# Stravinsqi/De Montfort University at the MediaEval 2014 C@merata Task

Tom Collins
Faculty of Technology
De Montfort University
Leicester, UK
+44 116 207 6192
tom.collins@dmu.ac.uk

## ABSTRACT

A summary is provided of the Stravinsqi-Jun2014 algorithm and its performance on the MediaEval 2014 C@merata Task. Stravinsqi stands for STaff Representation Analysed VIa Natural language String Query Input. The algorithm parses a symbolic representation of a piece of music as well as a query string consisting of a natural language expression, and identifies where event(s) specified by the query occur in the music. The output for any given query is a list of time windows corresponding to the locations of relevant events. To evaluate the algorithm, its output time windows are compared with those specified by music experts for the same query-piece combinations. In an evaluation consisting of twenty pieces and 200 questions, Stravinsqi-Jun2014 had recall .91 and precision .46 at the measure level, and recall .87 and precision .44 at the beat level. Question categories and examples are analysed where the Stravinsqi algorithm was less successful. Important potential applications of this work in web-based music notation software and musicological research are also discussed.

## 1. INTRODUCTION

There is a long-standing interest in querying music represented as (or derived from) staff notation. Collections of themes/incipits [2] provide an example of such an interest, and these predate even the introduction of computing into musicological research. With the development of digital formats such as kern [8] came algorithmic theme-finders [9], and research based on algorithms that identified scale degrees and melodic intervals automatically [1, 20]. Currently, some of this functionality lives on in a Python package for computational musicology called music21 [5]. It is fair to say, however, that technology does not yet exist capable of:

1. Accepting a music-analytic query in the form of a natural-language string, such as "perfect fifth followed by a D4";
2. Reliably retrieving instances of higher-level music-theoretic concepts from staff notation, such as functional harmonies (e.g., "Ib") or cadences (e.g., "interrupted cadence").

The C@merata task [16] challenges researchers to address both of these issues. Given a natural language query and a piece of music in digital staff notation representation, the C@merata task evaluates an algorithm's ability to identify where one or more events specified by the query occur in the music. Time windows output by an algorithm are compared with time windows defined by a music theory expert. In the evaluation, algorithms that do not miss too many correct time windows will score well on recall, and those that do not return too many false-positive time windows will score well on precision.



Across the English-speaking world, there are many students already benefiting from interactions with web-based music notation software. Noteflight, for instance, enables users to create and share music notation online, with various educational features for high school and college settings.[1] One application of an algorithm that performs well on the C@merata task would be within such an interface, so that students could query and hear/see results for the pieces with which they are working, in order to develop their understanding of various music-theoretic terms. Other applications include continued and enhanced support of musicological research [1, 20], and informing solutions to various tasks in music informatics research, such as expressive rendering of staff notation [7] or automatic generation of music [4], both of which may benefit from knowledge of events such as cadences and/or changes in texture.

The remainder of this paper is devoted to describing an algorithm called Stravinsqi that was submitted to the C@merata task. Several annotated music examples demonstrate in more detail Stravinsqi's ability to extract instances of certain query types from staff notation (harmonic intervals, functional harmonies, cadences, and textures). The results of Stravinsqi on the C@merata task are then presented and discussed.

## 2. APPROACH

### 2.1 Overview

The Stravinsqi-Jun2014 algorithm that was entered in the C@merata task is embedded in a Common Lisp package called MCStylistic-Jun2014 (hereafter, MCStylistic), which has been under development since 2008 [4].[2] MCStylistic includes implementations of several algorithms from the fields of music information retrieval and music psychology, such as the chord-labelling algorithm HarmAn [12], the Krumhansl-Schmuckler key-finding algorithm [10], keyscapes [14], and the Structure Induction Algorithm [11].

From a natural language perspective, there are two types of queries: compound queries such as "a Bb followed a bar later by a C followed by a tonic triad", and ordinary queries such as "perfect cadence". Stravinsqi checks the query string for compound queries and splits it into *N query elements* if necessary, e.g., "a Bb" and "a bar later by a C" and "tonic triad".

