Musical Variation and Improvisation based on Multi-Resolution Representations

Johan Loeckx

Artificial Intelligence Lab
Vrije Universiteit Brussel
Pleinlaan 2, 1050 Brussel, Belgium
jloeckx@ai.vub.ac.be

Abstract. Musical creativity is one of the show pieces of Artificial Intelligence and during the last decades, many paths have been explored to capture musical style and to generate new music. One of the approaches exploits the similarities between music and language. In this paper, Fluid Construction Grammar, a state-of-the-art computational grammar is used to parse/analyse an existing piece, in order to create a variation on the song and generate an improvisation in the same style, using the same bi-directional grammar. A novel multi-resolution time representation to model musical melodies is presented.

Keywords: Fluid Construction Grammar, Improvisation, Musical Style, Composition

1 Introduction

Many paths for representing and creating music have been explored in the last decades, including Hidden Markov Models [12], Machine Learning, Prediction Suffix Trees [9], n-gram models [13], schema-based approaches [6] and connectionist models [19]. Although a lot of these techniques have achieved many successes, most computational representations still lack the expressive power to model the complex subtleties and interactions in music. On of the running debates is the question what the analogies between music and language are [3] and whether paradigms and concepts of language can be successfully applied to music. In the past, for example, generative context-free grammars have been used [10, 15], in which *constructions* (the rules of the grammar) are written in a generic form of recursive rewrite rules. The term "context-free" stems from the fact that the constructions can be applied *regardless of the context of their left-hand side*.

There are, however, some fundamental problems with these grammars in music as much as in language. First, music is heavily context-dependent. Though previous research efforts have attempted to generalize or extended these kinds of grammars to account for some of the context [14], they lack the intrinsic expressive power to model the complex context-aware interactions that constantly occur in music. Second, the single-aspect view on notes or degrees as atomic entity, combined with the limited expressive power of the rules, are not satisfying in the domain of music and multiple-viewpoint approaches are required [2]. For example, (a) the same note can have two degrees in a modulation sequence, depending whether you take the first or second key as a reference; (b) in a polyphonic piece, a note is both part of a "musical phrase" in the same voice (e.g. the highest note in the phrase) as part of a chord with regard to the other voices.

For this reason, the author opted for Fluid Construction Grammar (FCG), a state-of-the-art computational grammar that allows the expression of context in a sophisticated way and preserves the intrinsic multi-viewpoint nature of musical artefacts. In this paper, a novel methodology to create improvisations and variations on existing songs is proposed, based on a multi-resolution representation of music. This work is structured as follows. First, the concepts of Fluid Construction Grammar are introduced. Next, the methodology is explained and the implementation discussed. The approach is finally validated on two example, one for improvisation and one for variation.

2 Fluid Construction Grammar

2.1 Paradigm

As explained above, it is essential that a computational representation for music provides the necessary formalisms and processing mechanisms, powerful enough to handle and describe the complexities present in human compositions, as expressed for example in professional musical literature. Fluid Construction Grammar (FCG) is such a formalism that has repeatedly proved its power and flexibility in the domain of linguistics [17, 4, 20]. FCG was conceived in 1998 as an underlying formalism for modelling language evolution using language games played by autonomous robots [7]. It allows parsing a (natural language or musical) utterance into a semantic interpretation, and vice versa (during production), using the same bi-directional rule set of constructions. It uses as many as possible existing widely accepted notions in theoretical and computational linguistics, including but not limited to:

- *feature structures* for the representation of musical symbolic information;
- *abstract templates or constructions* for the representation of grammatical patterns;
- *general operators* for building up syntactic and semantic structures.

FCG represents all information during parsing and production using *transient structures*, which are extended coupled feature structures consisting of two sets of addressable units, called poles: a syntactical one that corresponds to structure and a semantic one that represents the "meaning" of a musical phrase. One of the strong point is that the feature structures provide information from different linguistic or musical perspectives. A note, for example, can have different viewpoints (an absolute pitch and a degree with respect to the key and chord) and relations, creating a sophisticated context to which constructions can refer.

