A Three-Layer Approach for Music Retrieval in Large Databases

Kjell Lemström
Dept. of Computer Science
University of Helsinki
PO Box 26
FIN-00014 Helsinki
Finland
klemstro@cs.helsinki.fi

Geraint A. Wiggins
Dept. of Computing
City University, London
Northampton Square
London EC1V 0HB
England
geraint@soi.city.ac.uk

David Meredith
Dept. of Computing
City University, London
Northampton Square
London EC1V 0HB
England
dave@city.ac.uk

Abstract

An effective music information retrieval (MIR) system should provide fast queries to music databases taking into account musical features relevant to the task, such as *transposition invariance*, *polyphony* of music and the fact that there might be some 'extra intervening musical elements' (such as grace notes) within the database occurrence of the query pattern. The importance of efficiency is due to the need to find musical documents in possibly enormous databases. In this paper, we introduce an approach using three different matching layers each of which is possible to find transposition invariant occurrences of given musical query patterns in polyphonic music databases. The advantage of the layers is in sorting them in an order of decreasing efficiency as regards to the speed, but in an increasing order of carefulness put in the searching process. Thus, a repeatedly occurring musical pattern used as a query pattern should be found very efficiently, while searching for a query pattern corresponding to a rare database occurrence with arbitrarily many extra intervening elements is allowed to take more time.

Key words: music retrieval, multi-layer approach, polyphonic databases.

1 Introduction

The rapid growth of Internet-based services, multimedia technology, and development of new standards, such as MPEG-7, have made content-based retrieval and analysis of multimedia information an increasingly important field of research. Traditionally, the research has focused on the indexing and retrieval of texts and images. Nevertheless, music is at least as important a part of our culture and, consequently, as important a constituent of the multimedia domain, which can be seen in the current effort being put into developing theories and practical methods for its retrieval for use in, for example, Internet search and digital libraries. Content-based retrieval of music requires specific techniques that have not been employed for other media.

In this paper, we suggest a multi-layer approach for music information retrieval (MIR) in large-scale databases. Our approach contains three different matching layers, each of which is capable of finding transposition invariant occurrences of a given query pattern within a polyphonic database. The matching layers are executed in order of decreasing efficiency. The same ordering, however, puts the layers in order of increasing number of possibilities inspected. Having executed one layer, the algorithm of that layer outputs the occurrences it has found (if any), after which it is down to the user whether the next matching layer is to be executed.

A specific property of two of the layers (the first and the third) is that in a constituent of a musical pattern there may appear any finite number of other events between any two events included in the query pattern. Thus, an occurrence could be found even if the database version had, for example, some kind of musical decoration (such as different ornamentations or grace notes) that is absent in the query pattern. Next we briefly describe the three layers one by one.

2 Description of the Layers

The First Layer The first matching layer is based on indexing. A particularly time-efficient indexing technique, based on *suffix structures*, for the matching can be performed in linear time with respect to the length of the query pattern. There are several ways to implement a suffix structure. An attractive choice is the so-called *suffix tree* because its space complexity is linear in the size of the database. Moreover, it can be constructed in linear time (7). However, a suffix tree storing all the MIDI files available on the Internet would require an impossible amount of main memory (4). Because of this problem, some MIR researchers have started to consider the possibility of finding *musically meaningful* patterns in the musical documents residing in the database. One of the first ideas was to extract only the melodies, or only the melodies of the choruses of the musical documents, these being the parts of musical documents most frequently used in content based queries (1).

Another current idea for indexing music (3) is to use a multiple viewpoint system (2). The idea is to put in the structure such subsequences from any of the considered viewpoints whose frequency, in practice, exceeds significantly the expected frequency. Both of the methods above, as well as all the others previously suggested for MIR indexing, have been alike in that they have been based on sequences of musical events, and the sequences to be considered have been fixed beforehand.

Our current pattern induction algorithm, SIATEC (Structure Induction Algorithm with Translational Equivalence Classes) (6), can be used to find the frequently occurring musical patterns. SIATEC works in two phases. The first phase, SIA, computes every maximal repeated pattern in the given music database. The second phase takes the output of SIA as input and then computes all the occurrences of each of the maximal repeated patterns computed by SIA. The patterns to be inserted in the indexing structure are selected out of these occurrences of maximal repeated patterns, and can be seen as the *Longest Common Transposition Invariant Subsequences* (or LCTS) (4) of subsequences appearing in the database. Thus, we are able to find in this most efficient first phase patterns that are musically meaningful (since they are those that recur in the database) even if they are decorated unexpectedly. We cannot afford, however, to index all such recurring patterns of a large-scale database, because of the huge amount of main memory required for the indexing structure. Therefore, the following matching layers would be needed in many situations.

