular example.

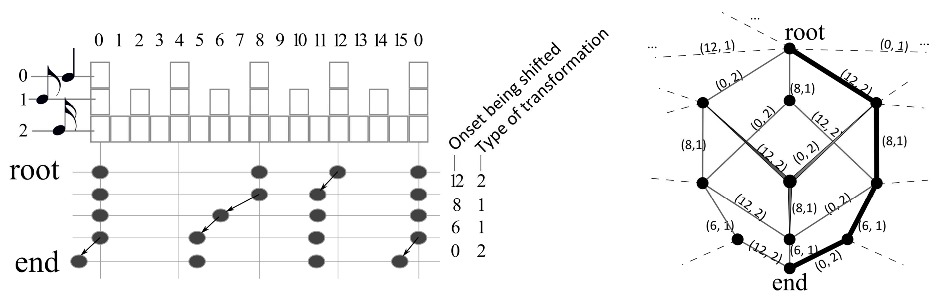

Fig. 7. Left: An example of a branch starting from the root pattern and finishing to an end pattern that cannot be further syncopated. Right: Example of a syncopation tree (partially shown). The patterns are shown as dots and the lines connecting them correspond to the indicated transformations. The thick line corresponds to the branch shown on the left. The patterns shown are connected with the same set of syncopation transformations applied in a different order.

The entire tree is formed by several branches like the ones shown in Fig. 7 and has several ends. The number of ends depends on the number of possible transformations that can be performed. In our example, the 3 onsets belong to different beats and can undergo 2 different transformations each. Therefore, we have $2^3 = 8$ different ends in this tree. In Fig. 7, we show only the branches that lead to the end pattern shown on the left. The rest of the ends are connected to the patterns of the figure through the dashed lines. Generating branches that connect specific patterns is equivalent to analyzing the relations between them.

4.3 Recursive Algorithms

De-syncopation. A recursive de-syncopation algorithm is provided here as a means of completely de-syncopating a binary pattern and finding its root. The corresponding branch consists in a series of transformations expressed as a branch array, i.e. an order array coupled to a type array.

The de-syncopation process recursively applies a de-syncopation transformation on each silent pulse that is found to syncopate until there is no syncopating pulse. Using the output of the algorithm one can generate all intermediate patterns from the corresponding root pattern to the original input pattern. The root pattern is independent of the order in which the pattern is de-syncopated.

Fig. 8 illustrates the process through an example. Pulses are examined in a certain order and when a silent pulse is preceded by an onset in a faster metrical level, the onset is shifted as described in section 4.1. In the example the pattern is scanned from right to left. However, the order in which the pulses are examined can be, in principle, freely chosen.

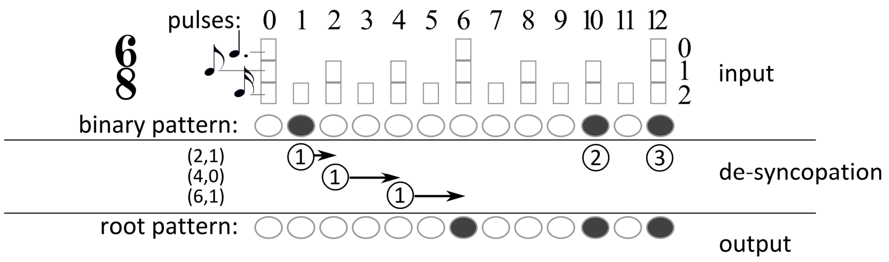

Fig. 8. Illustration of a recursive de-syncopation process. The metrical template shown corresponds to a 6/8 meter at 180bpm (quarter note = 1000ms). Each onset of the binary pattern is numbered in a circle. The de-syncopated process is shown as arrows.

The following pseudo code illustrates the basic steps in the de-syncopation algorithm:

```
##inputs
bool PATTERN = {O1, O2, ..., On}
int LEVELS = {L1, L2, ..., Ln}
int ORDER = {P1, P2, ..., Pm}
##outputs
int outORDER = {empty array}
int outTYPE = {empty array}
## DE-SYNCOPAION LOOP
Repeat
    For each position p in ORDER
        if (O[p]==FALSE AND L[p]<=max(LEVELS)
            r = Find_Preceding_Pulse(p)
            if (O[r to p-1 r]==FALSE)
              O[r]=FALSE; O[p]=TRUE//Shift onset r->p
              Output PATTERN
              Append p in outORDER
              Append (L[p]-L[r])in outTYPE
    end
Until no onsets can be shifted
Output outORDER, outTYPE
```

