Comparing Harmonic Similarity Measures

W. Bas de Haas[1], Matthias Robine[2], Pierre Hanna[2], Remco C. Veltkamp[1], and Frans Wiering[1]

[1] Utrecht University, Department of Information and Computing Sciences
Padualaan 14, 3584 CH, Utrecht, The Netherlands
{bas.dehaas,remco.veltkamp,frans.wiering}@cs.uu.nl
[2] LaBRI - Université de Bordeaux
F-33405 Talence cedex, France
{pierre.hanna,matthias.robine}@labri.fr

Abstract. We present an overview of the most recent developments in polyphonic music retrieval and an experiment in which we compare two harmonic similarity measures. In contrast to earlier work, in this paper we specifically focus on the symbolic chord description as the primary musical representation and the similarity between sequences of these descriptions. In the experiment we compare a geometrical and an alignment approach to harmonic similarity, and measure the effects of chord description detail and a priori key information on retrieval performance. For this experiment a large new chord sequence corpus is assembled. The results show that a computational costly alignment approach significantly outperforms a much faster geometrical approach in most cases, that a priori key information boosts retrieval performance, and that using a triadic chord representation yields significantly better results than using more simple or more complex chord representations.

Key words: MIR, Harmony, Polyphony, Chord, Similarity, Evaluation, Ground-truth data.

1 Introduction

In the last two decades Music Information Retrieval (MIR) has evolved into a broad research area in which two main directions can be discerned: symbolic music retrieval and the retrieval of musical audio. The first direction traditionally uses score-based representations and musical structures that resemble notes to tackle typical retrieval problems. One of the most important and most intensively studied of these is probably the problem of determining the similarity of a specific musical feature, e.g. melody, rhythm, etc. The second direction–musical audio retrieval–extracts features from the audio signal and uses these features for estimating whether two pieces of music share certain musical properties. In this paper we focus on a musical representation that is symbolic but can be estimated reasonably well from audio: chord sequences.

Only recently, partly motivated by the growing interest in audio chord finding, MIR researchers have started using chords descriptions as principal representation for modeling music similarity. Naturally, these representations are

ISBN: 978-84-614-0853-5
7th International Symposium on Computer Music Modeling and Retrieval
Málaga (Spain), June 21-24, 2010

specifically suitable for capturing the harmonic similarity of a musical piece. However, determining the harmonic similarity of sequences of chords gives rise to three questions. First, what is harmonic similarity? Second, why do we need harmonic similarity? Last, do chord descriptions provide a valid and useful abstraction of the musical data for determining music similarity? The first two questions we will address in this introduction; the third question we will answer empirically in a retrieval experiment. In this experiment we will compare a geometrical and an alignment based harmonic similarity measure.

The first question–what is harmonic similarity–is difficult to answer. We strongly believe that if we want to model what makes two pieces of music similar, we must not only look at the musical data, but especially at the human listener. It is important to realize that music only becomes music in the mind of the listener, and probably not all information needed for good similarity judgment can be found in the data alone. Human listeners, musician or non-musician, have extensive culture-dependent knowledge about music that needs to be taken into account when judging music similarity.

In this light we consider the harmonic similarity of two chord sequences to be the degree of agreement between structures of simultaneously sounding notes and the agreement between global as well as local relations between these structures in both sequences as perceived by the human listener. With the agreement between structures of simultaneously sounding notes we denote the similarity that a listener perceives when comparing two chords in isolation and without surrounding musical context. However, chords are rarely compared in isolation and the relations between the global context–the key–of a piece and the relations to the local context play a very important role in the perception of tonal harmony. The local relations can be considered the relations between functions of chords within a limited time frame, for instance the preparation of a chord with a dominant function with a sub-dominant. All these factors play a role in the perception of tonal harmony and should be shared by two compared pieces up to certain extent to if they are considered similar.

The second question about the usefulness of harmonic similarity is easier to answer, since music retrieval based on harmony sequences offers various benefits. It allows for finding different versions of the same song even when melodies vary. This is often the case in cover songs or live performances, especially when these performances contain improvisations. Moreover, playing the same harmony with different melodies is an essential part of musical styles like jazz and blues. Also, variations over standard basses in baroque instrumental music can be harmonically very related.

The application of harmony matching methods is broadened further by the extensive work on chord label extraction from musical audio data within the MIR community, e.g. [1, 2]. Chord labeling algorithms extract symbolic chord labels from musical audio: these labels can be matched directly using the algorithms covered in this paper.

If you would ask a jazz musician to answer the third question–whether chord descriptions are useful–he will probably agree that they are, since working with

chord labels is everyday practice in jazz. However, we will show in this paper that they are also useful for retrieving pieces with a similar but not identical chord sequence by performing a large experiment. In this experiment we compare two harmonic similarity measures, the Tonal Pitch Step Distance (TPSD) [3] and the Chord Sequence Alignment System (CSAS) [4], and test the influence of different degrees of detail in the chord description and the knowledge of the global key of a piece on retrieval performance.

The next section gives a brief overview of the current achievements in chord sequence similarity matching and harmonic similarity in general, Section 3 describes the data used in the experiment and Section 4 presents the results.