The piece is converted from its MusicXML format to kern format using the `xml2hum` script [15].[3] The kern file is parsed by import functions in MCStylistic to give the following representations, which are referred to as *point sets*:

[1] http://www.noteflight.com

[2] http://www.tomcollinsresearch.net

[3] http://extras.humdrum.org/bin/osxintel64/

- Instrument/staff and clef names at the beginning of each staff;
- Bar numbers where time signatures are specified, together with the number of beats per bar, the type of beat, and the corresponding ontime. Ontime is the incrementing time in staff notation measured in crotchet beats, with 0 for bar 1 beat 1. A note beginning at measure 3 beat 1 in a piece with four crotchets per bar has ontime 8, for example;
- A point-set representation of the piece, where each point represents a note. The five-dimensional point consists of the ontime of the note, its MIDI note number, its morphetic pitch number [11], its duration in crotchet beats, and its numeric staff number (with zero for the top staff). Morphetic pitch number is a numeric encoding of staff height, with 60 for C♭4, C♮4, C♯4, 61 for D♭4, D♮4, D♯4, etc. There is a bijection between pitch and (MIDI note, morphetic pitch)-pairs [4], meaning if the MIDI note and morphetic pitch numbers of a note are known, then so are its correct pitch spelling and octave number. And vice versa;
- A point-set representation of the piece with three extra dimensions, one each for articulation, dynamics, and lyrics information. If, for example, a note is marked marcato, then the dynamics entry of the point will contain ">";
- A point-set representation of the piece, where each point represents a notated rest.

Each query element is passed to several sub-functions (e.g., `harmonic-interval-of-a`, `duration&pitch-time-intervals`, `rest-duration-time-intervals`, etc.), along with the appropriate point set(s). For example, the function `rest-duration-time-intervals` takes a query element, the point set of notated rests, and the point set of instrument/staff and clef names as its arguments, because these three information sources are sufficient for locating rests of specific duration (perhaps also constrained to specified voices). In general, if the query is relevant to the Stravinsqi sub-function, the sub-function should output time intervals where events corresponding to that query occur. As an example, if the query element is "semiquaver F♯", then the function `duration&pitch-time-intervals` should output all time intervals where a semiquaver F♯ occurs, whereas for this same query the output of the function `harmonic-interval-of-a` should be empty.

Since there is not space here to describe all of the sub-functions called by Stravinsqi, the next subsection will address a straightforward sub-function `harmonic-interval-of-a`, and then more involved sub-functions for identifying functional harmonies, cadences, and textures.

If a query string is ordinary (contains one element only), then the time windows in the first nonempty sub-function's output are passed to a final function that converts the time windows into the XML format required by the task. Calls to the sub-functions are ordered by most to least specific (e.g., `duration&pitch-time-intervals` is more specific than `pitch-time-intervals`), so that if both an earlier output and a later output are nonempty, the output for the more specific rather than the less specific sub-function is returned.

If a query string is compound, then sub-function $f_m$ will have a set of output time windows $T_{m,n}$ for query element $n$. Stravinsqi will allow any plausible sequence of time windows $\tau_1 \in T_{m(1),1}$, $\tau_2 \in T_{m(2),2}, \ldots, \tau_N \in T_{m(N),N}$ to be passed on for further processing. Plausibility here means that two time windows $[a, b]$ and $[c, d]$ are adjacent ($b = c$) if the original query contained "followed by" or "then", or that two time windows are the appropriate distance apart if the original query contained, say, "followed two bars later by". A plausible sequence of time windows $\tau_1 = [a_1, b_1]$, $\tau_2, \ldots, \tau_N = [a_N, b_N]$ is merged into one time window $[a_1, b_N]$ and passed to the final function for conversion into the XML format required by the task.

## 2.2 Harmonic Intervals

To identify instances of a specified harmonic interval, the point-set representation of a piece is converted into minimal segments [12], delimited by times at which notes begin and end. The first few minimal segments for the piece shown in Figure 1 are

$S_0 = \{(-1, \text{G2}, 1, 3), (-1, \text{G3}, 1, 2), (-1, \text{D4}, 1, 1), (-1, \text{D4}, 1, 0)\}$,

$S_1 = \{(0, \text{G2}, 2, 3), (0, \text{B3}, 1, 2), (0, \text{D4}, 1, 1), (0, \text{G4}, 1, 0)\}$,

$S_2 = \{(0, \text{G2}, 2, 3), (1, \text{B3}, 1/2, 2), (1, \text{D4}, 1, 1), (1, \text{G4}, 1/2, 0)\}$,

$S_3 = \{(0, \text{G2}, 2, 3), (1, \text{D4}, 1, 1), (3/2, \text{C4}, 1/2, 2), (3/2, \text{A4}, 1/2, 0)\}$.

