BIG CHORD DATA EXTRACTION AND MINING

Mathieu Barthet[1], Mark D. Plumbley[1], Alexander Kachkaev[2], Jason Dykes[2], Daniel Wolff[2], Tillman Weyde[2]

[1] *Centre for Digital Music, Queen Mary University of London*

[2] *City University London*

Correspondence should be addressed to: m.barthet@qmul.ac.uk

Abstract: **Harmonic progression is one of the cornerstones of tonal music composition and is thereby essential to many musical styles and traditions. Previous studies have shown that musical genres and composers could be discriminated based on chord progressions modeled as chord n-grams. These studies were however conducted on small-scale datasets and using symbolic music transcriptions.**

In this work, we apply pattern mining techniques to over 200,000 chord progression sequences out of 1,000,000 extracted from the I Like Music (ILM) commercial music audio collection. The ILM collection spans 37 musical genres and includes pieces released between 1907 and 2013. We developed a *single program multiple data* parallel computing approach whereby audio feature extraction tasks are split up and run simultaneously on multiple cores. An audio-based chord recognition model (Vamp plugin Chordino) was used to extract the chord progressions from the ILM set. To keep low-weight feature sets, the chord data were stored using a compact binary format. We used the CM-SPADE algorithm, which performs a vertical mining of sequential patterns using co-occurence information, and which is fast and efficient enough to be applied to big data collections like the ILM set. In order to derive key-independent frequent patterns, the transition between chords are modeled by changes of qualities (e.g. major, minor, etc.) and root keys (e.g. fourth, fifth, etc.). The resulting key-independent chord progression patterns vary in length (from 2 to 16) and frequency (from 2 to 19,820) across genres. As illustrated by graphs generated to represent frequent 4-chord progressions, some patterns like circle-of-fifths movements are well represented in most genres but in varying degrees.

These large-scale results offer the opportunity to uncover similarities and discrepancies between sets of musical pieces and therefore to build classifiers for search and recommendation. They also support the empirical testing of music theory. It is however more difficult to derive new hypotheses from such dataset due to its size. This can be addressed by using pattern detection algorithms or suitable visualisation which we present in a companion study.

1. INTRODUCTION

In Western tonal music, chord progressions form one of the fundamental building blocks of musical structure. A chord progression is a series of two or more chords the qualities and order of which contribute to establishing or changing the tonality of a musical piece founded on a given key. Although the space of chord progressions is more constrained than the space of melodies in a tonal system, due to perceptual considerations and harmonic rules [1], determining which of all possible chord progressions are sensible ones is not straightforward. In this study, we adopt a bottom-up, data-driven, approach to chord progression analysis, as in [2], as opposed to a top-down approach which would rely on a predefined set of rules fitting a chord progression model. Assuming that adequate data mining techniques and large enough datasets are employed, such approach offers the potential to uncover exemplar or idiomatic chord progressions directly from empirical data analyses.

Building a chord progression model that would encompass every styles and traditions is challenging, if not impossible. As a matter of fact, the nature of chord progressions used by composers varied and evolved across musical eras, styles and cultures in complex ways. For instance, chords containing the dissonant tritone (augmented fourth or diminished fifth) known colloquially as the "Devil's interval" were, successively, rejected until the end of the Renaissance, used in the Baroque and Classical eras, albeit in a controlled way, and finally, heavily exploited in Romantic, Modern and Jazz music. Hence, the evolution of musical styles has led to the emergence of different chord progressions. In addition to musical and psychoacoustical principles, the context in which music is created is also of importance when considering which chords were used and which weren't. The development of robust analysis methods for empirical data can help understanding how musical styles have evolved. We aim to detect frequent chord progressions, i.e. chord progressions that appear in a systematic way given a specific context. For instance, the I-V-vi-IV[1] progression which has been used in countless Popular music songs (e.g. Adele's *"Someone Like You"*), as funnily illustrated in the "Four-Chord Song" by comedy group The Axis of Awesome[2], was already used in Baroque music (e.g. Mozart's *Pachelbel's Canon* is a variant of this progression). Other progressions are highly idiosyncratic of certain musical genres, such as the III7-VI7-II7-V7 circle progression (e.g. E7-A7-D7-G7 in C) or so-called "ragtime progression", which often appears in the bridge section of Jazz standards. As authors from several previous works (e.g. [2], [3], [4], [5]), we believe that chord progressions contain precious information to find the commonalities and specificities of musical styles and composers. We don't attempt in this work to explain how progressions are constructed or why some progressions have been used but we are concerned with the problem of detecting significant chord progressions based on given datasets.