Contribution. This paper presents an overview of chord sequence based harmonic similarity and two harmonic similarity approaches are compared in an experiment. For this experiment a new large corpus of 5028 chord sequences is assembled and both algorithms are subjected to six tasks. All tasks use the same dataset, but differ in the amount of chord detail and in the use of a priori key information. The results show that a computational costly alignment approach significantly outperforms a much faster geometrical approach in most cases, that a priori key information boosts retrieval performance, and that using a triadic chord representation yields significantly better results than using more simple or more complex chord representations.

2 Overview: Similarity Measures for Chord Sequences

The harmonic similarity of symbolic music has been investigated by many authors, but the number of systems that focus solely on similarity chords sequences is much smaller. Of course it is always possible to convert notes into chords and vice versa, but this is not a trivial task. Nowadays, several algorithms can correctly segment and label approximately 80 percent of a symbolic dataset (see for a review [5]). Within the audio domain hidden Markov Models are frequently used for chord label assignment, e.g. [1, 2]. The algorithms considered in this paper abstract from these labeling tasks and focus on the similarity between chord progressions only. As a consequence, we assume that we have a sequence of symbolic chord labels describing the chord progression in a piece of music.

The systems currently known to us that are designed to match these sequences of symbolic chord descriptions are: the TPSD [3], the CSAS [4] and a harmony grammar approach [6]. The first two are quantitatively compared in this paper and are introduced in the next two subsections, respectively. They have been compared before, but all previous evaluations of TPSD and CSAS were done with relatively small datasets (<600 songs) and the data in [4] did not match the one used in [3]. The harmony grammar approach could, at the time of writing, not compete in this experiment because in its current state it is yet unable to parse all the songs in the used dataset. We expect this issue to be resolved in the near future.

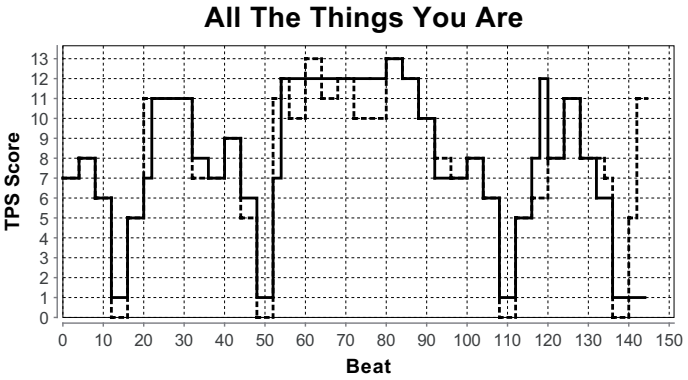

Fig. 1. A plot demonstrating the comparison of two similar versions of *All the Things You Are* using the TPSD. The total area between the two step functions, normalized by the duration of the shortest song, represents the distance between both songs. A minimal area is obtained by shifting one of the step functions cyclically.

The next section introduces the TPSD and the improvements over the implementation used for the experiment here and the implementation in [3]. Section 2.2 highlights the different variants of the CSAS. The main focus of this paper is on the similarity of sequences of chord labels, but there exist other relevant harmony based retrieval methods: some of these are briefly reviewed in Section 2.3.

2.1 Tonal Pitch Step Distance

The TPSD uses Lerdahl's [7] Tonal Pitch Space (TPS) as its main musical model. TPS is a model of tonality that fits musicological intuitions, correlates well with empirical findings from music cognition [8] and can be used to calculate a distance between two arbitrary chords. The TPS model can be seen as a scoring mechanism that takes into account the number of steps on the circle of fifths between the roots of the chords, and the amount of overlap between the chord structures of the two chords and their relation to the global key.

The general idea behind the TPSD is to use the TPS to compare the change of chordal distance to the tonic over time. For every chord the TPS distance between the chord and the key of the sequence is calculated, which results in a step function (see Figure 1). As a consequence, information about the key of the piece is essential. Next, the distance between two chord sequences is defined as the minimal area between the two step functions over all possible horizontal circular shifts. To prevent that longer sequences yield larger distances, the score is normalized by dividing it by the duration of the shortest song.

The TPS is an elaborate model that allows to compare every arbitrary chord in an arbitrary key to every other possible chord in any key. The TPSD does not use the complete model and only utilizes the parts that facilitate the comparison

of two chords within the same key. In the current implementation of the TPSD time is represented in beats, but generally any discrete representation could be used.

The TPSD version used in this paper contains a few improvements compared to the version used in [3]: by applying a different step function matching algorithm from [9], and by exploiting the fact that we use discrete time units that enable us to sort in linear time using counting sort, a running time of $O(nm)$ is achieved where n and m are the number of chord symbols in both songs. Furthermore, to be able to use the TPSD in situations where a priori key information is not available, the TPSD is extended with a key finding algorithm.

Key finding. The problem of finding the global key of piece of music is called key finding and in the current case this is done on the basis of chord information only. The rationale behind the key finding algorithm that we present here is the following: we consider the key that minimizes the total TPS distance and best matches the starting and ending chord, the key of the piece.

