Automatic Semantic Annotation of Music with Harmonic Structure

Tillman Weyde, Jens Wissmann
Music Informatics Research Group, Department for Computing, City University, London, United Kingdom

***Abstract*—This paper presents an annotation model for harmonic structure of a piece of music, and a rule system that supports the automatic generation of harmonic annotations. Musical structure has so far received relatively little attention in the context of musical metadata and annotation, although it is highly relevant for musicians, musicologists and indirectly for music listeners. Activities in semantic annotation of music have so far mostly concentrated on features derived from audio data and file-level metadata. We have implemented a model and rule system for harmonic annotation as a starting point for semantic annotation of musical structure.**

Our model is for the musical style of Jazz, but the approach is not restricted to this style. The rule system describes a grammar that allows the fully automatic creation of an harmonic analysis as tree-structured annotations. We present a prototype ontology that defines the layers of harmonic analysis from chords symbols to the level of a complete piece. The annotation can be made on music in various formats, provided there is a way of addressing either chords or time points within the music. We argue that this approach, in connection with manual annotation, can support a number of application scenarios in music production, education, and retrieval and in musicology.

***Keywords*—harmonic analysis, semantic description, automatic semantic annotation, grammar, ontology.**

I. Introduction

Computer music software and standard data formats provide little means for representing harmonic structure of music. The interest in the semantic annotation of music, especially automatic annotation, has in recent years mainly focused on music information retrieval based on audio-related features and the classification of genres, styles, or artists. The musical content and structure as it is represented in traditional musical notation and analysis is rarely dealt with.

The harmonic structure of music is important to both music theory and practice, harmonic analysis is taught to music and musicology students, and it has been the subject of computer based research (e.g. [1], [2]). Chord symbols are used by musicians in jazz and pop music, they appear in figured bass in baroque music, and they are used in music theory for all styles. Chords symbols can be found in a broad range of data on the web, e.g. song collections in text format (alternating lines of text and chord symbols), music notation with chord symbols, guitar tablatures, and there is even a specific markup language for adding chord symbols to lyrics, called ChordML [3]. There is no formal standard for symbols and their meaning in music theory, but there are some widely accepted conventions in mainstream music theory, such as using scale degrees and functions. The ontology we present here is based on these conventions and adds some elements for completeness and consistency.

In music theory, an analysis of the harmonic structure of a piece is started by subsuming simultaneous or successive notes to chords classes, chords classes to functions and functions to progression patterns. This structure provides condensed information on the harmonic characteristics of a piece, helping musicians to understand, memorise, and play the music. Therefore, the creation of these kinds of annotation is very useful for applications in musicology, music education, and music information retrieval.

Music theorists often speak of the 'rules' of harmony, e.g. Fux' gradus [4], or preference rules as used by Lerdahl and Jackendoff [5] or Temperley [6]. Although these rules are not formally specified (the works mentioned before are exceptions insofar as they do formalise their rules to some degree), it is desirable for a musical knowledge representation to represent these rules formally. In music as a form of art rules can be broken, but they can still give a description of relevant harmonic structures at least within a given style and define better the meaning of the represented concepts. Appropriately formalised, rules can be specified to a degree where they can be used as the basis for automatic music analysis to generate harmonic structural descriptions. Annotating music manually can be tedious, therefore a method for automatic annotation can support a musical expert in creating a machine readable representation of musical knowledge, and it helps understand the structure of music better.

II. Harmonic Structure

Harmony in general describes the sounding of several notes simultaneously. For analytical purposes, the harmonic dimension of music is represented by symbols that abstract from the actual notes to classes of chords. This abstraction occurs on the pitch dimension, as octave positions are mostly not taken into account, and on the time dimension, as the harmonic symbols are assigned to measures or parts of measures, which may have notes arranged differently over time. The first abstraction is to classify a set of notes or a time period by assigning a chord symbol, which we assume to be given in this paper. The chord information is often provided in scores for Jazz or Pop Music. The higher levels of analysis put these symbols into a structural context. In tonal music,

this context consists of progression patterns and cadences, which are modelled in the rule system we describe here in this paper.