In contrast to traditional musicological studies conducted on small datasets, the proposed method - here applied to chord progression analysis - addresses large-scale music corpora. This is only made possible by the development of appropriate computational models and music information retrieval techniques. Hence our work follows an empirical musicology approach as advocated in [6]. We used an audio-based chord recognition algorithm, the Chordino Vamp plugin [7], to automatically predict the chord progressions from over one million musical pieces part of I Like Music[3]'s commercial music library (denoted ILM dataset in the following). Although chord progression analysis and modeling has been the focus of previous studies in the field of music informatics (see Section 2), none had relied on such large-scale corpus, to our knowledge. We developed for this purpose a single program mutliple data (SPMD) parallel computing technique for fast feature extraction from audio signals. Detecting frequent chord progressions in this "big data" context is well suited to pattern mining which aims at discovering hidden knowledge in very large amounts of data, regardless of its form. Sequential pattern mining (SPM) arose as a sub-field of data mining to focus on the detection of frequent subsequences from sequences of events occurring in an ordered way [8]. SPM has attracted a great deal of interest in recent years as it can be applied to many situations such as web user analysis, stock trend prediction, DNA sequence analysis, etc. [9]. To the best of our knowledge, SPM had not been applied to musical chord sequence analysis previously. In this study, we use one of the state-of-the-art's SPM algorithm, CM-SPADE, that relies on a pruning mechanism based on co-occurrence information [10]. We focus on the analysis of frequent chord patterns in six musical genres (Blues, Classical, Folk, Jazz, Reggae and Rock'n'roll) based on over 200,000 tracks from the ILM dataset and present

[1] We use a classic notation based on Roman numerals to refer to the modes of the seven-note major scale with upper and lower cases for major and minor modes, respectively: I (ionian), ii (dorian), III (phrygian), IV (lydian), V (mixolydian), vi (aeolian), vii^o (locrian); ^{o}: diminished triad.

[2] https://www.youtube.com/watch?v=oOlDewpCfZQ#t=56

[3] http://www.ilikemusic.com/

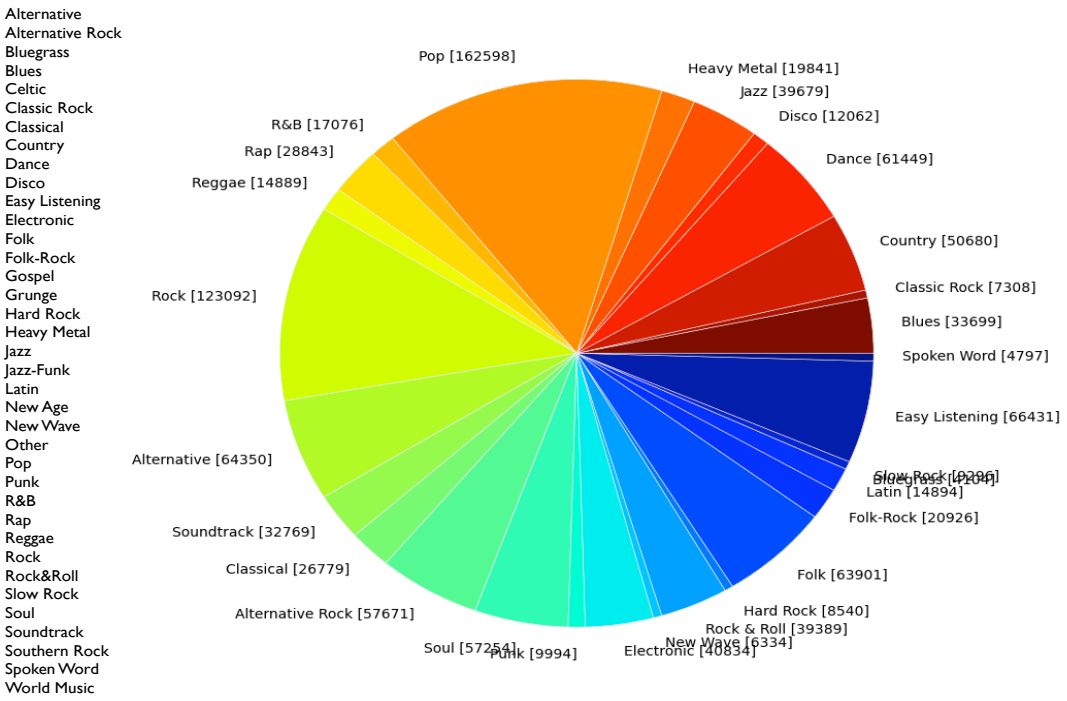


Figure 1: Overview of the I Like Music (ILM) dataset across musical genre. The total number of pieces in each genre is reported in square brackets. ILM's genres are defined according to the ID3 tag standard (`http://id3.org/`).

some preliminary analytical results. In a companion study [11], we developed visualisation algorithms adapted to the analysis of frequent chord progression patterns.

The study of chord progressions presents interests in fields such as musicology, music retrieval and music composition. We hope that in the longer-term such study will contribute to creating additional evidence to existing musicological findings (e.g. "Jazz musicians started to frequently play Lydian augmented chords from the 60s" [12]) and help discovering new ones, for instance to (i) understand the similarities or discrepancies between pieces, composers, and (ii) to study how styles have evolved over time. From a retrieval perspective, many use-cases of chord progressions can be thought of, such as genre and artist recognitions, retrieval of pieces based on a reference chord sequence, the recommendation of pieces for recreational listening or pedagogical applications, etc. The analysis of frequent chord patterns may also be of help to learn music composition and to find out which chord progressions are regularly used, probably because they "hook" listeners (see e.g. Hooktheory's online "Trends" tool[4] which presents data-driven chord transition statistics). This work is part of a larger project, the Digital Music Lab[5] whose goals are to develop research methods and software infrastructure for exploring and analysing large-scale music collections, and to provide researchers and users with datasets and computational tools to analyse music audio, scores and metadata.