For minimizing the total TPS distance, the TPSD key finding uses TPS based step functions as well. We assume that when a song matches a particular key, the TPS distances between the chords and the tonic of the key are relatively small. The general idea is to calculate 24 step functions for a single chord sequence, one for each major and minor key. Subsequently, all these keys are ranked by sorting them on the area between their TPS step function and the x-axis; the smaller the total area, the better this key fits the piece, and the higher the rank. Often, the key at the top rank is the correct key. However, among the false positives at rank one, non-surprisingly, the IV, V and VI relative to the ground-truth key[1] are found regularly. This makes sense because, when the total of TPS distances of the chords to C is small, the distances to F, G and Am might be small as well. Therefore, to increase performance, an additional scoring mechanism is designed that takes into account the IV, V and VI relative to the ground-truth key. Of all 24 keys, the candidate key that minimizes the following sum S is considered the key of the piece.

$$S = \alpha r(I) + r(IV) + r(V) + r(VI) + \begin{cases} \beta \text{ if the first chord matches the key, and} \\ \gamma \text{ if the last chord matches the key} \end{cases}$$

Here $r(.)$ denotes the rank of the candidate key, a parameter α determines how important the tonic is compared to the other frequently occurring scale degrees, and β and γ control the importance of the key matching the first and last chord, respectively. The parameters were tuned by hand and an α of 2, a β of 4, and a γ of 4, were found to give good results. Clearly, this simple key-finding algorithm is biased towards western diatonic music, but for the corpus used in this paper it performs quite well. The algorithm scores 88.8 percent correct on

[1] The roman numbers here represent the diatonic interval between the key in the ground-truth and the predicted key.

a subset of 500 songs of the corpus used in the experiment below for which we manually checked the correctness of the ground-truth key. The above algorithm takes $O(n)$ time, where n is the number of chord symbols, because the number of keys is constant.

Root interval step functions. For the tasks where only the chord root is used we use a different step function representation (See Section 4). In these tasks the interval between the chord root and the root note of the key defines the step height and the duration of the chord again defines the step length. This matching method is very similar to the melody matching approach by Aloupis et al. [10]. Note that the latter was never tested in practice. The matching and key finding methods are not different from the other variants of the TPSD. Note that in all TPSD variants chord inversions are ignored.

2.2 Chord Sequence Alignment System

The CSAS algorithm is based on local alignment and computes similarity scores between sequences of symbols representing chords or distances between chords and key. Approximate string matching techniques allow to quantify the differences between two such sequences. Among several existing methods, Smith and Waterman's approach [11] consists of detecting local similar areas between two sequences of symbols. This *local alignment* or *local similarity* algorithm locates and extracts a pair of regions, one from each of the two given strings, that exhibit high similarity. A similarity score is calculated by performing elementary operations transforming the one string into the other. The operations used to transform the sequences are deletion or insertion of a symbol, and substitution of a symbol by another. The total transformation from the one string into the other can be solved with a dynamic programming in quadratic time.

The following example illustrates local alignment by computing a distance between the first chords of two variants of the song *All The Things You Are* considering only the root notes:

string 1	_	F	B♭	E♭	A♭	_	D♭	D	G	C	C
string 2	F	F	B♭	A	A♭	A♭	D♭	D	_	C	C
operation	I	M	M	S	M	I	M	M	D	M	M
score	−1	+2	+2	−2	+2	−1	+2	+2	−1	+2	+2

Algorithms based on local alignment have been successfully adapted for melodic similarity [12–14] and recently it has been used to determine harmonic similarity [4] as well. Two steps are necessary to apply the alignment technique to the comparison of chord progressions: the choice of the representation of a chord sequence, and costs of the elementary operations between symbols. To take the durations of the chords into account, we represent the chords at every beat. The algorithm has therefore a complexity of $O(nm)$, where n and m are the sizes of the compared songs in beats. The cost function can either be adapted to the

chosen representation or can simply be binary, i.e. the cost is positive (+2) if the two chords described are identical, and negative (−2) otherwise. The insertion or deletion cost is set to −1.

Absolute representation. One way of representing a chord sequence is to simply represent the chord progression as a sequence of absolute root notes and in that case prior knowledge of the key is not required. An absolute representation of the chord progression of the 8 first bars of the song *All The Things You Are* is then:

$$F, B\flat, E\flat, A\flat, D\flat, D, G, C, C$$

In this case, the substitution costs may be determined by considering the difference in semitones, the number of steps on the circle of fifths between the roots, or by the consonance of the interval between the roots, as described in [13]. For instance, the cost of substituting a C with a G (fifth) is lower than the substitution of a C with a D (Second). Taking into account the mode in the representation can affect the cost function as well: a substitution of a C for a Dm is different from a substitution of a C for a D, for example. If the two modes are identical, one may slightly increase the similarity score, and decrease it otherwise. Another possible representation of the chord progression is a sequence of absolute pitch sets. In that case one can use musical distances between chords, like Lerdahl's TPS model [7] or the distance introduced by Paiement et al. [15], as a cost function for substitution.

Key-relative representation. If key information is known beforehand, a chord can be represented as a distance to this key. The distances can be expressed in various ways: in semitones, or as the number of fifths between the roots of the chords and the tonic of the key of the song, or with more complex musical models, such as TPS. If in this case the key is A♭ and the chord is represented by the difference in semitones, the representation of the chord progression of the first eight bars of the song *All The Things You Are* will be:

$$3, 2, 5, 0, 5, 6, 1, 4, 4$$

If all the notes of the chords are taken into account, the TPS or Paiement distances can be used between the chords and the triad of the key to construct the representation. The representation is then a sequence of distances, and we use an alignment between these distances instead of between the chords themselves. This representation is very similar to the representation used in the TPSD. The cost functions used to compare the resulting sequences can then be binary, or linear in similarity regarding the difference observed in the values.