Importantly, a note can belong to more than one minimal segment. For example, the second D4 in Violin II b.1 belongs to both $S_2$ and $S_3$. The second step in identifying the relevant harmonic intervals is to convert the written interval into one or more *target pairs* of MIDI note and morphetic pitch intervals. For example, a "major second" would correspond to 2 MIDI note intervals (2 semitones) and 1 morphetic pitch interval (1 step on the staff), so the target pair would be (2, 1). A "second" would require two target pairs: (2, 1) for "major second", and (1, 1) for "minor second". For each minimal segment $S$, the maximal translatable pattern (MTP) [11] of the target pair(s) is calculated. The MTP is

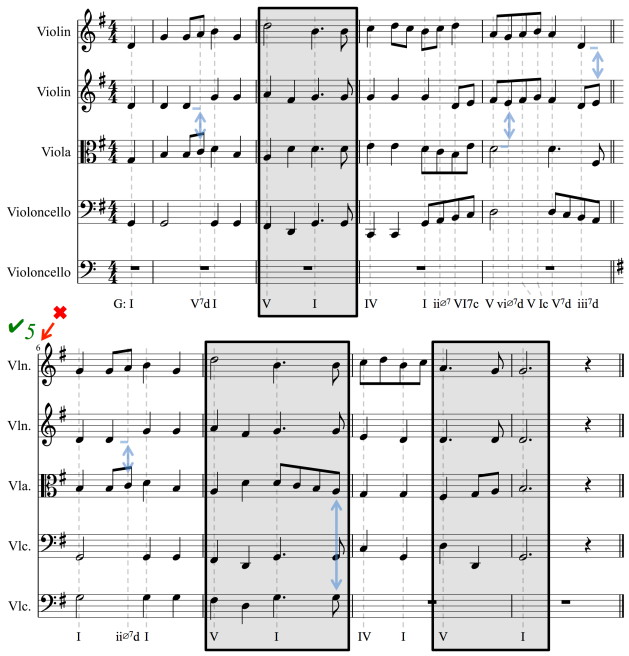



**Figure 1. Prelude to *Te Deum* H146 by Marc-Antoine Charpentier (1643-1704), annotated with a bar number error (tick and cross), intervals of a harmonic second (arrows), functional harmonies below each staff, and three perfect cadences (black boxes).**

the set of all points that translate by the target pair on to some other point in the minimal segment, which corresponds to the set of all notes that form a harmonic interval corresponding to the target pair. If a minimal segment *S* has a nonempty MTP for the target pair, then the time window where *S* sounds contains the specified interval. This approach to calculating harmonic intervals is thorough, in the sense that it considers notes within and between voices (as required by the task description), and notes that sound together at a specified interval but do not necessarily begin together. Almost all the harmonic seconds in Figure 1 (indicated by arrows) are of this variety.

## 2.3 Harmonic Analysis and Cadences

Example output of the Stravinsqi algorithm's sub-function for functional harmonic analysis is shown below each staff in Figure 1. The algorithm is over-labelling in bb.3-4, but otherwise the output is quite promising. Recent remarks in musicological circles suggest that an algorithm capable of sensible harmonic analyses does not yet exist and may be some way off [17, 18]. This could be lack of dialogue between musicological and music information retrieval communities, however. The HarmAn algorithm [12] is over a decade old, and while, admittedly, it provides chord labels (e.g., G maj, $D^7$) rather than functional harmonic labels (e.g., I, $V^7$), HarmAn is a useful starting point. Included in MCStylistic, the Stravinsqi algorithm's sub-function is called `HarmAn->Roman`, an extension of the HarmAn algorithm, and it does provide functional harmonic labels and inversions, as shown in Figure 1.

The HarmAn algorithm [12] begins by converting a point-set representation of a piece into minimal segments (cf. subsection 2.2). A segment is defined as a sequence of adjacent minimal segments, and so there are $n(n-1)/2$ possible segments for $n$ minimal segments. Each segment is compared with templates representing different chords (e.g., C maj, D♭ maj, D maj,…, $C^7$, D♭$^7$, $D^7$, etc.), and scored according to how similar it is to each template. The template with highest score is assigned as a provisional chord label. The final part of the HarmAn algorithm consists of a linear-time method for traversing these segment scores in temporal order and outputting a set of chord labels, without requiring an exhaustive search over all possible segment combinations [12]. My extension to HarmAn, called `HarmAn->Roman`, involves estimating the overall key of the input piece/excerpt, using this to convert chord labels to functional harmonic labels, and adding information about inversions.

