Discovery and Utilization of Jazz Motifs for Computer-Generated Solos

Joseph Yaconelli[1] and Robert M. Keller[2] *

[1] Department of Computer and Information Science
University of Oregon
Eugene, Oregon, USA
[2] Department of Computer Science
Harvey Mudd College
Claremont, California, USA
josephy@cs.uoregon.edu, keller@cs.hmc.edu

Abstract. Building on previous work in computer generated jazz solos using probabilistic grammars, this paper describes research extending the capabilities of the current learning process and grammar representation used in the Impro-Visor educational music software with the concepts of motifs and motif patterns. An approach has been developed using clustering, best match search techniques, and probabilistic grammar rules to identify motifs and incorporate them into computer generated solos. The abilities of this technique are further expanded through the use of motif patterns. Motif patterns are used to induce coherence in generated solos by learning the patterns in which motifs were used in a given set of transcriptions. This approach is implemented as a feature of the Impro-Visor software.

Keywords: motif, motive, jazz, improvisation, generative grammar, grammar learning, machine learning, Impro-Visor

1 Introduction

By *motif* (also known as *motive*) we mean a recurrent melodic idea. Motifs are widely used in both composed and improvised music. In composing, an entire composition can be based on a single motif, variations of which are produced by transposition, contraction, expansion, inversion, etc., with said variations knit together to form complete melodies. In jazz compositions, for example, various blues (such as "Sonnymoon for Two") and riff tunes (such as "Lester Leaps In"), motifs may be repeated unaltered or with minor alterations to suit the harmonic background to form melodies. Motifs are also common in improvisations, where they can sometimes be heard recurring in multiple performances of different pieces, indicating that they have been practiced. They may also originate spontaneously, then be reused by a soloist in the same piece. Also common

* This work was supported in part by NSF CISE REU award number 1359170 to Harvey Mudd College.

is incorporation of a motif from the original melody (the "head") of the song (known as "playing off the melody") or from an entirely different song (known as "quoting").

It has been long understood that well-known and highly-regarded jazz musicians use practiced, as well as spontaneous, motifs in their playing. For example, Owens [1] opens his chapter *Motives* by stating "Every mature jazz musician develops a repertory of motives and phrases which he uses in the course of his improvisations. His 'spontaneous' performances are actually precomposed to some extent." He continues with eighteen pages of analysis of motives of a single musician, Charlie Parker. Martin [2] challenges Owens' assertion that Parker's solos are constructed mostly from isolated preconceived ideas without reference to the head melody by providing in-depth analysis of several solos.

In a related vein, noted educators such as Haerle [3], who of necessity are also players, recommend learning a variety of motives from which solos can be partially or totally constructed. Jazz educators have also published many volumes of patterns and licks, which are related to motifs. A pattern may be considered a motif constructed from a specific formula, such as 1-2-3-5 degrees of a scale, while a lick is generally longer and might incorporate one or more motifs.

Terefenko [4], in a five-page section entitled *Motivic Development*, states "There are certain characteristics, however, that a motif should have to lend itself for musical development. These characteristics include a strong rhythmic profile, an interesting melodic shape, a clear harmonic structure, and a relatively pitch short duration." The first two characteristics are subjective and we do not presuppose any means of objectively quantifying them currently, while we do contend that our methods address the second two.

2 Goal of This Work

A known lack of realism in improvisations created by machine learning software, including grammars, Markov chains, and neural networks, is the lack of any sort of *global coherence* in the solos. While short segments of created music are typically plausible, there is generally a lack of reference between earlier and later improvised segments. We claim that use of motifs is a partial solution to the global coherence issue.

The ability to create motifs and then reuse them within improvisational software can help make the overall melodies improvised by the software more convincing. Our objective is to provide simple grammatical constructs that will enable dynamic capture, then reuse, of one or more motifs within a solo. The source of the motif can be either generated artificially, pre-specified, or captured from another player, such as a human using the program as an educational companion.

Toward this goal, we developed two techniques that will enhance coherence. The first is to extend the grammar formalism developed in [5] for Impro-Visor [6] to provide an easy means for exploiting motifs, which are either represented by

specific grammar productions or which are captured when a motif is generated dynamically. The second is to provide a mechanism for recognizing motifs during the grammar learning process, so that the exploitation mechanism has a set of motifs to use as a basis.