Transposition invariance. In order to be robust to key changes, two identical chord progressions transposed in different keys have to be considered as similar. The usual way to deal with this issue [16] is to choose a chord representation which is transposition invariant. A first option is to represent transitions

between successive chords, but this has been proven to be less accurate when applied to alignment algorithms [13]. Another option is to consider a key relative representation, like the representation described above which is by definition transposition invariant. However, this approach is not robust against local key changes. With an absolute representation of chords, we use an adaptation of the local alignment algorithm proposed in [17]. It allows to take into account an unlimited number of local transpositions and can be applied to representations of chord progressions to account for modulations.

According to the choice of the representation and the cost function, several variants are possible in order to build an algorithm for harmonic similarity. In Section 4 we explain the different representations and scoring functions used in the different tasks of the experiment and their effects on retrieval performance.

2.3 Other Methods for Harmonic Similarity

The third harmonic similarity measure using chord descriptions is a generative grammar approach [6]. The authors use a generative grammar of tonal harmony to parse the chord sequences, which results in parse trees that represent harmonic analyses of these sequences. Subsequently, a tree that contains all the information shared by the two parse trees of two compared songs is constructed and several properties of this tree can be analyzed yielding several similarity measures. Currently a parser can reject a sequence of chords as being ungrammatical. For this reason, not all pieces can be parsed and therefore this harmonic approach cannot be evaluated here.

Another interesting retrieval system based on harmonic similarity is the one developed by Pickens and Crawford [18]. Instead of describing a musical segment with one chord, they represent a musical segment as a vector describing the 'fit' between the segment and every major and minor triad. This system then uses a Markov model to model the transition distributions between these vectors for every piece. Subsequently, these Markov models are ranked using the Kullback-Liebler (KL) divergence. It would be interesting to compare the performance of these systems to the algorithms tested in here in the future.

3 A Chord Sequence Corpus

The Chord Sequence Corpus used in the experiment consists of 5,028 unique human-generated Band-in-a-Box files that are collected from the Internet. Band-in-a-Box is a commercial software package [19] that is used to generate musical accompaniment based on a lead sheet. A Band-in-a-Box file stores a sequence of chords and a certain style, whereupon the program synthesizes and plays a MIDI-based accompaniment. A Band-in-a-Box file therefore contains a sequence of chords, a melody, a style description, a key description, and some information about the form of the piece, i.e. the number of repetitions, intro, outro etc. For extracting the chord label information from the Band-in-a-Box files we have extended software developed by Simon Dixon and Matthias Mauch [20].

|Fm7 . . .	|Bbm7 . . .	|Eb7 . . .	|AbMaj7 . . . |
|DbMaj7 . . .	|Dm7b5 . G7b9 .	|CMaj7 . . .	|CMaj7 . . . |
|Cm7 . . .	|Fm7 . . .	|Bb7 . . .	|Eb7 . . . |
|AbMaj7 . . .	|Am7b5 . D7b9 .	|GMaj7 . . .	|GMaj7 . . . |
|A7 . . .	|D7 . . .	|GMaj7 . . .	|GMaj7 . . . |
|Gbm7 . . .	|B7 . . .	|EMaj7 . . .	|C+ . . . |
|Fm7 . . .	|Bbm7 . . .	|Eb7 . . .	|AbMaj7 . . . |
|DbMaj7 . . .	|Dbm7 . Gb7 .	|Cm7 . . .	|Bdim . . . |
|Bbm7 . . .	|Eb7 . . .	|AbMaj7 . . .	|. . . . |

Table 1. A leadsheet of the song *All The Things You Are*. A dot represents a beat, a bar represents a bar line, and the chord labels are presented as written in the Band-in-a-Box file.

Throughout this paper we have been referring to chord labels or chord descriptions. To rule out any possible vagueness, we adopt the following definition of a chord: a chord always consist of a root, a chord type and an optional inversion. The root note is the fundamental note upon which the chord is built, usually as a series of ascending thirds. The chord type (or quality) is the set of intervals relative to the root that make up the chord and the inversion is defined as the degree of the chord that is played as bass note. One of the most distinctive features of the chord type is its mode, which can either be major or minor.

Although a chord label always describes these three properties, root, chord type and inversion, musicians and researchers use different syntactical systems to describe them, and also Band-in-a-Box uses its own syntax to represent the chords. Harte et al. [21] give an in depth overview of the problems related to representing chords and suggests a unambiguous syntax for chord labels[2]. An example of a chord sequence as found in a Band-in-a-Box file describing the chord sequence of *All the Things You Are* is given in Table 1.

All songs of the chord sequence corpus were collected from various Internet sources. These songs were labeled and automatically checked for having a unique chord sequence. All chord sequences describe complete songs and songs with fewer than 3 chords or shorter than 16 beats were removed from the corpus in an earlier stage. The titles of the songs, which function as a ground-truth, as well as the correctness of the key assignments, were checked and corrected manually. The style of the songs is mainly jazz, latin and pop.