A construction is an abstract schema that can be used to expand any aspect of a transient structure from any perspective and it can consult any aspect of the transient structure to decide how to do so. Constructions are not merely intended to be descriptive, they should support parsing and production processes as active computational entities. FCG-constructions use the same representation as transient structures, i.e. they are formulated in terms of named units, features and values. A construction may also introduce new units, and establish new relations between units. They are more abstract however than transient structures because its elements can be variables instead of concrete elements, and not all the features should be specified.

One of the main features of construction grammar is that the basic unit of grammar, the construction, deals not only with syntactic issues but also with semantics, pragmatics and meaning. In this sense, a construction is a pairing of form and meaning/function. It is important to note that the same construction can be used for parsing as well as production.

2.2 Differences with context-free grammars

For those familiar with existing work on generative grammars for music production and representation, it is crucial to understand the fundamental differences between FCG and more "traditional" context-free grammars.

1. *FCG marks no sharp distinction between idiomatic and general rules:* a grammatical construction can be either highly specialised, e.g. applicable to a single case, or very abstract, covering a wide range of uses.
2. There is a *continuum in the hierarchy and domain of rules:* constructions can involve different levels of abstraction. For example, the same rule can for example apply to one note or to a motif and there no distinction in the formulation or processing of melodic or harmonic entries.
3. FCG allows *schematisation through variable binding and categorisation:* Grammatical constructions can be made more abstract (or schematic) than concrete instantiations of a musical phrase in three ways: parts of the structure can be left out (widening the context of application), variables can be used instead concrete instantiations and categories are introduced.
4. *Constructions can be combined* – several constructions all matching with different parts of the musical phrase – *or integrated*, using hierarchical constructions that combine partial structures into larger wholes.

2.3 Parsing and producing music

In comprehension or parsing, the transient structure starts from information derived from the input utterance and is progressively expanded using constructions until the meaning can be derived. In formulation, the transient structure contains initially the meaning to be expressed and then consecutive operations expand this structure until all the information is there to render an utterance. In this paper, we will not go deeper into detail into the semantics of music, which is a controversial topic on its own [11, 8, 16], but use a pragmatic interpretation that is not based on philosophical grounds but serves our purpose: to create a new song in the same (rhythmical) style; or to create the same song in a different style.

To facilitate further explanations, we'll briefly go into deeper detail on how the parsing process works and how FCG builds higher order abstractions when presented with a basic symbolic representation of a musical melody. Let us start from a simple list-based musical notation in which every extra level of brackets introduces a double timing resolution or quantization (e.g. from fourth to eighth note). Parsing the two bars at Layer 0 pictured in Fig. 1 to obtain the Layer 3 abstract representation, is achieved by executing following command in FCG/LISP:

```
(parse '(((1 2 3 4) 5 5 6 6) 5) *constructions*)
```

in which every number represents the degree of the note. The `*constructions*` variable contains a set of constructions (the grammar), which is essentially a set of templates that can be applied to the current *transient structure*. Applying such templates or constructions occurs in two phases:

– *Matching:* The features present in a template are matched with features in units in the transient structures; variables can represent any kind of structure (as everything is a list) and thus occur at different levels of the hierarchy.
– *Merging:* When the matching phase succeeded, the construction and transient structure are merged and all variables bound. New units can be created using the so-called ”J” operator.

FCG is particularly suited for music because the complex short- and long-term and transversal interactions that frequently occur in music demand a representation that is able to capture these hierarchical elements while not being constrained by them [18]. More generally, music consists of many dependencies between different aspects and multiple views at different abstraction levels, as musical structures are not simple linear or purely hierarchical structures [2] but many sophisticated interactions between entities and across different levels of abstraction, occur [1].

3 Methodology & Results

Time to get to the core of our methodology. The creation of a new piece of music in the style of another, occurs in 4 phases:

1. *Abstraction:* in a first stage, a given song is being parsed by FCG to create a multi-resolution representation of the piece by progressively making more and more abstraction of time. Layer 0 corresponds to the original input.
2. *Parsing:* at each resolution, (grammatical) constructions are applied that transform the representation (Layer N) to a new one with higher resolution (layer $N-1$). It can be considered as a ”trace” of how to arrive to the original song, starting from a time-abstracted representation. We define the set of applied generic constructions as *the musical style* of the song.
3. *Introduction of novelty:* In case of variation, a different set of constructions (different musical style) is applied to the highest abstracted song. In case of improvisation, a different abstract melody is used as input for the same construction set (style).
4. *Production:* The reverse process from parsing is applied (remember that all constructions are operational for parsing as well as production).