3 Related Work

The focus of our research is on the discovery and subsequent utilization of jazz motifs and motif patterns. Musical motif discovery has been investigated by many researchers such as Grachten [7], Hsu, et al. [8], and Weiss and Bello [9]. Rolland and Ganascia [10] used a dynamic programming model based on edit distance in a musicological investigation. Pinto [11] represented musical scores as fully connected graphs, using eigenvector techniques to determine likely motifs. Lartillot [12] uses musical listening strategies to develop a computational model for motif discovery.

Motif utilization is a less researched field. However, there has been work in utilization by several researchers. Chiu and Shan [13] use a multi-layered approach to music generation and motif utilization. Explicitly repeated motifs are discovered in a corpus of music along with the patterns in which they are used most prominently. A musical score is then generated by building an overarching structure, which is then filled with phrases of melody and motif learned from the corpus.

Gjerdingen [14] shows how self-organizing neural-like networks are capable of finding stable, yet plastic, groupings of musical phrases that hold musical meaning. These groupings then have a prototype representing an average or general concept of its members. While not immediately applied to motif utilization, it shows potential for generating novel motivic ideas based on those present in a musical score.

4 Motifs in Jazz

Motifs are short melodies that are repeated throughout a piece of music. They are one of the fundamental building blocks of jazz improvisation. The style of motifs and context in which they are used are key in defining a musician's style and therefore also key in emulating that style. Motifs are also used to define the overarching structure of a solo. Motifs can be repeated at the end of a phrase to give a set of otherwise dissimilar ideas a sense of connectedness.

Figure 1 shows the red circled motif repeated three times in the jazz standard "Autumn Leaves." Note that when the motif is repeated, it is transposed and played over different chords. This is because a motif is a musical "idea" rather than a specific set of notes. A motif is encoded in the rhythm, shape or contour, and consonance of a melody. These more abstract parts of a melody allow a motif to be adapted to the chords over which it is played. Thus, motifs partially create intrigue in a composition or performance. By repeating similar patterns already heard earlier in a piece of music, the listener is given the pleasure of recognizing a

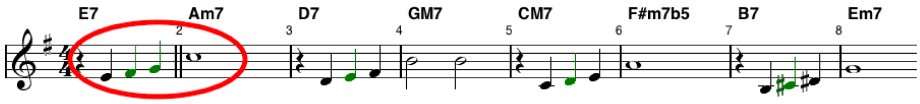


Fig. 1. A motif repeated in the melody of "Autumn Leaves"

familiar melody. Figure 1 incidentally demonstrates how note heads are colored automatically by Impro-Visor as an option. Black note heads indicate pitches that are in the chord, while green note heads indicate pitches that are "color tones," i.e. tones that are sonorous over the chord but not actually in the chord. For example, the G natural in the first measure is a raised ninth relative to the E7 chord, which also functions as the leading tone into the next chord A minor seventh. Two other colorations that may occur are blue, indicating an approach tone, defined to be one resolving to a chord or color tone that follows, and red, which indicates none of the other categories. These categories also play a role in *abstract melodies*, to be discussed.

4.1 Grammars for Melody Generation

In the current work, motifs are manipulated using one of the methods by which Impro-Visor improvises: *probabilistic generative grammars*, as described in [5]. In such a grammar, filling melodic space is governed by grammatical productions. A *production* consists of a left-hand side non-terminal symbol and a right-hand side, which is a sequence of terminal symbols, non-terminal symbols, and other linguistic constructs. Ultimately, a non-terminal symbol becomes, through a series of replacements of left-hand side non-terminals by right-hand side sequences, a string of terminal symbols.

Each production has an associated *weight*, which can be viewed as an unnormalized probability. When a non-terminal is to be expanded, all productions with that non-terminal as a left-hand side are collected, their weights summed, and each weight is divided by the sum to obtain a normalized value between 0 and 1. A random number is generated and the system selects a rule with probability proportion to the rule's normalized weight value.

Here are few examples of how grammars work. We use the notation of [5] for increasing readability of the exposition. However, in the Impro-Visor software, grammars are represented textually using S-expressions.