Within the collection, 1775 songs contain two or more similar versions, forming 691 classes of songs. Within a song class, songs have the same title and share a similar melody, but may differ in a number of ways. They may, for instance, differ in key and form, they may differ in the number of repetitions, or have a special introduction or ending. The richness of the chords descriptions may also diverge, i.e. a $C^{7\flat9\flat13}$ may be written instead of a C^7, and common sub-

[2] The software that was used to read the Band-in-a-Box files is also capable of translating the Band-in-a-Box chord label into a label according to syntax proposed in Harte et al. [21].

Class Size	Frequency	Percent
1	3,253	82.50
2	452	11.46
3	137	3.47
4	67	1.70
5	25	.63
6	7	.18
7	1	.03
8	1	.03
10	1	.03
Total	5028	100

Table 2. The distribution of the song class sizes in the Chord Sequence Corpus.

stitutions frequently occur. Examples of the latter are relative substitution, i.e. Am instead of C, or tritone substitution, e.g. $F\#^7$ instead of C^7. Having multiple chord sequences describing the same song allows for setting up a *cover-song* finding experiment. The the title of the song is used as ground-truth and the retrieval challenge is to find the other chord sequences representing the same song.

The distribution of the song class sizes is displayed in Table 2 and gives an impression of the difficulty of the retrieval task. Generally, Table 2 shows that the song classes are relatively small and that for the majority of the queries there is only one relevant document to be found. It furthermore shows that 82.5% of the songs is in the corpus for distraction only. The chord sequence corpus is available to the research community on request.

4 Experiment: Comparing Retrieval Performance

We compared the TPSD and the CSAS in six retrieval tasks. For this experiment we used the chord sequence corpus described above, which contains sequences that clearly describe the same song. For each of these tasks the experimental setup was identical: all songs that have two or more similar versions were used as a query, yielding 1775 queries. For each query a ranking was created by sorting the other songs on their TPSD and CSAS scores and these rankings and the runtimes of the compared algorithms were analyzed.

4.1 Tasks

The tasks, summarized in Table 3, differed in the level of chord information used by the algorithms and in the usage of a priori global key information. In tasks 1-3 no key information was presented to the algorithms and in the remaining 3 tasks we used the key information, which was manually checked for correctness, as stored in the Band-in-a-Box files. The tasks 1-3 and 4-6 furthermore differed in the amount of chord detail that was presented to the algorithms: in tasks 1

Task nr.	Chord Structure	Key Information
1	Roots	Key inferred
2	Roots + triad	Key inferred
3	Complete Chord	Key inferred
4	Roots	Key as stored in the Band-in-a-Box file
5	Roots + triad	Key as stored in the Band-in-a-Box file
6	Complete Chord	Key as stored in the Band-in-a-Box file

Table 3. The TPSD and CSAS are compared in six different retrieval tasks.

and 4 only the root note of the chord was available to the algorithms, in tasks 2 and 5 the root and the triad were available and in tasks 3 and 6 the complete chord as stored in the Band-in-a-Box file was presented to the algorithms.

The different tasks required specific variants of the tested algorithms. For tasks 1-3 the TPSD used the TPS key finding algorithm as described in Section 2.1. For the tasks 1 and 4, involving only chord roots, a simplified variant of the TPSD was used, for the tasks 2, 3, 5 and 6 we used the regular TPSD, as described in Section 2.1 and [3].

To measure the impact of the chord representation and substitution functions on retrieval performance, different variants of the CSAS were built also. In some cases the choices made did not yield the best possible results, but they allow the reader to understand the effects of the parameters used on retrieval performance. The CSAS algorithms in tasks 1-3 all used an absolute representation and the algorithms in tasks 4-6 used a key relative representation. In tasks 4 and 5 the chords were represented as the difference in semitones to the root of the key of the piece and in task 6 as the Lerdahl's TPS distance between the chord and the triad from the key (as in the TPSD). The CSAS variants in tasks 1 and 2 used a consonance based substitution function and algorithms in tasks 4-6 a binary substitution function was used. In tasks 2 and 5 a binary substitution function for the mode was used as well: if the mode of the substituted chords matched, no penalty was given, if they did not match, a penalty was given.

A last parameter that was varied was the use of local transpositions. The CSAS variants applied in tasks 1 and 3 did not consider local transpositions, but the CSAS algorithm used in task 2 did allow local transpositions (see Section 2.2 for details).

The TPSD was implemented in Java and the CSAS was implemented in C++, but a small Java program was used to parallelize the matching process. All runs were done on an Intel Xeon quad-core CPU at a frequency of 1.86 GHz. and 4 Gb of RAM running 32 bit Linux[3]. Both algorithms were parallelized to optimally use the multiple cores of the CPUs.

[3] Due to time constrains the CSAS run on task 6 was performed on a different server. This server ran a 64 bit Linux OS, had 2 Xeon quad-core CPUs that were clocked at a frequency of 2.5 GHz and had 8 Gb of RAM.