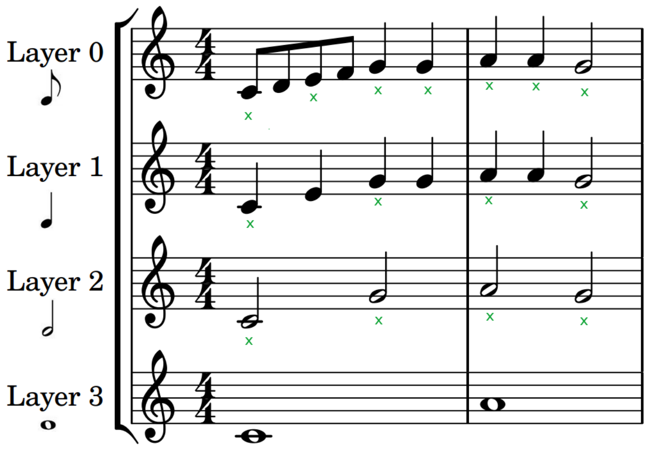


Fig. 1. Multi-resolution view on a simple melody. Layer 0 captures all fine-grained rhythmic information, while higher layers create rhythmic abstractions based on reduction. In this example reduction is obtained by selecting the "strong beat" at each layer, indicated by a cross, for each layer. The resolution (quantization) is indicated for each layer at the left. All different layers are stored simultaneously to create a multi-resolution representation of a musical phrase, model for a hierarchy of time and allowing to capture short-term details as well as long-term dependencies and discourse structures.

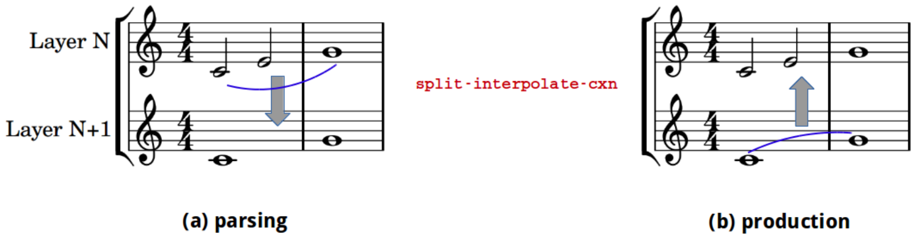


Fig. 2. The split-interpolate-cxn construction applied both in parsing and in production. During parsing, the construction creates a generic operator that can restore the abstraction performed by moving to a different resolution (higher layer or quantization). In production, application of the construction restores the lower layer representation. It is important to notice that the operator is a generic operator, which means that it can be applied to different notes as well. For example, in production, applying the same construction to the two whole notes of C and E at layer N, would yield the sequence C D E at layer $N - 1$.

We'll now look into deeper detail into each of these phases. Fig. 1 illustrates how a multi-resolution representation is derived. At each time-resolution or quantization-layer (indicated on the left), the amount of rhythmical information is reduced by summarizing two notes at layer N into one at layer $N + 1$ (for example two eighth notes into one fourth). A rudimentary way to this is to withhold the strong beat only, as illustrated. Please note that this particular implementation is chosen only to illustrate the concepts and far too simplistic to yield satisfactory results in general. A better approach would be to capture relevant rhythmical and harmonic information, for example by means of melodic reduction [5].

In a second stage, constructions are applied that specify the transformation from one layer to the other in a generic manner. An example is given in Fig. 2. The two half notes C and E, summarized to C by the abstraction process, can be reconstructed by "interpolating" E between C and G. Please note that different constructions can apply to the same abstraction. Table 1 gives an overview of the implemented constructions in our example (that capture musical information).