A. Chord Symbols

At the lowest level, chord symbols classify the notes sounding simultaneously or within a short time period. Chords symbols describe the root and type (or mode) of chords and alterations or extensions of the basic structure of these types. Although these notes may be arranged differently with regards to their distribution over voices and octaves, this specification is sufficient to ensure that it fits with a given melody. This level of description is used in Jazz and Pop music (e.g. in leadsheets and guitar songbooks).

A typical example of a chord symbol is `Am7`, which means that the chord has the root A, is of type minor (that defines it contains the pitches A, C, and E) and a 7th is added (the note G). For the rest of this paper, we will treat chord symbols as atomic units, as we are mainly interested in higher level structures.

B. Higher Structural Levels

Based on the low level chord symbols, structural patterns have been identified in music theory, that describe the relations between different harmonies in a given context. The first level of these are scale degrees, that put a chord in relation to a key, respectively its root note. E.g. IIm7 describes a chord on the second degree of a scale, again in minor with an added 7th. The chord on the first degree (i.e. the root note) is called the tonic.

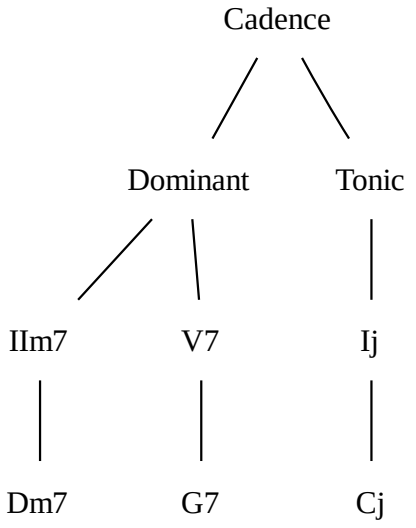
In a piece of music, the chord progression usually departs from the tonic and returns to it is several times. The departing and returning to the tonic is called a *cadence* according to [7]. Although music theorists use the term cadence in different ways, it will be used only with this meaning here. Within a cadence, chords are classified as having certain functions. The tonic is the most stable function, while the dominant (mainly the V7 chord) implies a sense of tension leading to the tonic.

A harmonic analysis starts with the marked up chords symbols (which are often given on metrical units such as whole or half bars) and interprets them as a sequence of cadences. By doing that, a key is implied. This key is often stable but in many pieces of music it will change at some point in a modulation. A modulation is created by chords that cannot be interpreted in the current key and imply therefore a change of key that is often designed in a way that the key is ambiguous over some chords.

A simple typical example of a cadence in jazz is the following:

```
Dm7 G7 Cj
```

The `m7` after the `D` indicates a minor triad with a minor 7th added. The `j` after the `C` indicates a major triad with a major seventh to be added, which makes it a typical tonic chord. In Jazz, the first `m7` chord is in this constellation seen as closely coupled with the following dominant chord as a dominant function to the key a fifth below the dominant chord [8], [9]. This chord sequence is therefore normally (depending on the context) analysed as a cadence in C major, consisting of the degrees II, V, and I. The analysis can be represented as a tree in this form:



As there are more patterns, which occur with variations and extensions, it is necessary to include additional levels, which may seem redundant here. E.g. a dominant can be repeated (with its dependant IIm7), so that an intermediate level has to be included that we called a dominant area.

The recognition of the cadences can be realised as a parsing process based on a grammar. We use here the model defined by Weyde in [10]. This includes ii-V patterns, substitutions, the recursive circle-of-fifth cadences. It has variables that regulate the use of style-specific features, such as the use of the bVII7 in Bebop, cf. [11], or the use of dominant-7 chords on the tonic in Blues.

III. Jazz-OWL Harmony Ontology

To make the knowledge created by a harmonic analysis accessible in the Semantic Web, it is necessary to make annotations, which have well-defined meanings, for which an ontology is used. We have defined an ontology for jazz harmony in the Web Ontology Language (OWL), cf. [12], named *Jazz-OWL*. An OWL ontology specifies the vocabulary and structure for a description that is written in the Resource Description Framework (RDF, see [13] and [14])

A. Described Objects

A description encoded in RDF describes resources that are identified by Uniform Resource Identifiers (URIs), cf. [15]. Although it is possible in an ontology to restrict the kind of URIs, we feel that it is more beneficial to leave it open whether Jazz-OWL is applied on chord symbols, sets of notes, or physical or metrical time intervals.