The above code makes use of a subroutine for finding the preceding onset, if any:

```
Find_Preceding_Pulse(p)
  LMIN = L[p]
  p--;  CurrentML = L[p]
  While (L[p] > LMIN AND O[i]==FALSE)
      p--;  CurrentML = min(L[p], CurrentML)
  End
  If (O[p]==TRUE AND L[p]== CurrentML AND L[p]>LMIN)
    Return p
  Else Return -1
End
```

The above code receives as input three arrays: 1) the pattern as a binary string (PATTERN), 2) the metrical template as an array of the metrical indexes (LEVELS) and 3) the ORDER array which represents the order in which the pulses should be scanned and the found syncopations should eliminated. The algorithm outputs the new binary pattern with each shift of an event. At the same time it stores the details of each shift in two arrays. The outORDER array contains the pulse of each shift and the outTYPE array contains the metrical level difference. The two arrays together comprise the branch arrays generated by the de-syncopation and describe the transformations applied to the input pattern. They are output at the end of the loop.

In the example of Fig. 8 the two arrays would be:

```
outORDER  = 2, 4, 6
outTYPE   = 1, 0, 1
```

A couple of examples of default values for the input order array would be the pulses in their natural order, from left to right (0, 1, 2, etc.), or in their order of importance, e.g. according to their metrical level or the indispensability values of Clarence Barlow [22, 23]. Not all pulses need to be contained in the order array, since not all pulses carry onsets that need to be de-syncopated. However, it is important to ensure that all pulses that will syncopate in any of the intermediate steps of the de-syncopation process will be included in the input order array. As long as this is ensured, the order of transformations only affects the intermediate steps. The root pattern and the instances of syncopation found are the same for any order. The number of elements in the output branch arrays, as well as the values of each pair in the arrays, is unaffected by the input order. Only their position in the arrays could in certain cases change, depending on the pattern. In the example of Fig. 8, the result is always the same; all output arrays and patterns will be exactly the same, independently of the input order.

Syncopation. In order to automatically syncopate an input pattern, one can apply repeatedly a series of syncopation transformations until no onsets can be further shifted. The branch arrays for the process need to include all pulses that carry onsets in the original input pattern and any of the output patterns of the applied transformations. That will ensure that when the transformations are applied recursively all onsets will be shifted in syncopating positions and the end of the branch will be reached.

The process is similar to the one shown on the left side of Fig. 7. However, in that example, the transformations were applied once and were particular to the generated branch. In a recursive syncopation process, the branch arrays can be generic and can automatically generate a complete branch from root to end pattern. For example, a default order of transformations can be generated by starting from the pulses in the faster metrical levels and continue to slower ones until all pulses are included (the fastest metrical level can be ignored). After applying the possible transformations to all the pulses in the array, the process repeats until no onset can be displaced.

For each pulse, a type of transformation should also be assigned. Several options for default values are available. The simplest is a constant value for all pulses. The types can also be chosen according the metrical level of each pulse. For example, all notes could be shifted to the preceding sixteenth note. That would be a type 2 for the quarter notes and a type 1 for the eight notes.

In cases when the level difference expressed in the type of syncopation results to a metrical level faster than the fastest metrical subdivision included in the metrical template, then this fastest metrical subdivision is used disregarding the type. For example, if the type is set to 2 for an onset at the eight note level, it should be shifted to the preceding thirty-second note. If the metrical template only goes as fast as the sixteenth note level, then the onset should be shifted to the preceding sixteenth note, disregarding the type value found in the array.

If one wishes to reverse the transformations generated by the de-syncopation algorithm, he needs to reverse the two branch arrays, outORDER and outTYPE, produced by the recursive de-syncopation process. Performing the transformation once on the root pattern generates the exact steps of the de-syncopation.