The remainder of this article is organised as follows. In Section 2 we review previous computational approaches for the study of chord progressions. Section 3 presents the ILM dataset and our parallel computing technique for audio-based chord extraction. The sequential pattern mining methods developed for chord data are presented in Section 4. In Section 5 we present and discuss some of the obtained results, and in Section 6 we give our conclusions and perspectives.

[4] `http://www.hooktheory.com/trends`
[5] `http://dml.city.ac.uk/`

2. Related works

Most of the previous works on chord progressions have been carried out using symbolic score data. Paiement *et al.* proposed a distributed representation for chords designed such that Euclidean distances roughly correspond to psychoacoustic dissimilarities between chords [13]. The model parameters are learnt based on the expectation-maximisation and junction tree algorithms. The model captures the chords' structure in given musical styles using symbolic MIDI (Musical Instrument Digital Interface) data. The representation is directly tied to the psychoacoustic properties of chords, rather than on their qualities (minor, major, diminished, etc.). This model can be used to interpret the structure of chord progressions and to generate new ones for computational creativity purposes but does not directly address the detection of frequent chord progressions given a large amount of data. Several studies have applied natural language processing (NLP) techniques for chord progression analysis. [3] modeled chord progressions as n-grams and strings in order to conduct automatic genre recognition from symbolic data. Their experiments, conducted on 761 tracks from Popular, Jazz, and Common Practice music, showed that classification rates as high as 85% could be obtained for a three-genre class discrimination problem using a naïve Bayes classifier with the multivariate Bernoulli statistical model. These results, together with those from [5], give evidence that chord progressions characterise some of the commonalities and specificities of musical genres. [4] went further and proposed a model to characterise the harmonic choices of composers using a weighted n-grams of chord progressions. They define as n-gram profile of a chord progression the collection of all chord n-grams appearing in the progression given a weight proportional to the number of beats it lasts. Experiments were conducted on 218 pieces by 8 Jazz composers from various eras using symbolic notations extracted from the Jazz standards' Real Books. Similarity between composers were obtained by computing a cosine-based distance measure between composers' n-gram profiles. The obtained results present a hierarchical clustering of the 8 composers which seem to be consistent with Jazz composition styles from an historical perspective.

The study [2] shares common goals with the present work, i.e. to find frequent chord progressions from empirical data, but relies on symbolic notations rather than audio predictions and only presents results for four-chords progressions. The authors developed a method to extract common subsequences of chord classes from symbolic data, independent of key and context. Frequent chord progressions or so-called "chord idioms" are obtained by ordering the common subsequences by frequency of occurrence in the dataset. In total 424 pieces were analysed from two collections of annotations, Harte's Beatles transcriptions [14] and some of the Real Books. The results show that the prevailing idioms in the Beatles songs present major chords alternating in fifths or fourths, appearing in more than 40% of the songs (e.g. I-IV-I-IV and V-I-V-I.). The most common chord sequences for the Jazz songs follow the circle of fifths (ascending in fourths/descending in fifths), e.g. VI-ii-V-I and vi-ii-V-I, appearing in 28% and 25% of the songs respectively.

Other studies have focused on big music data problems. [15] devised content-based retrieval and algorithmic analysis techniques to parse and index hundreds of thousands of music scores from the *Petrucci Music Library (IMSLP)*. N-grams of melodies up to a length of 16 notes were identified from over 65,000 IMSLP scores. The indexing and metadata processing were run in the Hadoop and HBase environments [16]. A web-based search engine[6] was then built to find melodies that occurred three or more times in scores published or composed during specific years. We are interested in developing a similar search engine for chord progressions as part of the Digital Music Lab project.

3. Big Chord Data Extraction

3.1. Dataset

The I Like Music (ILM) commercial library we analysed comprised 1,293,049 musical pieces from 37 different musical genres with release years ranging from 1907 to 2013. Fig. 1 shows the number of pieces in the ILM dataset in each genre. Additional figures describing the ILM dataset can be found on our *Big Chord Data Extraction and Mining webpage*[7]. Almost all the tracks are encoded in an high quality uncompressed WAV format. Amongst the total number of pieces, about 22% were not available at the time of the analysis (e.g. broken link, track not ingested in the database yet), 0.2% were only available in the MP3 format, and 0.2% were rejected as their durations were either too short or too long (e.g. whole albums such as compilations instead of single piece). The chord feature extraction process described in Section 3.5 was hence conducted on 997,580 pieces in total. The results from Section 5 were obtained for pieces from six genres, Blues (31,618 pieces), Classical (21,446 pieces), Folk (45,194 pieces), Jazz (35,991 pieces), Reggae (13,421 pieces) and Rock'n'roll (36, 654 pieces), corresponding to a total of 209,204 pieces.