In practice it is intended that the URI is used to annotate an existing chord symbol, that can be provided MPEG-

SMR[1], MusicXML[2], or MUSITECH [16], [17], or it can provide the chord classification for a metrical or physical time positions in a piece of music. Although MusicXML and MPEG-SMR do currently not provide URIs for chord identification, references can be made using XPath expressions. MUSITECH explicitly addresses objects using unique identifiers that can be expressed as URIs.

B. Class definitions

The Jazz-OWL ontology provides classes for all types of symbols needed on the different levels of harmonic analysis. The classes have properties that are used to define the tree relations as well as harmonic characteristics.

On the lowest level of harmonic analysis chord symbols are annotated. We have defined the following classes and properties, for chord symbols[3]:

```
Namespace( jowl =
 <http://mi.soi.city.ac.uk/smusitech/jowl>)

OWLClass( jowl:ChordSymbol)
    DataProperty( jowl:hasChordSymbol
    range(xsd:string) )
```

A chord symbol can be represented by the OWL class *jowl:ChordSymbol*. Its optional data property *jowl:hasChordSymbol* can contain the concrete string representation of a chord symbol, e.g. `Cj` or `G7`. In cases where the string representation of a chord symbol can be retrieved from the annotated document itself this may be redundant. Nevertheless, this representation is useful when the URI of a *jowl:ChordSymbol* refers to representations like audio data or sets of notes.

```
OWLClass( jowl:DegreeChord)
ObjectProperty( jowl:hasRoot
   domain(jowl:DegreeChord)
   range(jowl:PitchClass))
OWLClass( jowl:PitchClass)
Individual( jowl:C type(owl:PitchClass))
Individual( jowl:D type(owl:PitchClass))
...

/* classes that represent specific
   chord degrees */
OWLClass( jowl:Ij7
   super( jowl:DegreeChord))
OWLClass( jowl:V7
   super( jowl:DegreeChord))
...
```

The *jowl:realizationOfDegree* property associates a chord symbol with the degree chord that it realizes. This property refers to subclasses of the class *jowl:DegreeChord*. These classes constitute a vocabulary of specific degree chords[4] like *jowl:Ij*, *jowl:V7* or *jowl:IIm7*. The root of the chord (e.g. 'C'or 'G') can be stored in the property *jowl:hasRoot* and is represent *jowl:PitchClass*. On the lowest level of harmonic analysis chord symbols are identified. As the key relationships is a main characteristic of harmonic models, all classes in "layers of analysis" (cf. Figure 1) have an *jowl:hasRoot* property.

Following this pattern, for the next level of analysis vocabulary is provided to describe a *jowl:DegreeChord* as realizing a function (*jowl:FunctionChord*) using the property *jowl:realizationOfFunction*. Different kinds of function chords are again expressed as subclasses:

```
OWLClass( jowl:Function)

/* classes that represent specific
   chord degrees */
OWLClass( jowl:Dominant
   super( jowl:Function))
OWLClass( jowl:Tonic
   super( jowl:Function))
...
```

