DMUN: A Textual Interface for Content-Based Music Information Retrieval in the C@merata task for MediaEval 2016

Andreas Katsiavalos
De Montfort University
Leicester, UK
andreas.katsiavalos@gmail.com

ABSTRACT

This paper describes a text-based Question-Answering (QA) system for content-based music information retrieval (MIR) according to the C@merata task description [12,13].

1. INTRODUCTION

Content-based search of music information is an active research area [4] with applications in education and general musicological tasks. Apart from collections of music data such as KernScores [11], even traditional library catalog servers can be searched based on their content [7]. To access these data and extract content-based information we developed a text query parser that, given a sentence such as a C@merata question, generates a script for music operations. The script contains the music concepts and their relations as described in the query, but in a structured form in such a way that workflows of specific music data operations are formed. A parser then reads the script and calls the corresponding functions from a framework we created on top of music21 [6]. The questions tested are a sub-set of 28 random selections from the complete set of questions.

An overview of the query system is given in section 2 with more detailed descriptions of important concepts and procedures. In section 3 we present the results of the algorithm with detailed description and discuss them. Last, the conclusions are presented in section 4.

2. APPROACH

2.1 Overview

Query parsing and music content operations are kept separate and the only connection between them is through an intermediate layer.

There are three major components of this approach are:

- A query interpreter
- The script language
- A music information workflow interpreter

The query interpreter resolves the query text into a script that describes a music information workflow. This is a layered process that required hard-coded knowledge about valid query terms and types (see 2.2).

The script language consists of “information request” statements that are formed by the clauses: “select”, “from” and “where” having similar functionality as that described by the Structured Query Language (SQL) (see 2.3).

MediaEval 2016 Workshop, October 20-21, 2016, Amsterdam.

The music information workflow interpreter connects the script with a set of music-related functions that are built on top of the music21 framework (see 2.4). discuss the development of the question types over the past three years and in particular focus on the more sophisticated methods adopted for question generation this year. We will then present the participating systems for this year and discuss the results which they obtained.

2.2 The Query Interpreter

The query interpreter is a class that is initalised with a “language” file that contains information about valid terms, their types, composite types and, composite type relations. Composite types are music concepts and will be referred to as entities. This file stores generic terms, but some values, e.g. names of the parts are extracted from music data.

The terms of a query phrase can be:

- **values**,
- music concept/**entity keywords, E**,
- music concept/**entity relation keywords, R**.

For example, “dotted quarter note dominant 7th” is a chord entity. Entities are further categorized into “content” and “context” types. Although in the question set we tested, context entities are the parts and measures and content entities are note, rest, chord and simultaneity, it is the relation keywords that define what is the search context and what is the target content. Relations enable the transformation of the query into a structured request by defining the context-content relation. The conditions are just the entity attributes.

Some of the relation types that were identified in the tested question set are shown below (the “<>” symbol means any type of entity):

< > (" ") < > , <(duration, pitch, note, chord)>
< > ("followed by") < >, <(duration, pitch, note, chord)>
< > ("in", "in the") < > contextual and conditional
< > ("of", "of a") < >
< > ("parallel")
< > ("repeated") <> ("time","times")
< > ("between”, “between the") < > ("and") < >
< > ("against", "only against") < >
…

The terms of the query phrase are processed in layers starting by identifying the type of each one. Next, composite types and words are grouped into entities. After all the types are matched, the entity relations are identified. Last, the query is converted into an information request using “select-from-where” statements.

1. Load the language file
2. Parse the query
2.1 First pass: terms to types
2.2 Second pass: type groups and relations
2.3 Third pass: Content and Context identification
2.4 Fourth pass: Make information request
3 Run information request script with music framework

14 seven-note chord in the harpsichord

context : parts, condition: instrument
get type : chord
condition : cardinality value

Figure 1. Example query analysis

2.3 Information Request using a Script

After the query phrase analysis a script that contains a structured information request is generated by converting the identified entities and their relations into a sequence of "select-from-where" statements.