Here is a production representing a motif that was learned from a John Coltrane Solo on "Giant Steps":

$$M0 \rightarrow C8\ (\Delta\ -4\ -3\ C8\ C8\ C8)(\Delta\ 3\ 5\ C8\ L8)(\Delta\ -2\ -1\ C8\ C8)$$

The left-hand side of the production, M0, is a non-terminal that can be expanded into the right-hand side. The first symbol on the right-hand side is C8, which is *abstract melody* notation[5] for a Chord tone with duration of an 8th note. C8 is followed by three *slopes*, as indicated by the Δs. A slope is a sequence

of abstract notes that are separated by a minimum and maximum number of semitones between each. These values are the two numbers following each Δ. For example, (Δ 3 5 $C8$ $L8$) means a **C**hord tone followed by a co**L**or tone, each 8th notes, separated by an interval of between 3 and 5 semitones. Unsigned numbers represent upward slopes, while negative numbers represent downward slopes. For example, (Δ -4 -3 $C8$ $C8$ $C8$) represents three Chord tones, each 8th notes, separated by intervals of between 3 and 4 semitones in the downward direction. By sequence slopes in this way, we can represent arbitrary *melodic contours*.

When such a production is used, it expands into actual notes by instantiating the chord and color tones. Figure 2 shows examples of melodies produced by this rule. Since the melody starts out descending, we refer to this as the *descending* version of M0.

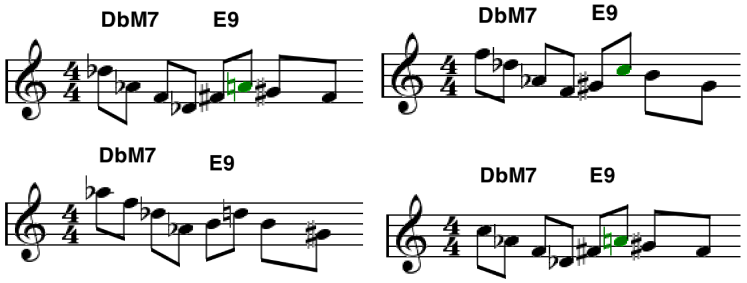


Fig. 2. Instances of a grammar production M0 for a *descending* motif, as produced from an abstract melody, over the first two chords of "Giant Steps". Notice that each instance starts on one of the four chord tones of DbM7: Db, F, Ab, C

Now consider a second production with the same left-hand side:

$$M0 \rightarrow C8\ (\Delta\ 2\ 3\ L8\ C8\ C8)(\Delta\ -4\ -4\ C8)(\Delta\ 2\ 3\ C8\ C8\ C8)$$

When this second production is used, it expands into melodies shown in Figure 3. We will call this motif *ascending* for reference purposes.

Either production with M0 on the left-hand side can be used to expand non-terminal M0. Each production is given a probability. Now consider a production for a pattern P0 that uses M0 four times on the right-hand side:

$$P0 \rightarrow M0\ M0\ M0\ M0$$

Applying this production will result in four instances of M0. However, because there are two separate productions with M0 on the left-hand side, each time M0 is expanded, a different production may be used, resulting in a line such as the one shown in Figure 4. In this line, the descending motif is used first, then the

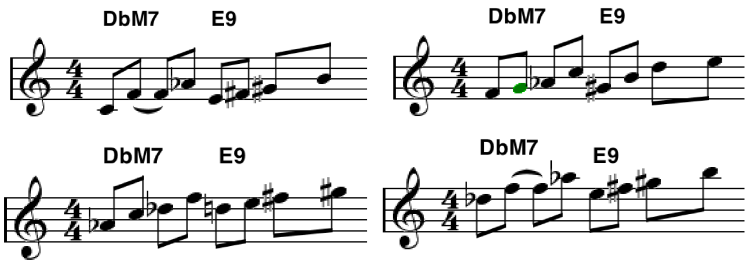


Fig. 3. Instances of a grammar production for an *ascending* motif, as produced from an abstract melody, over the first two chords of "Giant Steps". The tied notes are due to automatic merging of repeated notes that takes place in Impro-Visor to make the line more interesting. (This feature can be turned off.)

ascending motif, then the descending motif is used two more times. However, the choice of descending versus ascending is made randomly each time M0 is expanded.

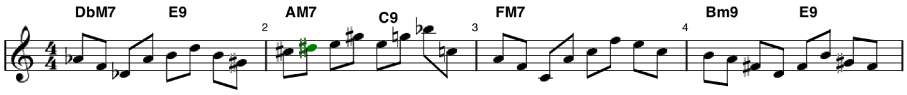


Fig. 4. Generation from P0, wherein the motif choices are descending, ascending, then descending (twice) (Note: Impro-Visor will wrap by shifting the next note up or down an octave if it would otherwise be outside a specified range.)