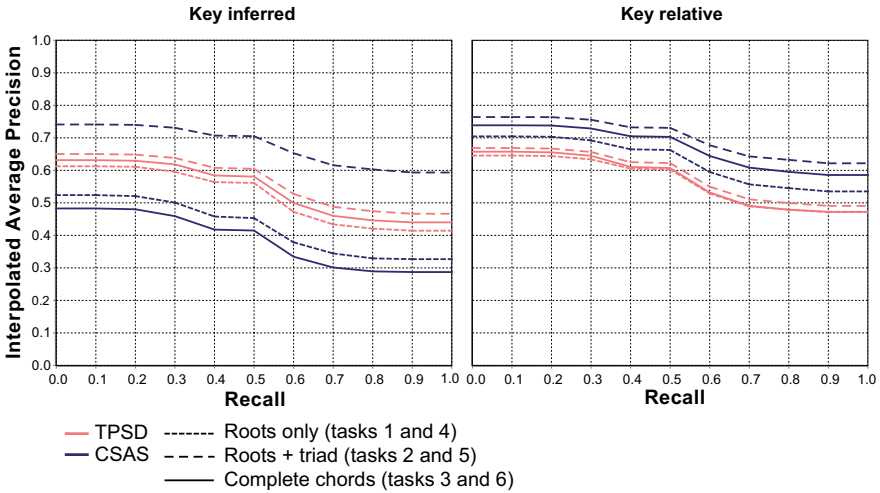

Fig. 2. The 11-point precision and recall charts for the TPSD and the CSAS for tasks 1–3, on the left, and 4–6 on the right.

4.2 Results

For each task and for each algorithm we analyzed the rankings of all 1775 queries with 11-point precision recall curves and Mean Average Precision (MAP). Figure 2 displays the interpolated average precision and recall chart for the TPSD and the CSAS for all tasks listed in Table 3. We calculated the interpolated average precision as in [22] and probed it at 11 different recall levels. In all evaluations the query was excluded from the analyzed rankings. In tasks 2 and 4-6 the CSAS outperforms the TPSD and in tasks 1 and 3 the TPSD outperforms the CSAS. The curves all have a very similar shape, this is probably due to the specific sizes of the song classes and the fairly limited amount of large song classes (see Table 2).

In Figure 3 we present the MAP and the runtimes of the algorithms on two different axes. The MAP is displayed on the left axis and the runtimes are shown on right axis that has an exponential scale doubling the amount of time at every tick. The MAP is a single-figure measure, which measures the precision at all recall levels and approximates the area under the (uninterpolated) precision recall graph [22]. Having a single measure of retrieval quality makes it easier to evaluate the significance of the differences between results. We tested whether the differences in MAP were significant by performing a non-parametric Friedman test, with a significance level of $\alpha = .05$. We chose the Friedman test because the underlying distribution of the data is unknown and in contrast to an ANOVA the Friedman does not assume a specific distribution of variance. There were significant differences between the runs, $\chi^2(11, N = 1775) = 2,618$, $p < .0001$. To determine which of the pairs of measurements differed significantly

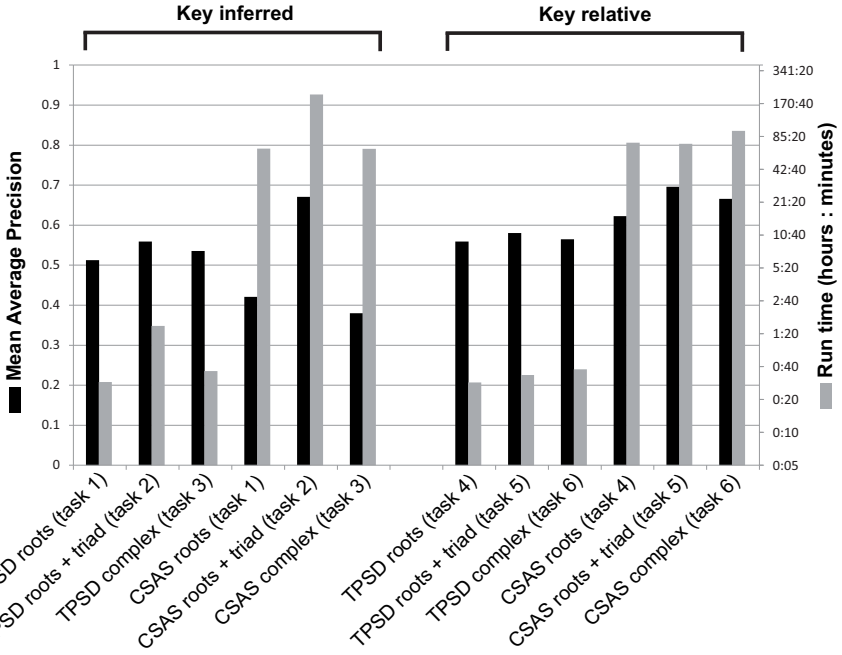

Fig. 3. The MAP and Runtimes of the TPSD and the CSAS. The MAP is displayed on the left axis and the runtimes are displayed on an exponential scale on the right axis. On the left side of the chart the key inferred tasks are displayed and the key relative tasks are displayed on the right side.

we conducted a post hoc Tukey HSD test[4]. Opposed to a T-test, the Tukey HSD test can be safely used for comparing multiple means. A summary of the analyzed confidence intervals is given in Table 4. Significant and non-significant differences are denoted with +'s and –'s, respectively.

