Master's Thesis

Data structure for multi-layered digital score

Shuichi Shimosaka

Supervisor Satoshi Tojo

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

December, 2022

Abstract

Content-Based Musical Retrieval (CBMR) intends to figure out the methods for performing search and retrieving information efficiently from digitized music scores. Since the current objective of CBMR is primarily to get similar musical patterns using monophonic or polyphonic queries, no method has been proposed to extract similar occurrences using the structural information on music such as a key or a harmonic progression. (e.g. find the measures from digitized scores that have a II-V-I harmonic progression)

In this thesis, multi-layered data structure containing scores, key and harmonic information will be proposed along with algorithms that realize the search. The proposed data structure and the algorithms allow searching through a progression of key or harmony, which satisfies the needs to explore music that has a similar musical structure. In particular, people will be able to view the music score as a search result of a query of key or harmony by using the data structure which contains information about the opus number, the measure number, the key and the harmony.

While current CBMR research only provides the functionality for searching for the surface of music by executing the exact matching or the fuzzy matching for given music notes, the proposed method aims to capture the needs to search for music not by music notes but by musical structure, namely "Are there examples where Bach used a V-IV harmonic progression, which should be rare? If yes, how many cases do we see in his works and how did he address this progression in his compositions?" or, "Can we see all the past examples of a II-V-I progression created by great composers?", which have been common questions amongst musicologists, composers and professional musicians.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Musicologists, composers and professional musicians have had the desire to find similar musical occurrences that have similar musical structure. For example, they may ask "Are there examples where Bach used a V-IV harmonic progression, which should be rare? If yes, how many cases do we see in his works and how did he address this progression in his compositions?" or, "Can we see all the past examples of a II-V-I progression created by great composers?" The reason why they respect the examples of composers of the past is that the compositional technique of Western music consists intrinsically of the accumulation of the previous realization of great composers. This means that a thorough understanding of past compositions should be the prerequisite for their musical analyses or compositions. Despite the fact that these information-seeking needs are primitive and largely rule-based, these tasks are essentially carried out manually, even now. Therefore, information technology can offer a more efficient way of streamlining the process.

The existing research has focused on finding the methods to perform the search for digitized scores, as in MEI or MusicXML, queried by music notes. It can be said that the existing methods take only care of searching for music notes, which are the surface of the music and are merely one of the many components of the music. Western music in particular has a layered structure on a background, such as key or harmony. To the best of the author's knowledge, there have been no proposed methods, data structure or tools that enable search by using music's structural information to fulfill the needs such as "finding measures that have a V-IV harmonic progression" or "listing all the composition examples for the II-V-I harmonic progression".

In this thesis, multi-layered data structure containing scores, key and harmonic information will be proposed along with algorithms that realize the search. The proposed data structure and the algorithms allow searching through a progression of key or harmony, which satisfies the needs to explore

music that has a similar musical structure. In particular, people will be able to view the music score as a search result of a query of key or harmony by using the data structure which contains information about the opus number, the measure number, the key and the harmony. While current CBMR research only provides the functionality for searching for the surface of music by executing the exact matching or the fuzzy matching for given music notes, the proposed method aims to capture the needs to search for music not by music notes but by musical structure. (Figure 1.1)

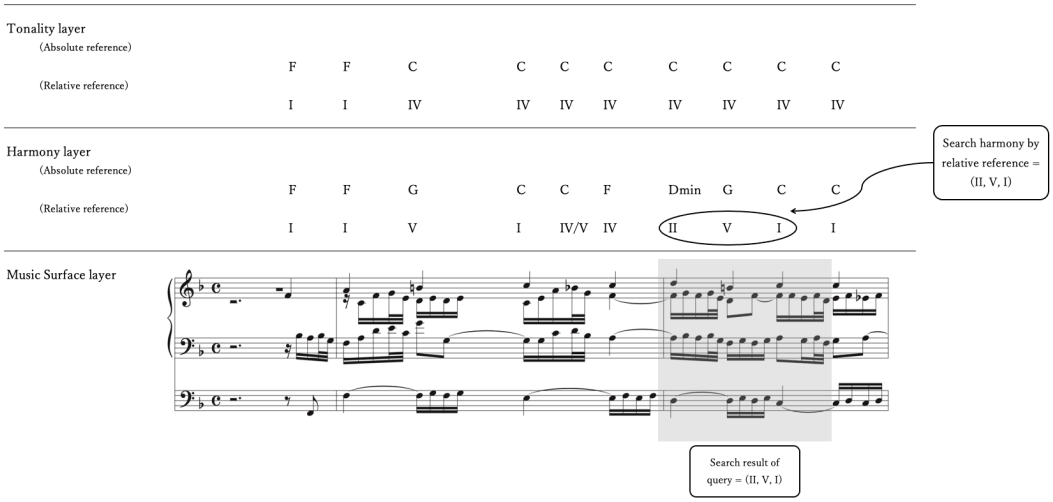

Figure 1.1: Concept of multi-layered score and searching for the upper layer of music

This thesis employs annotated data of J.S. Bach's Little Organ Book, which contains a Roman numeral analysis produced by Japanese professional organists as part of the KAKEN project "Computation model for understanding music based on statistical grammar model and constructive semantics". Since there have been few annotation efforts done by professional musicians which analyzed great composer's work and interpreted its structure, the database can be regarded as a valuable information resource of digital musicology, and therefore the database was adopted in this thesis for the experimental purposes. In the meantime, the method proposed in this thesis can be applied to other annotated musical corpora.

Chapter 2

Related Works

2.1 Research area of MIR

Music Information Retrieval (MIR) refers to an area of research that seeks to extract information from the digitized music data resource. According to the definition by Downie [1], "Music Information Retrieval (MIR) is a multidisciplinary research endeavor that strives to develop innovative content-based searching schemes, novel interfaces, and evolving networked delivery mechanisms in an effort to make the world's vast store of music accessible to all.", which attempts to provide effective measures to allow users to access the enormous digital music resources available.

There are mainly two major areas of focus that MIR considers: one is the audio information retrieval and another is the symbolic information retrieval (Velard et al. [2]). The audio information represents digital audio information such as in WAV or MP3 format and the symbolic information represents digitized music score such as in MIDI, MusicXML or MEI format. The audio information retrieval basically applies the signal processing technique, while the symbolic information retrieval has an affinity for the Natural Language Processing (NLP). Since both approaches have advantages and disadvantages, a suitable method should be selected each time in accordance with the purpose of the data extraction. For example, the audio information retrieval technique should be used when the user wants to search for music from his or her humming, whilst the symbolic information retrieval technique corresponds to the need to search for scores from music notes populated by the user.

Among the methods that use the symbolic information, Content-Based Musical Retrieval (CBMR) aims in particular to research how to conduct a search for digitized music scores to acquire the information that the user

may want. Garfinkle et al. [3] defines that "Content-Based Music Retrieval (CBMR) for symbolic music aims to find all similar occurrences of a musical pattern within a larger database of symbolic music.". To put it simply, CBMR aims to obtain relevant music scores when a query such as music notes or a melody is given. CBMR attempts to accommodate various social demands such as finding a song title from a particular tune, studying examples of past composition of well-known themes or examining the possibility of plagiarism. As discussed in more detail in the next section, the search procedure should have the flexibility to cover the wide range of requirements. For example, it should meet multiple needs such as allowing exact matching and fuzzy matching, allowing not only monophonic, but also polyphonic entries, and so on.

2.2 Issues in searching for music scores

As our modern life is inseparably connected to the Internet, it is indisputable that search technology, which aims to efficiently gather the required data from a huge amount of information on the Internet, occupies an essential position in information technology. Needless to say, search engines such as Google have been constantly evolving in recent decades. It is important to note that when people talk about search technology, it usually refers to a search by natural language, with some exceptions such as searching for images. Therefore, when we say "search for music scores", it frequently means searching for scores using the meta-data such as the name of the composer, the title of the song, the genre and so on. Meanwhile, there has been a huge demand to search for scores by musical elements such as music notes or a melody, because such functionality can greatly assist musicians, composers and musicologists to collect examples of past music. However, unlike natural language querying, there are two major hurdles to performing a search by musical elements. The first issue is an absolute data shortage. Although there are abundant music scores appearing on the Internet, many of them are stored as image files, which provide limited information to the computer. Therefore, even at present, there are few data storages that provide machine-readable music scores such as in MEI or MusicXML format. The second issue is the difficulty of seeking music symbolically, which is a key focus of CBMR. For example, even when people simply want to find a tune that moves as "C-D-E-F-G-A-B", there are multiple factors to take into consideration such as "Which pitch of C-D-E-F-G-A-B?", "Is a consecutive move only allowed (e.g. C4-D4-E4-F4-G4-A4-B4) or is a jump permitted (e.g. C4-D3-E5-F2-G5-A2-B4)?", "What is the duration of each note?", "Are we searching for a

melody appearing on outer voices or do we want to look for inner voices as well?", "Is there a designated instrument or timbre?". The symbolic music search system should take these considerations into account so that it can provide appropriate results consistent with the user's intent. Lemström [4] established definitions of these diverse music search requirements.