Due to the temporal nature of music, ordered sequences are natural and ubiquitous data type in the musical data models. Up to this level, our ontology does just contain elements that are used to identify the type of an individual chords. On higher analysis levels, these are related to each other. A piece is understood as a cadence sequence that can contain cadences that contain functional areas that contain individual elements, i.e. chord functions which are related to chord symbols. Unfortunately, OWL has no built-in support for ordering. It is not possible to use the *rdf:List* construct for modelling sequences, as OWL uses *rdf:List*s for its RDF-serialization[5]. This shortcoming has been noted by different authors. Drummond et. al. [18] propose an OWL pattern for lists, which we use to model sequences in Jazz-OWL. The pattern is based on the standard pattern for linked lists:

```
Namespace(lst =
 <http://www.co-ode.org/ontologies/lists>)

Class(lst:OWLList partial
    restriction(isFollowedBy
        allValuesFrom(lst:OWLList)))
Class(lst:EmptyList complete
    lst:OWLList
        restriction(hasContents
            maxCardinality(0)))
lst:EquivalentClasses(EmptyList
    intersectionOf(lst:OWLList
        NOT restriction(lst:isFollowedBy
            SOME owl:Thing)))
ObjectProperty(lst:hasListProperty
    domain(lst:OWLList))
ObjectProperty(lst:hasContents
    Functional
    super(lst:hasListProperty))
ObjectProperty(lst:hasNext
    Functional
    super(lst:isFollowedBy))
ObjectProperty(lst:isFollowedBy
    Transitive
    super(lst:hasListProperty)
```

[1]The MPEG Symbolic Music Representation MPEG-SMR is currently in the process of standardization, cf. http://www.interactivemusicnetwork.org/mpeg-ahg/ .

[2]See http://www.recordare.com/xml.html .

[3]The definitions are given in concrete abstract syntax, cf. http://owl.man.ac.uk/2003/concrete/latest.

[4]As this set is comparatively large, only a few illustrative examples are given here.

[5]http://www.w3.org/TR/owl-semantics/mapping.html#rdf_List_mapping

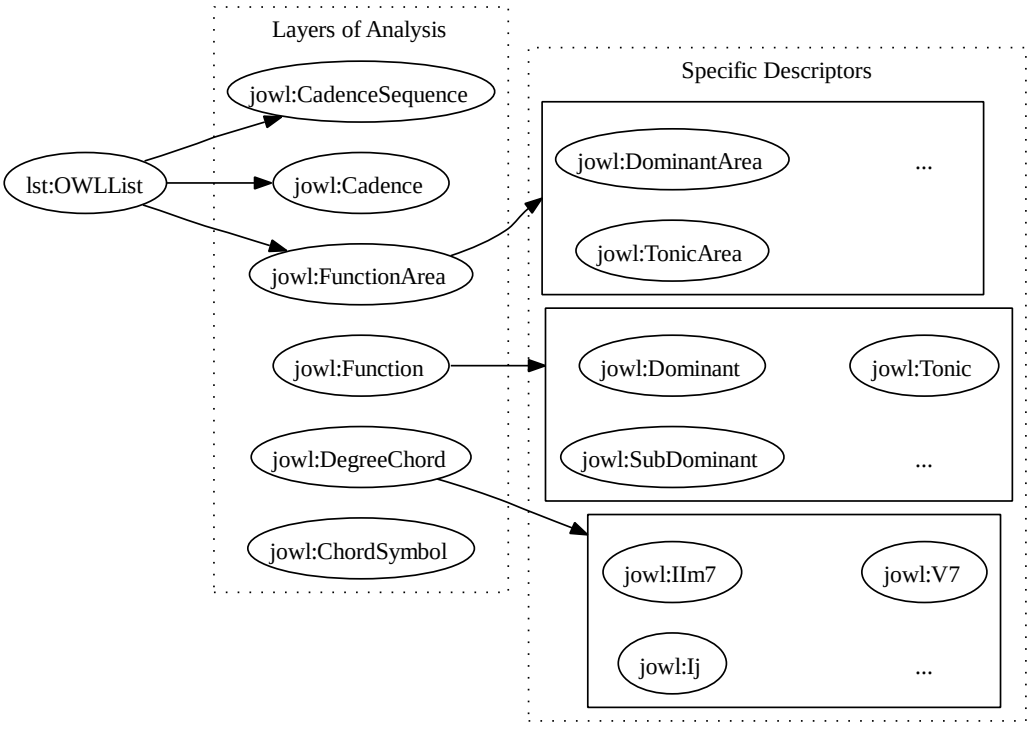


Fig. 1. Jazz-OWL class structure (arrows indicate subclass relationship)

```
   range(lst:OWLList))
```

This is a recursive definition of an ordered sequence, where a list (*lst:OWLList*) contains a first element (referred to using *lst:hasContents*) and followed is by some rest (referred to by *lst:hasNext*) that is itself defined as *lst:OWLList*. The intention is that cells should be directly linked by the functional property *lst:hasNext*. The class *lst:EmptyList* acts as terminal symbol at the end of a sequence. The transitive superproperty *lst:isFollowedBy* can be used in definitions and queries, where the order of elements that are not direct neighbors is of interest.