Suppose that, instead of random choices at each step, we desire a behavior in which the first instance of M0 is chosen randomly, then the next three instances are to use the *same choice* that was made in the first instance. The addition, as part of the present work, of *sharing* notation to the grammar provides for this. The production is changed to

$$P0 \rightarrow (share\ M0)(share\ M0)(share\ M0)(unshare\ M0)$$

The new keywords **share** and **unshare** serve to cache the chosen instance of M0 for reuse. The first **share** chooses an instance of M0 as normal, while subsequent instances of **share** use the original instance, rather than selecting a new instance. The **unshare** also uses the original instance, but releases the memory of the instance from the cache, so that the next time M0 is expanded, a new instance will be chosen. Figure 5 shows a possible result, where P0 is used twice. In the first use, the descending motif was chosen, and in the second use, the ascending motif was chosen. The sharing construct thus enables emulation of behavior in a jazz performance, wherein the player chooses or invents a motif on the fly,

then repeats it. Although we use only motifs in the above examples, the right-hand side of a production can intersperse motifs with arbitrary other generated material, including melodies the length of which is not predetermined. So a motif used early in a solo could first reappear much later.

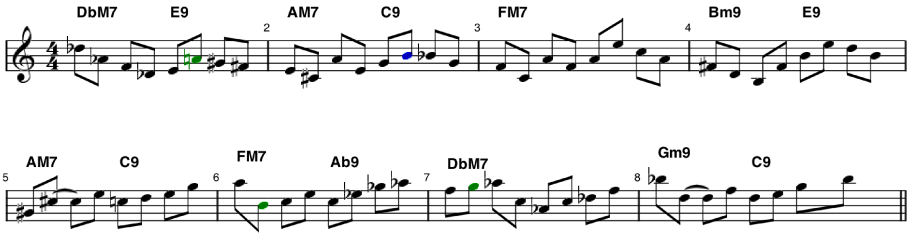


Fig. 5. Generation from P0 with sharing. In the first four bars, the descending motif was chosen and shared, while in the second four bars, the ascending motif was chosen and shared. (Note: Impro-Visor will wrap by shifting the next note up or down an octave if it would otherwise be outside a specified range.)

5 Motif Learning

5.1 Motif Representation for Machine Learning

As described in the preceding section, learned motifs are represented as short abstract melodies extracted from a solo transcription that includes the underlying chord progression. A transcription is represented by a lead sheet (a musical document consisting of a chord progression and melody line) encoded in Impro-Visor's leadsheet notation [21]. Extraction involves moving a fixed-size *window* over the transcription, which creates fixed-size melodic segments from which abstract melodies are formed. In our motif learning, motifs are accompanied by additional information on their original location in the transcription and the number of *equivalent* (in the sense of "nearly identical") melodies exist in the same transcription.

We use abstract melodies as a possible device for capturing how a motif exists in the mind of a musician, even if created extemporaneously. See Berliner [16], Chapter 8, "Thinking in the Moment," for a broad exploration of such ideas. For a scientific investigation of psychological aspects of whether motifs are preconceived or created spontaneously, please see Norgaard [17]. Representing a motif as an abstract melody frees it from being based on any specific key, mode, or chord sequence. This allows the same motif to be actualized as melodies in many different musical situations. We contend that this resembles the manner in which a musician might reuse motifs, i.e. using the abstract concept of the motif rather than consciously transposing or otherwise altering a previous melody to fit over a new chord. We do not claim that our version of abstract melody is

the only such device, nor does our particular notion cover all possible notions of abstract melody. For example, we currently do not capture various notions of time warping at present.

5.2 Clustering

After melodies are extracted from a transcription, they are clustered using the unsupervised k-means clustering algorithm [18], using the same metrics as in Gillick et al. [5], namely:

(1) number of notes in the abstract melody
(2) location of the first note that starts within the window
(3) total duration of rests
(4) average maximum slope of ascending or descending groups of notes
(5) whether the window starts on or off the beat
(6) order of the contour (how many times it changes direction)
(7) consonance

The k-means algorithm divides motifs into k clusters based on these metrics. Each cluster has a *centroid* which is a single representative point for all elements (motifs) in the cluster. The clustering algorithm ensures that motifs in a cluster are closer to their cluster's centroid than to any other cluster's centroid.