2.3 Basic theory of CBMR

The purpose of CBMR is not only to retrieve music scores that perfectly match a given query for music notes, but to retrieve all similar patterns from a query by allowing some sort of fluctuations. A variety of methods have been proposed to cope with the fuzziness of this type of data extraction. Velardo et al. [2] summarized the existing approaches by 1. Category (Mathematics/Music theory/Hybrid), 2. Type of melody (Monophony/Polyphony), 3. Genre (General/Folk songs), 4. Similarity function, 5. Musical Parameters (Pitch/Duration/Harmony etc.), 6. Musical Representation (String of symbols/Graph etc.). Most notably, the approach proposed by Orio and Rodà [5] can be regarded as a major study of CBMR that took advantage of music theory. Inspired by Schenkerian analysis [6] and Generative Theory of Tonal Music (GTTM) [7], Orio proposed to create a graph from music scores that simplifies melodies. The graph has multiple layers according to the reduction level, which is intended to be used as a basis for the fuzzy search.

2.4 Implementation of CBMR

Apart from the basic theory of CBMR, a prominent example of the project which aspires to implement digital music data storage with search and analysis capabilities is the SIMSSA (Single Interface for Music Score Searching and Analysis) project initiated by Fujinaga et al. [8]. By forming divisional cooperative subgroups called "Axis", the SIMSSA project streamlines a set of workflows of 1. image recognition of scores, 2. storing scores in XML format and 3. making searchable information available to the public. A subgroup called "Content Axis" pursues the sophistication of OMR (Optical Music Recognition) by utilizing the machine learning and the image recognition technology so that they can store the scores in machine-readable format. Meanwhile, a subgroup called "Analysis Axis" seeks to construct efficient methods to search for stored music scores made by the Content Axis subgroup. Furthermore, they also aim to establish a user interface to share analytic information such as statistics of each musical composition. The

SIMSSA project offers one reasonable response to resolve the complexity of data storage of music scores and search methods, in a way that it is an end-to-end approach ranging from performing XML encoding of music scores to producing a searchable interface of the music data storage. Hopkins et al. [9] explicates the data structure used by the SIMSSA project.

The most well-known tools developed by the Analysis Axis subgroup as part of the SIMSSA project are PatternFinder [3] by Garfinkle and jSymbolic [10] by McCay. PatternFinder is a score search platform, which is built utilizing music21 [11]. The tool realized various types of music score search such as exact matching and fuzzy matching of monophonic/polyphonic queries given by the user. PatternFinder is unique in that it offers a distributable package which implemented CBMR. On the other hand, jSymbolic is a statistical analysis platform for music scores. By loading MEI or MIDI files into the tool, jSymbolic extracts feature values such as the pitch class histogram, the average duration of notes, the range of the piece and so on. The tool is intended to be used for musicological analysis, such as music classification, more commonly known as the estimation of a composer from an anonymous piece.

The NEUMA [12] project is another example of implementing CBMR other than the SIMSSA project. Like as the SIMSSA project, the goal of the NEUMA project is to establish a digital music library by taking advantage of OMR technology. The NEUMA website (http://neuma.irpmf-cnrs.fr) offers a simple music score searching function that allows the user to search from monophonic music notes provided through the simple keyboard layout displayed on the screen.

Apart from the comprehensive approaches described above, several standalone score search tools exist. For example, PEACHNOTE developed by Viro [13] provides a built-in melody search function in IMSLP, whereas Ask Toskanini! developed by Bahraini and Tilevich [14] specializes in searching for scores from the highest/lowest pitches, the melody, the tempo and so on.

2.5 Topics CBMR has not covered

The tools mentioned in the previous section only provide the functionality to search for music scores by using music notes as a query. Western music consists not only of music notes, but also of a background frame, such as key (e.g. progression from C-Major to a-minor) or harmony (e.g. progression from G-Major's V to I). Although there have been needs amongst musicians, composers and musicologists to search for music scores using the background information of music (e.g. "find the location that modulates from C-Major to

a-minor", "find the location that has the chord progression from G-Major's V to I"), existing CBMR approaches have not undertaken to cover such a demand. There is little existing research that has focused on extracting information using the upper layer structure of music. Although it is not directly related to the score search, Haas et al. [15] attempted to compute the similarity of harmonic structure between two pieces of music by comparing the trees generated from a proposed algorithm.

Chapter 3

Database used in the thesis

This thesis utilizes the annotated database of J.S. Bach's Little Organ Book, which contains a Roman numeral analysis created in partnership with Japanese professional organists as part of the KAKEN project 16H01744: "Computation model for understanding music based on statistical grammar model and constructive semantics". Although several annotated data sets are available that include a Roman numeral analysis performed by professional musicians, such as Neuwirth et al. [16]'s annotated data for Beethoven's pieces, the publicly accessible data corpus for the Little Organ Book was never produced.

This chapter describes the abstract of this database. In the meantime, the background and details of the data is accessible to everyone on the KAKEN project site, https://www.jaist.ac.jp/is/labs/tojo-lab/kiban-A/.

The targeted pieces are the total 45 works of J.S. Bach's Little Organ Book, BWV599-BWV644 (except BWV634, which is deemed to be the earlier version of BWV633). As described in the KAKEN project website, existing research for music information retrieval focuses primarily on simple structured music, such as Bach's 4-voice chorales. Since the Little Organ Book is regarded as a direct expansion of his 4-voice chorales in terms of composition technique, the annotation is made to these works so that the computer can process various musical expressions and the diversity of ornaments in the musical works.

The first step in compiling data was the creation of MusicXML data for each piece. The MusicXML data was created from actual scores by computer processing with human inspection. The scores were from Montréal: Les Éditions Outremontaises (2008) edited by Pierre Gouin, obtained from International Music Score Library Project (IMSLP) / Petrucci Music Library. (Licence: Creative Commons Attribution Non-commercial 3.0)

The next step was the addition of a Roman numeral analysis to the score by professional musicians. Although computer-generated harmonic analysis

is an area that has long been desired, as indicated by Gotham et al. [17] and Ju et al. [18], human intervention is at present unavoidable because there are many ambiguities in interpreting harmony.

With regards to the method, when modern people refer to harmonic analysis, it tends to be based upon the theory of functional harmony, advocated by Hugo Riemann (1849-1919), which rooted in Jean-Philippe Rameau (1863-1764)'s music theory. However, it should be noted that it is not possible to interpret perfectly the harmony of the Little Organ Book according to the theory of functional harmony. This is for the most part because Bach's music was created before this theory was established. Moreover, the Little Organ Book includes especially works that were written in church mode, which is outside the tonality. Meanwhile, the analysis was intended to answer the question such as "how do modern people, who are accustomed to tonal music, understand the harmony and tonality of the Little Organ Book, and how does modern music theory interpret the pieces?". In other words, the analysis provides one interpretation for the structure of Bach's works in terms of tonality and harmony by venturing to use the expressions of the theory of functional harmony. The goal of the analysis is to discover how modern people deal with sound and perceive tonality.

The annotated scores are available in PDF and MusicXML on the website. Figure 3.1 describes a PDF example of the BWV609 annotation. The first and second measures are analyzed as follows:

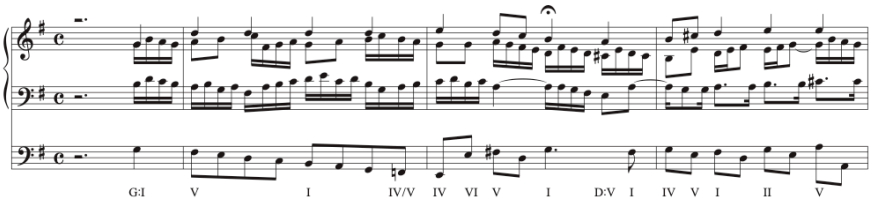

Figure 3.1: Example of the BWV609 annotation

1. The pickup measure: G-Major's I
2. The 1st-2nd beat of measure 1: G-Major's V
3. The 3rd-4th beat of measure 1: G-Major's I
4. The upbeat of the 4th beat of measure 1: G-Major's IV/V (identical to C-Major's V)
5. The 1st beat of measure 2: G-Major's IV and VI

6. The 2nd beat of measure 2: : G-Major's V

7. The 3rd beat of measure 2: : G-Major's I

8. The 4th beat of measure 2: : D-Major's V and I

As noted as a cautionary note on the site, the analysis does not identify inversion of chords and does not strictly differentiate a borrowed chord with modulation. This is because the primary purpose of the analysis is to determine the most appropriate interpretation of the degree of key and harmony perceived by modern people. The example in Figure 3.1 interprets the upbeat of the 4th beat of measure 1 as a borrowed chord (IV/V) while the 4th beat of measure 2 is interpreted as modulation.

With regards to MusicXML, the annotation is incorporated into the <lyric> attribution, which enables computational analysis. For example, the upbeat of the 4th beat of measure 1 is shown in Figure 3.2.

```
▼<note default-x="379">
 ▼<pitch>
    <step>A</step>
    <octave>3</octave>
  </pitch>
  <duration>1</duration>
  <voice>3</voice>
  <type>16th</type>
  <stem default-y="-40">down</stem>
  <staff>2</staff>
  <beam number="1">continue</beam>
  <beam number="2">continue</beam>
 ▼<lyric default-y="-193" number="1">
    <syllabic>single</syllabic>
    <text>IV/V</text>
  </lyric>
</note>
```

Figure 3.2: MusicXML expression for the upbeat of the 4th beat of measure 1 of BWV609

Before using the data, the following points need particular attention as some of the expressions presented in the analysis differ slightly from the common harmonic analysis style.

- The German expression is adopted. Thus, “B’ requires special care because B/b in the German expression represents B♭ Major/Minor in the English expression. Upper case characters represent major keys while lower case characters represent minor keys.

- Roman numerals are all represented with upper case characters. (e.g. III, IV, V) As we can identify major or minor with the key and the degree of the chord, it is considered unnecessary to use upper case characters for major chords and lower case characters for minor chords.