Name	Explanation
hold	keep a note (a whole stays a whole)
split-interpolate	insert the note in the middle of the two adjacent notes
split-repeat	repeat the note (e.g. a whole becomes two quarter notes)
split-leadto	fill in the note leading to the next note (a step below)
split-movefrom	split a note into the original note and a step above
split-totonic	add the tonic
split-todominant	add the dominant

Table 1. List of bi-directional constructions, explained in the context of production. Please remark that the same constructions are used in parsing as well.

In a third phase, novelty is introduced. In the case of improvisation (Fig. 3) in the style of a given song, the abstracted melody is altered, a different song is chosen or a completely new melody is generated. In the case of variation (Fig. 4), a different set of constructions is applied to the same abstracted melody. One is free to choose the starting resolution, depending on how close or far you want to deviate from the original song. Lastly, the grammar is applied in reverse direction (production), but using the same set of constructions. Fig. 5 shows the LISP code for the split-interpolate construction, suitable for parsing as well as production. Note that variables in FCG are preceded by a question mark. As discussed above, each construction consists of two poles, a semantic one representing meaning and a syntactic one representing form. In *production*, the semantic pole is matched with the abstract representation and the transient structure is enriched with information from the syntactic pole. ?n3-1 and ?n3-2 represent two notes at layer 3, with degrees ?d3-1 and ?d3-2 respectively. If the semantic pole of the transient structure has been tagged with split-interpolate, the construction applies and the syntactic pole is constructed accordingly. In that case, two notes are constructed at layer 2 – indicated with `(length 2)` – and a degree equal to `(?d3-1+?d3-2)/2`. The calculation is performed by the :relation-interpolate? function. The footprints fea-

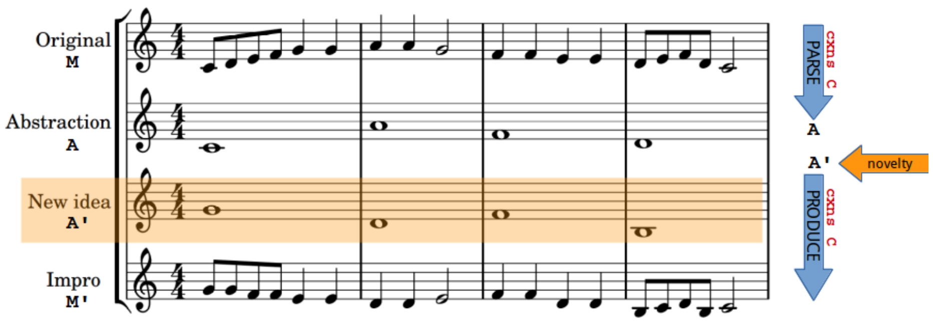


Fig. 3. The same style (set of constructions C), obtained from parsing an existing melody `M` is applied to a different abstract melody `A′` to obtain an *improvisation* `M′` in the same style as `M`. The grammar should be constructed in such a way that all relevant harmonic and melodic musicality is captured, so that any abstracted melody leads to a 'sensible' melody. One possibility is to start from the rules of counterpoint to express common musical knowledge.

ture is used to keep track of all applied constructions. In parsing, the process is reversed and the syntactic pole is matched with the multi-resolution representation to enrich the semantic pole of the transient structure, in this case with the split-interpolate.

