The C-BRAHMS Project

Lemström, Kjell

2003


http://hdl.handle.net/10138/27545

Downloaded from Helda, University of Helsinki institutional repository.

This is an electronic reprint of the original article.

This reprint may differ from the original in pagination and typographic detail.

Please cite the original version.
The C-BRAHMS Project *

Kjell Lemström, Veli Mäkinen, Anna Pienimäki, Mika Turkia, Esko Ukkonen
Department of Computer Science, University of Helsinki
PO Box 26 (Teollisuuskatu 23)
FIN-00014 University of Helsinki, Finland
{klemstro,vmakinen,apimmone,turkia,ukkonen}@cs.Helsinki.FI

Abstract
The C-BRAHMS project develops computational methods for content-based retrieval and analysis of music data. A summary of the recent algorithmic and experimental developments of the project is given. The search engine developed by the project is available at http://www.cs.helsinki.fi/group/cbrahms.

1 Introduction
Content-Based Music Retrieval, or CBMR for short, is a research topic studied rather extensively during the last half decade. One of its famous instances is the so-called “query by humming” or WYHIWYG (What You Hum Is What You Get) application. Given a large database of music called the source, the task is to find excerpts in the database that resemble the most (in a musical way) the hummed query pattern.

This paper introduces our CBMR project called C-BRAHMS (Content-Based Retrieval and Analysis of Harmony and other Music Structures) and its output, the C-BRAHMS engine. The project aims at designing and developing efficient methods for computational problems arising from music comparison, analysis, data mining and retrieval. Currently the project has a focus on retrieving polyphonic music in large scale music databases of symbolically encoded music.

The C-BRAHMS project was formally established in January, 2002. C-BRAHMS is part of the From Data to Knowledge (FDK) research unit hosted by the Department of Computer Science at University of Helsinki. FDK has been selected as a centre of excellence funded by the Academy of Finland. The group collaborates with several researchers and research groups abroad.

2 Music retrieval algorithms
Symbolic music data can be seen as strings of symbols, and string-matching-based methods have been applied to CBMR problems. These methods are designed for handling one-dimensional data, and hence they do not apply on polyphonic music without modifications.

The C-BRAHMS project has developed some generalized string matching algorithms that can deal with polyphonic music. For instance the algorithm by Lemström & Tarhio (2003) uses bit-parallelism and precomputed offline data structure containing pitch interval classes for each chord. This structure is then scanned to filter out match candidates, which are checked with another, slower, algorithm.

Our other string-matching-based algorithms allow efficient transposition-invariant approximate searching (Mäkinen, Navarro & Ukkonen, 2003; Lemström & Navarro, 2003); here approximation means insertions and deletions of notes but not small variations of pitch levels. The efficiency is achieved by using sparse dynamic programming and bit-parallel techniques. Similar techniques are used for finding the minimum splitting of a pattern in a multi-track musical work (Lemström & Mäkinen, 2003).

The project has put effort in developing a recent methodology interpreting music data as geometric objects in an Euclidean space (Wiggins, Lemström & Meredith, 2002; Ukkonen, Lemström & Mäkinen, 2003). In geometric representation both the query pattern and the source are represented by objects in a multidimensional space. For instance, in a 2-dimensional time-pitch space, point objects give the onset time and the pitch, while line segment objects (the well-known piano-roll representation) give the duration of the associated notes, as well. To include additional note parameters the dimensionality can be increased without the need to modify algorithms. The approach is inherently transposition-invariant, and dealing with monophonic and polyphonic music is equally straightforward. Moreover, musical decorations, such as ornamentations for instance, do not deteriorate its working.

One of the basic ideas of the geometric approach is to calculate difference vectors between each source point object and pattern point object, sort the vectors, and calculate the frequencies of all such difference vectors. There is a complete match if the frequency of some difference vector equals the number of point objects in the pattern. Then the onset times must match, which means that rhythmic information is taken into account in the geometric approach; note that the string matching approach usually loses this because only the relative order of notes is taken into account. A more efficient version sorts the difference vectors online by using a hash table (Wiggins, Lemström & Meredith, 2002).