The overall retrieval performance of all algorithms on all tasks can be considered good, but there are some large differences between tasks and between algorithms, both in performance and in runtime. With a MAP of .70 the overall best performing setup was the CSAS using triadic chord descriptions and a key relative representation (task 5). The TPSD also performs best on task 5 with an MAP of .58. In tasks 2 and 4-6 the CSAS significantly outperforms the TPSD. On tasks 1 and 3 the TPSD outperforms the CSAS in runtime as well as performance. For these two tasks, the results obtained by the CSAS are significantly lower because local transpositions are not considered. These results show that taking into account transpositions has a high impact on the quality of the retrieval system, but also on the runtime.

[4] All statistical tests were performed in Matlab 2009a.

		Key Inferred					Key Information available					
		task1 TPSD	task2 TPSD	task3 TPSD	task1 CSAS	task2 CSAS	task3 CSAS	task4 TPSD	task5 TPSD	task6 TPSD	task4 CSAS	task5 CSAS
key	task2 TPSD	+										
inferred	task3 TPSD	–	+									
	task1 CSAS	+	+	+								
	task2 CSAS	+	+	+	+							
	task3 CSAS	+	+	+	+	+						
key	task4 TPSD	+	–	+	+	+	+					
information	task5 TPSD	+	+	+	+	+	+	+				
available	task6 TPSD	+	–	+	+	+	+	–	+			
	task4 CSAS	+	+	+	+	+	+	+	–	–		
	task5 CSAS	+	+	+	+	+	+	+	+	+	+	
	task6 CSAS	+	+	+	+	–	+	+	+	+	+	–

Table 4. This table shows for each pair of runs if the mean average precision, as displayed in Figure 3 differed significantly (+) or not (–).

The retrieval performance of the CSAS is good, but comes at a price. On average over six of the twelve runs, the CSAS runs need about 136 times as much time to complete as the TPSD. The TPSD takes about 30 minutes to 1.5 hours to match all 5028 pieces, while the CSAS takes about 2 to 9 days. Due to the fact that the CSAS run in task 2 takes 206 hours to complete, there was not enough time to perform a run on task 1 and 3 with the CSAS variant that takes local transpositions into account.

In task 6 both algorithms represent the chord sequences as TPS distances to the triad of the key. Nevertheless, the TPSD is outperformed by the CSAS. This difference as well as other differences in performance might well be explained by the insertion and deletion operations in the CSAS algorithm: if one takes two identical pieces an inserts one arbitrary extra chord somewhere in the middle of the piece, an asynchrony is created between the two step functions which has a large effect on the estimated distance, while the CSAS distance only gains one extra deletion score.

For the CSAS algorithm we did a few additional runs that are not reported here. These runs showed that the difference in retrieval performance using different substitution costs (binary, consonance or semi-tones) is limited.

The runs in which a priori key information was available performed better, regardless of the task or algorithm (compare tasks 1 and 4, 2 and 5, and 3 and 6 for both algorithms in Table 4). This was to be expected because there are always errors in the key finding, which hampers the retrieval performance.

The amount of detail in the chord description has a significant effect on the retrieval performance of all algorithms. In almost all cases, using only the triadic chord description for retrieval yields better results than using only the root or the complex chord descriptions. Only the difference in CSAS performance between using complex chords or triads is not significant in task 5 and 6. The differences between using only the root or using the complete chord are smaller and not always significant.

Thus, although colorful additions to chords may sound pleasant to the human ear, they are not always beneficial for determining the similarity between the harmonic progressions they represent. We think there might be a simple explanation for these differences. Using only the root of a chord already leads to good retrieval results, but by removing good information about the mode one looses information that can aid in boosting the retrieval performance. On the other hand keeping all rich chord additional information seems to distract the evaluated retrieval systems. Pruning the chord structure down to the triad might be seen as a form of syntactical noise-reduction, since these additions, if they do not have a voice leading function, have a rather arbitrary character and can only add some harmonic spice.

5 Concluding Remarks

We performed a comparison of two different chord sequence similarity measures, the TPSD and the CSAS, on a large newly assembled corpus of 5028 symbolic chord sequences. The comparison consisted of six different tasks in which we varied the amount of detail in the chord description and the availability of a priori key information. The CSAS variants outperform the TPSD significantly in most cases, but is in all cases far more costly to use. The use of a priori key information improves performance and using only the triad of a chord for similarity matching gives the best results for the tested algorithms. Nevertheless, we can positively answer the third question that we have asked ourselves in the introduction–do chord descriptions provided a useful and valid abstraction– because the experiment presented in the previous section clearly shows that chord descriptions can be used for retrieving harmonically related pieces.

The retrieval performance of both algorithms is good, especially if one considers the size of the corpus and the relatively small class sizes (see Table 2), but there is still room for improvement. Both algorithms cannot deal with large structural changes, e.g. adding repetitions, a bridge, etc. A prior analysis of the structure of the piece combined with partial matching could improve the retrieval performance. Another important issue is that the compared systems treat all chords as equally important. This is musicologically not plausible. Considering the musical function in the local as well as global structure of the chord progression, like is done in [6] or with sequences of notes in [23], might improve the retrieval results.

With runtimes that are measured in days, the CSAS is a costly system. The runtimes might be improved by using GPU programming [24], or with filtering steps using algorithms such as BLAST [25].

The harmonic retrieval systems and experiments presented in this paper consider a specific form of symbolic music only. Nevertheless, the application of the methods here presented is not limited to symbolic music and audio applications are currently investigated. Especially the recent developments in chord label extraction are very promising because the output of these methods could be matched directly with the systems here presented. The good performance of the

proposed algorithms lead us to believe that also in other musical domains, such as audio, retrieval systems will benefit from chord sequence based matching in the near future.

