# GUIDO/MIR — an Experimental Musical Information Retrieval System based on GUIDO Music Notation

**Holger H. Hoos**[*] [†] and **Kai Renz**[†] and **Marko Görg**[†]

## Abstract

Musical databases are growing in number, size, and complexity, and they are becoming increasingly relevant for a broad range of academic as well as commercial applications. The features and performance of musical database systems critically depend on two factors: The nature and representation of the information stored in the database, and the search and retrieval mechanisms available to the user. In this paper, we present an experimental database and retrieval system for score-level musical information based on GUIDO Music Notation as the underlying music representation. We motivate and describe the database design as well as the flexible and efficient query and retrieval mechanism, a query-by-example technique based on probabilistic matching over a clustered dataset. This approach has numerous advantages, and based on experience with a first, experimental implementation, we believe it provides a solid foundation for powerful, efficient, and usable database and retrieval systems for structured musical information.

## 1 Introduction

Multimedia databases play an important role, especially in the context of online systems available on the World Wide Web. As these databases grow in number, size, and complexity, it becomes increasingly important to provide flexible and efficient search and retrieval techniques. When dealing with musical data, two main difficulties are encountered: Firstly, the multidimensional, often complex structure of the data makes both the formulation of queries and the matching of stored data with a given query difficult. Secondly, there is often a considerable amount of uncertainty or in accuracy in the query and/or the data, stemming from limitations of the methods used for obtaining queries, such as "Query-By-Humming" [12], or for acquiring musical data, such as automated performance transcription, as well as from simple human error when entering data.

While there is a strong and increasing interest in database and retrieval systems for sound and sound-level descriptions of music, many application contexts (particularly in musical analysis, composition, and performance) benefit from or require higher-level, structured music representations. Consequently, there is a growing body of research on musical databases and music information retrieval based on structured, score-level music representations (see, e.g., [3; 21; 8]). In this work, we focus on content-based music information retrieval from a database of score-level musical data based on the query-by-example approach [2]. The main contributions of our work, can be summarised as follows:

1. We use GUIDO Music Notation [16] as the music representation underlying the database as well as for formulating queries. Compared to the use of MIDI and various other music representation formats, this approach has a number of conceptual and practical advantages which will be discussed in detail in the following sections. We find that GUIDO is particularly suitable for formulating queries in a query-by-example approach, and we outline how a small and natural extension of GUIDO allows the explicit and localised representation of uncertainty associated with a given query.
2. We introduce a novel music retrieval mechanism based on probabilistic models and a hierarchically clustered musical database. Using probabilistic models for musical information retrieval has the advantage of offering natural, elegant, and flexible ways of scoring exact and approximate matches between pieces in the database and a given query. While in this work, we introduce and illustrate this general concept using rather simple probabilistic models, the approach can be easily generalised to

[*]Corresponding author. University of British Columbia, Department of Computer Science, 2366 Main Mall, Vancouver, BC, V6T 1Z4, Canada, hoos@cs.ubc.ca

[†]Technische Universität Darmstadt, Fachbereich Informatik, Wilhelminenstr. 7, D-64283 Darmstadt, Germany, renz@iti.informatik.tu-darmstadt.de

more complex probabilistic models.

3. We present an experimental database and retrieval system which implements the design and techniques proposed in this paper. This prototypical system, which is available on the WWW, supports various combinations of melodic and rhythmic query types for retrieving information from a database of pieces of varying complexity. The system is implemented in Perl [1] and highly portable; the underlying, object-oriented and modular design facilitates the implementation of different search and retrieval techniques and the investigation of their behaviour.

In the following, we present and discuss our overall approach in more detail. We start with a brief introduction of GUIDO Music Notation and discuss its use in the context of musical database and retrieval systems. In Section 3, we outline our experimental musical database design and implementation. Section 4 is the core of our work; it motivates and describes our approach to music information retrieval. Related approaches are briefly discussed in Section 5, and Section 6 presents some conclusions and outlines a number of directions for future research.