An issue faced when designing chord labelling algorithms is the balance to be struck between under- and over-labelling progressions. An example of the `HarmAn->Roman` solution to this issue is shown in Figure 2. An under-labelled solution (prefaced by UL) places labels at each beat, whereas an over-labelled solution (preface OL) is busier for the last five beats of the passage in particular. The `HarmAn->Roman` output (prefaced by AO for algorithm output) is more similar to the under-labelled solution than it is to the over-labelled solution. A harmony instructor might prefer the under-labelled solution to the over-labelled solution in this instance, as the end of the latter seems overly fussy. In this sense, `HarmAn->Roman` does well. If the passage is queried for an occurrence of Ib, however, `HarmAn->Roman` does less well, because it has not split b.3 beat 4 into two labels (I and Ib), and so the occurrence of Ib is not found.

The Stravinsqi algorithm's sub-function for cadence identification is based in large part on its use of `HarmAn->Roman` for

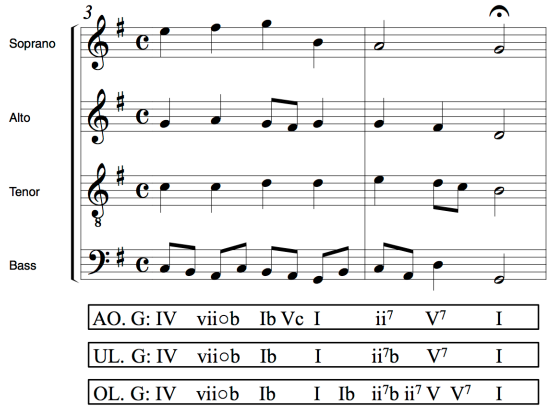



**Figure 2. Bars 3-4 of the chorale setting "Straf mich nicht in deinem Zorn" BWV115.6 (R38) by Johann Sebastian Bach (1685-1750), annotated with algorithm-output (AO) functional harmonies, and under-labelled (UL) and over-labelled (OL) expert annotations.**

functional harmonic analysis. Three perfect cadences identified by the cadential sub-function, `cadence-time-intervals`, are shown in Figure 1, indicated by black boxes with grey shading. The function `cadence-time-intervals` forms bigrams from a harmonic labelling (e.g., the labels I, $V^7$d, I, V, I, IV become (I, $V^7$d), ($V^7$d, I), (I, V), (V, I), (I, IV)). If a perfect cadence is to be sought, then indices of bigrams relevant to perfect cadences are returned. (E.g., occurrences of (V, I) or ($V^7$, I) or (V, i), etc. would be relevant.) If (*A*, *B*) is a relevant bigram and *B* has ontime *t*, then *t* must correspond to either beats one or three of the bar in a piece in common time, in order for (*A*, *B*) to be admitted as a perfect cadence. Since, for a perfect cadence, it is only "occasionally stipulated that the final chord must have the tonic in the highest part" [13], no further requirements are stipulated. Looking at the three perfect cadences identified in Figure 1, the last is of the variety with the tonic in the highest part, and the first two have the mediant in the highest part.

## 2.4 Textural Analysis

The Stravinsqi algorithm's sub-function for textural analysis outputs quadruplets: the first element in a quadruplet is the beginning of a time window where a texture label applies; the second is the end of that time window; the third is a texture label (monophonic, homophonic, melody with accompaniment, polyphonic, contrapuntal) or empty, and the fourth is a value in [0, 1] expressing the confidence with which the texture label was assigned. Thresholds for these labels can be altered so that, for instance, a piece consisting entirely of monophony apart from the final bar might be considered entirely monophonic if the monophony threshold is lenient, or monophonic up until the last bar and then homophonic if the monophony threshold is more strict.

The texture sub-function operates by identifying points (representing notes) that belong to windows $[0, s]$, $[h, h + s]$, $[2h, 2h + s]$, where the window size $s = 4$ crotchet beats and the hop size $h = 1$ crotchet beat, and sending these point collections for independent textural analysis. When a label has been assigned to each collection, adjacent windows that share labels are elided.

Independent textural analysis of a windowed excerpt begins with testing whether the label "monophonic" applies, by calculating whether $|X \cap Y|/|Y|$ is greater than or equal to the monophonic threshold, where $X$ is the set of (ontime, MIDI note)-pairs in the

