Longest common subsequence techniques for aligning audio and symbolic music: A case study in Western classical music

Iman S. H. Suyoto and Alexandra L. Uitdenbogerd
School of Computer Science and Information Technology, RMIT
GPO Box 2476v, Melbourne, Victoria 3001, Australia

Abstract

An important goal of music retrieval research is the successful retrieval of music recordings using note-based queries. In this report, we demonstrate the feasibility of searching for recorded music by using a symbolic version of the same work as a query. Our approach makes use of the longest common subsequence algorithm to match automatically transcribed musical audio with a symbolic rendition of the same work. Longest common subsequence yields the best results when matching is applied to an absolute pitch representation and the score is normalised by the answer length. About 90% of queries returned a relevant answer in the top 10, and precision at 1 was about 80% for a collection of 1 808 recordings of western classical music—a substantial improvement (80%) on previous techniques.

1 Introduction

The ability to find music that is in the form of a recorded performance, when the query consists of notes, has thus far been elusive, or limited at best. However, several promising alternatives have arisen, such as cover version detection in audio using feature matching of whole recorded versions [Gómez and Herrera, 2006], and finding specific recordings via a noisy sample of the recording [Wang, 2003].

Our approach has been to consider the case of a query consisting of a symbolic rendition of the desired piece, and a collection consisting of typical commercial-quality audio recordings.

In our previous work [Suyoto and Uitdenbogerd, 2007], we applied velocity- and pitch-based filtering to notes in the automatic transcriptions, and then matched the lowest part extracted from both the query and the transcribed works in the collection—on the assumption that low frequency transcribed notes are more likely to be real notes than harmonics. The matching was most successful when using global alignment on relative pitch representations (directed modulo 12) of the note sequences. This approach worked well for pieces that were mostly monophonic, and some that were largely monotimbral, but failed for a substantial proportion of the query set.

In this work, we continue to investigate matching polyphonic audio music with its symbolic equivalent. In brief, our current technique employs the longest common subsequence (LCS) algorithm [Gusfield, 1997, pp. 227–228]. This is described in more detail in Section 3. In Section 4, our experimental setup is described with the results shown in Section 5. Conclusions and suggestions for future work are discussed in Section 6.

As an overview, our previous approaches using global and local alignments on relative pitch sequences [Suyoto and Uitdenbogerd, 2007] only enabled about 50% of the queries to yield a relevant answer in the top 10 returned results, with precision at 1 at around 40%. Normalisation of the longest common subsequence scores between absolute pitch sequences using the answer length is proven to be more effective, with about 90% queries yielding a relevant answer in the top 10 and precision at 1 of about 80%.

Accompanying materials for this report can be found at `http://mirt.cs.rmit.edu.au/pubs/sy/`.

2 Related work

We are unaware of other work besides ours that attempts this precise problem. Polyphonic matching of symbolic queries and collection, and audio to audio matching are not directly comparable. The work by Gómez and Herrera [2006] on cover version detection had accuracy at around 50% with a collection of just 90 pieces, but the problem is likely to be harder than matching a perfect symbolic representation to an audio one. We have observed that automatically transcribed pieces differ markedly from each other, that is, the "noise" that is transcribed is not consistent across versions of the same work, despite having the same instrumentation. A similar problem with audio-to-audio matching is also discussed in Marolt [2006], but this makes use of a melody-based representation. The collection consisted of 1 820 pieces with 36 of them as the queries. Out of the devised approaches, the most effective one achieved 27% of hits being in the top 5 returned answers.

The Shazam system[1] retrieves specific recordings that match the same version of the song fed as the query, although the audio quality is different [Wang, 2003]. The system uses audio signatures. The limitation of the system is that it does not find cover versions of the query music. There are other attempts to find cover versions by using long-term structure, such as variation in loudness or volume [Foote, 2000] or timbral texture [Aucouturier and Sandler, 2001, 2002]. Those techniques have only been tested on collections of less than one hundred pieces, however. Works by Dannenberg et al. [2003], Hu and Dannenberg [2002], Mazzoni and Dannenberg [2001], and Shifrin and Birmingham [2003] show that matching MIDI files with monophonic hummed queries is more successful.