5.3 Motif Equivalence

As abstract melodies, motifs differing very slightly from each other will be considered equivalent, and thus regarded as the same motif. In order to capture the equivalence of motifs, we use two heuristic similarity measures. The first is a modified Levenshtein distance [19] from the abstract melody of one motif to the other normalized over L_d, the length of the longer of the two abstract melodies. The second is based on the Euclidean distances (D_1 *vs.* D_2) from each motif to the centroid of their common cluster. The triangle inequality provides an upper bound of $(D_1 + D_2)$ to the distance between the two motifs. Previous research [20] has indicated that these heuristics are good for best-match searching. We then use cutoffs ϵ_1 and ϵ_2 to set the tolerance of equivalence. Two motifs are considered equivalent if both of the following inequalities hold:

$$L_d \leq \epsilon_1$$
$$D_1 + D_2 \leq \epsilon_2$$

When two motifs are considered equivalent, they are both represented by a single representative abstract melody and the count of that motif increases by one. The abstract melody used to represent the two motifs is chosen, arbitrarily, as whichever came first in the original transcription.

5.4 Motif Clusters

Once melodies are clustered by the k-means algorithm, they are further triaged into *motif clusters*. The melodies from a k-means cluster are added to a motif cluster one at a time in a first-in-first-out (FIFO) manner. In order for a motif, m, to be placed into its corresponding motif cluster, C, the normalized log distance from that motif to the centroid of the corresponding k-means cluster must be less than a user-specified "motifness" parameter γ. Formalized, this means:

$$\frac{log(D_C^m) - min(log(D_C))}{max(log(D_C)) - min(log(D_C))} \leq \gamma$$

where $log(D_C^m)$ is the log of the distance of motif m in cluster C from the centroid of C, and min and max are taken over all motifs in cluster C. This ensures that all motifs in the motif cluster are sufficiently similar. Without these thresholds, all pieces of melody would be marked as motifs. Using a log distance ensures that only very close melodies qualify for the motif cluster. A similar tactic is used in fuzzy set theory wherein a set can be concentrated to reduce membership so that only "very similar" elements remain.

If a given motif does not satisfy the above inequalities, it is placed into a special-case motif cluster which contains all "non-motifs." This motif cluster is intended to contain all melodies that are not repeated enough to be considered a true motif.

If a motif does satisfy the inequalities, it is added to its respective motif cluster. Each motif cluster has an underlying max priority queue structure [21], using the motifs' counts as priorities. When a motif is place into the motif cluster, it is compared to all other members of the cluster. If a motif, m, is considered equivalent to another motif, n, in the cluster, the count of the motif n is increased by one and the motif m is represented by n. The increased count increases the priority of that motif. If the motif is not considered equivalent to any motifs already in the cluster, it is added to the max priority queue with its count (initially 1) as its priority.

5.5 Motif Cluster Representatives

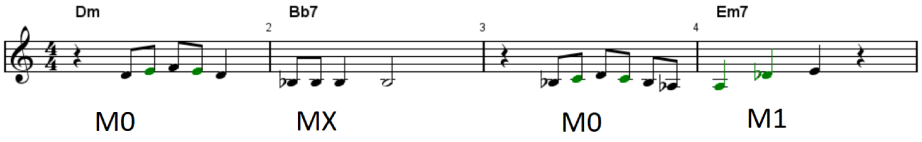
Within each motif cluster, some number or *representatives* are chosen. Currently, the algorithm choses the motifs from the cluster with up to the n highest priority counts, where n is a settable parameter, such as 3. These motifs are then added as productions to the grammar with a single left-hand side non-terminal $M01$, $M02$, etc., per motif cluster. When such a non-terminal is expanded, any one of the productions for that motif is chosen nondeterministically.

5.6 Motif Patterns

Once motif clusters and representatives are finalized, the motifs in each segment in the lead sheet are labeled by the motif cluster in which the segment was

placed. Many of the segments will end up in the "non-motif" cluster, labeled MX. Those phrases which made it into motif clusters are labeled with a unique label referring to their specific motif clusters, such as $M01$, $M02$, and so on. Once all phrases are labeled, a fixed-size *window* of a set number of bars (set to 4 by default) slides over the lead sheet, extracting the sequences of labels. Each set of labels becomes a *motif pattern*.