## 2 Why GUIDO?

GUIDO Music Notation[1] is a general purpose formal language for representing score level music in a platform independent, plain-text and human-readable way [16]. The GUIDO design concentrates on general musical concepts (as opposed to only notational, i.e., graphical features). Its key feature is *representational adequacy*, meaning that simple musical concepts should be represented in a simple way and only complex notions should require complex representations. Figure 1 contains three simple examples of GUIDO Music Notation and the matching conventional music notation.

The GUIDO design is organised in three layers: Basic, Advanced, and Extended GUIDO Music Notation. *Basic GUIDO* introduces the basic GUIDO syntactical structures and covers basic musical notions; *Advanced GUIDO* extends this layer to support exact score formatting and more sophisticated musical concepts; and *Extended GUIDO* introduces features which are beyond conventional music notation. GUIDO Music Notation is designed as a flexible and easily extensible open standard. Thus, it can be easily adapted and customised to cover specialised musical concepts as might be required in the context of research projects in computational musicology. GUIDO has not been developed with a particular application in mind but to provide an adequate representation formalism for score-level music over a broad range of applications. The intended application areas include notation software, compositional and analytical systems and tools, musical databases, performance systems, and music on the WWW. Currently, a growing number of applications is using GUIDO as their music representation format.

[1]GUIDO Music Notation is named after Guido d'Arezzo (ca. 992-1050), a renowned music theorist of his time and important contributor to today's conventional musical notation.

### GUIDO vs. MIDI

Currently, virtually every (content-based) MIR system works on MIDI files. The two main reasons for that are:

- the enormous amount of music available as MIDI files on the WWW
- the lack of a commonly used and accepted representation format for structured music

Although Standard MIDI File (SMF) format is the most commonly used music interchange format, it does not *adequately* support activities other than playback. MIDI was never intended to be the notation (and music) interchange format that it has become today.

There are several reasons, why MIDI is not very well suited for MIR. A MIDI file contains a low level-description of music which describes only the timing and intensity of notes. Since structural information such as chords, slurs or ties cannot be stored in a Standard MIDI file[2], a high- or multilevel description is not possible. Some of the basic limitations of a MIDI file are the lack of differentiation between enharmonic equivalents (e.g.. C-sharp and D-flat), and lack of precision in the durations between events (which are expressed in MIDI-ticks).

Our MIR system has been implemented using GUIDO as its underlying music representation language. To still be able to use the huge body of MIDI files on the WWW, our group has developed converters between GUIDO and MIDI[3].

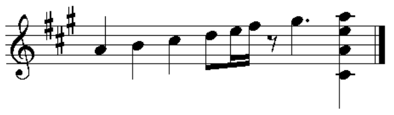
### GUIDO vs. XML

XML is a simplified subset of SGML, a general markup language that has been officially registered as a standard (ISO8879). Because of its increasing popularity, there have been quite a number of attempts to use it for storing musical data [14; 6] and as well as for Music Information Retrieval [26]. XML has obvious and undeniable strengths as a general representation language: it is platform independent, text-based, human-readable and extensible. Additionally, by using XML to represent music, one gains the advantage of using a standardised metalanguage for which a growing number of tools are becoming available. To our knowledge none of the approaches to music representation using XML published so far has yet gained wide acceptance. One

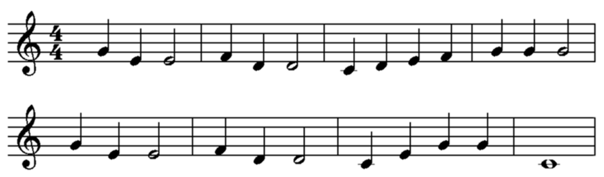
[2]Using non-standard techniques, it is possible to store additional information in MIDI files; however, these mechanisms are not part of the standard

[3]GMN2MIDI and MIDI2GMN are available at our web site `http://www.salieri.org/guido`

```
[ \key<"A"> a1/4 h c#2 d/8 e/16 f#16 _/8
  g#*1/4. {a1/4,c#,e2,a2} ]
```



```
[ \key<"C"> \meter<"4/4"> g1/4 e e/2
f/4 d d/2 c/4 d e f g g g/2 g/4 e e/2
f/4 d d/2 c/4 e g g c/1 ]
```



```
{ [ \tempo<"Vivace">
  \meter<"5/8"> \intens<"p"> \sl(\bm(g1*1/8 a b)
  \bm(b& c2) \bm(c# b1 a b& a&)) ],
[ \meter<"5/8"> \sl(g1*3/8 d/4 c#*3/8 d/4) ] }
```