Shalev-Shwartz et al. [2002] attempted a problem of matching polyphonic audio collection with monophonic symbolic queries. Their collection consisted of 832 opera performances with orchestra accompaniment. The length of each performance was one minute. (To contrast with our experiment, the pieces in our collection are not cut to any length.) Their technique makes use of a probabilistic approach to align the polyphonic audio with the monophonic symbolic queries. Temporal and spectral variations are used for matching. The method was tested with 50 queries with three different query lengths: 5 seconds, 15 seconds, and 25 seconds. With 25-second queries, average precision of 95% was achieved.

There has been a previous attempt at applying LCS to monophonic music matching [Guo and Siegelmann, 2004]. In that work, the LCS algorithm is modified to anticipate expansion and contraction. (In our work here, we do not modify the LCS algorithm.) The queries were produced by randomly choosing a subset of the collection and manipulating the songs in it. That was done by stretching the songs, shortening and lengthening randomly chosen notes (to simulate rhythmic inaccuracies), and inserting and deleting random notes. The best result—almost 90% of correct answers ranked top 5—was achieved when the queries were fractions of original songs with randomly inserted notes.

Similarly to our previous work [Suyoto and Uitdenbogerd, 2007], we do not use any low-level audio features for similarity calculation such as Mel-frequency cepstral coefficients.[2] Instead, we use the high-level music feature called *pitch*.

3 Matching

The matching process comprises three stages: transcription (of audio to symbolic data), standardisation, and alignment. These are discussed below.

3.1 Transcription

In this stage, symbolic data is obtained through the transcription of audio files using TS-AudioToMidi 3.30.[3] The transcription produced is stored in Standard MIDI file format.[4]

Noise in the form of extraneous notes is produced even with state-of-the-art transcription technology. This is caused by the difficulty of transcribing music produced by instruments whose timbre contains many harmonic components. This also makes polyphonic music more difficult to transcribe than monophonic music, in which the assumption of a single note at a time allows the fundamental frequency to be selected as "the note."

To illustrate the problem, we use one of the tracks in the collection used in our experiment, that is J. S. Bach's "BWV 1007 Prelude" performed by Carrai. Its first two bars are shown in Figure 1, and its transcription result is shown in Figure 2. As clearly seen in Figure 2, there are many extraneous notes.

3.2 Standardisation

Standardisation is the process of generating a string that represents a particular tune. A string representation allows a query and answers to be matched using approximate string matching techniques [Uitdenbogerd and Zobel, 1999, Suyoto and Uitdenbogerd, 2005].

Our previous work [Suyoto and Uitdenbogerd, 2007] uses the directed modulo-12 standardisation. This belongs to the class of representations that encode pitches as relative values (included in this class are contour standardisations) [Uitdenbogerd,

[1]See `http://www.shazam.com`.

[2]This report does not explain audio features. They are described elsewhere [Aucouturier and Pachet, 2002, Logan and Salomon, 2001, Pampalk et al., 2005, Somerville and Uitdenbogerd, 2005].

[3]See `http://audioto.com/eng/aud2midi.htm`.

[4]See `http://www.midi.org/about-midi/abtmidi2.shtml`.