We use the *lst:OWLList* pattern to model the upper levels of the analysis hierarchy. The top-level sequence is a *jowl:CadenceSequence* that can contain *jowl:Cadence*s that can contain *jowl:FunctionArea*s. A sequence is modelled by using *lst:OWLList* as superclass and by restricting the *lst:isFollowedBy* property to the sequence type. Further the content of the sequence is be defined by restriction:

```
OWLClass( jowl:CadenceSequence
   super( lst:OWLList)
   restriction lst:hasContents
   allValuesFrom(jowl:Cadence)
   restriction lst:isFollowedBy
   only( jowl:CadenceSequence))
OWLClass( jowl:Cadence
   super( lst:OWLList)
   restriction lst:hasContents
   allValuesFrom(jowl:FunctionArea)
   restriction lst:isFollowedBy
   only( jowl:Cadence))
OWLClass( jowl:FunctionArea
   super( lst:OWLList)
   restriction lst:hasContents
   allValuesFrom(jowl:Function)
   restriction
   lst:isFollowedBy
   only( jowl:FunctionArea))
```

IV. Automatic Harmonic Analysis

As mentioned above, the analysis can be created automatically, when a sufficient set of rules is defined. This can be done using a formal grammar, that defines, how symbols on different levels can be related. They have proven useful for describing natural and formal languages. We use a definite clause grammar (DCG). DCGs are an extension of context-free grammars with additional variables, which we use to represent key and mode.

A. A Grammar for Jazz Harmony

Grammars have been used in various musical contexts, see for instance [19], [20], and [21]. The grammar implemented here is a subset of that defined by [10], which extends the works by [22] and [23] on grammars for jazz harmony. The main difference to Ulrich's work is that his system tries to find regions within a piece, where one scale can be used. This is not necessarily identical to the cadences within one key. The grammar introduced by Steedman does introduce more harmonic structure, but it mixes harmony with metrical structure, and it is not strictly context-free, which makes the processing potentially more demanding.

In a basic grammar notation, a rule for simple cadences consisting of dominant and tonic could look like this:

```
Cadence -> DominantArea TonicArea
TonicArea -> Tonic
DominantArea -> Dominant
```

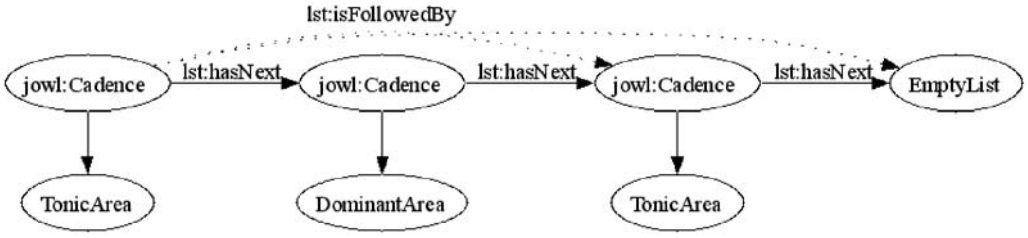


Fig. 2. Representation of a sequence of cadences using the OWLList pattern

```
DominantArea -> Dominant Dominant
```

The third rule uses two dominant symbols which may relate to different realisations of the dominant function, e.g. using a `D7` and a `D79`. To make sure that the chords in the cadence actually relate to the same key, information about the root needs to be added:

```
Cadence(Root) -> DominantArea(Root)
    TonicArea(Root)
DominantArea -> Dominant(Root)
   Dominant(Root)
DominantArea -> Dominant(fifth(Root))
   Dominant(Root)
```

The second rule now describes a sequence of two dominants, while the third contains the dominant to the dominant, which is a fifth higher.