These patterns are the means by which motifs are incorporated into a generated solo. Figure 6 shows a four measure section from Miles Davis's solo in "On Green Dolphin Street." The measures are labeled $M0$, MX, $M0$, and $M1$ respectively. This means the first measure and third measure were put into the same motif cluster and represent the same motif. The second measure, labeled MX, was put into the "non-motif" cluster and so is marked as thus. In the final grammar, the MX nonterminal will be able to be actualized into any arbitrary piece of melody from the original solo. The final measure, $M1$, was put into some other motif cluster.



$$P0 \rightarrow (share\ M0)\ MX\ (unshare\ M0)\ M1$$

Fig. 6. A section of a lead sheet with the labellings for each measure and corresponding motif pattern noted below

Motif patterns are extracted and generalized from a transcription, by scanning the transcription a second time. The order of motifs and "non-motifs" is noted and broken into 4-bar long *pattern* productions which are then added to the grammar.

Motif patterns give greater coherence to generated solos by enforcing compositional requirements learned from a transcription. These compositional requirements enable nondeterminism to occur while maintaining a sense of structure to the solo.

In addition to adding coherence to generated solos, motif patterns also are able to capture motifs of variable lengths. While the learning of motifs is currently limited to a single size, by letting that size be small, any motif that can be evenly divided by that motif size can be learned. This is done by requiring multiple motifs to be played one after the other through the use of motif patterns. A motif pattern can enforce an ordering of several motifs such that they are always played one after the other.

5.7 Nondeterminism

Nondeterminism can be incorporated into the system at two levels. At the top level, a choice can be made between using either the original grammar or a motif pattern to fill a section of music. The probabilities of one over the other is dictated by the generative-motifness, as mentioned in the next section. This ensures good variation within and between generated solos. With P as the start symbol for the grammar, the two probabilistic productions below show how each successive 16 beats may be filled using original (non-motif) vs. motif productions with probabilities p_1 and p_2 respectively. Here $Original$ is the would-be start symbol for the non-motif productions, while $UseMotifPattern$ is a separate start symbol that selects one of the motif patterns.

$$P \rightarrow (fill\ 16\ Original) \qquad [p_1]$$
$$P \rightarrow (fill\ 16\ UseMotifPattern) \qquad [p_2]$$

The second in which way nondeterminism is utilized is through having several motifs from each motif cluster with the same left-hand side non-terminal, as described earlier, and also through the set of "non-motifs." Each time a "non-motif" is discovered during the learning of motif patterns, it is represented in the rule by a special symbol, MX. At run-time, MX can be expanded into any non-motif melody.

5.8 Adjusting "Motifness"

There are two concepts of "motifness" in the use of motifs that permit a degree of adjustability. The first is in the learning phase. The user-specified "learning motifness" parameter γ specifies how close a motif must be to the centroid of a candidate cluster in order to be placed into the cluster. This loosely translates to how much variation each motif had in the original transcription.

The second concept, "generative-motifness" is at the time a solo is generated. Motifness at this point refers to how often to use a motif pattern (described in 5.6) versus standard grammar rules learned at the same time that do not use motifs. This was described in the previous section on nondeterminism.

6 Qualitative Evaluation

Learning motifs of a length of one measure and patterns that are four bars long, produces convincing solos, showing first that motifs can be effectively learned using our algorithm. Second, it shows that our method of utilizing motifs through motif patterns gives coherency and overarching structure to the generated solos. In Figure 7, we present a random example of a full generated chorus of "Giant Steps" with motifness set to 100%, to be compared to Figure 8, a random example that does not use motifs. In Figure 7, we have marked two evident motifs, the first occurring twice and the second three times. In contrast, no repeated motifs are identifiable in Figure 8. The grammars used were learned from a corpus of