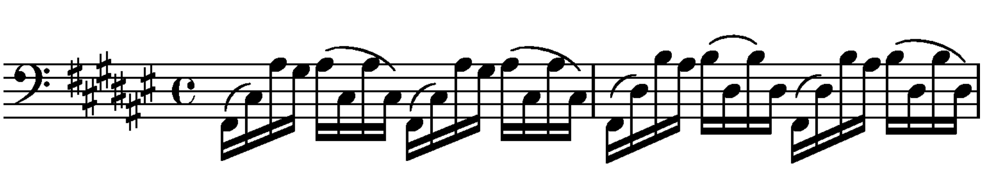

Figure 1: *J. S. Bach's "BWV 1007 Prelude."*

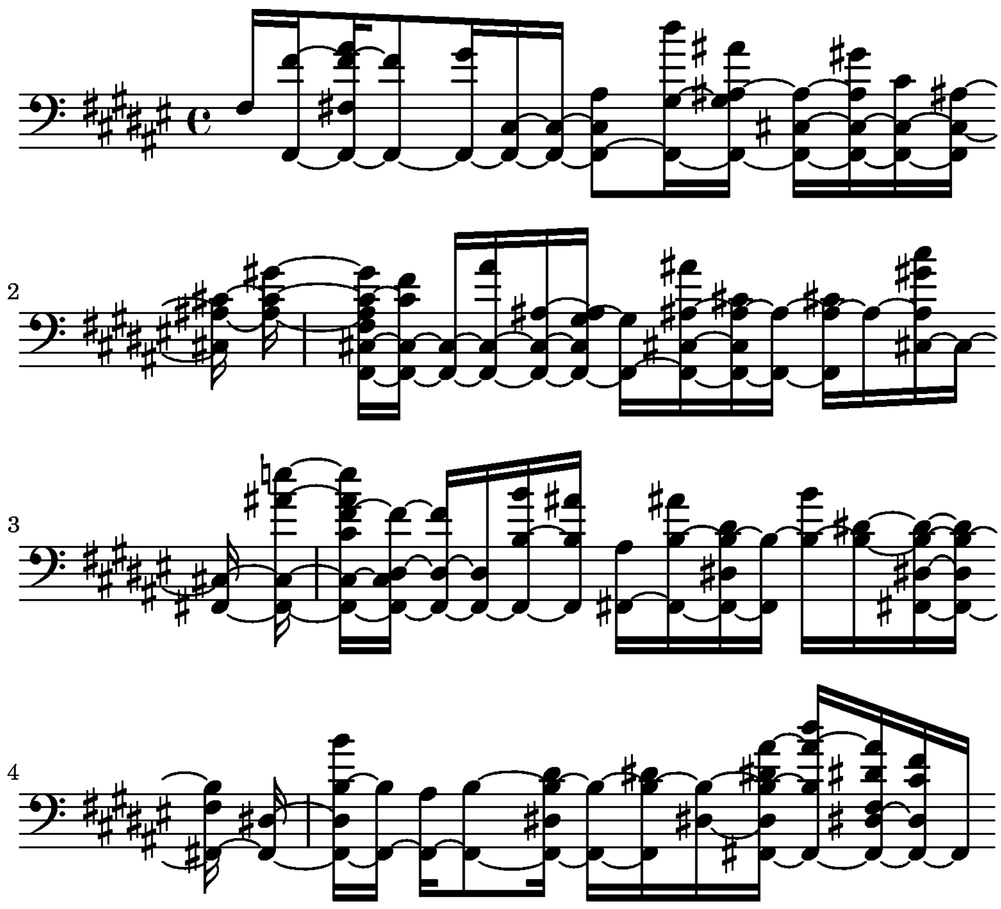

Figure 2: *The transcription result of J. S. Bach's "BWV 1007 Prelude" by Carrai for the bars shown in Figure 1. For reading comfort, notes are quantised to semiquavers/sixteenth notes.*

2002, pp. 78–86]. In this work, we use *limited absolute pitch standardisation*. In this standardisation, we encode pitches as absolute values. A note is represented as its pitch name. The idea is suggested in Uitdenbogerd [2002, pg. 81] and modified so that the octave is not retained. For example, the melody shown in Figure 3 is represented as "C G A C G# B D".

In the case of chords, we do not use any melody extraction algorithm like in previous work [Uitdenbogerd and Zobel, 1999, Suyoto and Uitdenbogerd, 2005, 2007]. Instead, the notes are sorted ascendingly by pitch. For example, the chord shown in Figure 4 is represented as "F A C". This is done because LCS is used, which does not penalise extraneous notes. The full presence of chords maximises the chance to match more symbols.

The disadvantage of this standardisation is that the string representation is not transposition-invariant. Therefore, in the alignment process, transposition is incorporated as part of matching. This issue is addressed in more detail in Section 3.3. As we shall see in Section 5, this approach supports much higher retrieval effectiveness compared to that used in Suyoto and Uitdenbogerd [2007].

3.3 Alignment

In the alignment phase, dynamic programming is used. The dynamic programming

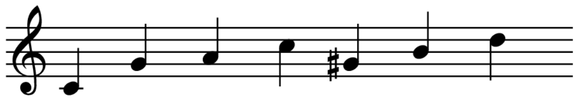

Figure 3: *This melody is represented as "C G A C G# B D" using limited absolute pitch standardisation.*

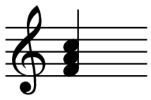

Figure 4: *This chord is represented as "F A C" using limited absolute pitch standardisation.*

technique on pitch sequences has been previously used in various works in symbolic music matching [Dannenberg et al., 2003, Suyoto and Uitdenbogerd, 2005, Uitdenbogerd and Zobel, 1999]. It has also been used for audio [Aucouturier and Sandler, 2001, Foote, 2000].