9 parallel thirds in measures 15-18

```
FROM CONTEXT:
        SELECT   measures
        FROM     parts.all
        WHERE    15 <= measure.number  <=18
SELECT CONTENT
        SELECT   chords
        FROM     CONTEXT
        WHERE    chord.type IS third
WHERE (RELATION)
        parallel
```

14 seven-note chord in the harpsichord

```
FROM CONTEXT:
        SELECT   parts
        FROM     parts.all
        WHERE    part.name == "harpsichord"
SELECT CONTENT
        SELECT   chords
        FROM     CONTEXT
        WHERE    chord.cardinality = 7
```

Figure 2. Text parsing examples of a function calls

By ordering and nesting such statements, all the queries that were tested were successfully converted into this workflow representation.

The use of a "language" file is a way to pass knowledge to the system about how to parse phrases. It contains:

- value collections grouped in primary types
  - e.g. 15-18 is type range.int
- primitive types grouped in music concepts/entities
  - "dotted quarter" is a *duration* entity
  - "first inversion of a triad" is a *chord* entity
- Relation definitions
  - groups of entities

2.4 Music Content Extraction

The structured information request that was described in the previous section is parsed from a music information retrieval interpreter that compiles an executable music21 script using music21 functions such as "getElementByClass()" and a plethora of features for music21.elements to compare with. Operating within the music21 ontology, we can perform conditional part selection, measure selection based on range, and get attribute values for basic elements such as note, rest and chord type.

One way to avoid over-analyzing the query into complicated information requests is to use more complex representations, such as note-sequences (VIS) [1], or Directed Interval Classes [3] and bypass low-level relations by transferring them to the representation.

3. RESULTS AND DISCUSSION

These are preliminary results and the approach is under development. In the rest of this section we discuss how queries resolve into information requests and the difficulties in the process.

3 octave leap in violin I
context : part, instrument type and number
get type : melodic interval, keyword "leap"
condition : interval value

5 Bb3, A3, G3, F3, E3

note,con:seq:comma, note, con:seq:comma, note, con:seq:comma, note, con:seq:comma, note

context : complete piece ? separate parts ?
get type : pitch sequence

9 parallel thirds in measures 15-18

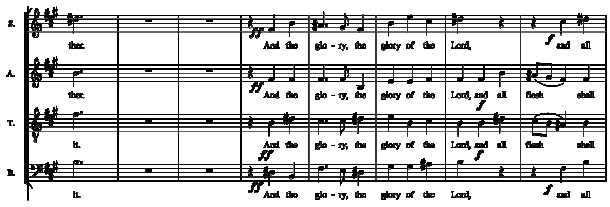
con:relation, interval_type, con:where:in, key, int:comp:range

context : measures
get type : chords:condition:thirds
condition : parallel

#10 authentic cadence in measures 14-18

cadence_type, key, con:where:in, key, int:comp:range

context : measures
get type : cadence
condition : cadence type



18 consecutive sixths between the Altos and Basses in measures 73-80

con:temp_relation, num:position, con:selection:between_the, term, con:and, term, con:where:in, key, num:comp:range

context : measures, int-range
relation : between X and Y
X type : part
Y type : part
content : melodic sequence
condition : interval type

#22 flute dotted half note only against strings

term, duration:exp, duration, key, ? , con:temp_relation:against, ? (find the string parts?) general_polyphony, pitch, on:where: in_the, term, con:where:in, key, num:int, rule:direction

context : parts, instrument
type : duration, composite
relation : only_against
term : part group conditions > not empty ?

#29 flute, oboe and bassoon in unison in measures 1-56

term, term, con:and, term. conection:where-condition:in, interval_type, context:where:in, int:comp,range

context : measures
context : parts, the instruments
type : notes
condition : same notes

#33 semibreve tied to a minim in the Bass clef

duration, con:notation:tied:tied_to_a, duration, con:where:in_the, term, key=type

context : parts ? or measures ?
relation : <a> tied_to <b>
a type : duration
b type : duration

44 four eighth notes in the bottom part
context : part, relative position
relation : sequence
: number <durations,pitches,notes>
type : note, conditions: duration

63 C D E F D E C in semiquavers repeated after a semiquaver
context : all
relation : X repeated after Y
X type : sequence, type: pitch-class
X cond : duration
Y type : duration

77 harmonic octave in the bass clef
context : measures, clef:
type : harmonic interval

Notice the assumption in defining the context that bass clef can appear anywhere in the score and it does mean a complete part.