- “+” symbol is added for major I or IV chords that may occur in a minor key. In addition, “-” symbol is added for minor I chords which occasionally takes place in a major key.

- Distinction of V and VII: VII which is part of the sequence (e.g. the descending fifth sequence) is treated as VII, while the other dominant chords are treated as V.

- The numbers in brackets represent harmony that have little or some ambiguous function. () is displayed for uninterpretable harmonies.

- The analysis does not dispute where exactly modulation happens. The objective of the analysis is to indicate what degree of which key is the most reasonable interpretation of harmony. It does not discuss each given location should be dealt with as a borrowed chord or modulation.

Finally, the analysis does not argue that the provided interpretation is an only answer. Rather, as syntax analysis of human language often offers several possibilities, there should be several interpretations for a given harmony.

This database will be referred to as *Little Organ Book analysis database* in subsequent chapters. In the *Little Organ Book analysis database*, each work has one file for its MusicXML file, with the
“BWV[Bach’s opus number]_Orig_wR.musicxml” naming convention.

Chapter 4

Problem settings and the proposed data structure

4.1 Problem settings

This section defines the upper layer search requirements for the musical piece. As illustrated below, H1 and H2 are defined for harmony search while T1 and T2 are defined for key search.

1. H1: Display the location(s) which correspond(s) to the harmonic sequences given in the absolute chord notation. (e.g. from Cmaj to Emin)

2. H2: Display the location(s) which correspond(s) to the harmonic sequences given in the relative chord notation. (e.g. from I to V)

3. T1: Display the location(s) which correspond(s) to the sequences given in the absolute key notation. (e.g. from C-Major to a-minor)

4. T2: Display the location(s) which correspond(s) to the sequences given in the relative key notation. (e.g. from Ist key to Vth key)

Table 4.1 shows the summary of the H1, H2, T1 and T2 search.

Table 4.1: Summary of the search requirements

Targeted information	Search type	
	Absolute	Relative
Harmony	H1	H2
Key	T1	T2

4.2 Proposed data structure

This section proposes a data structure that satisfies the search requirements H1, H2, T1 and T2, as shown in Table 4.2. This table contains information about the name of key and the name of harmony in the absolute and the relative reference for the place identified by the opus number, the measure number and the beat number.

Table 4.2: Proposed data structure

Column name (Physical name)	Column name (Logical name)	Format	Description
WorkNum	Work number	Numeral	Opus number of the piece
MeasNum	Measure number	Numeral	Measure number in the piece
BeatNum	Beat	Numeral	Beat number of the measure
TonAbs	Key name Absolute	String	Key name in the absolute reference in the German expression
TonRel	Key name Relative	String	Key name in the relative reference in Roman numerals
HarmAbs	Harmony name Absolute	String	Harmony name in the absolute reference in chord names
HarmRel	Harmony name Relative	String	Harmony name in the relative reference in Roman numerals

4.3 Creation of the corpus table

Based on the information provided by the *Little Organ Book analysis database*, the author has created a table in CSV format that complies with the data structure proposed in the previous section. The author made the following changes in order to meet the search requirements.

- The key name in the relative reference was created by the author because the information was not included in the *Little Organ Book analysis database*. There is no room for interpretation to obtain the relative reference name, as it is identifiable from the key name in the absolute reference, given the fact that the first key of the piece is I.
- The harmony name in the absolute reference was created by the author, as the information is not included in the *Little Organ Book analysis database*. There is no room for interpretation to obtain the absolute reference name, as it is identifiable from the combination of the key name in the absolute reference and the harmony name in the relative reference, which are included in the *Little Organ Book analysis database*.
- Borrowed chord expressions such as V/V (the Vth degree of harmony in the Vth key in the piece) are treated as if modulation occurred. For example, V/V of a-minor is shown as V of e-minor. This replacement can be done one by one without having room for interpretation.

With regards to the harmony name in the absolute reference, the following rules are adopted to avoid confusion and to avoid the use of special characters that can be detrimental to the calculation.

- "maj" is added for all major triads (e.g. Amaj)
- "min" is added for all minor triads (e.g. Emin)
- "min-5" is added for all diminished triads (e.g. Bmin-5)
- "+" represents ♯ (Sharp) (e.g. F+min for F♯ minor chord)
- "-" represents ♭ (Flat) (e.g. B-maj for B♭ major chord)

Table 4.3 shows the example corresponds to the analysis of the measure 1 and 2 of BWV599, which is displayed in Figure 4.1. This table will be referred to as *Little Organ Book corpus table* in the following chapters.

Table 4.3: Example of the *Little Organ Book corpus table* for the measure 1 and 2, BWV599

WorkNum	MeasNum	BeatNum	TonAbs	TonRel	HarmAbs	HarmRel
599	1	1	a	I	Amin	I
599	1	3	e	V	Bmaj	V
599	1	4	e	V	Emin	I
599	2	1	e	V	Bmaj	V
599	2	2	a	I	Emaj	V
599	2	2	a	I	Amin	I
599	2	3	a	I	Emaj	V
599	2	3	a	I	Bmin-5	II
599	2	4	a	I	Emaj	V

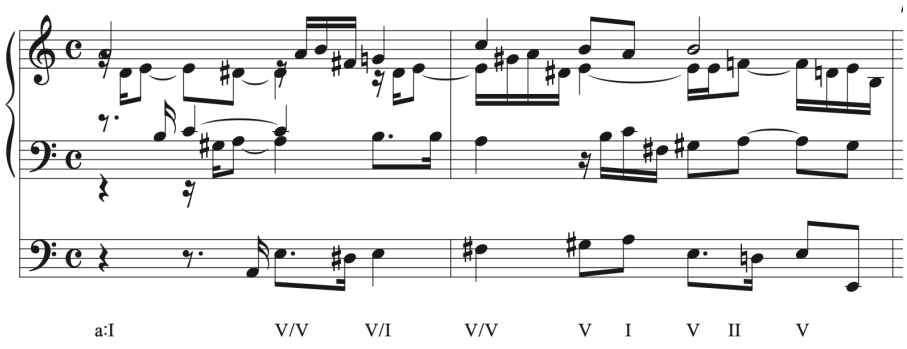

Figure 4.1: Analysis of the measure 1 and 2 of BWV599

4.4 Search algorithms

This section describes the algorithms for implementing the H1, H2, T1 and T2 search. Table 4.4 shows the environment used for implementation.

Table 4.4: Implementation Environment

Program language	Python 3
Library	music21 [11]
Music notation software	MuseScore 3
Music scores	The *Little Organ Book analysis database* in MusicXML
Search target	The *Little Organ Book corpus table* in CSV

Figure 4.2 describes a process flow for the H1, H2, T1 and T2 search. These searches can be performed in a similar manner as described below.

1. The user inputs a search query.
2. The Python program scans the *Little Organ Book corpus table* from the search query and retrieves the opus and measure number(s) which match(es) the query.
3. The MusicXML file(s) is/are retrieved from the *Little Organ Book analysis database* according to the opus number(s) fetched in the process #2.
4. For all retrieved MusicXML files, MuseScore displays measures fetched in the process #2.

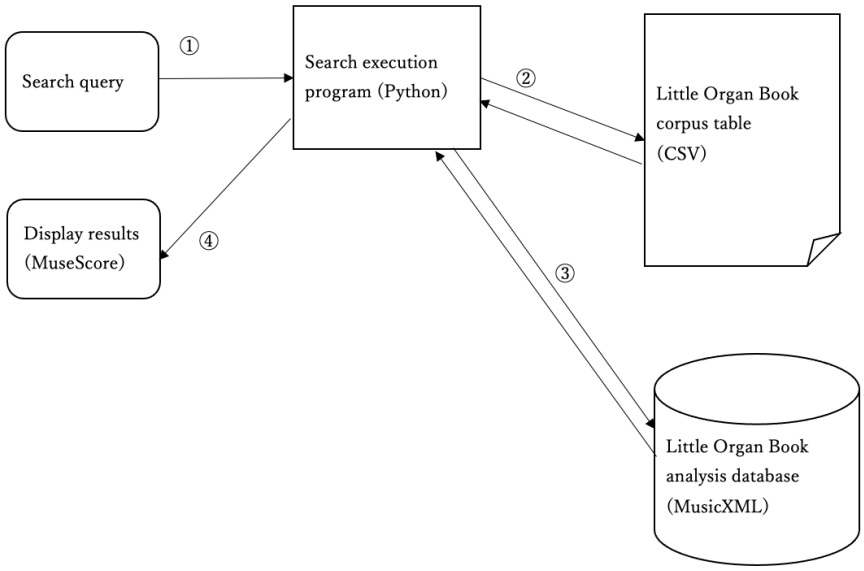

Figure 4.2: Process flow for performing a search

4.4.1 Harmony search in the absolute reference (H1)

Requirements for the H1 search are defined as follows.

1. Scan the HarmAbs column in the *Little Organ Book corpus table* and retrieve all rows that match the harmonic sequence given by the user query.

2. However, if the same harmony is repeated in the *Little Organ Book corpus table*, they are treated as if there were no duplicate rows. (e.g. if there are rows such as Cmaj, Gmaj, Gmaj, Cmaj, they are interpreted as Cmaj, Gmaj, Cmaj.)

3. If the retrieved rows have different WorkNum elements, such rows are not considered a search result.

Item #2 above is required because repeated harmony does not change harmonic functionality. For example, when a user enters a query as "Cmaj, Gmaj, Cmaj", he/she should also wish to see the case that is "Cmaj, Gmaj, Gmaj, Cmaj" in the *Little Organ Book corpus table*. Item #3 is needed to exclude the case that matches the user's query but has the different work numbers, which does not make any sense as a search result. For example, when a user enters a query as "Cmaj, Gmaj, Cmaj", "Cmaj (in BWV599),

Gmaj (in BWV599), Cmaj (in BWV600)" should not be retrieved as a search result because BWV600 has nothing to do with BWV599.