In this work, the *longest common subsequence* (LCS) [Gusfield, 1997, pp. 227–228] score between a query and an answer is used as a candidate score. The query is then transposed by one semitone. This is done 11 times. For example, if the query is "C E G C", it is transposed to "C# F G# C#", and then to "D F# A D", up to "B D# F# B". The answer remains untransposed. The transposition causes the answer to be scanned 12 times. This results in 12 candidate scores. The score with maximum similarity is picked as the final score for the answer. Mathematically, if $S(q,a)$ is the similarity between query q and answer a:

$$S(q,a) = \max_{s} \left(L \left(t \left(q,s \right), a \right) \right) \tag{1}$$

where $t(q,s)$ is a function transposing q by s semitones ($s \in \{0,1,2,\ldots,11\}$), and $L(T,a)$ is the longest common subsequence score between T and a. The full (unoptimised) algorithm is specified in Algorithm 1.

4 Experiment

Our experiment is intended as a direct comparison to our previous work [Suyoto and Uitdenbogerd, 2007], therefore the collection and the queries are the same. In brief, the previous technique (SBDG) involved the use of global alignment on a relative pitch representation that had initially been filtered to remove high pitch and low velocity notes. For further details, readers are redirected to it for more details.

As the similarity measurement, we use Equation 1. We also use the following similarity measurements:

$$S_I(q,a) = \frac{S(q,a)}{|a|} \tag{2}$$

$$S_K(q,a) = \frac{S(q,a)}{|a| \log_e |a|} \tag{3}$$

$$S_y(q,a,y) = \frac{S(q,a)}{\log_e^y |a|}; y = \{1.0, 1.1, 1.2, \ldots, 2.0\} \tag{4}$$

5 Results

To measure the effectiveness of our approach, we use the widely known *precision* information retrieval measure [Baeza-Yates and Ribeiro-Neto, 1999, pg. 75], which is defined as:

$$P \equiv \frac{|\mathbf{Rel} \cap \mathbf{Ret}|}{|\mathbf{Ret}|} \tag{5}$$

where **Rel** is the set of relevant answers and **Ret** is the set of retrieved answers. For presentational purposes, we use P_N (precision at N retrieved answers, $N = |\mathbf{Ret}|$); $N \in \{1,2,3,\ldots,20\}$.

The results for our experiments using $S(q,a)$, $S_I(q,a)$, and $S_K(q,a)$ similiarity measurements are shown in Table 1. The results for $S_y(q,a,y); y = \{1.0, 1.1, 1.2, \ldots, 2.0\}$ are shown in Table 2. It shows that $S_y(q,a,1.7)$ is the most effective similarity measurement when the desired answers are aimed to be in top 20, although $y = [1.3 \ldots 2.0]$ is approximately as good. The performance of $S_y(q,a,2.0)$ is remarkable for retrieving the correct answer in the first place. A visual comparison between the measurements and the SBDG method [Suyoto and Uitdenbogerd, 2007] as the baseline is depicted in Figure 5.

Using the LCS score only (the $S(q,a)$ similarity measurement) results into poor retrieval effectiveness. This is because some songs in the collection are significantly longer than others, and thus need to be represented using more symbols. As the LCS algorithm does not penalise mismatch, insertion, and deletion operations, the longer sequences can more easily yield high scores. Therefore, the long sequences cannot be distinguished from the actual correct answer(s). It is also confirmed in our experimental results that long sequences easily move up

Algorithm 1: *The algorithm to align two melodies p and q.*

```
1. Construct a matrix with $(|p|+1)$ rows and $(|q|+1)$ columns. Let us call a cell $D_{i,j}$.
2. $D_{0,0} \leftarrow 0$.
3. $D_{i,0} \leftarrow 0; i \in 0 \ldots |p|$.
4. $D_{0,j} \leftarrow 0; j \in 0 \ldots |q|$.
5. For $s$ in $0 \ldots 11$:
   1. $q' = t(q,s)$
   2. For $i$ in $1 \ldots |p|$:
      1. For $j$ in $1 \ldots |q|$:
         1. $D_i, j = \begin{cases} D_{i-1,j-1}+1 & ; \quad s_{i-1} = q'_{i-1} \\ \max(D_{i-1,j}, D_{i,j-1}); & s_{i-1} \neq q'_{i-1} \end{cases}$.
   3. $L_s = D_{|p|,|q|}$
6. Return $\max_s(L_s)$.
```