3.2. Audio-based automatic chord recognition

A number of approaches have been proposed for the automatic recognition of musical chords from audio signals (see e.g. [17], [18], [19] for comprehensive reviews on the topic). In this study we use an implementation of the model by Mauch described in [7] available as the Vamp plugin 'Chordino'[8]. This model relies on a chroma[9] extraction method using a non-negative least squares (NNLS) algorithm for prior approximate note transcription. The NNLS chroma features achieved a state-of-the-art accuracy of 80% when assessed using the dataset of the MIREX (music information retrieval evaluation exchange) challenge in 2009, mainly composed of Popular songs from the Beatles and Queen[10]. One of the parameters which affects the accuracy of chord recognition models is the number of chord classes (e.g. major, minor, dominant, etc.) which are considered. The results cited above were obtained in the simplified case of two chord classes (major, minor) leading to a total of 24 chords to be discriminated (12 chords per class associated with the 12 distinct semitones of the musical octave in the twelve-tone equal temperament). In the case of 10 chord classes (major, minor, major in first inversion, major in second inversion, major 6th, dominant 7th, major 7th, minor 7th, diminished and augmented), i.e. 120 chords to be discriminated, the accuracy of the Chordino models drops down to 63%. We use a recent version of the model (Chordino Vamp plugin v0.3) which incorporates a dictionary of 16 chord classes (see Tab. 1), leading to 192 different chords in total (as well as an additional "no chord" class). Unfortunately no formal evaluation of this version of the model, nor evaluations for other musical genres than Popular music, are available to date. An obvious drawback of chord progression analysis based on audio-based chord predictions as opposed to professionally-obtained transcriptions comes from the error rates of the audio-based chord recognition models in use. These error rates are far from being negligible, even for a state-of-the-art model, such as the one we use. However we assume that the most frequent patterns emerging from the analysis should be robust to noise (generated by erroneous predictions) given their high frequency of occurrence and the large-scale nature of the dataset.

Table 1: List of chord qualities and specific chord classes for the Chordino Vamp plugin (v0.3). The chord dictionary contains 192 chords in total (16 classes x 12 notes). A list of common harmonic functions for these chords, as defined by the major and minor natural scales are also reported: I (tonic), ii (supertonic), iii (mediant), IV (subdominant), V (dominant) and V7 (dominant 7), vi (submediant), vii, or vii^{o}, or vii^{ø} (leading tone); ^{o}: diminished triad, ^{ø}: diminished triad + b7; subst.: substitute.

Chord Quality	Chord Class	Harmonic funct.
Major	Major	I, IV, V
	Major (1st inversion)	I, IV, V
	Major (2nd inversion)	I, IV, V
	Major 6	I, IV, V
	Major 7	I, IV
	Major 9 (4th inversion, root: 2nd)	I, IV, V
Minor	Minor	ii, iii, vi, vii
	Minor 6	ii
	Minor 7	ii, iii, vi, vii
Dominant 7	Dominant 7	V7
	Dominant 7 (1st inversion)	V7
	Dominant 7 (3rd inversion)	V7
Half-diminished (7b5)	Half-diminished	vii^{ø}
	Half-diminished (3rd inversion)	vii^{ø}
Diminished	Diminished	subst. V7 (major scale), ii^{ø} (minor scale)
Augmented	Augmented	e.g. V^{+}, I^{+}, IV^{+}

3.3. Parallel computing

Given the very large number of musical pieces to analyse (∼1m), we developed a parallel computing approach adapted to audio feature extraction. This represents an extension of our previous work on large-scale music similarity feature extraction [20]. Parallel computing is a form of computation in which many calculations are carried out simultaneously. In a typical parallel program, multiple tasks run on multiple processors or cores. The computation problems are divided into a set of discrete "chunks" of work that can be distributed to the multiple tasks. The parallel computing approach to audio feature extraction we propose follows a single program multiple data (SPMD) architecture, according to Flynn's taxonomy [21]. In a SPMD program, tasks are split up and run simultaneously on multiple processors executing different threads (or processes) with different data input in order to speed up the computation. The chord feature extraction on the million tracks is divided into a set of parallel tasks, each of them working on portions of the dataset, i.e. batches of tracks (domain decomposition). We use a "pool of tasks" scheme as some tasks may run faster than others (the computation time of a task is a function of the track durations which vary). A master process holds the pool of tasks to

[6] `www.peachnote.com`

[7] http://isophonics.net/content/big-chord-data-extraction-and-mining

[8] `http://isophonics.net/nnls-chroma`

[9] Chroma here refers to an audio representation for which the entire spectrum is projected onto 12 bins representing a musical octave.

[10] `http://www.music-ir.org/mirex/wiki/2009:Audio_Chord_Detection`

be distributed to the worker processes and sends the workers a task when requested. The worker processes get tasks from the master process, perform the computation (feature extraction) and produce the results. This approach allows to dynamically balance the load between the workers at run time. The faster the workers the greater the number of tasks they will be able to complete. Feature extraction is performed by using the Vamp plugin host Sonic Annotator[11] together with the Vamp plugin Chordino (see Section 3.2). The features corresponding to a given batch of tracks are written in several output resource files. An important aspect in our program is managing the synchronisation between the parallel tasks at run time. This is important not to mix the audio features which have been extracted over the course of various tasks. Synchronisation is carried out using a lock so that the workers can acquire and then release an output resource (output feature file)[12]. For a given batch, the features are spread into as many output resource files as number of execution threads which are used. We typically set up the number of execution threads to the number of cores available on the server where the program is run.