B. Implementation

We have implemented the grammar rules above and all necessary auxiliary rules in SWI Prolog[6] A grammar implementation for Definite Clause Grammars is readily available, as with most Prolog systems, and it accepts grammar rules in a format very similar to the one described above. The Prolog inference provides directly a parsing facility. Also there is an RDF package available, which we use to generate the RDF description when a sequence is parsed.

A typical rule for harmonic analysis now looks like this:

```
i_chord(Root,Tree,URI,[T1,T2,T3|Triples])
-->
d7_chord(Root, ChordTag, ChordURI, Triples),
{atom_concat('Blues I',ChordTag,Tree),
 rdf_db:rdf_bnode(URI),
 T1=[URI,'rdf:type','jowl:iim7'],
 T2=[URI,'jowl:root',Root],
 T3=[URI,'jowl:realizedAsChord',ChordURI]
}.
```

Here the i_chord, which is used as part of the tonic area, is realised as a dominant 7th chord, which is typical for Blues style. This information is added to the textual output in Tree. The RDF information is created in T1, T2, and T3, which are all collected in the list passed as the last argument.

The implementation can be used to analyse sequences of Chord Symbols, and it outputs one or more analysis trees. When several analyses are possible the order of analyses depends on the order of the rules. A call for doing that will look like this:

```
analyse(['Dm7','G7','Cj'], Result,
   Root, Mode, URI, Triples).
```

Prolog then fills the variables with possible values according to the rules, and `Triples` contains the RDF description, if Prolog finds a solution, i.e. an interpretation of the chord symobols according to the grammar rules. The triples it produces have the graph structure shown in Figure 3.

Prolog can also be used to generate chord symbol sequences, it lists all possible sequences that the grammar allows, which may be a very large number. This is potentially interesting for Composers looking for inspiration of for students needing exercise material.

V. Discussion

As we have seen, the first hurdle is the current lack of support for sequential structures in OWL. The OWLList pattern [18] provides an ad hoc solution to this problem. Nevertheless, a standardized solution would be preferable, as the problem of missing support for sequences is also reflected in reasoning tools and other standards that build on OWL. The OWL ontology could be refined to include more specialized sequences types that resemble the grammar to a certain extend. By restricting the *lst:hasContents* and *lst:isFollowedBy* properties, we can define the elements of a list. For example, a specialized dominant area (*jowl:DominantArea2*) in which contains a suspension chord and the dominant could be defined as follows:

```
hasNext some (jowl:DominantArea2
   and (hasContents some jowl:Suspension)
and (hasNext some (jowl:DominantArea2
   and (hasContents some jowl:Dominant)
and (hasNext some EmptyList))))
```

Using this approach it is also possible to define only parts of a sequence, for instance to state that a sequence ends with the tonic. However, the propagation of root identification from the bottom level (chord symbols) to the top levels (cadence sequence level) would require the use of variables for which we have found no obvious solution in OWL. Further, it would be desirable to represent the rules using web standards as well. Unfortunately, rule languages for the Semantic web are still under

[6]Cf. http://www.swi-prolog.org/ .