Table 1: $\langle P_N \rangle$ *(mean precision at N) values for similarity measurements $S(q,a)$, $S_I(q,a)$, and $S_K(q,a)$.*

N	$S(q,a)$	$S_I(q,a)$	$S_K(q,a)$
1	0.029	0.000	0.000
2	0.015	0.000	0.000
3	0.010	0.000	0.000
4	0.007	0.000	0.000
5	0.006	0.000	0.000
6	0.010	0.000	0.000
7	0.013	0.000	0.000
8	0.011	0.000	0.000
9	0.010	0.000	0.000
10	0.012	0.000	0.000
11	0.011	0.000	0.000
12	0.010	0.000	0.000
13	0.011	0.000	0.000
14	0.010	0.000	0.000
15	0.010	0.000	0.000
16	0.011	0.002	0.000
17	0.010	0.002	0.000
18	0.010	0.002	0.000
19	0.011	0.002	0.000
20	0.010	0.001	0.000

to the higher ranks. Therefore, a penalty should be applied to long sequences. This was the rationale for testing various similarity measurements which normalise the LCS score with a function of answer length.

As shown in Table 1, our experiment with similarity measurements $S_I(q,a)$ and $S_K(q,a)$ did not result in higher effectiveness, however. On the contrary, the effectiveness is much lower, not being able to retrieve any correct answer in the top 15. On the other hand, Table 2 shows that using $S_y(q,a,y)$ yields high effectiveness, as is clearly demonstrated in Figure 5.

Symbolic queries that are optimised for listening are as effective in retrieving answers as those optimised for printing. This is normal, as there is not much difference between the two versions. Moreover, the LCS algorithm does not penalise extraneous notes, and they can actually increase the matching chance.

Table 2: $\langle P_N \rangle$ *(mean precision at* N*) values for* $S_y(q, a, y)$.

N	y										
	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0
1	0.676	0.676	0.706	0.765	0.794	0.794	0.794	0.794	0.765	0.794	0.824
2	0.529	0.529	0.559	0.574	0.603	0.559	0.574	0.574	0.574	0.574	0.574
3	0.490	0.490	0.500	0.500	0.529	0.519	0.500	0.500	0.490	0.500	0.510
4	0.375	0.382	0.390	0.390	0.404	0.404	0.397	0.390	0.382	0.390	0.382
5	0.300	0.306	0.312	0.318	0.324	0.324	0.324	0.312	0.312	0.318	0.312
6	0.250	0.255	0.265	0.275	0.275	0.270	0.270	0.270	0.275	0.275	0.260
7	0.214	0.219	0.227	0.235	0.235	0.231	0.231	0.231	0.235	0.235	0.227
8	0.188	0.191	0.199	0.210	0.206	0.202	0.202	0.202	0.206	0.206	0.202
9	0.167	0.170	0.176	0.186	0.183	0.186	0.183	0.180	0.186	0.186	0.180
10	0.153	0.153	0.162	0.168	0.165	0.168	0.165	0.165	0.168	0.168	0.162
11	0.139	0.139	0.147	0.153	0.150	0.153	0.150	0.150	0.153	0.153	0.147
12	0.127	0.127	0.135	0.140	0.137	0.140	0.140	0.137	0.140	0.140	0.137
13	0.118	0.118	0.127	0.129	0.131	0.129	0.129	0.127	0.129	0.129	0.127
14	0.109	0.111	0.117	0.119	0.122	0.119	0.119	0.119	0.119	0.119	0.117
15	0.102	0.104	0.110	0.112	0.114	0.112	0.112	0.112	0.112	0.112	0.110
16	0.096	0.097	0.103	0.105	0.107	0.105	0.105	0.105	0.105	0.105	0.103
17	0.090	0.092	0.097	0.099	0.100	0.099	0.099	0.100	0.099	0.099	0.097
18	0.085	0.089	0.092	0.093	0.095	0.093	0.093	0.097	0.093	0.093	0.092
19	0.082	0.084	0.087	0.088	0.090	0.088	0.088	0.092	0.090	0.088	0.087
20	0.078	0.081	0.082	0.085	0.085	0.084	0.084	0.087	0.085	0.084	0.084

Queries `B1010_B1` and `B1010_B2` [Suyoto and Uitdenbogerd, 2007] perform poorly compared to other queries. There is only one target answer for both queries, in which both parts of the song are combined into one song file. This makes the lengths for the two queries disproportional to that of the answer.