3.4. Light-weight binary feature format

Sonic Annotator allows to output audio features computed with Vamp plugins in various formats (Comma Separated Values, Resource Description Framework, etc.). With the standard CSV format, the chord annotations provided by the Chordino plugin are expressed as a series of string time stamps and chord labels (e.g. "26.145668934", "Dm7"[13]. We developed a light-weight binary format for Chordino chord annotations. We map each chord label to a single integer value. Hence Chordino's chord dictionary of 193 chords (192 chords and one "no chord" class, see Section 3.2) is represented by 193 integer values. Timestamps are represented by two integer values, one representing the number of seconds, the other the number of nanoseconds. The representation of a chord change at a specific moment in time across a piece is made using only three integer values. We adapted the binary feature writer for Sonic Annotator developed in [20] for such chord features. Once a feature batch has been processed, the binary output resource files produced by the various threads (see Section 3.3) are compressed[14]. The combined use of our binary feature format and of compression led us obtain feature compression ratios of about 4:1. The total size of the chord features for the million tracks in binary compressed format is about 407 MB (approx. 2MB per batch of 5,000 tracks). Given the light-weight nature of the chord feature computed on the server-side, we set up an automatic back up mechanism using the Mercurial version control system provided by our code.soundsoftware.ac.uk resource[15].

3.5. Results

The chord feature extraction process was launched on a virtual machine running Ubuntu 12.04.2 LTS (x86_64 architecture). The virtual machine had 8 cores running at 2.6 MHz with 8 GB of RAM and 12 MB of L3 cache. The processing of the ∼1m tracks took 6 weeks and 2 days. We estimate the speedup (wall-clock time of serial execution / wall-clock time of parallel execution) obtained thanks to our parallelised approach to be about the number of cores (8 in this case) which means that the process would have taken about a year without parallelisation. On a 24-core server, the time taken for the chord extraction process for the million tracks could be reduced to about two weeks.

[11] `http://www.vamp-plugins.org/sonic-annotator/`

[12] In our Python implementation, this is done using the *mutex* module which defines a class that allows mutual-exclusion.

[13] We use a standard letter notation for chords. For instance, Dm7 represents a D minor 7 chord (*m* stands for minor, *7* here implicitly stands for the b7).

[14] In our Python implementation, we used the *bz2* module which provides an interface for the bz2 compression library.

[15] Note that we use this resource in the first instance as a code repository which is the main purpose of the code.soundsoftware.ac.uk platform.

4. Big Chord Data Mining

4.1. Sequential Pattern Mining of Chord Progressions

In order to apply sequential pattern mining we first convert the audio-predicted chord annotations (list of time and chord labels, see Section 3.2) into a chord sequence (e.g. A7, D7, G7, etc.) database. The chord sequence database comprises the chord sequences for each piece from the ILM dataset (see Section 3.1). The normalised (or percentage) support of a sequential pattern is the number of sequences where the pattern occurs divided by the total number of sequences in the sequence database [22]. Open sequential patterns are patterns whose elements can be separated by other items in the sequence database (e.g. in the sequence A7-Bmin7-D7-G7, A7-D7-G7 is an open pattern). Closed sequential patterns are sequential patterns which are not strictly included in other patterns having the same support. As audio-predicted chord annotations are error-prone, we chose a technique looking for open rather than closed sequential patterns. Indeed, if the prediction of a sequence including a specific chord progression pattern (e.g. CP: A7-D7-G7) contained wrong chords between the correct ones (e.g. CS1: A7-N-D7-G7, where "N" refers to the non chord state from the model), and that the nature of these errors differed for other sequences including that pattern (e.g. CS2: A7, N, D7, N, G7), then a closed sequential pattern technique would not uncover the pattern of interest. Conversely, open sequential pattern techniques will uncover the relevant chord pattern independently of errors that may occur between the chords: in both chord sequences CS1 and CS2 above, the chord pattern CP will be detected with an open sequential pattern technique.

We applied the CM-SPADE sequential pattern mining algorithm for open patterns [10][16] to the chord sequence databases corresponding to the six genres described in Section 3.1 (Blues, Classical, Folk, Jazz, Reggae and Rock'n'roll). The algorithm yields a list of frequent chord patterns ranked according to their support. As we are interested in comparing the relative importance of the patterns across genres, we computed the percentage support for each pattern based on the total number of pieces in each genre.

4.2. Chord Progression Pattern Modeling

We aim to uncover frequent chord progression patterns independently of the key of the musical pieces. However we don't know the keys of the pieces from the ILM dataset. To tackle this issue, we post-processed the output of the CM-SPADE algorithm based on the following modeling. As in [2], we model a chord by a tuple (b_i, c_i) where $b_{i\in\{0,\ldots,11\}}$ refers to the bass note in semitones above the chord root (e.g. 0 for a bass note corresponding to the chord root, 4 for a bass note corresponding to the major third, etc.), and $c_{i\in\{0,\ldots,16\}}$ refers to the chord class, as described in Tab. 1 (0 corresponds to the non chord state, 1 to major, etc.). We model a chord transition by a tuple (($t^1 = (b_i^1, c_i^1)$, $t^2 = (b_i^2, c_i^2)$, d^i)) where t^1 and t^2 are the first and second chord tuples, and $d^{i\in\{0,\ldots,11\}}$ refers to the root differences in semitones (modulo 12) between the two chords. We model a chord progression by a sequence of chord transitions ((t^1,t^2,d^1), (t^2,t^3,d^2), ..., (t^n,t^{n+1},d^n)), where the length of the chord progression is given by n+1. By using such chord progression modeling, we are able to group together progressions with identical changes of chord class and root key. For instance, Dm-G-C and Em-A-D will both correspond to a chord progression of type: min $\xrightarrow{+fourth}$ maj $\xrightarrow{+fourth}$ maj, and which may be a ii-V-I progression (however we can't be certain of the latter as we don't have the key).