The Second Layer If the pattern being sought cannot be found in the index, the second layer of our approach can be invoked. This layer applies the fast bit parallel MONOPOLY filtering algorithm (5). With most patterns (any pattern whose length is shorter than the size of one computer word), the main phase scans through the database in linear time (with respect to the length of the database). The main phase reports possible locations for occurrences, which are to be checked with another, somewhat slower algorithm.

The advantage of this layer is that it does not need an excessively large main memory to be able to search the occurrences of the pattern anywhere in the database. Furthermore, even though it is not as efficient as the previous layer, it is still very fast compared to the conventional dynamic programming algorithm computing edit distance, frequently applied to MIR (see e.g. (4) for a summary of some current methods). The found occurrences, however, for this layer are always sequential. Therefore we have one further layer.

The Third Layer The final third layer applies the SIA(M)ESE matching algorithm based on the SIA algorithm (6). This layer is the slowest of the three, but it allows a broader definition of what counts as a possible match than the two previous layers. More precisely, it does the same kind of LCTS matching as the first layer, but because it does not need the indexing structure, it is not restricted to matching against frequently occurring patterns only. Moreover, SIA(M)ESE is capable of *multi-resolution searching*, *i.e.*, the matching can be done on any desired level of detail (*cf. e.g.*, Schenkerian analyses). This is obtained by definition without any extra cost on the performance; the resolution of the search is defined by the details of the user-given query pattern.

3 Concluding Remarks

In this paper, we have introduced a three-layer approach to MIR, which is capable of finding transposition invariant occurrences of a given query pattern within large-scale polyphonic music databases. The occurrences of the pattern can be found even if the database version has, for example, different musical decorations than the query pattern. The contribution of our approach is in having three distinct searching algorithms each of which are efficient and effective already in themselves, but that work particularly well together when sorted in an order of decreasing performance and in an increasing order of thoroghness and detail in the searching process. In this way, a repeatedly occurring pattern can be found very efficiently, whereas finding another query pattern corresponding to a rare database occurrence with possibly many intervening elements takes more time, but is still possible.

Denoting the number of musical events in the query pattern and music database by m and n, respectively, and the number of chords in the database and the maximum size of a chord by $\overline{n}$ and q, the running times of the different phases of our multi-layer algorithm are as follows. Before any query execution, a preprocessing phase is required; the indexing structure for the first layer is constructed in $O(n^2 \log n)$ time, and the structures required for the second layer are built in $O(\overline{n}q)$ time. The three matching layers are executable in $O(m)$, $O(\overline{n})$ ($O(\overline{n}q(q+m))$ for the checking phase), and $O(mn \log(mn))$ times, respectively, the second requiring the restriction that the length of the query pattern is no more than the size of the computer word in bits.

References

[1] D. Bainbridge. Private Communication. October 2000.

[2] D. Conklin and I. Witten. Multiple viewpoint systems for music prediction. *Journal of New Music Research* 24 (1995), pp. 51–73.

[3] D. Conklin and C. Anagnostopolou. Representation and Discovery of Multiple Viewpoint Patterns. In *Proceedings of the 2001 International Computer Music Conference*, Havana, Cuba, publ. ICMA, October 2001.

[4] K. Lemström. *String Matching Techniques for Music Retrieval*. PhD thesis, University of Helsinki, Faculty of Science, Department of Computer Science, 2000. Report A-2000-4.

[5] K. Lemström and J. Tarhio. Detecting monophonic patterns within polyphonic sources. In *Content-Based Multimedia Information Access Conference Proceedings (RIAO'2000)*, pages 1261–1279, Paris, 2000.

[6] D. Meredith, G.A. Wiggins and K. Lemström. Pattern induction and matching in polyphonic music and other multidimensional datasets. In *Proceedings of the Fifth World Multiconference on Systemics Cybernetics and Informatics (SCI2001)*, pages 61–66 (vol X), Orlando, FL, 2001.

[7] E. Ukkonen. On-line construction of suffix-trees. *Algorithmica* 14 (1995), pp. 249–260.