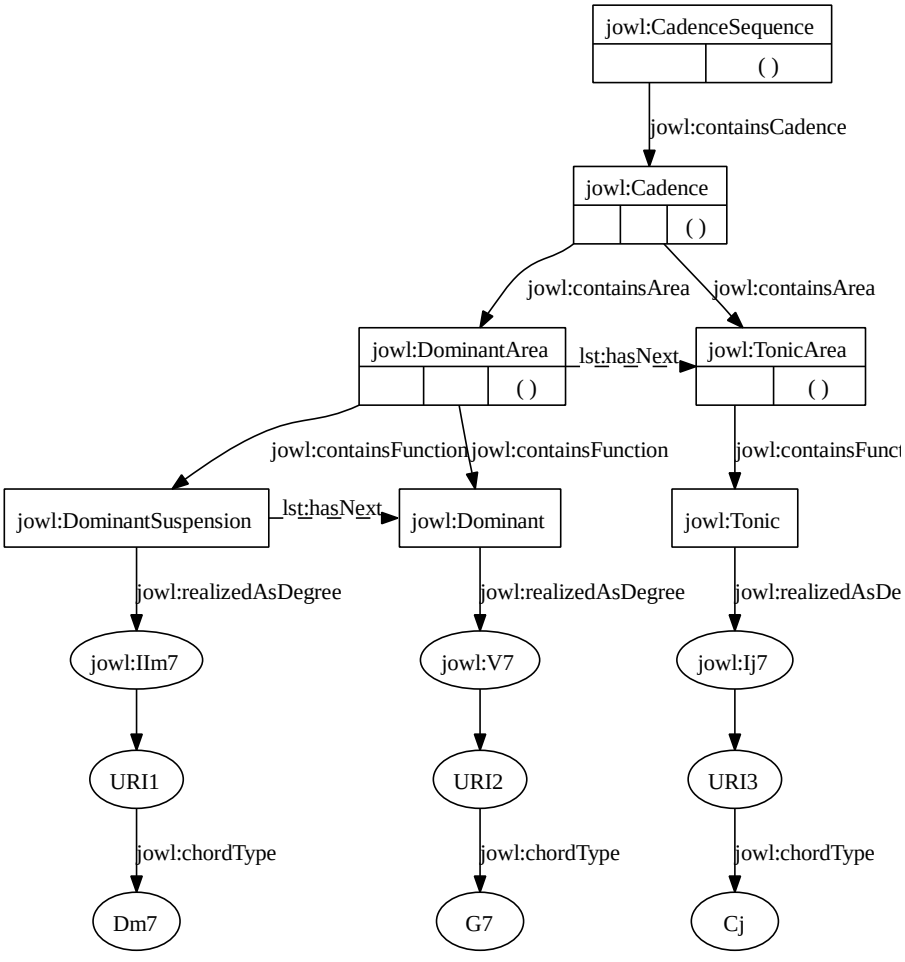


Fig. 3. Example Annotation of the cadence (Dm7, G7, Cj)

development and currently available reasoning engines seem to support only special subsets of the OWL. The most likely candidate, the Semantic Web Rule Language (SWRL) [24] that combines OWL-DL and OWL-Lite with RuleML, lacks some features that we require in order to translate our Prolog rules to SWRL: Most notably there is no notion of rule order. Rule order is a crucial element of in our approach as we use it to model musically preferred interpretations. That is, the topmost rules are the most preferred interpretations and are chosen first by Prolog's back-tracking algorithm. It has yet to be investigated whether other solutions such as modelling preference using weights are feasible. Further, problems of list representation and reasoning seems to be inherited from OWL and need further investigation.

From a musical point of view, the rules are useful to represent properties of music theory concepts and to allow the analysis of pieces conforming to these concepts. For pieces that do not fit exactly into these concepts, annotations can be made by hand. To automate this process, it would be useful to have approximate matching or different degrees of belonging to a concept.

A. Outlook

The semantic annotation of harmonic musical structure is a useful technique that can serve in several applications. The use of Semantic Web technologies allows to make the annotated information available together with an ontology that provides formal definitions of their meaning.

Future research and development should go into applications and the further transfer of representation into the Semantic Web standards. One goal is the integration of OWL and SWRL with Prolog, so that Prolog acts merely as an inference engine on the data in Semantic Web representation.

From the musical point of view, the extension of the rule system, the addition of further styles are desirable, this would include the definition of other chord and functional symbols as well as the corresponding rules. From the technical side, the integration of means for vague representation seem very interesting. E.g. using a fuzzy extension of OWL [25] could allow a more adequate form of music annotation.

REFERENCES

[1] D. Temperley, "An algorithm for harmonic analysis," *Music Perception*, vol. 15, no. 1, p. 31, 1997.