excerpt's *skyline* and $Y$ is the set of (ontime, MIDI note)-pairs in the excerpt.[4] If the skyline set is equal to (or very nearly equal to) the whole set of notes in an excerpt, then likely the windowed excerpt is monophonic. If the excerpt is not monophonic, textural analysis continues with testing whether the label "homophonic" applies, by calculating whether $|X \cap Y|/\max\{|X|, |Y|\}$ is greater than or equal to the homophonic threshold, where $X$ is the set of ontimes in the excerpt's skyline and $Y$ is the set of ontimes for notes that are in the excerpt but not in the skyline (the *accompaniment*). If the skyline's ontimes and the accompaniment's ontimes are very similar, then likely the excerpt is homophonic. If the excerpt is not homophonic, textural analysis continues with testing whether the label "melody with accompaniment" applies, by calculating two quantities: first, $r_1 = |X \cap Y|/\max\{|X|, |Y|\}$, where $X$ is the skyline calculation applied twice to the excerpt, and $Y$ is what remains; second, $r_2 = \rho_t/(3\rho_b)$, where $\rho_t$ is the *rhythmic density* (mean number of notes per crotchet beat, [6]) in the top staff and $\rho_b$ is the rhythmic density of the staff immediately below the top staff. If either $r_1$ or $r_2$ are greater than or equal to the melody-with-accompaniment threshold, then this label applies. The quantity $r_1$ is perhaps not easily interpreted, but it is the same idea for testing homophony, applied to just the accompaniment. The quantity $r_2$ tests whether the skyline is three or more times as active (in terms of notes per crotchet beat) than the accompaniment. If so then the excerpt's texture may be considered "melody with accompaniment". Finally, independent textural analysis tests whether the label "polyphonic" (vocal music) or "contrapuntal" (instrumental music) applies to an excerpt, by extracting the notes from each staff and testing whether these note collections would each be considered monophonic. If so then the excerpt is labeled polyphonic or contrapuntal. If not, then the excerpt does not have a texture label assigned.

Example output of Stravinsqi's textural analysis sub-function is shown in Figure 3. The piece begins with a sixteen-bar section that bisects into eight bars ending on an imperfect cadence, followed by eight similar bars ending on a perfect cadence. The excerpt begins with right-hand material that is more active in terms of notes per crotchet beat than the left hand. The label "melody with accompaniment" shown in Figure 3 for bb.1-8 seems appropriate, therefore, as does the maximal level of confidence in the label (= 1). The left hand becomes more active and independent of the right hand in bb.9-16, and so the change in label to "contrapuntal" seems appropriate also. The simpler left-hand material returns in b.17, as does the "melody with accompaniment" label.

## 3. RESULTS AND DISCUSSION

Figure 4 shows recall and precision results for the Stravinsqi algorithm on the 2014 C@merata task. The *measure* metrics reward an algorithm's output if it is in the same bar/measure as a ground-truth item, whereas the *beat* metrics require an algorithm's output to be in the same bar *and on the same beat* as a ground-truth item. The mean category in Figure 4 shows the overall results, with Stravinsqi having recall .91 and precision .46 at the measure level, and recall .87 and precision .44 at the beat level.[5]

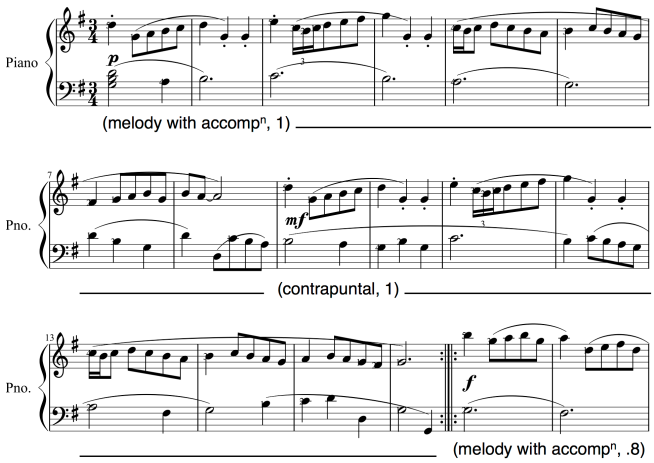



**Figure 3. Bars 1-18 of Minuet in G major BWV ANH 114 by Christian Petzold (1677-1733), annotated below each staff with texture labels and confidence ratings in [0, 1].**

Stravinsqi's strong performance on the first eight of twelve categories (pitch, duration,…, melodic interval) is encouraging, as is the small decrease in recall (.91 to .87) and precision (.46 to .44) with the change from measure- to beat-level granularity. The remainder of this section discusses the weaker aspects of the results, beginning with the drop in precision for compound queries. Poor precision here is due to the criteria used to select and combine time intervals for the different elements that comprise a compound query. As an example, suppose the compound query is "quaver followed by semiquaver", that $T_1$ is a set of time intervals for "quaver" and $T_2$ is a set of time intervals for "semiquaver". For subsequent processing, ideally the algorithm will select pairs $(\tau_1, \tau_2) \in T_1 \times T_2$ such that $\tau_1$ and $\tau_2$ are adjacent time intervals. Presently, Stravinsqi selects these pairs, as well as $(\tau_2, \tau_1) \in T_2 \times T_1$ such that $\tau_2$ and $\tau_1$ are adjacent time intervals. So the criteria for time interval selection were too lenient, but this is something that can be fixed in future work.