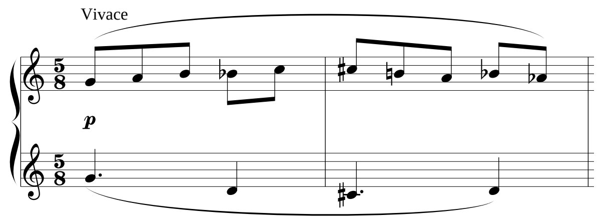


Figure 1: Simple examples of GUIDO Music Notation; more complex examples can be found in [17,18].

of the reasons for this seems to lie in the complexity of musical structure; just using a "new" format does not automatically lead to a simple and easy to use data structure.

To allow the use of XML-tools where needed, we have developed GUIDO/XML, a XML compliant format that completely encapsulates GUIDO within a XML structure. Using GUIDO/XML is simple: we provide tools that convert GUIDO Music Notation files into GUIDO/XML files and vice versa. Using this approach, we can continue to use GUIDO Music Notation and its associated tools (SALIERI, NoteAbility, NoteServer, ParserKit, etc.) but are also free to use any current or emerging XML tool.

One advantage of XML is its ability to store so called meta data. A piece of music can be associated with a composer, a title, a publisher, a publishing date and even version information. One can easily add new meta data fields encoding additional musical information (like for example performance-related data for a piece). Using GUIDO/XML in conjunction with a set of meta data information can lead to complete XML-compatible descriptions of structured music.

### Using GUIDO Music Notation for Musical Databases and MIR

As we have shown, GUIDO Music Notation offers an intuitive yet complete approach for representing musical data. Using GUIDO in musical databases is therefore a straight forward task: because it is a plain-text format, no additional tools are necessary to create, manipulate or to store GMN files. It is also possible to use standard text-compression tools to minimise storage space (the size of compressed GMN files compares to the size of MIDI files). By using existing tools like the GUIDO NoteServer[25], one can create conventional music notation from GUIDO descriptions quickly.

Because of its representationally adequate design, GMN is also very well suited for MIR: Queries can be written as (enhanced) GUIDO strings. Users with a background in GUIDO can specify even complex queries in an easy way. By using additional tools like a virtual piano-keyboard, even novice users are able to build queries quickly. In Section 4 it will be shown, how using GUIDO as the underlying music representation language simplifies the task of building query-engines and we also demonstrate, how a slight extension to GUIDO leads to an intuitive approach to approximate matching.

Other representation formats (like XML) do not provide this feature: a new query language has to be created in order to access the stored information. As there is no standard for musical queries (like SQL is for relational databases systems) a whole range of different musical query languages will be proposed in the future.

## 3 The Experimental GUIDO Database

As was shown in the previous section, GUIDO Music Notation is well suited as a general music representation language. Our prototypical MIR system is build on the basis of an experimental GUIDO Database that will be described in this section.

The GUIDO Database contains musical pieces stored as GMN files along with some additional information which is used for efficient retrieval (this will be discussed in more detail in the next section). Instead of building our musical database based on a conventional database system, we decided to implement it in Perl [1], using the regular file system for information storage. This design offers a numbers of advantages:

- The Perl language has good support for manipula-

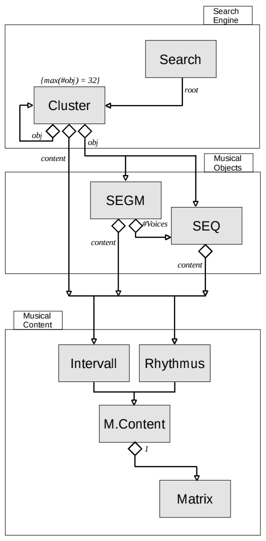


Figure 2: Overview of the object-oriented design of our experimental database and information retrieval system.

ting textual data (such as GUIDO or HTML data) and is well suited for rapid prototyping.

