A Survey of Query-By-Humming Similarity Methods

Alexios Kotsifakos
Department of Computer Science and Engineering
University of Texas at Arlington, USA

Panagiotis Papapetrou
Department of Information and Computer Science
Aalto University, Finland

Jaakko Hollmén
Department of Information and Computer Science
Aalto University, Finland

Dimitrios Gunopulos
Department of Informatics and Telecommunications
University of Athens, Greece

Vassilis Athitsos
Department of Computer Science and Engineering
University of Texas at Arlington, USA

ABSTRACT

Performing similarity search in large databases is a problem of particular interest in many communities, such as music, database, and data mining. Although several solutions have been proposed in the literature that perform well in many application domains, there is no best method to solve this kind of problem in a Query-By-Humming (QBH) application. In QBH the goal is to find the song(s) most similar to a hummed query in an efficient manner. In this paper, we focus on providing a brief overview of the representations to encode music pieces, and also on the methods that have been proposed for QBH or other similarly defined problems.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data Mining; H.2.4 [**Systems**]: Multimedia Databases

General Terms

Algorithms

Keywords

Query-By-Humming, Sequence Matching, Hidden Markov Models

1. INTRODUCTION

A problem of particular interest that has attracted the attention of music, database and data mining communities is that of searching a large database to find the most similar objects to a query object. Depending on the application domain this problem can be addressed in different ways. The application domain that we focus on in this paper is that of *Query-By-Humming (QBH)*.

Suppose you hear a song but cannot recall its name. A straightforward solution is to hum a part of this song and then perform a search on a large music repository to find this song, or songs similar to it. In a QBH system, the goal is to search a music database for the K most similar songs to a hummed query.

Due to the fact that QBH is a very noisy application domain, to perform meaningful matching the methods that address this problem have to a) consider slight or more serious errors occurring due to instant or temporary key or tempo loss, or else be error-tolerant, and b) choose a representation for music pieces that best fits their needs, without sacrificing efficiency and accuracy. In this paper, we present a brief overview of the methods that have been proposed, either for addressing QBH specifically or for other application domains that could be also of interest in QBH. The discrimination of the different categories is done based on whether the methods perform sequence, model, or n-gram based similarity search.

Next, we provide some basic terms used in QBH. Every piece of music (as shown in Figure 1) is a sequence of notes characterized by a *key*, that defines the standard pattern of allowed intervals that the sequence of notes should conform with, and a *tempo*, regulating the speed of the music piece. Each *note* consists of two parts, the *pitch* and the *duration*. A *pitch interval* is the distance between two pitches, and the interval of 12 semitones is called *octave*. Another important term is *transposition*, i.e., shifting a melody of a piece written in a specific key to another key. Finally, there is a discrimination between *monophonic* and *polyphonic* music, where the latter allows two or more notes sound simultaneously, in contrast to monophonic music as is QBH. The interested reader can look at [7, 27, 43, 47, 49, 50] for approaches in polyphonic music.

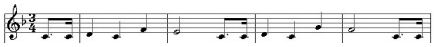

Figure 1: A part of the "Happy Birthday" music score.

It is important to understand the pros and cons of the different categories of methods (and certainly of the methods in particular), since music pieces are essentially sequences of values evolving in time, and thus the QBH problem is just an application accentuating the problematic aspects of methods that deal with time. With this in mind, improvements over existing methods can be introduced so that they can be used in other areas of interest, such as finding specific patterns in financial, weather, and sensor data observing the movements of humans. In the latter case assistive actions can be taken when particular patterns are observed.

Next, we present different ways of representing music pieces, along with the variety of methods that may be of interest in QBH categorized in sequence, model, and n-gram based methods.

PETRA '12, June 6 - 8, 2012, Crete Island, Greece.

2. REPRESENTING MUSIC PIECES

Although many approaches represent notes by encoding only pitch, it has been shown that the use of both pitch and duration information improves music retrieval [20] and the melody sequences are made more unique [30, 45].