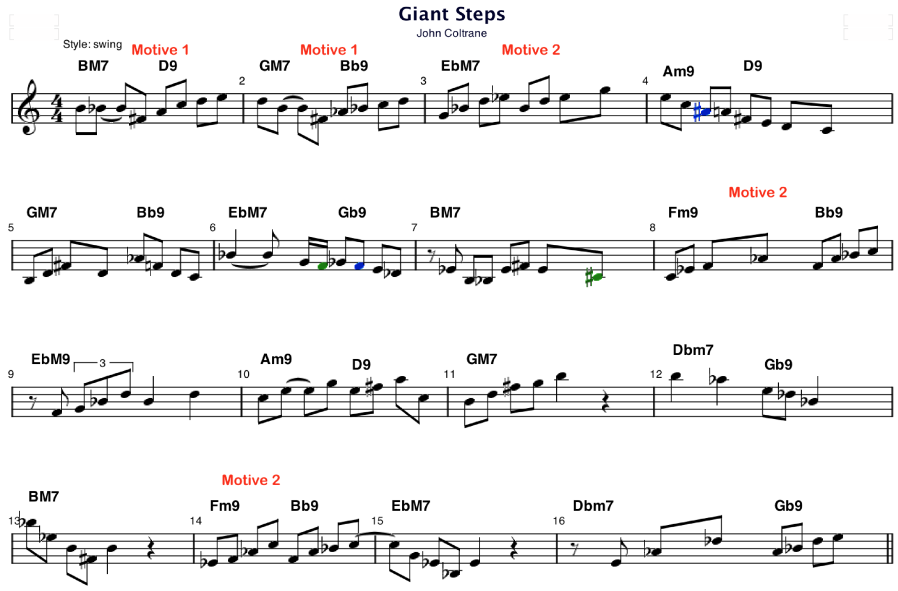


Fig. 7. A chorus generated with motifs

two John Coltrane solos, one over "Giant Steps" and the other over "Moment's Notice".

These examples demonstrate, as before- and after-snapshots, that our mechanisms of learning and exploiting motifs are effective. They do not, of course, prove that a solo exploiting motifs is more pleasing to the listener. While the contribution of motifs to coherence is evident, it could be that some listener aesthetics prefer greater or fewer repeated motifs, despite the contribution to coherence. We also do not claim that motifs are the only way to achieve coherence.

7 Conclusion

Given limited training data, our method is able to learn a wide variety of motifs from a corpus of transcriptions and use them in generated solos that share a similar sound but are different from any particular solos in the training data. Our research discovers motifs in a probabilistic way that allows for fuzziness in each motif, which is key in the discovery of jazz motifs due to plasticity of a motif through the course of a solo.

While it is difficult to describe explicitly the creative process relating to jazz solos, Johnson-Laird [22] gives a clear set of criteria, which he calls the "NONCE" definition of creativity. He states that for a result to be the product of creativity, it:

1. must be novel to the individual who created it;

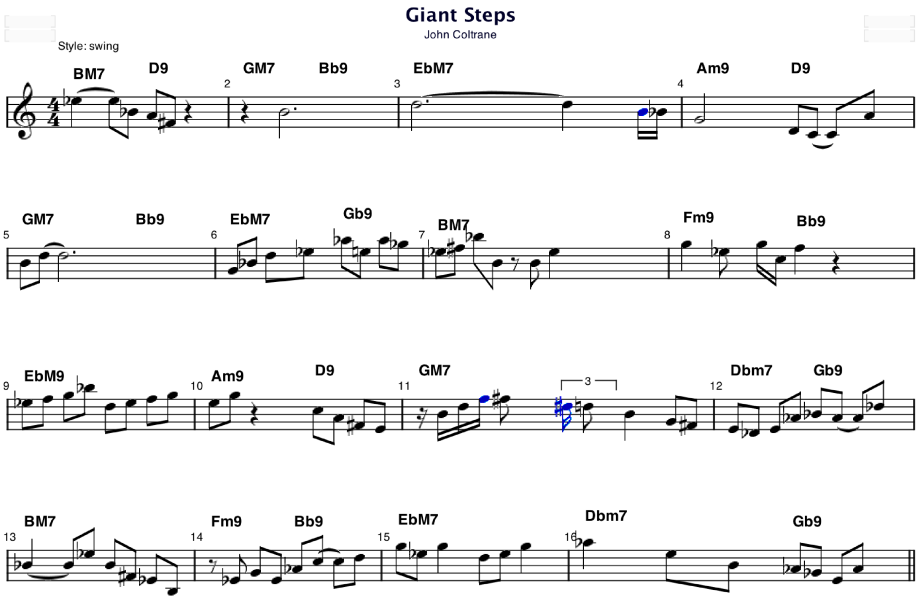


Fig. 8. A chorus generated without motifs

2. need not be novel to the society, but may optionally be so;
3. must be the product of a nondeterministic process;
4. must satisfy preexisting constraints or criteria; and
5. must be based on existing elements that are used as the building blocks or "raw material" for the end result.