In general, the C@merata task description was very clear, and the training/test data covered sensible question categories in a musically astute manner [16]. The following issues are small in comparison to the overall solid work, but they are raised because they may have had an effect on algorithm performance, and could be usefully addressed for future iterations of the task.

The triad and texture categories are somewhat underrepresented in the training and test data: neither the task description nor training data contains examples of texture queries [16]. The task description contains two examples of functional-harmonic queries, but there are none in the training data. Furthermore, there

[4] In most cases, the skyline is the set of points defined by the highest-sounding note in each minimal segment of an excerpt.

[5] Stravinsqi is labelled DMUN03 in the overview paper for the task [16]. The other submitted runs DMUN01 and DMUN02 are not remarkable: there were incorrect bar numbers in four pieces (see, for instance, the cross and correction in Figure 1), and this caused issues because the conversion function `xml2hum` overlooked incorrect bar numbers and began bar numbering from 1 for the first complete bar. DMUN02 adjusted for these errors compared with DMUN01, but still there was a further piece for which `xml2hum` would not work. The piece was re-encoded, and DMUN03 represents the submission that ran across all twenty pieces, adjusting for erroneous bar numbering also. Due to concerns about the bar numbering, the test data was opened in order to check for any knock-on effects. No changes were made to the Stravinsqi algorithm post-checking.

are only five texture questions and five functional-harmonic questions in 200 test data. It is possible to make some inferences about Stravinsqi's performance, however. For triad labelling, Stravinsqi suffered from the under/over-labelling issue (cf. subsection 2.3). In particular, it missed two first-inversion triads because the same triad in root position preceded them. The two events—root-position followed by first-inversion triad—were described by only one root-position label, and so the occurrence of the first-inversion triad was missed. Diagnosing errors in the texture queries is more difficult. The texture labels in Figure 3 seem reasonable, but these scored zero on all metrics for Question 10, "melody with accompaniment". Similarly poor results were returned for Question 130, "monophony", for Piece 13, the Prelude of Suite no.1 for Cello BWV1007 by J.S. Bach. The piece consists of 41 bars of monophony followed by a final chord in bar 42 (although this is often spread in performance). Stravinsqi returned a monophonic label for the entire piece. Depending on the value of the monophonic threshold, Stravinsqi will tolerate the occasional non-monophonic event in an otherwise monophonic texture. This monophonic label scored zero on all metrics. It is difficult to identify why this happened without knowing the details of the ground-truth texture annotation.

There were some inconsistencies between the task description/training collection, and test collection. On pp.5-6 of the task description, two examples suggest that a query containing an interval (e.g., "fourth") ought to be treated by default as a harmonic interval (simultaneously sounding notes), and only as a melodic interval (consecutive notes) if the word "melodic" is present. The score for Piece 20, "All praise to Thee my God" by Thomas Tallis (1505-1585) is entirely monophonic; it cannot contain simultaneously sounding notes and, by extension, cannot contain harmonic intervals. The query for one question about the Tallis (Question 200, "fourth"), a query about a harmonic interval by default, should result in an empty answer. Stravinsqi's output was empty for this question, but it received zero on all metrics.

On p.8 of the task description there was an example question about melodic intervals, reproduced in Figure 5 with relevant notes highlighted. It is evident from Figure 5 that melodic intervals can occur between consecutive chord notes on the same staff (see the first answer in b.19; B♭4, E5). Question 109, for Piece 11, the Largo cantabile from Concerto in G major op.7 no.2 RV299 by Vivaldi, requires that all instances of a "melodic rising fifth" be returned. Two of the nine answers are shown in Figure 6. As with Figure 5 there is an answer between two melody notes (b.1) and an answer between consecutive chord notes (bb.3-4). Of the remaining seven answers (not shown in the figure), one is of the first type (between melody notes) and six are of the second type (between consecutive chord notes). The Stravinsqi algorithm scored 1 on recall metrics for this question, but only 2/9 on precision, from which one infers the human, ground-truth annotation did not include melodic intervals between consecutive chord notes on the same staff. This contradicts the examples given in the task description.