86 whole-note unison E2 E3 E4
context : all parts
type : chord, from notes in all parts
condition : pitch content
condition : duration

#94 crotchet tied to crotchet
context : single parts
relation : X "tied to" Y
X type : duration
Y type : duration

186 whole-note chord
context : single part ? all parts ?
type : chord

4. CONCLUSION

The C@merata task became very demanding this year; however, this approach seems promising. The use of the intermediate information level created space for interpretations and generally allowed operations aimed at language understanding. Natural language was avoided but this approach seems to resemble natural language query patterns. Even if the query language stays in a limited dictionary and syntax, as long as it serves its purpose as an interface for information retrieval, it is worth attention.

The “segmentation ontology” (Fields et al., 2011) is an interesting idea. This work addresses large parts of the current approach’s need for an ontology, it provides implementations in RDF-OWL language for knowledge representations.

5. REFERENCES

[1] Antila, C., & Cumming, J. (2014). The VIS Framework: Analyzing Counterpoint in Large Datasets. In *Proceedings of the International Society for Music Information Retrieval*. Taipei, Taiwan.

[2] Arzt, A., Böck, S., & Widmer, G. (2012). Fast Identification of Piece and Score Position via Symbolic Fingerprinting. In *13th International Society for Music Information Retrieval* (pp. 433–438). Porto, Portugal.

[3] Cambouropoulos, E., Katsiavalos, A., & Tsougras, C. (2013). Idiom-independent harmonic pattern recognition based on a novel chord transition representation. In *3rd International Workshop on Folk Music Analysis*. Amsterdam, Netherlands.

[4] Casey, M., Veltkamp, R., Goto, M., Leman, M., Rhodes, C., Slaney, M., & others. (2008). Content-based music information retrieval: Current directions and future challenges. Proceedings of the IEEE, 96(4), 668–696.

[5] Collins, T. 2014. Stravinsqi/De Monfort University at the C@merata 2014 task. *Proceedings of the C@merata Task at MediaEval 2014*.

[6] Cuthbert, M. S., and Ariza, C. 2010. music21: a toolkit for computer-aided musicology and symbolic music data. In *Proceedings of the International Symposium on Music Information Retrieval* (Utrecht, The Nethlerands, August 09 - 13, 2010). 637-642.

[7] Dovey, M. J. (2001). Adding content-based searching to a traditional music library catalogue server. In *Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries* (pp. 249–250). ACM.

[8] Downie, J. S., & Cunningham, S. J. (2002). Toward a theory of music information retrieval queries: System design implications.

[9] Fields, B., Page, K., De Roure, D., & Crawford, T. (2011). The segment ontology: Bridging music-generic and domain-specific. In *Multimedia and Expo (ICME), 2011 IEEE International Conference on* (pp. 1–6). IEEE.

[10] Lewis, D., Woodley, R., Forth, J., Rhodes, C., Wiggins, G., & others. (2011). Tools for music scholarship and their interactions: a case study.

[11] Sapp, C. S. (2005). Online Database of Scores in the Humdrum File Format. In *ISMIR* (pp. 664–665).

[12] Sutcliffe, R. F. E., Fox, C., Root, D. L., Hovy, E., & Lewis, R. (2015). The C@merata Task at MediaEval 2015: Natural language queries on classical music scores. In Proceedings of the MediaEval 2015 Workshop, Wurzen, Germany, September 14-15 2015.

[13] Sutcliffe, R. F. E., Fox, C., Root, D. L., Hovy, E. and Lewis, R. (2015). Second Shared Evaluation of Natural Language Queries against Classical Music Scores: A Full Description of the C@merata 2015 Task. Proceedings of the C@merata Task at MediaEval 2015. http://csee.essex.ac.uk/camerata/.