We offer that our process for motif utilization satisfies Johnson-Laird's requirements. The motifs used and patterns in which they are used change on each generation of a solo, even if generating within the same chorus. The probability of the same motifs and patterns being used in the same way in two separate solos is nearly zero. The use of probabilities makes the process inherently nondeterministic. Generating melody over a preexisting chord structure ensures that the generated melody fits properly both in rhythm, length, and consonance, thus satisfying preexisting criteria and constraints. Finally, the generation process uses melodic and motivic ideas learned from a corpus, so that a sense of style is provided in improvisations.

8 Future Work

In the future, we hope to expand our criteria for motif clustering and equivalence to capture motifs invariant of common motivic development techniques such as changing a motif's rhythm or inverting a motif. In addition, we plan to adapt

this research to work in an online manner so that it can be integrated into Impro-Visor's active trading mode [23]. Finally, a subjective evaluation from unbiased third-parties is in order.

References

1. Owens, T.: Charlie Parker: Techniques of Improvisation. Ph.D. Dissertation UCLA (1974)
2. Martin, H.: Charlie Parker and Thematic Improvisation. Scarecrow Press (2001)
3. Haerle, D: Magic Motives. Jamey Aebersold Jazz (2015)
4. Terefenko, D. : Jazz Theory, Second Edition. Routledge (2018)
5. Gillick, J., Tang, K., Keller, R.M.: Machine Learning of Jazz Grammars. Computer Music Journal. 34, 56-66 (2010)
6. Impro-Visor (Improvisation Advisor) `https://www.cs.hmc.edu/~keller/jazz/improvisor/`
7. Grachten, M: JIG: Jazz Improvisation Generator. Workshop on Current Research Directions in Computer Music, 1-6. (2001)
8. Hsu, J-L, Liu, C-C, and Chen, A.L.P.: Discovering nontrivial repeating patterns in music data. IEEE Transactions on Multimedia. 3, 3, 311–325 (2001)
9. Weiss, R. J. and Bello, J.P.: Unsupervised Discovery of Temporal Structure in Music. IEEE Journal of Selected Topics in Signal Processing. 5, 6, 1240–1251 (2011)
10. Rolland, P. and Ganascia, J. Automated Motive-Oriented Analysis of Musical Corpuses: a Jazz Case Study. Proceedings of the 1996 International Computer Music Conference, ICMC, 240-243 (1996)
11. Pinto, A.: Relational motif discovery via graph spectral ranking. Proceedings of the Eighth Workshop on Mining and Learning with Graphs - MLG '10. 102–109 (2010)
12. Lartillot, O. A Musical Pattern Discovery System Founded on a Modeling of Listening Strategies. Computer Music Journal. 28, 3, 53–67 (2004)
13. Chiu, S-C and Shan, M-K: Computer Music Composition Based on Discovered Music Patterns. IEEE International Conference on Systems, Man and Cybernetics. 4401–4406 (2006)
14. Gjerdingen, R.O.: Categorization of Musical Patterns by Self-Organizing Neuron-like Networks. Music Perception: An Interdisciplinary Journal. 7, 4, 339–369 (1990)
15. `https://www.cs.hmc.edu/~keller/jazz/improvisor/LeadsheetNotation.pdf`
16. Berliner, P.: Thinking in Jazz – The Infinite Art of Improvisation. The University of Chicago Press. (1994)
17. Norgaard, M.: Descriptions of Improvisational Thinking by Artist-Level Jazz Musicians. Journal of Research in Music Education. 59, 2, 109-127 (2011)
18. `https://en.wikipedia.org/wiki/K-means_clustering`
19. `https://en.wikipedia.org/wiki/Levenshtein_distance`
20. Burkhard, W. A. and Keller, R. M.: Some Approaches to Best-Match File Searching. Communications of the ACM. 16, 4, 230–236 (1973).
21. `https://en.wikipedia.org/wiki/Priority_queue`
22. Johnson-Laird, P. N.: How Jazz Musicians Improvise. Music Perception: An Interdisciplinary Journal. 19, 3, 415–442 (2002)
23. Kondak, Z., Konst, M., Lessard, C., Siah, D., and Keller, R.M.: Active Trading with Impro-Visor. Proceedings of MUME 2016 - The Fourth International Workshop on Musical Metacreation. (2016)