For harmonic interval questions, it was surprising to see recall and precision of Stravinsqi at less than one. Regards recall, there were two rather wide intervals requested for which Stravinsqi was not prepared: "nineteenth" (Question 40), "major seventeenth" (Question 99). Question 80, "harmonic fifth", may not have been interpreted correctly during ground-truth annotation: the divisions value of 1 specified in the question suggests that crotchet-level granularity is sufficient for representing all fifths occurring in Piece 8, Sonata in F minor K466 by D. Scarlatti. Looking at the

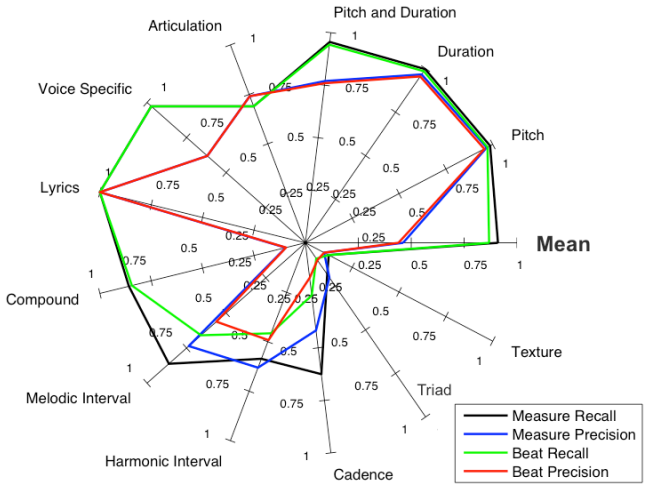
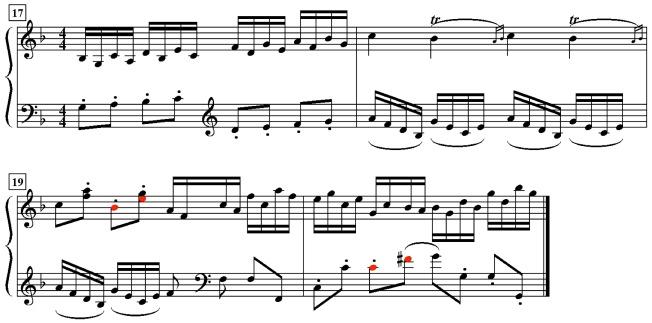



**Figure 4. Results of the Stravinsqi-Jun2014 algorithm on the MediaEval 2014 C@merata task. Overall results are indicated by the mean label, and followed by results for twelve question categories.**



**Q:** *augmented melodic fourth*
**A:** [4/4,2,19:3-19:4], [4/4,2,20:4-20:5]

**Figure 5. Bars 17-20 of Sonata in D minor K1 (L366) by Domenico Scarlatti (1685-1757), two augmented melodic fourths highlighted, and C@merata syntax below.**

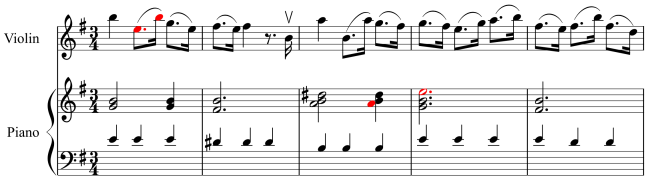



**Figure 6. Bars 1-5 of Sonata in E minor RV299 by Antonio Vivaldi (1678-1741), with two melodic perfect fifths highlighted.**

end of b.19, for instance, there is a fifth between quavers in the bass clef F4 and the treble clef C5, suggesting a divisions value of at least 2 is necessary for representing all fifths in the piece.

The only conceivable reason for less-than-perfect precision points to another potential issue with the ground-truth annotations. Question 18, for Piece 2 shown in Figure 1, requires that all instances of a "harmonic second" be returned. Arrows in Figure 1

indicate Stravinsqi's output, which scored 1/5 for precision. One assumes credit was assigned for the second between cello II (G3) and viola (A3) in b.6. The other four intervals identified in Figure 1 are instances of a note beginning in one voice and still sounding when another note begins and forms a harmonic second. Generally, it seems that the human, ground-truth annotation did not allow for harmonic intervals involving notes that sound together, but do not necessarily begin together. This is the main reason for Stravinsqi's drop in precision for harmonic interval questions, but is not due to an algorithmic fault.

# 4. CONCLUSION

While musicologists may not find all aspects of the C@merata task to be innovative and exciting (e.g., algorithms for finding melodic intervals and sequences of melodic intervals have existed for some time [1, 19]), the natural-language aspect of the task opens up new, interesting potential applications. That is, there are relatively few students/researchers willing to learn how to express a music query such as "perfect fifth" in the numeric format required by a pre-existing function in some programming environment, and fewer still who would be prepared to write a function that does not yet exist for some musicological query. There are many students/researchers, however, who would find it convenient to explore pieces of their choice with musicological string queries. Thus, algorithms that perform strongly on the C@merata task can have a significant impact in music education, and as a springboard for musicological research.