- Using PERL allows for very easy integration in online systems.
- Disk storage is cheap, and textual data can be compressed efficiently using general file compression techniques; furthermore, modern operating systems allow time-efficient file access through caching.
- It is easy and reasonably efficient to build index structures on a file system.
- Maintenance and updating of the database is relatively easy, since functionality of the operating system and underlying file system can be used.

One drawback of this approach is the fact that standard database functionality, such as concurrent write access, the implementation of access control, and transaction control would have to be implemented separately and are currently not supported. However, it should be noted that in the context of music information retrieval, write operations (i.e., modifications of or additions to the database) are relatively rare compared to read access (such as retrieval) and usually restricted to selected users. Interestingly, the same holds for many large and heavily used online biomedical and literature database systems. Our model is based on an off-line update mechanism, where pieces are added to the database by taking the database off-line and generating / updating the index structure while no other access is permitted.

Our implementation follows an object oriented design which is graphically summarised in Figure 2. Details of the implementation can be seen from the sources of our Perl modules, which are publicly available from `http://www.salieri.org/guido/mir`.

Currently, our database system contains about 150 files, most of which have been converted to GUIDO from other formats like abc and MIDI. Because the conversion from MIDI to GUIDO is a complex task that sometimes needs manual interaction, extending this corpus is time-consuming. However, we expect that based on recent improvements of our conversion tools, we will be able to extend our database to a much larger body of files. Our experimental system is not optimised for speed, and we are quite certain that we will need to increase its efficiency when operating on a much larger database.

## 4 The Experimental MIR Engine

Our music information retrieval approach is based on the "Query by Example" (QBE) paradigm [2]. QBE has the advantage that queries can be formulated in an easy and intuitive way. In many search and retrieval situations, users appear to prefer the QBE approach over the use of query languages, which support more complex queries like boolean expressions, wildcards, or regular expressions.

### Query Types

Many music information retrieval systems are primarily based on melodic, i.e., pitch-related information[4]. Types of melodic information that can be used for queries are absolute pitches, pitch-classes, intervals, interval classes (e.g., large/small intervals) and melodic trends (e.g., up/down/equal). Alternately or additionally, rhythmic information can be used as a basis for retrieval. Again, various types of rhythmic information can be distinguished: absolute durations, relative durations (or duration ratios), or trends (e.g., shorter, longer, equal).

Our prototypical MIR Engine supports queries that arbitrarily combine one out of five types of melodic information with one out of three types of rhythmic information. The melodic query features are the following: absolute pitch (such as c1, d#2, etc.), intervals (such as minor third, major sixth, etc.), interval types (such as second, fourth, etc.), interval classes (equal, small, medium, large), melodic trend (upwards, downwards, static). The three currently supported rhythmic features are absolute durations (such as 1/4, 1/8., etc.), relative durations (such as 1:2, 4:3, etc.), and rhythmic trends (shorter, longer, equal). All these features are determined for individual notes or pairs of notes, respectively, such that a query effectively specifies sequences of these features.

Since we are following a QBE approach, these various query types (and their combinations) correspond merely to different interpretations of the same musical query. For instance, the GMN fragment `[g1/4 e1/4 e1/4]` can be used as a purely melodic query, using ab-

[4]see [21] for an overview of MIR systems and their pitch representations

solute pitch. In this case, only the melodic sequence [g1 e1 e1] would be matched, regardless of rhythm. The same fragment, used as a purely rhythmical query would also match [e1/4 e1/4 e1/4], and even [_/4 _/4 _/4]. For information retrieval based on the QBE approach, this paradigm of "query = data + feature selection" is very natural; this applies particularly to multi-dimensional, complex data such as musical or graphical objects.

We can also distinguish exact retrieval, where the task is to find exact occurrences of the information specified in the query, or approximate (or error-tolerant) retrieval, where a certain amount of deviation between the query information and the data to be retrieved is permitted. Here, we first consider exact retrieval, and later discuss briefly an extension of our approach to approximate retrieval.