[2] Y. Hiraga, "A computational model of the cognition of melodic/harmonic progression," in *Proceedings of the First International Conference on Music Perception and Cognition*, Kyoto, 1989, p. 61.
[3] G. C. S. Frederico, "Actos: A peer-to-peer application for the retrieval of encoded music," in *First International Conference on Musical Applications Using XML*, 2002.
[4] J. J. Fux, *Gradus ad Parnassum*. 1725 (Neuauflage der Johann Joseph Fux Gesellschaft, Graz; Kassel Basel Paris London New York: Brenreiter und Akademische Druck- und Verlagsanstalt, 1967.
[5] F. Lerdahl and R. Jackendoff, *A Generative Theory of Tonal Music*. Cambridge, Mass.: The MIT Press, 1983.
[6] D. Temperley, *The Cognition of Basic Musical Structures*. Cambridge, Mass.: The MIT Press, 2001.
[7] H. Riemann, *Musikalische Syntaxis*. Leipzig: Breitkopf und Härtel, 1877.
[8] A. Jungbluth, *Jazzharmonielehre*. Mainz: Schott, 1981.
[9] D. Haerle, *The Jazz Language*. Miami, Fl: Studio 224, 1980.
[10] T. Weyde, "Grammatikbasierte harmonische Analyse von Jazzstandards mit Computerunterstützung," in *KlangArt-Kongreß 1995*, ser. Musik und Neue Technologie, B. Enders and N. Knolle, Eds. Osnabrück: Universitätsverlag Rasch, 1998, vol. 1, pp. 315–324.
[11] G. M. Potter, "The unique role of bvii7 in bebop harmony," *jazz forschung*, vol. 21, 1989.
[12] M. Smith, C. Welty, and D. McGuiness, "OWL web ontology language guide," Technical report, W3C recommendation, http://www.w3.org/TR/2004/REC-owl-guide-20040210, 2004.
[13] E. Miller, "Introduction to the resource description framework," *D-Lib Magazine*, 5 1998. [Online]. Available: http://www.dlib.org/dlib/may98/miller/05miller.html
[14] D. B. Ivan Herman, Ralph Swick, "Resource description framework," 2006. [Online]. Available: http://www.w3.org/RDF/
[15] N. W. Ian Jacobs, "Architecture of the world wide web," W3C, Tech. Rep., 2004. [Online]. Available: http://www.w3.org/TR/webarch/
[16] T. Weyde, "Case study: Representation of musical structure for music software," in *2nd MUSICNETWORK Open Workshop*, P. Nesi and et al., Eds., University of Leeds, Leeds, 2003.
[17] M. Gieseking and T. Weyde, "Concepts of the musitech infrastructure for internet-based interactive musical applications," in *Proceedings of the Wedelmusic Conference 2002*, N. e. al., Ed., Fraunhofer IGD, Darmstadt, 2002.
[18] N. Drummond, A. Rector, R. Stevens, G. Moulton, M. Horridge, H. H. Wang, and H. Seidenberg, "Sequences in protégé OWL," in *9th Intl. Protégé Conference - July 23-26, 2006 - Stanford, California*, 2006.
[19] R. u. J. P. F. Eberlein, *Kadenzwahrnehmung und Kadenzgeschichte - ein Beitrag zu einer Grammatik der Musik*. Franfurt a.M: Verlag Peter Lang, 1992.
[20] B. Sundberg, Johan und Lindblom, "Generative theories in language and music description," *Cognition*, 1976, nachdruck in Schwanauer, Stephan M. und Levitt, David A. (Hrsg.): Machine Models of Music. Cambridge Mass. 1993.
[21] C. Roads, "Grammars as representations for music," in *Foundations of Computer Music*, C. Roads and J. Strawn, Eds. Cambridge Mass.: The MIT Press, 1987.
[22] M. J. Steedman, "A generative grammar for jazz chord sequences," *Music Perception*, vol. 2, no. 1, p. 52, 1984.
[23] J. W. Ulrich, "The analysis and synthesis of jazz by computer," in *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, W. Kaufmann, Ed., Cambridge, MA, 1977.
[24] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A semantic web rule language combining owl and ruleml," Technical report, W3C recommendation, http://www.daml.org/2003/11/swrl/, 2003.
[25] S. Vembu, M. Kiesel, M. Sintek, and S. Baumann, "Towards bridging the semantic gap in multimedia annotation and retrieval," in *Proceedings of the 1st International Workshop on Semantic Web Annotations for Multimedia, SWAMM 2006 at the 15th International World Wide Web Conference*, 2006.