The Stravinsqi algorithm described above is one such strong performer, and has effectively solved seven of the twelve C@merata task categories shown in Figure 4 (pitch, duration, pitch and duration, articulation, voice specific, lyrics, and melodic interval). As for the remaining five categories, precision for one category (compound queries) can be improved by fixing a selection-criteria bug. The less-than-perfect performance on harmonic intervals can be addressed by resolving inconsistencies in the task description and test data. The labelling of functional harmonies is another area where Stravinsqi can be improved, and this will require more training and test data. More data are also required for the cadence and texture query categories. In future iterations of the task, it may be helpful to have at least two experts provide annotations for these higher-level music-theoretic concepts, to check that there are consistent answers towards which algorithm developers can aim. The addition of new, higher-level music-theoretic query categories (e.g., [3]) would be welcome in future iterations of C@merata as well, and will help to keep the task at the forefront of research in music computing.

# 5. REFERENCES


[1] Aarden, B. J. 2003. *Dynamic Melodic Expectancy*. Doctoral Thesis. School of Music, Ohio State University.

[2] Barlow, H., and Morgenstern S. 1948. *A Dictionary of Musical Themes*. Crown Publishers, New York, NY.

[3] Caplin, W. E. 2013. *Analyzing Classical Form: An Approach for the Classroom*. Oxford University Press, New York, NY.

[4] Collins, T. 2011. *Improved Methods for Pattern Discovery in Music, with Applications in Automated Stylistic Composition*. Doctoral Thesis. Faculty of Mathematics, Computing and Technology, The Open University.

[5] 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.

[6] Eerola, T., and North, A. C. 2000. Expectancy-based model of melodic complexity. In *Proceedings of the International Conference on Music Perception and Cognition* (Keele, UK, August 05 - 10, 2000). 7 pages.

[7] M. Grachten, and Krebs, F. 2014. An assessment of learned score features for modeling expressive dynamics in music. *IEEE T. Multimedia*. 16, 5, 1-8.

[8] Huron, D. 2002. Music information processing using the Humdrum toolkit: concepts, examples, and lessons. *Comput. Music J.* 26, 2, 11-26.

[9] Kornstädt, A. 1998. Themefinder: a web-based melodic search tool. In *Computing in Musicology*, vol.11, W. B. Hewlett and E. Selfridge-Field, Eds. Center for Computer Assisted Research in the Humanities, Stanford, CA, 231-236.

[10] Krumhansl, C. L. 1990. *Cognitive Foundations of Musical Pitch*. Oxford University Press, New York, NY.

[11] Meredith, D., Lemström, K., and Wiggins, G. A. 2002. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *J. New Music Res.* 31, 4, 321-345.

[12] Pardo, B., and Birmingham, W. P. 2002. Algorithms for chordal analysis. *Comput. Music J.* 26, 2, 27-49.

[13] Rockstro, W. S., Dyson, G., Drabkin, W., Powers, H. S., and Rushton, J. 2014. Cadence. In *Oxford Music Online*. Retrieved January 6, 2014 from www.oxfordmusiconline.com.

[14] Sapp, C. S. 2005. Visual hierarchical key analysis. *ACM Computers in Entertainment.* 3, 4, 1-19.

[15] Sapp, C. S. 2013. *Humdum Extras*. Retrieved March 3, 2014 from http://wiki.ccarh.org/wiki/Humdrum_Extras.

[16] Sutcliffe, R., Crawford, T., Fox, C., Root, D. L., and Hovy, E. 2014. Shared evaluation of natural language queries against classical music scores: a full description of the C@merata 2014 task. *Proceedings of the C@merata Task at MediaEval 2014*.

[17] Tymoczko, D. 2013. Review of Michael Cuthbert, music21: a toolkit for computer-aided musicology (http://web.mit.edu/music21/). *Music Theory Online*. 19, 3, 18 numbered paragraphs. Retrieved March 9, 2014 from http://mtosmt.org/issues/mto.13.19.3/mto.13.19.3.tymoczko.php.

[18] Tymoczko, D. 2014. Contribution to *Harmonic Analysis* discussion on the music21 list. May 3, 2014.

[19] von Hippel, P., and Huron, D. 2000. Why do skips precede reversals? The effect of tessitura on melodic structure. *Music Percept.* 18, 1, 59-85.