Expressing pitch can be done in three ways. First, by using absolute pitch, which is the frequency of the note [13, 15, 30, 44]. Secondly, the note number in the key of the piece [32] can be used. However, the key does not usually remain the same throughout the whole music piece, which makes this encoding very hard to use. Third, pitch can be expressed as pitch interval, that is the difference in the frequencies of two consecutive notes. Also, pitch intervals can be quantized in [-11,11] by applying modulo 12 [45], leading to two octaves, a reasonable range in which human pitch can fluctuate while singing. Regarding the encoding of duration there are three options [33]. Inter-Onset-Interval (IOI) is the difference in time onsets of two adjacent notes, IOI Ratio (IOIR), which is the ratio of IOIs of two adjacent notes, and Log IOI Ratio (LogIOIR), i.e., the logarithm of IOIR. A variation of LogIOIR is to quantize the LogIOIR values of the notes to the closest integer or the closest value in $[-2, 2]$ [33]. Considering all possible combinations of pitch and duration encoding, we can easily conclude that pitch interval and IOIR (or variations of them) leads to transposition and time invariance when comparing melodies.

3. SEQUENCE MATCHING

There are two broad categories of matching between sequences, the *whole sequence matching* and the *subsequence matching*. Next, we give a brief overview of methods of these categories. It should be mentioned that some of them have already been applied to music retrieval, while others not. It is of particular importance to identify the pros and cons of such methods, so that we can further decide if some of these methods can be improved to apply to QBH where the music pieces are represented as sequences of notes.

Referring to the first category of whole sequence matching, one well known dynamic programming (DP) method [2] is Dynamic Time Warping (DTW) [23]. DTW is a distance measure allowing to compare two sequences that may vary in time or speed. There are also several variants of DTW, among which cDTW [37], EDR [6], and ERP [5]) widely used in many application domains. These methods are robust to misalignments and time warps, and some (e.g., DTW, cDTW) can achieve high retrieval accuracy in applications such as time series mining and classification [21]. cDTW constrains the difference of the sequences' indices that can be matched, so as to avoid matching elements of the two sequences that are far away from each other. EDR[6] and ERP[5] are DP-based distance measures for time series similarity search and use the triangle inequality for pruning purposes; the latter is also a metric. It should be noted that ERP is more robust to noisy data, and that since both methods are designed for full sequence matching they are not directly applicable to a subsequence matching framework such as QBH. The method proposed in [1] operates using DTW and has been applied to music retrieval. It takes into account absolute pitches and deals with the problems of invariance by applying occasionally normalizations, depending on the proposed representation. Another approach that incorporates tempo variations was introduced in [32], while in [11] it was modified to deal with various representations as well. To deal with tempo variation the method presented in [29] scales the target sequences before applying DTW. Moreover, some approaches embed transposition invariance as a cost function in their DP computation [8, 25], though, with not very attractive runtime complexity. In addition, in [25] and [12] they aim at matching the whole query and no duration is incorporated. Transposition invariance can also be encoded by a proper representation [28].

Apart from these DTW-based whole sequence matching methods, the similarity between two sequences can be measured by finding their Longest Common SubSequence (LCSS) [3, 4, 45], i.e., the greatest number of elements that are common in both of them, and this can be done again using a DP approach. Thus, LCSS-based approaches can tolerate noise introduced by the user in the QBH application, while they allow for gaps in the alignment. However, since no bounds are imposed to the allowed gaps this may result in a large number of false positives when $|Q| << |X|$. Since errors in QBH systems are imposed by the users, tolerance in pitch and duration is highly eligible. Though, we should not be tolerant to skip too many elements in the target sequence, as in that case there is small possibility of finding a good match. With this direction in mind, in [18] an algorithm which accounts for a bounded number of gaps in the target sequence only is proposed. The algorithm also handles errors by using constant values as tolerances for each query element. Variations regarding the sum, differences, and length of consecutive gaps are proposed in [9] when matching two sequences, but as in [18] they are allowed to exist in the target sequence only. Notice that both approaches deal with *whole query matching* and absolute pitches, while none of them accounts for note duration, i.e., they are proposed for 1-dimensional sequences.

To efficiently address the QBH application, where it is more suitable to use subsequence matching methods, the aforementioned whole sequence matching methods could be retrofitted by performing a sliding window search over the database. The issue with such an approach is that it would be computationally expensive requiring one DP computation per window. The approach of Han et al. [16] is based on uniform segmentation and sliding windows, which requires the user to manually select the length of the segments and is therefore not sensitive to the actual behavior of the data and can efficiently handle only *near exact* matching.

Some DP methods reduce subsequence matching to full sequence matching, by cutting database sequences into small pieces, and requiring each query to correspond to an entire such piece. Such approaches though fail when the query corresponds to a database subsequence that is not stored as a single piece. One example is the Query-By-Humming system described by Zhu et al. [51], where each database song is cut into smaller, disjoint pieces, and they developed an efficient lower-bound for DTW. As a result, this method is not directly applicable to subsequence matching.