5. Frequent Chord Progression Patterns

5.1. Chord Progression Length and No of Frequent Patterns

Chord progressions may differ by their length and complexity. Amongst common progressions some only exhibit a few chord transitions, e.g. four chord progressions such as I-bVII-IV-IV (e.g.

[16] We used the implementation of the CM-SPADE algorithm provided by the SPMF toolbox available at: `http://www.philippe-fournier-viger.com/spmf/index.php`.

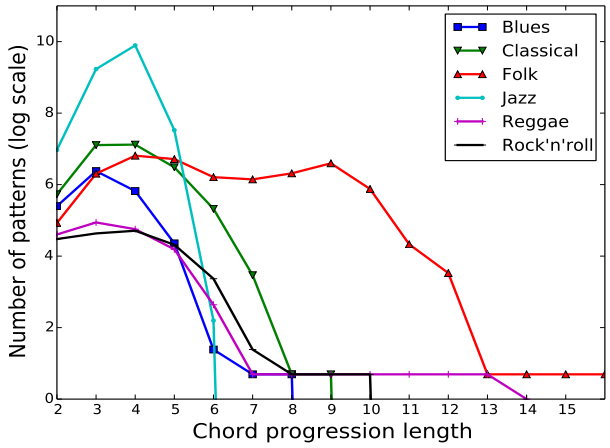


Figure 2: Number of frequent chord progression patterns as a function of the chord progression length in each genre.

Skynyrd's *"Sweet Home Alabama"*) and I-IV-V-IV (e.g. Taylor's *"Wild Thing"*), whilst others are built on a larger number of chord transitions, e.g. the sixteenth chord progression I-vi / ii-V / I-vi / ii-V / I-I7 / IV-iv / I-V / I-I (e.g. Gershwin's *"I Got Rhythm"*). In this study, we seek to uncover frequent chord patterns of different lengths, based on a set of lengths defined *a priori* (the minimal and maximal lengths of the patterns are input parameters of the CM-SPADE algorithm). We looked for frequent patterns of lengths ranging from 2 to 16. Fig. 2 shows the number of frequent patterns uncovered for each of these lengths for the six genres analysed in this study. The number of possible patterns for a given chord progression length is related to the variation of chord qualities and chord root keys in the progressions. The pattern frequency curves all have a bell-shape form with a maxima obtained for lengths of 3 or 4 chords. 3- and 4-chord progressions have been shown to be common in many genres (some interesting examples can be found on the Hooktheory website[17] for instance). The shape of the curves also demonstrates that the space of chord progressions is constrained by underlying rules as a curve counting all possible chord combinations as length increases would grow exponentially and rapidly tend towards infinity for 192 possible chords. For 4-chord progressions, the total number of obtained frequent patterns across genres ranks as follows: Jazz (19,820), Classical (1,235), Folk (906), Blues (337), Reggae (116) and Rock'n'roll (111). These results tend to show that a more diverse set of chords (and hence chord changes) is used in Jazz compared to the other genres. This is likely to be due to the fact that Jazz progressions often include chords with a tetrad (e.g. C7, Fmaj7) in addition to chords with a triad (e.g. C, F) increasing the number of possible patterns.

5.2. Genre-Based Frequent Patterns

Tables listing the 50 most frequent 4-chord patterns for ILM pieces in the Blues, Classical, Folk, Jazz, Reggae and Rock'n'roll genres are available on our *Big Chord Data Extraction and Mining webpage*. In order to facilitate the interpretation of the frequent chord progression patterns obtained with sequential pattern mining we developed a method to represent them based on directed graphs: nodes correspond to chord classes (e.g. major, minor, etc.) and oriented edges characterise the transition between chord classes. We express the key-invariant patterns (see Section 4.2) in the DOT graph description language and use the Graphviz visualisation software to generate the graphs. We represent the pattern percentage support (see Section 4.1) by the thickness of the edges (hence, the higher the support, the thicker the edge). Fig. 3 shows the graphs corresponding to the 100 most frequent 4-chord progressions in the Blues, Classical and Jazz genres for our audio-predicted chord sequences. Additional graphs can be visualised online on our *Big Chord Data Extraction and Mining webpage*. It can be seen from the tables and the graphs available online that the prevailing 4-chord patterns for the six genres are major chords alternating in fifths and fourths (+7 and + 5 semitones, respectively): (a) $maj \xrightarrow{+7} maj \xrightarrow{+5} maj \xrightarrow{+7} maj$ and (b) $maj \xrightarrow{+5} maj \xrightarrow{+7} maj \xrightarrow{+5} maj$. These progressions occurs in a very large proportion of the pieces: Blues (44%), Classical (14%), Folk (28%), Jazz (43%), Reggae (37%) and Rock'n'roll (68%). This in line with the results obtained from symbolic data for Popular and Jazz pieces in [2]. Although we don't know the key, it is very likely that progression (a) corresponds to a I-V-I-V progression, i.e. an alternation of perfect (V-I) and half cadences (I-V)[18], and progression (b) corresponds to a I-IV-I-IV progression, which includes a plagal cadence (IV-I). These progressions are not discriminative though as they appear in every genre. The progression alternating major and dominant chords in intervals of fourths and fifths, which is presumably a I7-IV7-I7-IV7 progression, is characteristic of the Blues genre (*pattern #45*, 15% of the pieces) as it's not one of the 50 most frequent patterns in other genres. Progressions including the relative minor (a minor third below the tonic) are more frequent in the Classical genre than in other genres (e.g. the *pattern #5*, I-vi-V-I, appears in 10% of the Classical music pieces). Progressions including tetradic minor 7 and major 7 chords are more frequent in the Jazz genre (see Fig. 3c).