Algorithm 1 shows the execution algorithm for the H1 search. The overview is outlined below.

1. Row 1-2 load a search query. The process retrieves harmony names in the absolute reference from the command line and stores them in args[]. m represents the length of the search query, which will be used to manipulate the array.

2. Row 3-4 load the *Little Organ Book corpus table*. The process stores the information from the *Little Organ Book corpus table* into csvdata array. After that, the WorkNum column in csvdata is loaded to the WorkNum array.

3. Row 5 sets up the search target. The index is given in the first column of cmpList and the HarmAbs column in csvdata is copied into the second column of cmpList.

4. Row 8-16 generate the search target. The process retrieves m rows from cmpList. The first element of the first column is copied into the tgtList array's first column. m elements of the second column are transposed and copied into the tgtList array's second to $m+1$th column. The basic concept of this procedure is illustrated in Figure 4.3. In this example, the 10th-12th elements of cmpList are not copied into tgtList because their second column elements are equivalent to the 9th element, which is Amin.

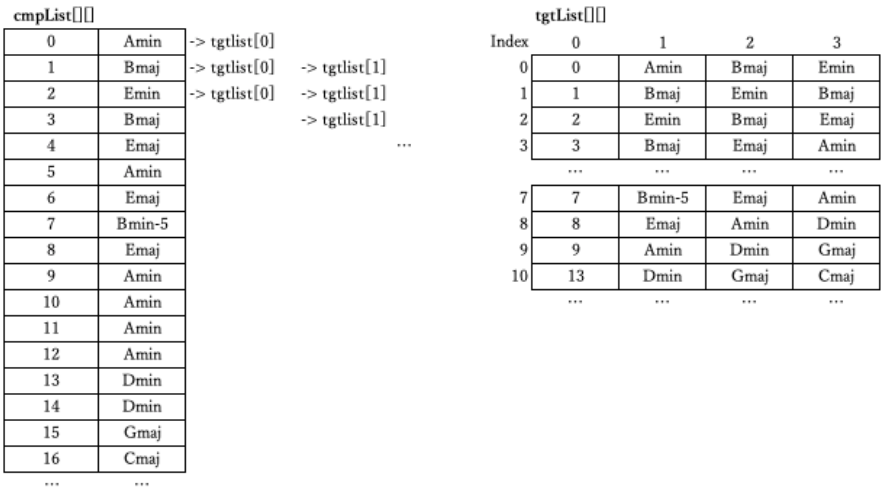

Figure 4.3: Overview of creating tgtList from cmpList (m=3)

5. The search is performed after row 18. Row 20 compares all rows in tgtList to args. If all elements in the second to $m+1$th column of the tgtList array match args, row k in tgtList is treated as a search result candidate.

6. Row 21 retrieves the start index of csvdata by using the tgtList element in the first column of the kth row.

7. Row 22 retrieves the end index of csvdata by using the tgtList element in the first column of the $k+m-1$th row.

8. Row 24 excludes search results with different WorkNum elements.

9. Row 25-28 prepare to display the search result. resultWorkNum (which denotes the opus number), resultStMeas (which denotes the minimum measure number) and resultEdMeasu (which denotes the maximum measure number) are retrieved from the csvdata rows which satisfy the condition in row 20 and row 24. Figure 4.4 shows the example where (Bmin-5, Emaj, Amin) is given as a search query.

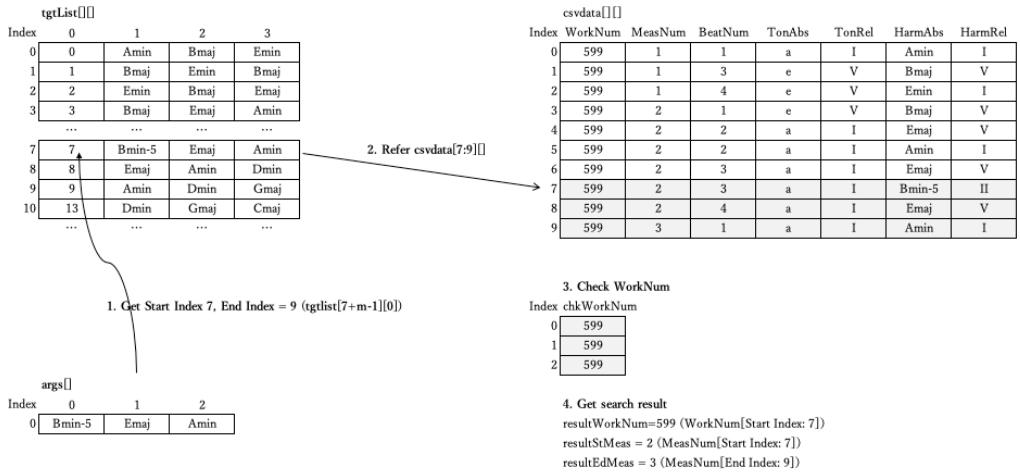

Figure 4.4: Overview of the process to display the search result (Search query: args[]=(Bmin-5, Emaj, Amin))

10. Row 29 loads MusicXML file which has the opus number equal to resultWorkNum. The measures from resultStMeas to resultEdMeas are derived. The loaded file is stored in mxmlfileexcerpt.

11. Row 30 calls MuseScore application to display mxmlfileexcerpt.

Algorithm 1 Execution algorithm for the H1 search (Harmony search in the absolute reference)

```
args[] ⇐ Get a search query (e.g.: Dmin, Gmaj, Cmaj) from the command line arguments
int m ⇐ Number of the command line arguments
csvdata[][] ⇐ Load the Little Organ Book corpus table (CSV)
WorkNum[] ⇐ Load the WorkNum column in csvdata
cmpList[][] ⇐ Give the index in the first column and load the HarmAbs column in csvdata
int n ⇐ Number of rows of cmpList
Declare two dimensional array tgtList[][]
for int i=0 to n-1 do
    if i==0 then
        tgtList[i][0] ⇐ cmpList[i][0]
        tgtList[i][1:m] ⇐ Transposition of cmpList[i+m][1]
    else if cmpList[i][1] is not equal to the preceding element AND cmpList[i][2] is equal to the preceding element then
        tgtList[i][0] ⇐ cmpList[i][0]
        tgtList[i][1:m] ⇐ Transposition of cmpList[i:i+m][1]
    end if
end for
int p ⇐ Number of rows of tgtlist
while k=0 < p do
    tmplist[] ⇐ tgtList[k][1:m]
    if args[]==tmplist[] then
        stIdx ⇐ tgtList[k][0]
        edIdx ⇐ tgtList[k+m-1][0]
        chkWorkNum[] ⇐ WorkNum[stIdx:edIdx]
        if All elements in chkWorkNum[] are equivalent then
            resultWorkNum ⇐ WorkNum[k]
            MeasNum[] ⇐ Load the MeasNum column of csvdata
            resultStMeas ⇐ MeasNum[stIdx]
            resultEdMeas ⇐ MeasNum[edIdx]
            mxmlfileexcerpt ⇐ Load the musicXML files that match resultWorkNum and derive the measures from resultStMeas to resultEdMeas.
            MuseScore displays mxmlfileexcerpt
        end if
    end if
end while
```

4.4.2 Harmony search in the relative reference (H2)

Requirements for the H2 search are defined as follows.

1. Scan the HarmRel column in the *Little Organ Book corpus table* and retrieve all rows that match the harmonic sequence given by the user query.

2. However, if the same harmony is repeated in the *Little Organ Book corpus table*, they are treated as if there were no duplicate rows. (e.g. if there are rows such as I, V, V, I, they are interpreted as I, V, I.)

3. If the retrieved rows have different WorkNum elements, such rows are not considered a search result.

4. If the retrieved rows have different TonAbs elements, such rows are not considered a search result.

Item #2 above is required as repeated harmony does not change harmonic functionality. For example, when a user enters a query as "I, V, I", he/she should also wish to see the case that is "I, V, V, I" in the *Little Organ Book corpus table*. Item #3 is needed to exclude the case that matches the user's query but has the different work numbers, as discussed in the H1 search algorithm. Item #4 aims at excluding the chord progression with multiple keys, because such search demand should be covered by T1 or T2. The H2 search only covers the search for harmony in the same key.

Algorithm 2 shows the execution algorithm for the H2 search. The overview is outlined below.

1. Row 1-2 load a search query. The process retrieves harmony names in the relative reference from the command line and stores them in args[]. m represents the length of the search query, which will be used to manipulate the array.

2. Row 3-5 load the *Little Organ Book corpus table*. The process stores the information from the *Little Organ Book corpus table* into csvdata array. After that, the WorkNum column in csvdata is loaded to the WorkNum array and the TonAbs column in csvdata is loaded to the WorkNum array.

3. Row 6 sets up the search target. The index is given to the first column of cmpList, the HarmRel column in csvdata is copied into the second column of cmpList and the TonAbs column in csvdata is copied into the third column of cmpList.

4. Row 9-17 generate the search target. The process retrieves m rows from cmpList. The first element of the first column is copied into the tgtList array's first column. m elements of the second column are transposed and copied into the tgtList array's second to $m+1$th column. The basic concept of this procedure is illustrated in Figure 4.5. In this example, the 10th element of cmpList is not copied into tgtList because their second column and third column elements are equivalent to the 9th element, which are I and a.