Regarding the second category, two similar methods for subsequence matching are presented in [17, 19], which implicitly account for tempo scaling. However, they are not transposition invariant, as absolute pitches are used. Furthermore, the Edit distance [26] has been used for music retrieval with several variations [8, 24, 25, 35]. Its most recent version used in QBH is presented in [48], where the cost function has been properly modified so that this distance measure uses both pitch and duration information. This variation of edit distance has been further evaluated in [22], where the authors tested several time series representations. Another DP-based method for finding the subsequences of evolving numerical streams that are closest to a query is presented in [38]. SPRING, is based on DTW and does not sacrifice accuracy, in spite of the fact that DTW is not a metric distance function. Another reason for making this method attractive is that it operates in constant space, and time linear to the database size (for short queries). Thus, it is a promising method to be applied to the problem of QBH [22].

Similarity methods can be also applied to the subsequence matching category as well. One such method is Smith-Waterman [42],

which compares segments of all possible lengths and optimizes the similarity measure. One application of this algorithm in QBH is presented at [45]. Finally, a recently proposed novel method for subsequence matching is SMBGT [22] that incorporates many intrinsic properties of the QBH application. It bounds the number of consecutive gaps allowed in both the query and target database sequences, allows for variable tolerance levels in the matching, imposes a constraint in the maximum match range, and optionally bounds the minimum number of matching elements. By allowing gaps in the compared sequences several errors that may occur while humming due to temporary key/tempo loss or significant instant note loss can be efficiently treated.

4. MODEL-BASED MATCHING

Several probabilistic methods (HMM-based) have been developed for speech recognition and music retrieval [14, 31, 34, 36, 39, 48]. In [31], an extended HMM is used where the target and query notes are associated through a series of hidden states, modeling the local and cumulative error in pitch and durations. HMMs also appear in [34], where they are used to compute the match score of the local alignment when both pitch and duration are used. An extended HMM architecture Factorial HMM has been proposed to model music, and more specifically Bach's chorales [14]. Factorial HMMs are based on a factored, distributed representation of the hidden state variable. Due to its complicated structure, inference and learning is intractable, and approximate learning is necessary. Although the model may be effective in capturing the statistical structure in the Bach's chorales, it is not built for any query processing as QBH. In [48] HMMs are used for the segmentation of the humming instances, followed by energy and pitch analysis to correct the segmentation errors. They use both pitch and duration and the retrieval stage is done by using salient information in the transcribed symbolic sequences (FingerPrints), but their method does not significantly outperform edit distance. Although the HMM-based methods incorporate the probabilistic behavior that is inherent in an application domain such as music retrieval, they are computationally expensive due to the required training, and creating models to represent all styles of music in a large database is a very tough task. Other approaches that exploit HMMs are presented in [40, 41], though their focus is to use HMMs as their front end to capture the location of notes in the input and not perform retrieval.

5. N-GRAMS

A method that has been successfully used in string matching is *n-grams*, according to which the number of matching substrings of a fixed length n is counted. Also, since it is more likely for long sequences to have a match, this count should be normalized. An efficient method for approximate string matching is proposed in [46], but is not suitable for the case of QBH where we have to deal with very noisy queries. n-gram-based methods proposed for music retrieval [10, 45] fail to handle noisy queries efficiently leading to poor performance as they are designed for *near exact* matching, although they may keep pitch and duration information.

6. OPEN PROBLEMS

Having studied the aforementioned methods for solving the similarity search problem in QBH, we observed several problems that need to be taken into account when trying to address this problem. First, there is no standard and preferred music format to store music pieces. MIDI and MP3 are the mostly used formats for this problem, and although MP3 is richer in information, it cannot be easily extracted and exploited as opposed to MIDI. Second, there is no music database widely available that can be used as a testbed to test the proposed methods on, and as a result the results reported are most of the times subjective, or at least cannot be generalized since the datasets are quite small and limited to a few genres of music or to songs of Beatles. Moreover, to the best of our knowledge, automatic music genre classification which would help efficiency and accuracy is not part of the proposed end-to-end systems. Last but not least, since QBH is a very noisy application domain, a method should allow for both slight and more serious key or tempo loss errors at the same time without affecting accuracy.