6. Conclusions and Future Work

In this study we propose a data-driven methodology for the analysis of frequent chord progressions from large-scale audio music collections. We developed a parallel computing software for audio feature extraction reducing the overall processing time by a factor roughly equal to the number of execution cores. This software was used together with the Chordino audio-based chord prediction model [7] in order to extract a million chord progressions for pieces from the I Like Music commercial musical library. Frequent chord progression patterns were uncovered using one of the state-of-the-art's sequential pattern mining algorithm (CM-SPADE [10]). Preliminary results obtained from the analysis of over 200,000 pieces from six musical genres highlighted 4-chord progressions that prevail in every genre, e.g. progressions with major chords alternating in fifths and fourths. Other progressions act as signatures of specific genres, e.g. frequent progressions with dominant 7 chords in Blues, with the relative minor in Classical music, etc.
In future work we aim to extend this study in several ways. We plan to extract additional features from audio using our parallel computing software in order to facilitate the interpretation of chord progressions; having knowledge of the key would for instance make functional harmony analysis possible, meter and tempo information would help characterising the rhythmic structure of the chord sequences. We also wish to develop other collection-level analysis techniques to investigate how chord progressions evolve over time within musical styles. Some of these analysis tools will hopefully be made available for users via an online platform which will include interactive visualisations such as that proposed in our companion study [11].

7. Acknowledgements

This project has been partly funded by the AHRC project *Digital Music Lab - Analysing Big Music Data* (grant AHL01016X1).

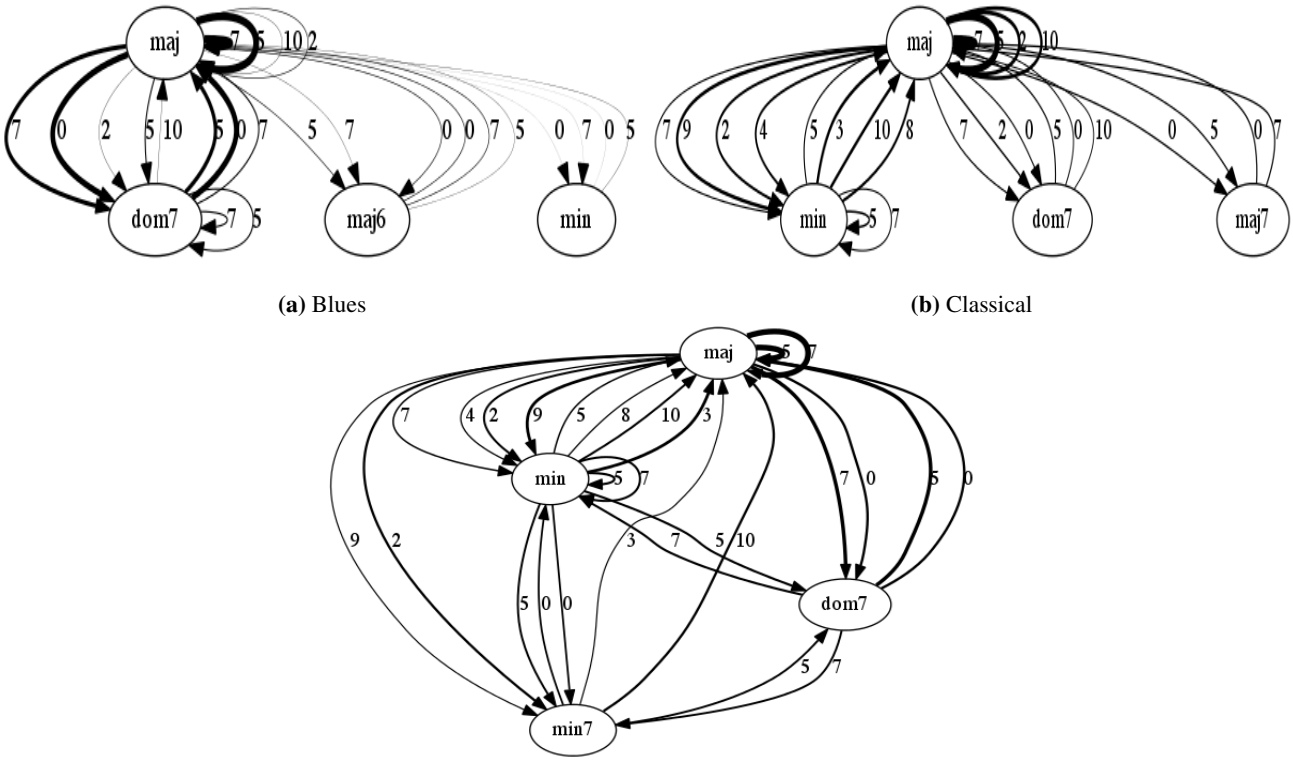
References

[1] W. Piston: *Harmony*. Norton, W. W. & Company, Inc., 5. edition, 1987.