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. ©2003 Johns Hopkins University.

*Supported by the Academy of Finland (grant 201560).
The demo engine uses MIDI files from the Mutopia project, allowing for greater flexibility and better maintainability. In practice, the overhead of using inter-}

gards the performance. In practice, the overhead of using inter-

sections to a distributed server implemented in Ruby language.

position required. The interface allows the matched part or the

mats; approximate bar number of the match; pitch class names

and content data, such as score files in PostScript and PDF for-

poser; title; opus number; date and genre of the music piece,

The query results include metadata, such as name of the com-

poser; title; opus number; date and genre of the music piece,

and content data, such as score files in PostScript and PDF for-

mats; approximate bar number of the match; pitch class names

( and octaves) of the matching notes; and the amount of trans-

mats; approximate bar number of the match; pitch class names

( and octaves) of the matching notes; and the amount of trans-

position required. The interface allows the matched part or the

whole music piece to be played, and to view various histograms

of the note data.

The algorithms have been implemented as C language exten-
sions to a distributed server implemented in Ruby language.

The mixing of interpreted and compiled code allows for greater

productivity in implementing parts that are not essential as re-
gards the performance. In practice, the overhead of using inter-

preted language is negligible. A reduced number of code lines

allows for greater flexibility and better maintainability.

The demo engine uses MIDI files from the Mutopia project.

Files are released either as public domain or under a Mutopi-
aBSD license. Each MIDI file is converted to a string contain-
ing variable-length chords which, subsequently, contain notes.

This string is included in a object representing the music piece,

and searches are performed separately for each piece. Accord-
ing to our experiments, this procedure minimizes the amount of

used working memory in contrast to the approach of merging

all data into one large string and performing a single search on

it (the latter approach would also require removing matches that
go beyond music piece boundaries). This is an essential issue

with algorithms requiring a large amount of space, when work-

ing on large databases and large amounts of concurrent queries

are allowed.

According to our experiments, the former arrangement also im-

proves the performance of memory-allocation-intensive algo-

rithms over the latter approach. However, algorithms that do not

need memory allocation during search, such as our bit-parallel

algorithms, suffer minor performance degradation.

4 Future Directions

In every culture, music is an important part of human communi-
cation, and musicologists have been analyzing written music for

centuries. During the last decades some musical analysis tools

have been developed, which might be further formalized to de-
scribe them as computational problems. Thus, suitable com-
puter programs could be developed to replace the tedious manu-

al work. Moreover, some work on music psychology about

what makes a musical work pleasing to listen to have been de-
scribed precisely enough to be applicable in computerized mu-

sic analysis. In the future, the project will further attempt to use

findings in musicology and music psychology to achieve better

computational methods and results. Data mining methods such

as in Pienimäki (2002) will also be used.

References


splitting of pattern in multi-track string matching. In Proceed-
ings of the 14th Annual Symposium on Combinatorial Pattern


bit-parallel techniques for transposition invariant approximate

matching in music retrieval. To appear in Proceedings of the

10th International Symposium on String Processing and Infor-
mation Retrieval.


matching for multi-track strings. To appear in Nordic Journal

of Computing.


for transposition invariant string matching. In Proceedings of

the 20th International Symposium on Theoretical Aspects of


Pienimäki, A. (2002). Indexing music databases using auto-
matic extraction of frequent phrases. In Proceedings of the

Third International Conference on Music Information Re-

trieval, (pp. 25-30).

Ukkonen, E., Lemström, K. & and Mäkinen, V. (2003). Geo-

metric algorithms for transposition invariant content-based mu-

sic retrieval. To appear in Proceedings of the 4th International

Symposium on Music Information Retrieval.


phonic content-based music retrieval. In Proceedings of the

Third International Conference on Music Information Re-

trieval, (pp. 283-284).

http://www.cs.helsinki.fi/group/cbrahms

http://www.mutopiaproject.org