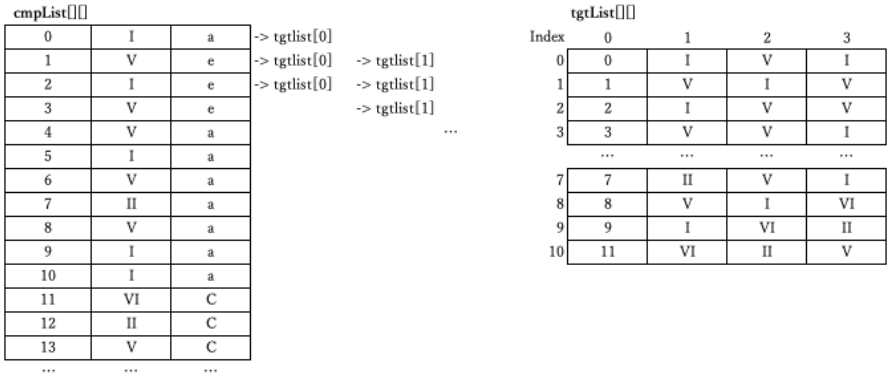

Figure 4.5: Overview of creating tgtList from cmpList (m=3)

5. The search is performed after row 19. Row 21 compares all rows in tgtList to args. If all elements in the second to $m+1$th column of the tgtList array match args, row k in tgtList is treated as a search result candidate.

6. Row 22 retrieves the start index of csvdata by using the tgtList element in the first column of the kth row.

7. Row 23 retrieves the end index of csvdata by using the tgtList element in the first column of the $k+m-1$th row.

8. Row 26 excludes search results with different WorkNum or TonAbs elements.

9. Row 27-30 prepare to display the search result. resultWorkNum (which denotes the opus number), resultStMeas (which denotes the minimum measure number) and resultEdMeasu (which denotes the maximum measure number) are retrieved from the csvdata rows which satisfy the condition in row 21 and row 26. Figure 4.6 shows the example where (II, V, I) is given as a search query.

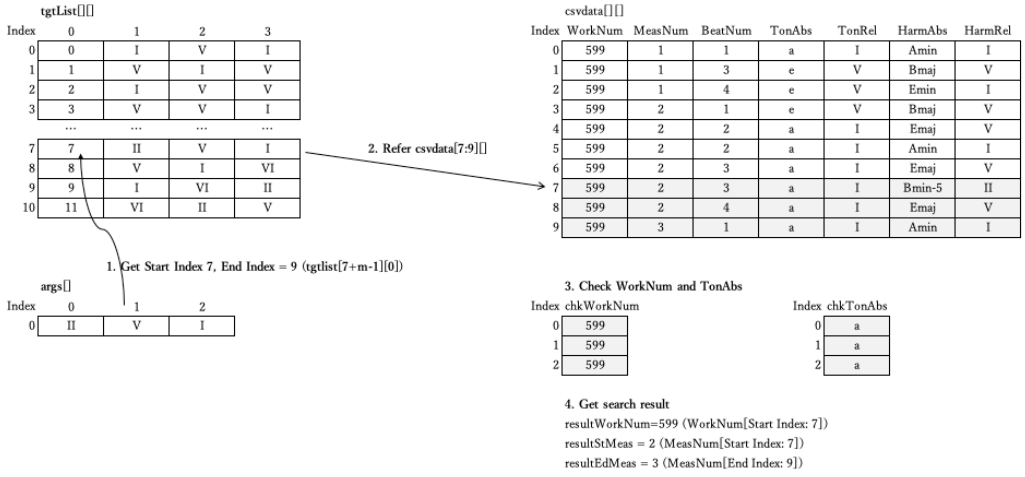

Figure 4.6: Overview of the process to display the search result (Search query: args[]=(Bmin-5, Emaj, Amin))

10. Row 31 loads MusicXML file which has the opus number equal to resultWorkNum. The measures from resultStMeas to resultEdMeas are derived. The loaded file is stored in mxmlfileexcerpt.

11. Row 32 calls MuseScore application to display mxmlfileexcerpt.

Algorithm 2 Execution algorithm for the H2 search (Harmony search in the relative reference)

```
args[] ⇐ Get a search query (e.g.: I, V, I) from the command line
int m ⇐ Number of the command line arguments
csvdata[][] ⇐ Load the Little Organ Book corpus table (CSV)
WorkNum[] ⇐ Load the WorkNum column in csvdata
TonAbs[] ⇐ Load the TonAbs column in csvdata
cmpList[][] ⇐ Give the index in the first column and load the HarmAbs column in csvdata
int n ⇐ Number of rows of cmpList
Declare two dimensional array tgtList[][]
for int i=0 to n-1 do
  if i==0 then
    tgtList[i][0] ⇐ cmpList[i][0]
    tgtList[i][1:m] ⇐ Transposition of cmpList[i+m][1]
  else if cmpList[i][1] is not equal to the preceding element AND cmpList[i][2] is equal to the preceding element then
    tgtList[i][0] ⇐ cmpList[i][0]
    tgtList[i][1:m] ⇐ Transposition of cmpList[i:i+m][1]
  end if
end for
int p ⇐ Number of rows of tgtlist
while k=0 < p do
  tmplist[] ⇐ tgtList[k][1:m]
  if args[]==tmplist[] then
    stIdx ⇐ tgtList[k][0]
    edIdx ⇐ tgtList[k+m-1][0]
    chkWorkNum[] ⇐ WorkNum[stIdx:edIdx]
    chkTonAbs[] ⇐ TonAbs[stIdx:edIdx]
    if All elements in chkWorkNum[] are equivalent AND All elements in chkTonAbs[] are equivalent then
      resultWorkNum ⇐ WorkNum[k]
      MeasNum[] ⇐ Load MeasNum column of csvdata
      resultStMeas ⇐ MeasNum[stIdx]
      resultEdMeas ⇐ MeasNum[edIdx]
      mxmlfileexcerpt ⇐ Load the musicXML files that match resultWorkNum and derive the measures from resultStMeas to resultEdMeas.
      MuseScore displays mxmlfileexcerpt
    end if
  end if
end while
```

4.4.3 Key search in the absolute and the relative reference (T1 and T2)

Requirements for the T1 and T2 search are defined as follows. The T1 and T2 search can be performed by the same algorithm.

1. Scan the TonAbs (in case of T1) or the Ton Rel (in case of T2) column in the *Little Organ Book corpus table* and retrieve all rows that match the key sequence given by the user query.

2. However, if the same key is repeated in TonAbs or TonRel, they are treated as if there were no duplicate rows. (e.g. if there are rows such as a, a, a, e, C, they are interpreted as a, e, C.)

3. If the retrieved rows have different WorkNum elements, such rows are not considered a search result.

Item #2 above is required to identify where the key changes, as the *Little Organ Book corpus table* repeats the same TonAbs or TonRel elements while the same key continues. Item #3 is needed to exclude the case that matches the user's query but has the different work numbers, as discussed in the H1 search.

Algorithm 3 shows the execution algorithm for the T1/T2 search. The overview is outlined below.

1. Row 1-2 load a search query. The process retrieves key names from the command line and stores them in args[]. m represents the length of the search query, which will be used to manipulate the array.

2. Row 3-4 load the *Little Organ Book corpus table*. The process stores the information from the *Little Organ Book corpus table* into csvdata array. After that, the WorkNum column in csvdata is loaded to the WorkNum array.

3. Row 5 sets up the search target. The index is given in the first column of cmpList and the TonAbs (in case of T1) / TonRel (in case of T2) column in csvdata is copied into the second column of cmpList.

4. Row 8-16 generate the search target. The process retrieves m rows from cmpList. The first element of the first column is copied into the tgtList array's first column. m elements of the second column are transposed and copied into the tgtList array's second to $m+1$th column. The basic concept for this procedure is illustrated in Figure 4.7. In this example,

the 3rd-4th elements and the 6th to 12th elements of cmpList are not copied into tgtList because their second column elements are equivalent to the 2nd or the 5th elements, which are e and a.

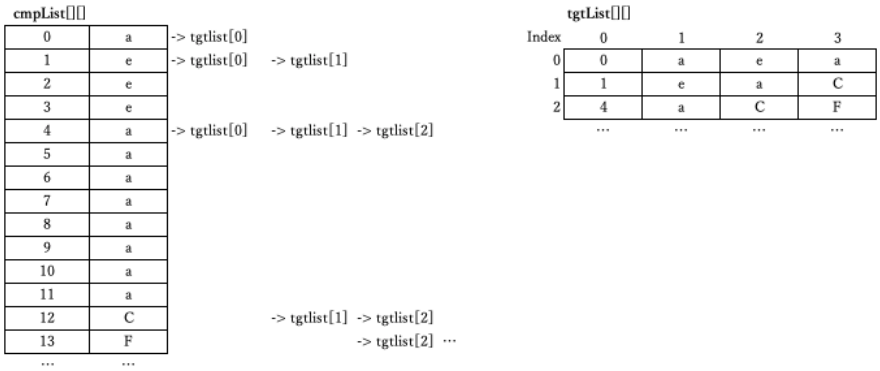

Figure 4.7: Overview of creating tgtList from cmpList (m=3)

5. The search is performed after row 18. Row 20 compares all rows in tgtList to args. If all elements in the second to $m+1$th column of the tgtList array match args, row k in tgtList is treated as a search result candidate.