References

1. Mauch, M., Noland, K., Dixon, S.: Using Musical Structure To Enhance Automatic Chord Transcription. In: Proceedings of the Tenth International Society for Music Information Retrieval Conference (ISMIR). (2009) 231–236
2. Bello, J., Pickens, J.: A Robust Mid-Level Representation for Harmonic Content in Music Signals. In: Proceedings of the International Symposium on Music Information Retrieval. (2005) 304–311
3. de Haas, W.B., Veltkamp, R.C., Wiering, F.: Tonal Pitch Step Distance: A Similarity Measure for Chord Progressions. In: Proceedings of the Ninth International Society for Music Information Retrieval Conference (ISMIR). (2008) 51–56
4. Hanna, P., Robine, M., Rocher, T.: An Alignment Based System for Chord Sequence Retrieval. In: Proceedings of the 2009 Joint International Conference on Digital Libraries, ACM New York, NY, USA (2009) 101–104
5. Temperley, D.: The Cognition of Basic Musical Structures. Cambridge, MA, MIT Press (2001)
6. de Haas, W.B., Rohrmeier, M., Veltkamp, R.C., Wiering, F.: Modeling Harmonic Similarity Using a Generative Grammar of Tonal Harmony. In: Proceedings of the Tenth International Society for Music Information Retrieval Conference (ISMIR). (2009) 549–554
7. Lerdahl, F.: Tonal Pitch Space. Oxford University Press (2001)
8. Krumhansl, C., Kessler, E.: Tracing the Dynamic Changes in Perceived Tonal Organization in a Spatial Representation of Musical Keys. Psychological Review **89**(4) (1982) 334–68
9. Arkin, E., Chew, L., Huttenlocher, D., Kedem, K., Mitchell, J.: An Efficiently Computable Metric for Comparing Polygonal Shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence **13**(3) (1991) 209–216
10. Aloupis, G., Fevens, T., Langerman, S., Matsui, T., Mesa, A., Nuñez, Y., Rappaport, D., Toussaint, G.: Algorithms for Computing Geometric Measures of Melodic Similarity. Computer Music Journal **30**(3) (2004) 67–76
11. Smith, T., Waterman, M.: Identification of Common Molecular Subsequences. Journal of Molecular Biology **147** (1981) 195–197
12. Mongeau, M., Sankoff, D.: Comparison of Musical Sequences. Computers and the Humanities **24**(3) (1990) 161–175
13. Hanna, P., Ferraro, P., Robine, M.: On Optimizing the Editing Algorithms for Evaluating Similarity Between Monophonic Musical Sequences. Journal of New Music Research **36**(4) (2007) 267–279
14. van Kranenburg, P., Volk, A., Wiering, F., Veltkamp, R.C.: Musical Models for Folk-Song Melody Alignment. In: Proceedings of the Tenth International Society for Music Information Retrieval Conference (ISMIR). (2009) 507–512
15. Paiement, J.F., Eck, D., Bengio, S.: A probabilistic model for chord progressions. In: Proceedings of the Sixth International Conference on Music Information Retrieval (ISMIR), London, UK (2005) 312–319
16. Uitdenbogerd, A.L.: Music Information Retrieval Technology. PhD thesis, RMIT University, Melbourne, Australia (July 2002)

17. Allali, J., Hanna, P., Ferraro, P., Iliopoulos, C.: Local Transpositions in Alignment of Polyphonic Musical Sequences. In: Proceedings of the String Processing and Information Retrieval Symposium (SPIRE). (2007) 26–38
18. Pickens, J., Crawford, T.: Harmonic Models for Polyphonic Music Retrieval. In: Proceedings of the Eleventh International Conference on Information and Knowledge Management, ACM New York, NY, USA (2002) 430–437
19. Gannon, P.: Band-in-a-Box. PG Music (1990)
20. Mauch, M., Dixon, S., Harte, C., Casey, M., Fields, B.: Discovering Chord Idioms Through Beatles And Real Book Songs. In: Proceedings of the Eighth International Society for Music Information Retrieval Conference (ISMIR). (2007) 255–258
21. Harte, C., Sandler, M., Abdallah, S., Gómez, E.: Symbolic Representation of Musical Chords: A Proposed Syntax for Text Annotations. In: Proceedings of the Sixth International Society for Music Information Retrieval Conference (ISMIR). (2005) 66–71
22. Manning, C., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press New York, NY, USA (2008)
23. Robine, M., Hanna, P., Ferraro, P.: Music Similarity: Improvements of Edit-based Algorithms by Considering Music Theory. In: Proceedings of the ACM SIGMM International Workshop on Multimedia Information Retrieval (MIR), Augsburg, Germany (2007) 135–141
24. Ferraro, P., Hanna, P., Imbert, L., Izard, T.: Accelerating Query-by-Humming on GPU. In: Proceedings of the Tenth International Society for Music Information Retrieval Conference (ISMIR). (2009) 279–284
25. Altschul, S., Gish, W., Miller, W., Myers, E., Lipman, D.: Basic Local Alignment Search Tool. Journal of Molecular Biology **215** (1990) 403–410