The results also demonstrate that for this task, notes should be represented by absolute pitch. Comparing to the results in Suyoto and Uitdenbogerd [2007], relative pitch representation is lower in effectiveness, as the noise captured in the transcription often diminishes the reliability of an interval-based representation.

6 Conclusions and Future Work

This report explores the use of the LCS algorithm to match manually constructed queries and audio transcription. Our experimental results demonstrate that:

- It is possible to match audio with symbolic music.
- Representing notes by absolute pitch is more suited to this task than using relative pitch.

We plan to test our approach on other genres of music to see the scope of its applicability.

7 Acknowledgements

We thank Falk Scholer for proof-reading a draft version of this report.

References

J.-J. Aucouturier and F. Pachet. Music similarity measures: What's the use? In M. Fingerhut, editor, *Proceedings of the Third International Conference on Music Information Retrieval*, pages 157–163, Paris, France, Oct. 2002. IRCAM-Centre Pompidou.

J.-J. Aucouturier and M. Sandler. Using long-term structure to retrieve music: Representation and matching. In Downie and Bainbridge [2001], pages 1–2.

J.-J. Aucouturier and M. Sandler. Finding repeating patterns in acoustical musical signals: Applications for audio thumbnailing. In *Proceedings of the Audio Engineering Society 22nd International Conference on Virtual, Synthetic and Entertainment Audio*, pages 412–421, Espoo, Finland, June 2002.

R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, New York, USA, 1999.

R. Dannenberg, K. Lemström, and A. Tindale, editors. *Proceedings of the Seventh International Con-*

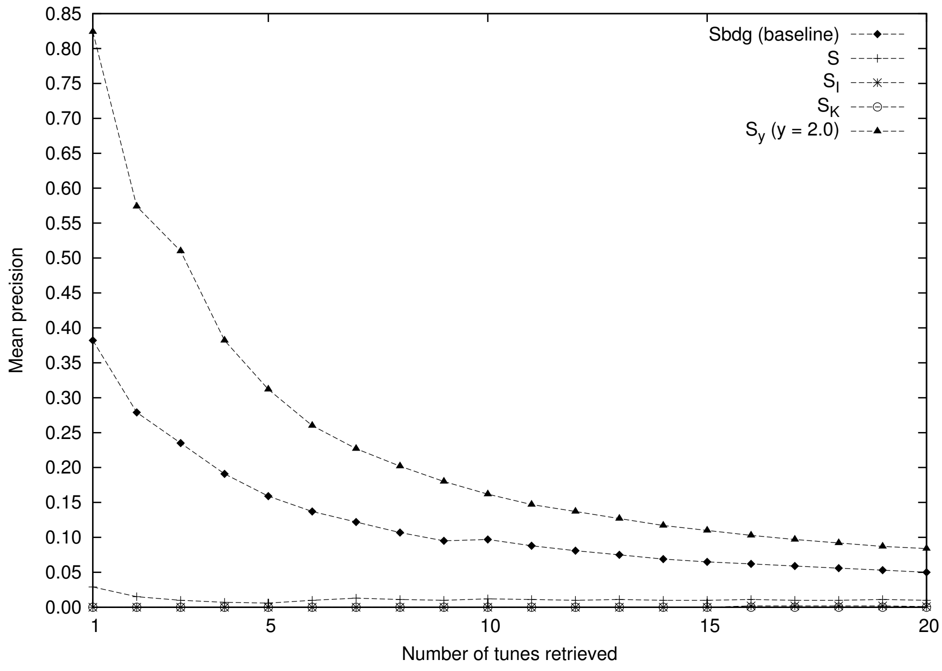

Figure 5: *Comparison between various similarity measurements.*

ference on Music Information Retrieval, Victoria, Canada, Oct. 2006. University of Victoria.