7. CONCLUSIONS

In this paper we presented a brief overview of the ways to encode music pieces, and also of the methods that have been proposed in the literature for the QBH application or that would be of interest to study in order to better understand the problematic aspects of applying them to similar application domains.

Acknowledgments

This work has been partially supported by grants from the National Science Foundation, IIS-0812601, IIS-1055062, IIS-1238660, CNS-0923494, CNS-1035913, and CNS-1059235, by the Academy of Finland ALGODAN Center of Excellence, and the MODAP and DISFER projects.

8. REFERENCES

[1] N. Adams, M. Bartsch, J. Shifrin, and G. Wakefield. Time series alignment for music information retrieval. In *Proceedings of ISMIR*, pages 303–311, 2004.

[2] R. Bellman. The theory of dynamic programming. *Bull. Amer. Math. Soc.*, 60(6):503–515, 1954.

[3] L. Bergroth, H. Hakonen, and T. Raita. A survey of longest common subsequence algorithms. In *SPIRE*, pages 39–48, 2000.

[4] B. Bollobás, G. Das, D. Gunopulos, and H. Mannila. Time-series similarity problems and well-separated geometric sets. In *Symposium on Computational Geometry*, pages 454–456, 1997.

[5] L. Chen and R. Ng. On the marriage of l_p-norms and edit distance. In *VLDB*, pages 792–803, 2004.

[6] L. Chen and M. T. Özsu. Robust and fast similarity search for moving object trajectories. In *SIGMOD*, pages 491–502, 2005.

[7] M. Clausen, R. Engelbrecht, D. Meyer, and J. Schmitz. Proms: A web-based tool for searching in polyphonic music. In *ISMIR*, 2000.

[8] T. Crawford, C. Iliopoulos, and R. Raman. String matching techniques for musical similarity and melodic recognition. *Computing in Musicology*, 11:73–100, 1998.

[9] M. Crochemore, C. Iliopoulos, C. Makris, W. Rytter, A. Tsakalidis, and K. Tsichlas. Approximate string matching with gaps. *Nordic Journal of Computing*, 9(1):54–65, 2002.

[10] R. Dannenberg, W. Birmingham, B. Pardo, N. Hu, C. Meek, and G. Tzanetakis. A comparative evaluation of search techniques for query-by-humming using the MUSART testbed. *Journal of the American Society for Information Science and Technology*, 58(5):687–701, 2007.

[11] R. Dannenberg and N. Hu. Understanding search performance in query-by-humming systems. In *ISMIR*, pages 232–237, 2004.

[12] S. Deorowicz. Speeding up transposition-invariant string matching. *Information Processing Letters*, 100(1):14–20, 2006.

[13] J. Downie. The musifind music information retrieval project, phase iii: evaluation of indexing options. In *Connectedness: Information, systems, people, organizations: In conference of the Canadian Association for Information Science, 1995*, pages 135–146, 1995.

[14] Z. Ghahramani and M. I. Jordan. Factorial hidden markov models. *Machine Learning*, 29:245–275, 1997.

[15] A. Ghias, J. Logan, D. Chamberlin, and B. Smith. Query by humming: Musical information retrieval in an audio database. In *ACM Multimedia*, pages 231–236, 1995.

[16] T. Han, S.-K. Ko, and J. Kang. Efficient subsequence matching using the longest common subsequence with a dual match index. In *Machine Learning and Data Mining in Pattern Recognition*, pages 585–600. 2007.

[17] N. Hu, R. Dannenberg, and A. Lewis. A probabilistic model of melodic similarity. In *ICMC*, pages 509–515, 2002.

[18] C. Iliopoulos and M. Kurokawa. String matching with gaps for musical melodic recognition. In *PSC*, pages 55–64, 2002.

[19] J. Jang and M. Gao. A query-by-singing system based on dynamic programming. In *International Workshop on Intelligent Systems Resolutions*, pages 85–89, 2000.

[20] T. Kageyama, K. Mochizuki, and Y. Takashima. Melody retrieval with humming. In *ICMC*, pages 349–349, 1993.

[21] E. Keogh. Exact indexing of dynamic time warping. In *VLDB*, pages 406–417, 2002.

[22] A. Kotsifakos, P. Papapetrou, J. Hollmén, and D. Gunopulos. A Subsequence Matching with Gaps-Range-Tolerances Framework: A Query-By-Humming Application. *PVLDB*, 4(11):761–771, 2011.

[23] J. B. Kruskall and M. Liberman. The symmetric time warping algorithm: From continuous to discrete. In *Time Warps*. Addison-Wesley, 1983.