6. Row 21 retrieves the start index of csvdata by using the tgtList element in the first column of the kth row.

7. Row 22 retrieves the end index of csvdata by using the tgtList element in the first column of the $k+m-1$th row.

8. Row 24 excludes search results with different WorkNum elements.

9. Row 25-28 prepare to display the search result. resultWorkNum (which denotes the opus number), resultStMeas (which denotes the minimum measure number) and resultEdMeasu (which denotes the maximum measure number) are retrieved from the csvdata rows which satisfy the condition in row 20 and row 24. Figure 4.8 shows the example where (a, e, a) is given as a search query.

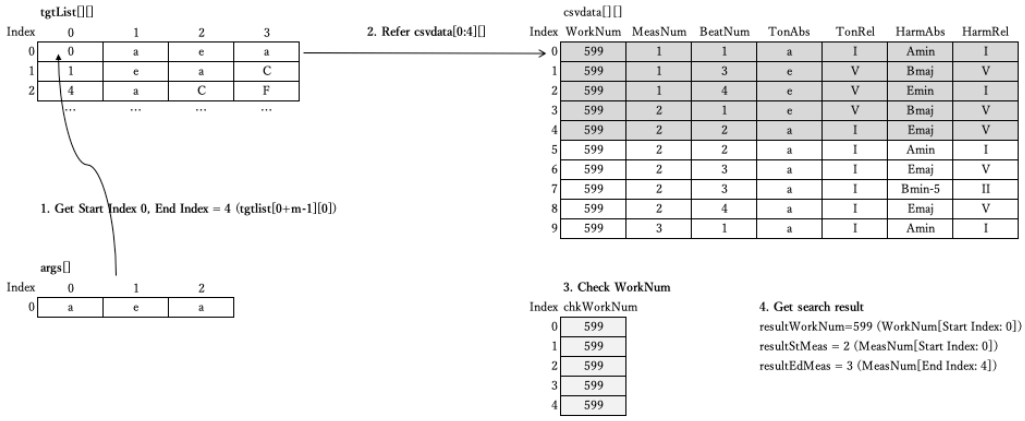

Figure 4.8: Overview of the process to display the search result (Search query: args[]=(a, e, a))

10. Row 29 loads MusicXML file which has the opus number equal to resultWorkNum. The measures from resultStMeas to resultEdMeas are derived. The loaded file is stored in mxmlfileexcerpt.

11. Row 30 calls MuseScore application to display mxmlfileexcerpt.

Algorithm 3 Execution algorithm for the T1 and T2 search (Key search in the absolute and the relative reference)

```
1: args[] ⇐ Get a search query (e.g.: a, e, C) from the command line arguments
2: int m ⇐ Number of the command line arguments
3: csvdata[][] ⇐ Load the Little Organ Book corpus table (CSV)
4: WorkNum[] ⇐ Load the WorkNum column in csvdata
5: cmpList[][] ⇐ Give the index in the first column, load the TonAbs (in case of T1) / TonRel (in case of T2) column in csvdata to the second column
6: int n ⇐ Number of rows of cmpList
7: Declare two dimensional array tgtList[][]
8: for int i=0 to n-1 do
9:    if i==0 then
10:      tgtList[i][0] ⇐ cmpList[i][0]
11:      tgtList[i][1:m] ⇐ Transposition of cmpList[i+m][1]
12:   else if cmpList[i][1] is not equal to the preceding element AND cmpList[i][2] is equal to the preceding element then
13:      tgtList[i][0] ⇐ cmpList[i][0]
14:      tgtList[i][1:m] ⇐ Transposition of cmpList[i:i+m][1]
15:   end if
16: end for
17: int p ⇐ Number of rows of tgtlist
18: while k=0 < p do
19:   tmplist[] ⇐ tgtList[k][1:m]
20:   if args[]==tmplist[] then
21:      stIdx ⇐ tgtList[k][0]
22:      edIdx ⇐ tgtList[k+m-1][0]
23:      chkWorkNum[] ⇐ WorkNum[stIdx:edIdx]
24:      if All elements in chkWorkNum[] are equivalent then
25:         resultWorkNum ⇐ WorkNum[k]
26:         MeasNum[] ⇐ Load the MeasNum column of csvdata
27:         resultStMeas ⇐ MeasNum[stIdx]
28:         resultEdMeas ⇐ MeasNum[edIdx]
29:         mxmlfileexcerpt ⇐ Load the musicXML files that match resultWorkNum and derive the measures from resultStMeas to resultEdMeas.
30:         MuseScore displays mxmlfileexcerpt
31:      end if
32:   end if
33: end while
```

Chapter 5

Further possibility for leveraging the proposed data structure

The proposed data structure can be used in addition to the search requirements defined in the previous chapter. This section discusses two cases to show the further applicability of the proposed data structure.

1. Fuzzy search for H2: enable the H2 search regardless of whether the harmony is expressed in a borrowed chord or modulation.
2. Find the location where the sequence takes place, especially for the descending fifth sequence.

5.1 Fuzzy search for H2

5.1.1 Issue of the H2 search

When a harmonic sequence is given in H2 as a search query, it is intended to find the location that exactly matches the query. Therefore, when the HarmRel elements are stored as Table 5.1, the query shown below should be given to select the rows.

- H2 search query I - IV - V - I - IV/I - IV/V - IV/I

However, when the HarmRel elements are stored as Table 5.2, the H2 search will not select rows because it does not exactly match the given query. The difference between Table 5.1 and 5.2 is whether the harmony is treated

as a borrowed chord or modulation, indicating the same harmony with a different expression. There is no concrete definition in musicology in which case would have to be treated as a borrowed chord and which case would have to be treated as modulation. As there is room for interpretation even from musicologist's point of view, the *Little Organ Book analysis database* also has an ambiguity in the use of borrowed chords and modulation. This section will discuss how the H2 search can be improved to enable robust search to the notation variants.

Table 5.1: Corpus table example (shown as the search result in the H2 search)

TonRel	HarmRel
I	I
I	IV
I	V
I	I
I	IV/I
I	IV/V
I	IV/I

Table 5.2: Corpus table example (*not* shown as the search result in the H2 search)

TonRel	HarmRel
I	I
I	IV
I	V
I	I
IV	I
IV	V
IV	I

5.1.2 Relation between the degree of key and the degree of harmony

Recalling the scale structure in Western music, the combinations of key and harmony listed below represent the same harmony, ignoring the existence of major/minor triads and a leading note.

1. The Ist degree of the Ist key
2. The VIIth degree of the IInd key
3. The Vth degree of the IIIrd key
4. ...
5. The IInd degree of the VIIth key

Although the VIIth degree of the IInd key and the IInd degree of the VIIth key do not exactly match the Ist degree of the Ist key due to a leading

note, this chapter will purposely treat them as identical in order to simplify mathematical calculations. This simplification will not be harmful because the fuzzy search should get the result candidates broadly.

For example, a harmony which consists of C-E-G is C-major's Ist degree, but it can also be e-minor (the IIIrd key)'s VIth degree, F-major (the IVth key)'s Vth degree or G-major (the Vth key)'s IVth degree. The important point to note here is that the sum of the degree of harmony and the degree of key is 2 (e.g. I+I) or 9 (III+VI or V+IV) in this case. In addition, since a scale in Western music consists of 7 notes repeated in cycles (e.g. C, D, E, F, G, A and B for the C-major scale), when we subtract 7 for summation results which are 9, the sum of the degree of harmony and the degree of key should all be 2.

When we apply the same calculation for the IVth degree of the Ist key, we can see all summation results are 5.

1. The IVth degree of the Ist key (Sum: 5)
2. The IIIrd degree of the IInd key (Sum: 5)
3. The IInd degree of the IIIrd key (Sum: 5)
4. The Ist degree of the IVth key (Sum: 5)
5. ...
6. The Vth degree of the VIIth key (Sym: 5 [7+5-7])

Furthermore, when we apply the same calculation for the Vth degree of the Ist key, we can see all summation results are 6.

1. The Vth degree of the Ist key (Sum: 6)
2. The IVth degree of the IInd key (Sum: 6)
3. The IIIrd degree of the IIIrd key (Sum: 6)
4. The IInd degree of the IVth key (Sum: 6)
5. The Ist degree of the Vth key (Sum: 6)
6. ...
7. The VIth degree of the VIIth key (Sum: 6 [7+6-7])

5.1.3 Method for the fuzzy search

This subsection examines the method of conducting the fuzzy search based on the characteristics of the degree of key and harmony discussed in the previous subsection. Let us add one column to the *Little Organ Book corpus table*, which is called TonHarm. The values are given to the TonHarm column according to the rules described below:

1. If the element of HarmRel is *not* a borrowed chord (i.e. a Roman numeral without "/"), set the sum of TonRel+HarmRel in Arabic number.
2. If the element of HarmRel is a borrowed chord (i.e. Roman numerals with "/"), set the sum of two Roman numerals in HarmRel in Arabic number.
3. In all of these cases, subtract 7 if the value equals or exceeds 9.

Table 5.3 and Table 5.4 show the enhancement of Table 5.1 and Table 5.2, which added the TonHarm column according to the rule described above. When comparing Table 5.3 and Table 5.4, both have the same TonHarm numbers regardless of the HarmRel expression.

Table 5.3: Enhancement of Table 5.1

TonRel	HarmRel	TonHarm
I	I	2
I	IV	5
I	V	6
I	I	2
I	IV/I	5
I	IV/V	2
I	IV/I	5

Table 5.4: Enhancement of Table 5.2

TonRel	HarmRel	TonHarm
I	I	2
I	IV	5
I	V	6
I	I	2
IV	I	5
IV	V	2
IV	I	5

The search query should also be converted to Arabic numbers as shown below, in accordance with the rule applied to the TonHarm column. For example,

- H2 fuzzy search query: 2 - 5 - 6 - 2 - 5 - 2 - 5

should be created from

- H2 search query: I - IV - V - I - IV/I - IV/V - IV/I

By scanning the TonHarm column using the H2 fuzzy search query, we can retrieve the result regardless of the expression of harmony.