The following pseudo code illustrates the basic steps in the syncopation algorithm:

```
##inputs
bool PATTERN = {O1, O2, ..., On}
int LEVELS = {L1, L2, ..., Ln}
int ORDER = {P1, P2, ..., Pk}
int TYPE = {T1, T2, ..., Tk}
int MAXrepeat = maximum number of repetitions
int step = 0;
##outputs
int outORDER = {empty array}
int outTYPE = {empty array}
## SYNCOPATION LOOP
Repeat
    For each position p[i] in ORDER //at index i of ORDER
        if O[p[i]]==TRUE AND L[p[i]]!=maximum[LEVELS]
            Find preceding position r: L[r]=L[p[i]]+T[i]
            if O[p[i]-1 to r]==FALSE
              O[p[i]]=FALSE; O[r]=TRUE//Shift p[i]-> r
              Output PATTERN
              Append p[i]in outORDER
              Append (L[r]-L[p[i]])in outTYPE
    end
step += 1
Until no onsets can be shifted OR step >= MAXrepeat
Output outORDER, outTYPE
```

A “*find preceding position*” subroutine is used to find the preceding pulse that belongs to the faster metrical level. The subroutine is similar to the one used in the de-syncopation algorithm with the difference that it is looking for a silent pulse and that the pulse belongs to specific metrical level determined by the type of transformation.

At the end of the algorithm shown above, a pair of branch arrays is output. These arrays can differ from the input ones. As the pattern is scanned according to the input order, some transformations might be blocked or skipped when an onset is not found resulting in a different final order. The two arrays, outORDER and outTYPE, contain the actual transformations that were performed.

Fig. 9 presents a schematic overview of the entire process of generating a complete branch that passes through a particular pattern. The process is divided in an analysis and a generation stage. First, the de-syncopation of the input pattern to its root provides a detailed analysis of the syncopation instances in the given pattern. Second, a

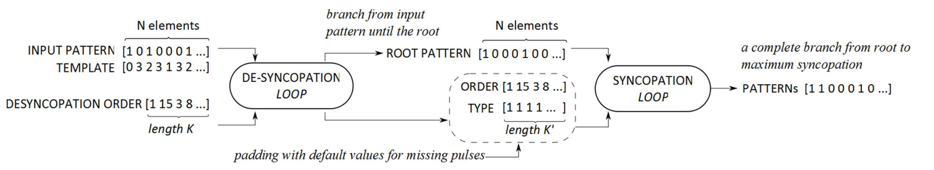
N elements
INPUT PATTERN [1 0 1 0 0 0 1 ...]
TEMPLATE [0 3 2 3 1 3 2 ...]
DESYNCOPATION ORDER [1 15 3 8 ...]
length K
DE-SYNCOPATION LOOP
branch from input pattern until the root
N elements
ROOT PATTERN [1 0 0 0 1 0 0 ...]
ORDER [1 15 3 8 ...]
TYPE [1 1 1 1 ...]
length K'
padding with default values for missing pulses
SYNCOPATION LOOP
a complete branch from root to maximum syncopation
PATTERNs [1 1 0 0 0 1 0 ...]

Fig. 9. Overview of the generation of a branch of syncopation transformations that passes through a given input rhythmic pattern

complete branch is generated, starting with the root pattern, reaching the input pattern and expanding to the other end, to a maximally syncopated pattern. The first part, the analysis, is performed by the de-syncopation algorithm and the second, the generation, by the syncopation algorithm.

It is important to note that the de-syncopation process functions as an analysis stage, even though it actually generates all the intermediates between the root and the input pattern as part of the analysis. The root pattern together with the branch arrays of order and types gives a complete picture of the generated patterns and their relations in a compressed form. The mere collection of the actual intermediate patterns can be thought of as a bi-product of the process.

The second part of the process generates the entire branch beginning with the root. By altering the order of elements in the syncopation arrays, one can generate a different branch that might not pass by the given input pattern but it will end at the same end pattern. The patterns found in that branch will share the same types of syncopations, although the patterns in their entirety would not be identical. However, altering the type values leads to a completely different branch.

5 Conclusions

In this paper we present a set of generic transformations that can serve as the basis for a formalized model of syncopation. The generative nature of the transformations makes them suitable for both analysis and generation of syncopation in rhythmic patterns. In fact the two processes are combined under the concept of the syncopation tree; a way of generating transformation paths between patterns.

The rhythmic patterns found on the branches of the tree are interconnected through specific sequences of transformations. Any two patterns on a tree can be transformed from one to the other following a branch of single-step transformations. One can navigate from one pattern to another unveiling the relations between the patterns. In each tree, it exists only one pattern with no syncopation, called the root pattern. The particular branch that connects a pattern to its root provides with a detailed description of the syncopation in the pattern, such as the number of syncopating events, their positions and the metrical levels involved. The transformations and the tree can serve as the basis for rhythmic similarity or distance measures.