[24] K. Lemström and S. Perttu. Semex-an efficient music retrieval prototype. In *ISMIR*, pages 23–25, 2000.

[25] K. Lemström and E. Ukkonen. Including interval encoding into edit distance based music comparison and retrieval. In *AISB*, pages 53–60, 2000.

[26] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics*, 10(8):707–710, 1966.

[27] A. Lubiw and L. Tanur. Pattern matching in polyphonic music as a weighted geometric translation problem. In *ISMIR*, pages 289–296, 2004.

[28] V. Makinen, G. Navarro, and E. Ukkonen. Algorithms for transposition invariant string matching. *Lecture notes in computer science*, pages 191–202, 2003.

[29] D. Mazzoni and R. Dannenberg. Melody matching directly from audio. In *ISMIR*, pages 17–18, 2001.

[30] R. McNab, L. Smith, I. Witten, C. Henderson, and S. Cunningham. Towards the digital music library: Tune retrieval from acoustic input. In *International Conference on Digital Libraries*, pages 11–18, 1996.

[31] C. Meek and W. Birmingham. A comprehensive trainable error model for sung music queries. *Journal of Artificial Intelligence Research*, 22(1):57–91, 2004.

[32] M. Mongeau and D. Sankoff. Comparison of musical sequences. *Computers and the Humanities*, 24(3):161–175, 1990.

[33] B. Pardo and W. Birmingham. Encoding timing information for musical query matching. In *ISMIR*, pages 267–268, 2002.

[34] B. Pardo, J. Shifrin, and W. Birmingham. Name that tune: A pilot study in finding a melody from a sung query. *Journal of the American Society for Information Science and Technology*, 55(4):283–300, 2004.

[35] S. Pauws. Cubyhum: A fully operational query by humming system. In *ISMIR*, pages 187–196, 2002.

[36] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[37] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Transactions on ASSP*, 26:43–49, 1978.

[38] Y. Sakurai, C. Faloutsos, and M. Yamamuro. Stream monitoring under the time warping distance. In *ICDE*, pages 1046–1055, 2007.

[39] J. Shifrin, B. Pardo, C. Meek, and W. Birmingham. HMM-based musical query retrieval. In *CS joint Conference on Digital Libraries*, pages 295–300, 2002.

[40] H. Shih, S. Narayanan, and C. Kuo. An HMM-based approach to humming transcription. In *2002 IEEE International Conference on Multimedia and Expo (ICME2002)*, 2002.

[41] H. Shih, S. Narayanan, and C. Kuo. A statistical multidimensional humming transcription using phone level hidden Markov models for query by humming systems. In *Proceedings of IEEE 2003 International Conference on Multimedia and Expo*, volume 1, pages 61–64, 2003.

[42] T. Smith and M. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147:195–197, 1981.

[43] R. Typke, P. Giannopoulos, R. Veltkamp, F. Wiering, and R. Van Oostrum. Using transportation distances for measuring melodic similarity. In *ISMIR*, pages 107–114, 2003.

[44] A. Uitdenbogerd and J. Zobel. Manipulation of music for melody matching. In *ACM Multimedia*, pages 235–240, 1998.

[45] A. Uitdenbogerd and J. Zobel. Melodic matching techniques for large music databases. In *ACM Multimedia (Part 1)*, page 66, 1999.

[46] E. Ukkonen. Approximate string-matching with q-grams and maximal matches. *Theoretical Computer Science*, 92(1):191–211, 1992.

[47] E. Ukkonen, K. Lemström, and V. Mäkinen. Geometric algorithms for transposition invariant content-based music retrieval. In *ISMIR*, pages 193–199, 2003.

[48] E. Unal, E. Chew, P. Georgiou, and S. Narayanan. Challenging uncertainty in query by humming systems: a fingerprinting approach. *Transactions on Audio Speech and Language Processing*, 16(2):359–371, 2008.

[49] G. Wiggins, K. Lemström, and D. Meredith. SIA(M)ESE: An algorithm for transposition invariant, polyphonic content-based music retrieval. In *ISMIR*, pages 13–17, 2002.

[50] C. Yang. Efficient acoustic index for music retrieval with various degrees of similarity. In *International Conference on Multimedia*, page 591, 2002.

[51] Y. Zhu and D. Shasha. Warping indexes with envelope transforms for query by humming. In *SIGMOD*, pages 181–192, 2003.