5.1.4 Supplemental notes for the fuzzy search

If a minor amendment is applied, the fuzzy search method described in the previous subsection can also be applied when the base key is not the Ist key.

For example, let us consider the case shown in Table 5.5. This case represents the example that the I-II-V-I harmonic movement takes place in the IIIrd key in the middle of the music. After that, another I-II-V-I harmonic movement occurs in the Ist key.

Table 5.5: Example of the corpus table (Modulation from the IIIrd key to the Ist key)

TonRel	HarmRel	TonHarm
III	I	4
III	II	5
III	V	8
III	I	4
I	I	2
I	II	3
I	V	6
I	I	2

It is important to note that the Ist key in Table 5.5 is the VIth key from the IIIrd key perspective. For instance, if the Ist key in Table 5.5 is C-major, the IIIrd key is e-minor. However, from an e-minor standpoint, e-minor is the Ist key and C-major is the VIth key.

Therefore, the H2 search query shown below should need to obtain the case of Table 5.5. However, the H2 fuzzy search query corresponds to the H2 search query cannot retrieve the rows in Table 5.5 because the TonHarm column is not equivalent.

- H2 search query: I - II - V - I - VI/I - VI/II - VI/V - VI/I
- H2 fuzzy search query: 2 - 3 - 6 - 2 - 7 - 2 - 4 - 7

One way to solve this problem is to transpose the H2 search query from the IInd key to the VIIth key. Namely, the H2 search query should be extended to the queries described below.

1. H2 search query (Transposed into the IInd key): II/I - II/II - II/V - II/I - VII/I - VII/II - VII/V - VII/I

2. H2 search query (Transposed into the IIIrd key): III/I - III/II - III/V - III/I - I - II - V - I

3. H2 search query (Transposed into the IVth key): IV/I - IV/II - IV/V - IV/I - II/I - II/II - II/V - II/I

4. ...

5. H2 search query (Transposed into the VIIth key): VII/I - VII/II - VII/V - VII/I - V/I - V/II - V/V - V/I

The H2 fuzzy search queries correspond to the H2 search queries are:

1. H2 fuzzy search query (Transposed into the IInd key): 3 - 4 - 7 - 3 - 8 - 2 - 5 - 8

2. H2 fuzzy search query (Transposed into the IIIrd key): 4 - 5 - 8 - 4 - 2 - 3 - 6 - 2

3. H2 fuzzy search query (Transposed into the IVth key): 5 - 6 - 2 - 5 - 3 - 4 - 7 - 3

4. ...

5. H2 fuzzy search query (Transposed into the VIIth key): 8 - 2 - 5 - 8 - 6 - 7 - 3 - 6

By using all these fuzzy search queries, the case of Table 5.5 can be retrieved as a search result of the H2 fuzzy search query that is transposed into the IIIrd key.

5.2 Finding a sequence using a harmonic progression

5.2.1 Overview of the descending fifth sequence

"Sequence" in musicology is defined as "the more or less exact repetition of a passage at a higher or lower level of pitch" [19]. In many sequence models, the descending fifth sequence is one of the more widely used stylized movements of the Baroque period. For example, this sequence has a harmonic progression which descends several times by Vth degree, such as V-I-IV-VII-III-VI-II. The MILNE Library [20] explains in more detail the sequence examples, such as BWV593 illustrated in Figure 5.1. The *Little*

Organ Book corpus table can be utilized to find where the descending fifth sequence occurs, because we can deduce the existence of the descending fifth sequence by simply verifying the progression of harmony, without seeing the melodic movement.

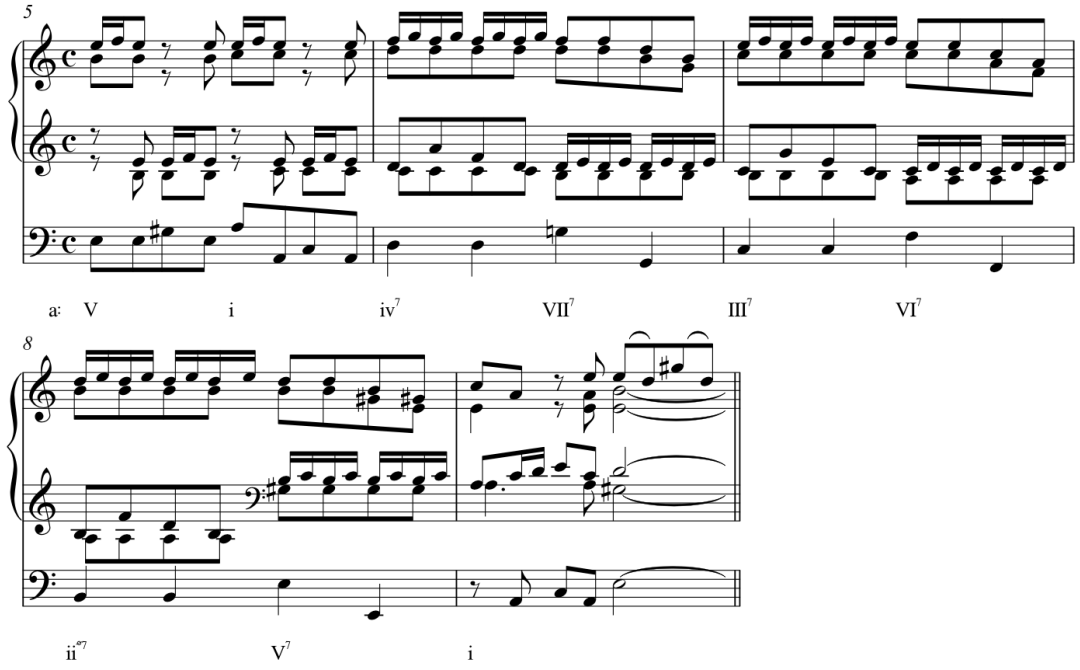

Figure 5.1: Example of the descending fifth sequence (BWV593, cited from MILNE Library [19])

5.2.2 Method to find the descending fifth sequence

As described in the previous section, a scale in Western music consists of 7 notes which are repeated in cycles. When the G-major chord (triad of G, B and D) descends by Vth, it reaches the C-major chord (triad of C, E and G). Likewise, when the C-major chord descends by Vth, it reaches the F-major chord (triad of F, A and C).

Let us examine how to deal with the characteristic of the scale in a mathematical way. Firstly, let us list the chords from A to G and set the index from 1 to 7, ignoring the sharps/flats and Major/Minor chords, which is shown in Table 5.6.

When we see a movement of two consecutive harmonies, there is a relation shown below when the harmony descends by the Vth degree. In this case,

Table 5.6: Chord names with index

Chord name	Index
A	1
B	2
C	3
D	4
E	5
F	6
G	7

X denotes the number of the first chord and Y denotes the number of the second chord.

$$\begin{cases} |X - Y - 1| = 3 & \text{if } X - Y > 0, \\ |X - Y| = 3 & \text{if } X - Y < 0 \end{cases} \tag{5.1}$$

For example, when harmony shifts from G to C, the difference between two chords is $|7 - 3 - 1| = 3$ because $X = 7$, $Y = 3$ and $X - Y > 0$. Meanwhile, when harmony shifts from A to D, the difference between two chords is $|1-4| = 3$ because $X = 1$, $Y = 4$ and $X-Y < 0$. Therefore, we can get candidates for the descending fifth sequence by subtracting the numbers from two consecutive harmonies of all HarmRel column elements in the *Little Organ Book corpus table* and finding the location where 3 is repeated.

Table 5.7 shows an example of this manipulation, where we find 3 is repeated in the first half of the measure. In fact, it moves as II-V-I-IV, which implies the descending fifth sequence as shown in Figure 5.2.

Table 5.7: Chord name, Index and the difference of two consecutive chords of the 4th measure of BWV599

Chord name	Index	Difference between two consecutive chords
Dmin	4	-
Gmaj	7	3
Cmaj	3	3
Fmaj	6	3
Dmin	4	2
Gmaj	7	3
Cmaj	3	3

By applying this method to all pieces in J.S. Bach's Little Organ Book, we can find the very long descending fifth sequence from the 10th measure to

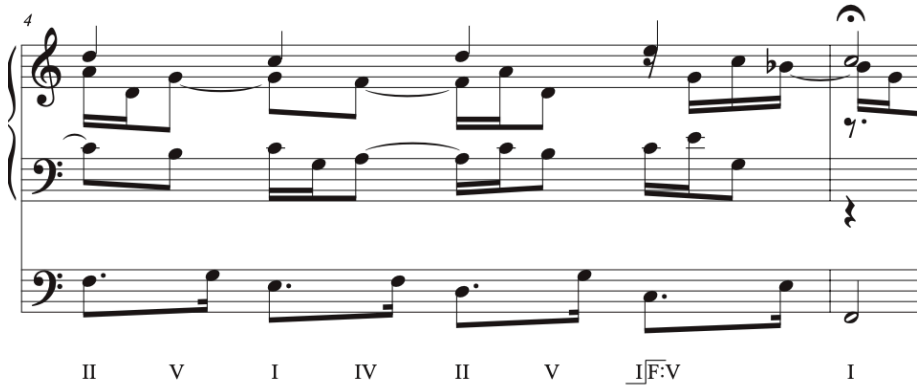

Figure 5.2: The 4th measure of BWV599

the 13th measure of BWV625, which starts with F-major's V and ends with d-minor's I+. By using the *Little Organ Book corpus table*, we can efficiently gather candidates of the sequence from a very simple calculation.