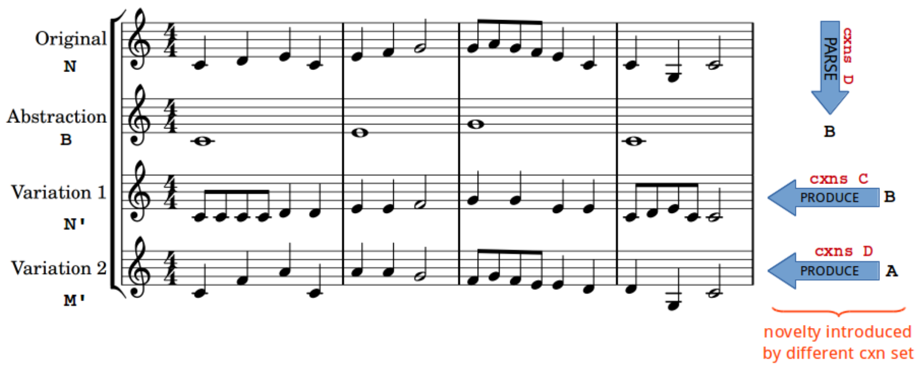
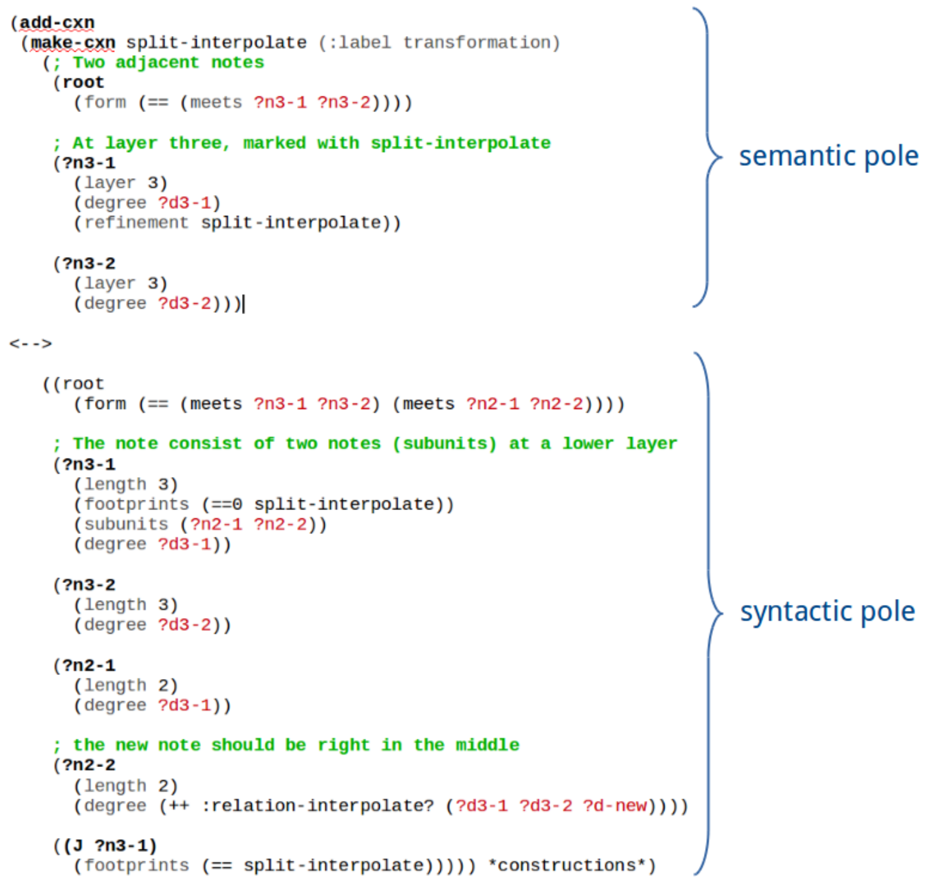


Fig. 4. The style extracted from melody `M` in Fig. 3 is applied to the time-abstracted version `B` of Frère Jacques without repeats (`N`), to yield a *variation* `N′`.

4 Conclusions and Future Work

First steps towards deeper symbolic representations of music for the modelling of style and generation of music were presented in this paper. The methodology has shown its capability to capture the style of a song and it was shown how a bi-directional grammar,



```
(add-cxn
 (make-cxn split-interpolate (:label transformation)
   (; Two adjacent notes
    (root
      (form (== (meets ?n3-1 ?n3-2))))

    ; At layer three, marked with split-interpolate
    (?n3-1
      (layer 3)
      (degree ?d3-1)
      (refinement split-interpolate))

    (?n3-2
      (layer 3)
      (degree ?d3-2)))|

<-->

   ((root
      (form (== (meets ?n3-1 ?n3-2) (meets ?n2-1 ?n2-2))))

    ; The note consist of two notes (subunits) at a lower layer
    (?n3-1
      (length 3)
      (footprints (==0 split-interpolate))
      (subunits (?n2-1 ?n2-2))
      (degree ?d3-1))

    (?n3-2
      (length 3)
      (degree ?d3-2))

    (?n2-1
      (length 2)
      (degree ?d3-1))

    ; the new note should be right in the middle
    (?n2-2
      (length 2)
      (degree (++ :relation-interpolate? (?d3-1 ?d3-2 ?d-new))))

    ((J ?n3-1)
      (footprints (== split-interpolate))))) *constructions*)
```