The way each transformation is encoded and described in this model helps in creatively exploring its uses. For example, one can think of modeling the syncopation of one particular music style as specific transformations on certain root patterns. These transformations could then be applied on root patterns that do not belong to the modeled styled. The modeling of a musical style can be done by automatically de-syncopating a number of patterns characteristic to the style, e.g. rhythmic patterns performed simultaneously by different instruments. The resulted transformations could then be combined applied to the root of any other pattern effectively "copying" the syncopation style.

The transformation can be employed in automatic music generation algorithms. For example, one can imagine a simple algorithm that generates root patterns in a certain meter and then a series of transformations is applied to generate more complex syncopated patterns. The generated root patterns can include pitch information such as a melody or baseline in a certain key.

The applications of the syncopation transformations are left to be explored in the future work of our group. However, we developed a software application as a small example of a creative application of the transformations. It has the form of a Max4Live midi device that manipulates the syncopation of midi clips. The Max4Live devices and related externals are available for download at our website: http://smc.inescporto.pt/shakeit/

References

1. Gómez, F., Thul, E., Toussaint, G.T.: An experimental comparison of formal measures of rhythmic syncopation. Proceedings of the International Computer Music Conference. pp. 101–104 (2007).
2. Huron, D.: Sweet anticipation: music and the psychology of expectation. The MIT Press (2006).
3. Gabrielsson, a: Adjective ratings and dimension analyses of auditory rhythm patterns. Scand. J. Psychol. 14, 244–60 (1973).
4. Randel, D.M.: The Harvard Dictionary of Music. Belknap Press of Harvard University Press, Cambridge, MA (1986).
5. Kennedy, M., Bourne, J. eds: Oxford Ditionary of Music. Oxford University Press (1994).
6. Huron, D., Ommen, A.: An Empirical Study of Syncopation in American Popular Music, 1890-1939. Music Theory Spectr. 28, 211–231 (2006).
7. Fitch, W.T., Rosenfeld, A.J.: Perception and Production of Syncopated Rhythms. Music Percept. 25, 43–58 (2007).
8. Palmer, C., Krumhansl, C.L.: Mental representations for musical meter. J. Exp. Psychol. 16, 728–741 (1990).
9. Keith, M.: From Polychords to Polya : Adventures in Musical Combinatorics. Vinculum Press, Princeton (1991).
10. Gómez, F., Melvin, A., Rappaport, D., Toussaint, G.T.: Mathematical measures of syncopation. Proc. BRIDGES: Mathematical Connections in Art, Music and Science. pp. 73–84. Citeseer (2005).
11. Temperley, D.: Syncopation in rock: a perceptual perspective. Pop. Music. 18, 19–40 (1999).

12. Volk, A., Haas, W. de: A Corpus-Based Study on Ragtime Syncopation. 14th International Society for Music Information Retrieval Cinference. , Curitiba, Brazil (2013).
13. Sioros, G., Miron, M., Cocharro, D., Guedes, C., Gouyon, F.: Syncopalooza : Manipulating the Syncopation in Rhythmic Performances. 10th International Symposium on Computer Music Multidisciplinary Research. pp. 454–469. Laboratoire de Mécanique et d'Acoustique, Marseille (2013).
14. Jones, M.R.: Musical Time. In: Hallam, S., Cross, I., and Thaut, M. (eds.) The oxford handbook of music psychology. pp. 81–92. Oxford University Press (2009).
15. Longuet-Higgins, H.C., Lee, C.S.: The rhythmic interpretation of monophonic music. Music Percept. 1, 424–441 (1984).
16. Parncutt, R.: A perceptual model of pulse salience and metrical accent in musical rhythms. Music Percept. 11, 409–464 (1994).
17. Lerdahl, F., Jackendoff, R.: A Generative Theory of Tonal Music. The MIT Press, Cambridge (1983).
18. Yeston, M.: The Stratification of Musical Rhythm. Yale University Press, New Haven, CT (1976).
19. Sioros, G., Guedes, C.: A formal approach for high-level automatic rhythm generation. Proceedings of the BRIDGES 2011 – Mathematics, Music, Art, Architecture, Culture Conference. , Coimbra, Portugal (2011).
20. Repp, B.H.: Rate Limits of Sensorimotor Synchronization. Adv. Cogn. Psychol. 2, 163–181 (2006).
21. London, J.: Hearing in Time. Oxford University Press (2012).
22. Barlow, C.: Corrections for Clarence Barlow ' s Article: Two Essays on Theory. Comput. Music J. 11, 10 (1987).
23. Barlow, C., Lohner, H.: Two essays on theory. Comput. Music J. 11, 44–60 (1987).