[2] M. Mauch, S. Dixon, C. Harte, M. Casey, and B. Fields: *Discovering Chord Idioms Through Beatles and Real Book Songs*. In *Proc. of the Int. Society for Music Information Retrieval (ISMIR) Conference*. 2007.

[17] `http://www.hooktheory.com/theorytab/common-chord-progressions`

[18] For a definition of cadences, see e.g. `http://en.wikipedia.org/wiki/Cadence_(music)`



(a) Blues

(b) Classical

(c) Jazz

Figure 3: Graph representations of the 100 most frequent 4-chord progressions in the ILM dataset for various genres. The nodes represent the chord classes predicted from audio (Chordino Vamp plugin). The oriented edges describe the transitions between two chords. The thickness of the edges is proportional to the pattern support. The interval between chord roots is expressed in number of semitones above the chord root of the first chord and reported next to the edges (e.g. 5 semitones corresponds to a 4th, 7 semitones to a 5th, etc.).

[3] C. Perez-Sancho, D. Rizo, and J. M. Inesta: *Genre classification using chords and stochastic language models*. In *Connection Science*, volume 0(0):1–13, 2008.

[4] M. Ogihara and T. Li: *N-gram chord profiles for composer style representation*. In *Proc. of the International Society for Music Information Retrieval (ISMIR) Conference*, pages 671–676. 2008.

[5] A. Anglade, E. Benetos, M. Mauch, and S. Dixon: *Improving Music Genre Classification Using Automatically Induced Harmony Rules*. In *Journal of New Music Research*, volume 39(4):349–361, 2010.

[6] E. Clarke and N. Cook: *Empirical Musicology: Aims, Methods, Prospects*. Oxford University Press, 2008.

[7] M. Mauch and S. Dixon: *Approximate Note Transcription for the Improved Identification of Difficult Chords*. In *Proc. of the Int. Society for Music Information Retrieval (ISMIR) Conference*. 2010.

[8] C. H. Mooney and J. F. Roddick: *Sequential Pattern Mining - Approaches and Algorithms*. In *ACM Computing Surveys (CSUR)*, volume 45(2), 2013.

[9] M. Esmaeili and F. Gabor: *Finding Sequential Patterns from Large Sequence Data*. In *IJCSI International Journal of Computer Science Issues*, volume 7(1), 2010.

[10] P. Fournier-Viger, A. Gomariz, M. Campos, and R. Thomas: *Fast Vertical Mining of Sequential Patterns Using Co-occurrence Information*. In *PAKDD Part I*, LNAI 8443, pages 40–52. Springer, 2014.

[11] A. Kachkaev, D. Wolff, M. Barthet, M. D. Plumbley, J. Dykes, and T. Weyde: *Visualising Chord Progressions in Music Collections: A Big Data Approach*. In *Proc. of the 9th Conference on Interdisciplinary Musicology (CIM)*. 2014.

[12] M. Levine: *The Jazz Theory Book*. Sher Music, 1995.

[13] J.-F. Paiement, D. Eck, S. Bengio, and D. Barber: *A Graphical Model for Chord Progressions Embedded in a Psychoacoustic Space*. In *Proc. of the 22nd International Conference on Machine Learning*. 2005.

[14] C. Harte, M. Sandler, S. A. Abdallah, and E. Gomez: *Symbolic representation of musical chords: A proposed syntax for text annotations*. In *Proc. of the Int. Society for Music Information Retrieval (ISMIR) Conference*. 2005.

[15] V. Viro: *Peachnote: Music score search and analysis platform*. In *Proc. of the Int. Society for Music Information Retrieval (ISMIR) Conference*. 2011.

[16] J. Dean and S. Ghemawat: *MapReduce: Simplified data processing on large clusters*. In *ACM*, volume 51(1):107–113, 2008.

[17] K. Lee and M. Slaney: *Automatic Chord Recognition from Audio Using an HMM with Supervised Learning*. In *Proc. of the 1st ACM workshop on audio and music computing multimedia*. 2006.

[18] C. Harte: *Towards Automatic Extraction of Harmony Information from Music Signals*. Ph.D. thesis, Queen Mary University of London, 2010.

[19] M. Mauch: *Automatic Chord Transcription from Audio Using Computational Models of Musical Context*. Ph.D. thesis, Queen Mary University of London, 2010.

[20] G. Fazekas, M. Barthet, and M. Sandler: *Demo paper: The BBC Desktop Jukebox music recommendation system: A large scale trial with professional users*. In *Multimedia and Expo Workshops (ICMEW), 2013 IEEE International Conference on*, pages 1–2. 2013.

[21] M. J. Flynn: *Some Computer Organizations and Their Effectiveness*. In *IEEE Trans. Comput.*, volume C–21(9):948–960, 1972.

[22] D. Perera, J. Kay, I. Koprinska, K. Yacef, and O. Zaiane: *Clustering and Sequential Pattern Mining of Online Collaborative Learning Data*. In *IEEE Trans. on Knowledge and Data Engineering*, volume 21(6):759–772, 2009.