Table 5.8: The 10th-13th measures of BWV625 and the difference of two consecutive chords

TonAbs	HarmRel	HarmAbs	Index	Difference between two chords
F	V	Cmaj	3	-
F	I	Fmaj	6	3
d	VI	B-maj	2	3
d	II	Emin-5	5	3
d	V	Amaj	1	3
d	I	Dmin	4	3
F	II	Gmin	7	3
F	V	Cmaj	3	3
F	I	Fmaj	6	3
F	IV	B-maj	2	3
d	II	Emin-5	5	3
d	V	Amaj	1	3
d	I+	Dmaj	4	3

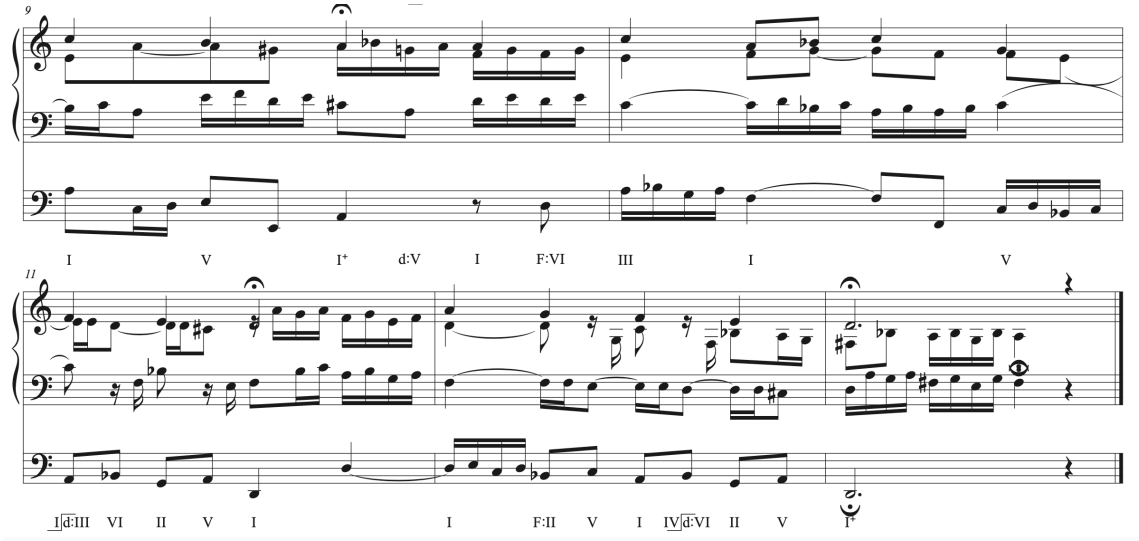

Figure 5.3: The 9th-13th measures of BWV625

Chapter 6

Conclusion

6.1 Achievements of the thesis

This thesis proposed multi-layered data structure containing scores, key and harmonic information. The proposed data structure allows searching through a progression of key or harmony, which has not been covered by existing CBMR research. For example, the proposed data structure and the algorithms fulfill requirements such as "Find measures from music scores by key that moves from C-major to a-minor" or "Find measures from music scores by harmony that moves as I, II, V and I".

The principal contributions of this thesis are as follows.

1. Contribution to musical education: students learning composition will easily find examples of great composers of the past with the same key or harmonic movement.

2. Contribution to musical interpretation: by listing all the musical pieces that have the similar key or harmonic movement, people can be aware of the similarity in the upper layer of music, which is not easily visible on the surface of the music. This will lead to a new understanding of musical pieces when people perform a musical analysis.

3. Contribution to musicology: musicologists used to collect examples manually when he/she wants to see the similar key/harmonic movement in various musical pieces. The proposed data structure and algorithms will streamline such time-consuming work. People will be able to easily answer questions such as "find locations that have a V-IV harmonic progression" or "list all the compositional examples for the progression of II-V-I".

6.2 Future work

Three points listed below are the topics to be addressed in the future work.

1. Automatic creation of corpus tables: In this thesis, the corpus table called *Little Organ Book corpus table* was created manually by the author. In order to create a corpus table from larger digitized music score storage, automation of the process using digitized information (MusicXML, MEI) is a must.

2. Implement functionality other than search: CBMR does not aim only to consider the effective way to search but also to gather analytical information about the music. For example, jSymbolic [10] delivers statistics related to given music notes such as the pitch class histogram or the average duration of notes. Similar approaches can be taken for upper layers of music. For example, further contribution to musicology is expected by providing the transition probability of key or harmony.

3. Further enhancement of the fuzzy search: the H1-H2 and the T1-T2 search algorithms are an exact match search. In Chapter 5, this thesis also proposed a fuzzy search algorithm that is robust to notation variants of borrowed chords and modulation. However, there may be other requirements for the fuzzy search such as retrieving the "IV-V-I" harmonic movements from a "II-V-I" search query, which has a feature very similar to "IV-V-I". The algorithms proposed in this thesis are not able to cover such a search demand. One possible solution is to introduce a concept of harmonic distance advocated in Tonal Pitch Space (TPS) [21], which can retrieve similar harmonic movements from the function of harmonies.

Bibliography

[1] Downie, J. Stephen. "The Scientific Evaluation of Music Information Retrieval Systems: Foundations and Future." *Computer Music Journal* 28, no. 2 (2004): 12-23. muse.jhu.edu/article/169382.

[2] Velardo, V, M. Vallati, S. Jan, "Symbolic Melodic Similarity: State of the Art and Future Challenges." *Computer Music Journal* (2016): 40 (2): 70–83.

[3] Garfinkle, D., C. Arthur, P. Schubert, J. Cumming and I. Fujinaga, PatternFinder: Content-Based Music Retrieval with music21: *Proceedings of the 4th International Workshop on Digital Libraries for Musicology.* (2017): 5-8

[4] Lemström, K. *String matching techniques for music retrieval.* Ph.D. Dissertation. University of Helsinki. (2000).

[5] Orio, N., and A. Rodà. "A Measure of Melodic Similarity Based on a Graph Representation of the Music Structure." *Proceedings of the International Conference for Music Information Retrieval*, (2009): pp. 543–548.

[6] Schenker, H., *Der Freie Satz, Neue musikalische Theorien und Phantasien.* Universal Wien, O. Jonas, 1956 edition (1935).

[7] Lerdhal, F. and R. Jackendoff. *A Generative Theory of Tonal Music.* The MIT Press, Cambridge, MA (1983).

[8] Fujinaga, I., A. Hankinson, and J. Cumming. "Introduction to SIMSSA (Single Interface for Music Score Searching and Analysis)." *Proceedings of the International Workshop on Digital Libraries for Musicology*, (2014): 100–102. London, UK.

[9] Hopkins, E., Y. Ju, G. Polins Pedro, C. McKay, J. Cumming, and I. Fujinaga. SIMSSA DB: Symbolic music discovery and search. *Poster*

presentation at the International Conference on Digital Libraries for Musicology. (2019)

[10] McKay, C. jSymbolic: A software application for music information retrieval and analysis. Invited Speaker. *CESEM*, Nova University of Lisbon, Lisbon, Portugal. 8 March 2018.

[11] Cuthbert, M. Scott and C. Ariza. "music21: A toolkit for computer-aided musicology and symbolic music data." *Proceedings of the 11th International Society for Music Information Retrieval Conference.* (2010): 637–642.

[12] Abrouk, L., H. Audéon, N. Cullot, C. Davy-Rigaux, Z. Faget, D. Gross-Amblard, P. Rigaux, A. Tacaille, E. Gavignet, and V. Thion-Goasdoué. "The Design and Implementation of neuma, a Collaborative Digital Score Library." In Submitted., (2010). Available at http://neuma.irpmf-cnrs.fr.

[13] Viro, V., "Peachnote: Music Score Search and Analysis Platform," *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR)*, (2011): pp. 359–362.

[14] Bahraini, A. and E. Tilevich. "Ask toscanini!: architecting a search engine for music scores beyond metadata." *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing* (2019): n. pag.

[15] de Haas, W.B., M. Rohrmeier, R.C. Veltkamp, F. Wiering, Modeling Harmonic Similarity Using a Generative Grammar of Tonal Harmony.: *Proceedings of the Tenth International Society for Music Information Retrieval Conference (ISMIR).* (2009): 549–554

[16] Neuwirth M., D. Harasim, FC. Moss, M. Rohrmeier, The Annotated Beethoven Corpus (ABC): A Dataset of Harmonic Analyses of All Beethoven String Quartets. *Frontiers Dig Human.* (2018): 5(16). https://doi.org/10.3389/fdigh.2018.00016.

[17] Gotham, Mark R. H., D. Tymoczko and M. Cuthbert. "The RomanText Format: A Flexible and Standard Method for Representing Roman Numerial Analyses." *ISMIR* (2019).

[18] Yaolong J., S. Howes, C. McKay, N. Condit-Schultz, J. Calvo-Zaragoza, I. Fujinaga: An Interactive Workflow for Generating Chord Labels for Homorhythmic Music in Symbolic Formats. *ISMIR* (2019): 862-869

[19] Kennedy, M. and J.B. Kennedy, The Concise Oxford Dictionary of Music (5 ed.), Oxford University Press (2007).

[20] MILNE Library, https://milnepublishing.geneseo.edu/fundamentals-function-form/chapter/25-diatonic-descending-fifth-sequences/

[21] Lerdahl, F. *Tonal Pitch Space*. Oxford University Press (2001).