Fig. 5. LISP code for the FCG construction `split-interpolate`, musically illustrated in Fig. 2. The specification contains all required information for parsing (moving from the syntactic to the semantic pole) and production (reverse direction). The `split-interpolate` construction transforms a note at level N into two notes at level $N-1$ (smaller quantization level), in which the first note is equal to the original note and the second note interpolates between the first note and the next.

representing a musical piece, can be derived from a multi-resolution representation. The approach builds a bridge between corpus-based and knowledge-based systems. Two examples were given, illustration variation and improvisation. A lot of improvements can be made to the proposed methodology. Currently, it only supports monophonic binary music in one key. Also, the current grammar ignores chord structure, leading to "odd sounding" compositions. More research is to be done on the abstraction operator that constructs the multi-resolution representation, so it accounts for chord structure, the tonality of notes and syncopated rhythms.

Acknowledgements. This research has been supported by the EU FP7 PRAISE project #318770.

References

1. Christopher Alexander. A city is not a tree. *Architectural Forum*, 1965.
2. Darrell Conklin and Ian H Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research*, 24(1):51–73, 1995.
3. S Feld and A.A. Fox. Music and language. *Annual Review of Anthropology*, pages 25–53, 1994.
4. Kateryna Gerasymova, Luc Steels, and Remi Van Trijp. Aspectual morphology of russian verbs in fluid construction grammar. In *Proceedings of the 31th Annual Conference of the Cognitive Science Society*, pages 1370–1375. Cognitive Science Society gerasymova-09a. pdf Google Scholar, 2009.
5. Édouard Gilbert and Darrell Conklin. A probabilistic context-free grammar for melodic reduction. 2007.
6. Robert Gjerdingen. *Music in the galant style*. Oxford University Press, 2007.
7. Thomas Hoffmann and Graeme Trousdale. *The Oxford handbook of construction grammar*. Oxford University Press, 2013.
8. Stefan Koelsch, Elisabeth Kasper, Daniela Sammler, Katrin Schulze, Thomas Gunter, and Angela D Friederici. Music, language and meaning: brain signatures of semantic processing. *Nature neuroscience*, 7(3):302–307, 2004.
9. Olivier Lartillot, Shlomo Dubnov, Gérard Assayag, and Gill Bejerano. Automatic modeling of musical style. In *Proceedings of the 2001 International Computer Music Conference*, pages 447–454, 2001.
10. F. Lerdahl and R. Jackendoff. An overview of hierarchical structure in music. *Music Perception*, pages 229–252, 1983.
11. Marvin Minsky. *Music, mind, and meaning*. Springer, 1982.
12. Francois Pachet. The continuator: Musical interaction with style. *Journal of New Music Research*, 32(3):333–341, 2003.
13. Marcus T Pearce and Geraint A Wiggins. Expectation in melody: The influence of context and learning. *Music Perception*, 23(5):377–405, 2006.
14. Martin Rohrmeier. A generative grammar approach to diatonic harmonic structure. In *Proceedings of the 4th Sound and Music Computing Conference*, pages 97–100, 2007.
15. Martin A Rohrmeier and Stefan Koelsch. Predictive information processing in music cognition. a critical review. *International Journal of Psychophysiology*, 83(2):164–175, 2012.
16. L Robert Slevc and Aniruddh D Patel. Meaning in music and language: Three key differences: Comment on towards a neural basis of processing musical semantics by stefan koelsch. *Physics of life reviews*, 8(2):110–111, 2011.

17. Luc Steels. *Design patterns in fluid construction grammar*, volume 11. John Benjamins Publishing, 2011.
18. Luc Steels and Joachim De Beule. Unify and merge in fluid construction grammar. In *Symbol grounding and beyond*, pages 197–223. Springer, 2006.
19. Peter M Todd and D Gareth Loy. *Music and connectionism*. Mit Press, 1991.
20. Remi van Trijp. Feature matrices and agreement: A case study for german case. *Design Patterns in Fluid Construction Grammar*, 11:205, 2011.