R. B. Dannenberg, W. P. Birmingham, G. Tzanetakis, C. Meek, N. Hu, and B. Pardo. The Musart testbed for query-by-humming evaluation. In Hoos and Bainbridge [2003], pages 41–47.

J. S. Downie and D. Bainbridge, editors. *Proceedings of the Second International Symposium on Music Information Retrieval*, Bloomington, USA, Oct. 2001.

J. Foote. Arthur: Retrieving orchestral music by long-term structure. In D. Byrd, J. S. Downie, T. Crawford, W. B. Croft, and C. Nevill-Manning, editors, *Proceedings of the First International Symposium on Music Information Retrieval*, Plymouth, USA, Oct. 2000.

E. Gómez and P. Herrera. The song remains the same: Identifying versions of the same piece using tonal descriptors. In Dannenberg et al. [2006], pages 180–185.

A. Guo and H. Siegelmann. Time-warped longest common subsequence algorithm for music retrieval. In C. L. Buyoli and R. Loureiro, editors, *Proceedings of the Fifth International Conference on Music Information Retrieval*, pages 258–261, Barcelona, Spain, Oct. 2004. Universitat Pompeu Fabra.

D. Gusfield. *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press, Cambridge, UK, 1997.

H. H. Hoos and D. Bainbridge, editors. *Proceedings of the Fourth International Conference on Music Information Retrieval*, Baltimore, USA, Oct. 2003. Johns Hopkins University.

N. Hu and R. B. Dannenberg. A comparison of melodic database retrieval techniques using sung queries. In G. Marchionini and W. Hersh, editors, *Proceedings of the Second ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 301–307, Portland, USA, July 2002.

B. Logan and A. Salomon. A music similarity function based on signal analysis. In *Proceedings of the 2001 IEEE International Conference on Multimedia and Expo*, pages 745–748, Tokyo, Japan, Aug. 2001.

M. Marolt. A mid-level melody-based representation for calculating audio similarity. In Dannenberg et al. [2006], pages 280–285.

D. Mazzoni and R. B. Dannenberg. Melody matching directly from audio. In Downie and Bainbridge [2001], pages 17–18.

E. Pampalk, A. Flexer, and G. Widmer. Improvements of audio-based music similarity and genre classification. In J. D. Reiss and G. A. Wiggins, editors, *Proceedings of the Sixth International Conference on Music Information Retrieval*, pages 628–

633, London, UK, Sept. 2005. Queen Mary, University of London.

S. Shalev-Shwartz, S. Dubnov, N. Friedman, and Y. Singer. Robust temporal and spectral modeling for query by melody. In K. Järvelin, M. Beaulieu, R. Baeza-Yates, and S. H. Myaeng, editors, *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 331–338, Tampere, Finland, Aug. 2002.

J. Shifrin and W. P. Birmingham. Effectiveness of HMM-based retrieval on large databases. In Hoos and Bainbridge [2003], pages 33–39.

P. Somerville and A. L. Uitdenbogerd. Classification of music based on musical instrument timbre. In *Proceedings of the Fourth Australasian Conference on Knowledge Discovery and Data Mining*, pages 173–188, Sydney, Australia, Dec. 2005.

I. S. H. Suyoto and A. L. Uitdenbogerd. Effectiveness of note duration information for music retrieval. In L. Zhou, B. C. Ooi, and X. Meng, editors, *Proceedings of the Tenth International Conference on Database Systems for Advanced Applications*, pages 265–275, Beijing, China, Apr. 2005. Springer-Verlag.

I. S. H. Suyoto and A. L. Uitdenbogerd. Aligning musical audio with symbols: A case study of western classical music. Technical Report TR-07-1, School of Computer Science and Information Technology, RMIT, 2007.

A. L. Uitdenbogerd. *Music Information Retrieval Technology*. PhD thesis, School of Computer Science and Information Technology, RMIT, Melbourne, Australia, 2002.

A. L. Uitdenbogerd and J. Zobel. Melodic matching techniques for large music databases. In D. Bulterman, K. Jeffay, and H. J. Zhang, editors, *Proceedings of the 1999 ACM Multimedia Conference*, pages 57–66, Orlando, USA, Nov. 1999.

A. Wang. An industrial-strength audio search algorithm. In Hoos and Bainbridge [2003], pages 7–13. Invited talk.