## Probabilistic Models

The music information retrieval approach taken here is based on the general idea of characterising and summarising musical structure using probabilistic models. Searching for a fragment with a specific musical structure (specified in a query) can then be done by probabilistic matching using these models. Here, we propose a rather simple approach, which is based on first-order Markov chains for modeling the melodic and rhythmic contours of a monophonic piece of music [15; 9]. Currently, we focus on horizontal queries only, i.e. queries which only involve monophonic music, and treat pieces with multiple voices (or chords) as collections of monophonic pieces.

Intuitively, a (discrete time) first-order Markov chain is a probabilistic model for a process which at each time is in a state, and at each time step probabilistically changes into a successor state (which can be the same as the current state) with a probability that only depends on the present state. Hence, first-order Markov chains are characterised by the transition probabilities $t_{ab}$ for entering state $b$ as the next state, when the current state is $a$.[5] The transition probabilities characterising a first-order Markov chain can be written in form of a square matrix $T = (t_{ab})$ whose rows and colum indices correspond to the states of the chain. It should be noted that first-order Markov chains with a finite set of states correspond to non-deterministic finite state machines (FSMs), and can also be seen as a special case of Hidden Markov Models (HMMs) where emissions for all states are deterministic [24].

In the application considered here, we conceptually use one first-order Markov chain for each melodic and rhythmic query type and each given monophonic piece. The states of these chains correspond to pitches for absolute pitch queries, to intervals for interval queries, to relative durations for relative rhythmic queries, etc. The corresponding transition probabilities are determined from frequency counts over neighbouring pitches, intervals, note durations, etc. which are normalised to obtain proper probabilities.

Figure 3 shows the transition probability matrices for the Markov chains characterising the sequences of absolute pitches and durations for the "Hänschen Klein"[6] example from the second row of Figure 1 as well as the corresponding representations as non-deterministic finite state machines; the latter representation is often more intuitive and concise.

The transition probabilities of these first-order Markov chains summarise statistical properties of the pieces in the musical database. When trying to find exact matches between a given query and pieces in the database, we can make use of the following simple observation: If for a given piece $p$, a transition that is present in the query (e.g., an upward fifth followed by a downward third) has probability zero, there is no exact match of the query in $p$. Unfortunately, the converse is not true: There are cases where there is no exact match of the given query in $p$, yet for any neighbouring features in the query, the corresponding transition probabilities in $p$ are greater than zero.

Generally, the key idea of information retrieval based on probabilistic models is the following: Given a piece $p$ and a probabilistic model $M(p)$ for this piece, this model can be used to generate pieces $p'$ with properties similar to $p$. Here, these properties are the transition probabilities of the first-order Markov chains we use for characterising sequences of features. To assess the potential of a match given a query sequence $q = q_1, q_2, ..., q_n$ (where the $q_i$ are individual features such as pitches) and a candidate piece $p$ from the database, we determine the probability $P(q|M(p))$ that the probabilistic model of $p$, denoted $M(p)$, generates the feature sequence corresponding to the given query $q$. For our simple probabilistic model, $M(p)$ is characterised by a matrix of transition probabilities $t_{ab}$, and the probability of generating the query sequence given that model is given by the product of the transition probabilities $t_{ab}$ which correspond to all neighbouring features in the query sequence:

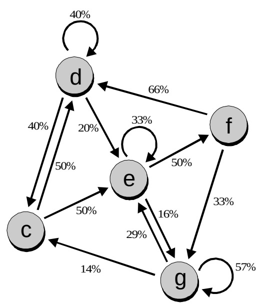
$$P(q|M(p)) = \prod\{t_{ab}|a = q_i, b = q_{i+1}, 1 \leq i < n\}$$

Intuitively, this probability score will be higher for pieces which contain many exact matches than for pieces which contain few exact matches, and as explained above, it will be zero for pieces which do not contain any exact matches at all. Since the probabilistic model we use is very simplistic and is certainly far from capturing all relevant (statistical) features of the pieces in the database, we cannot expect this intuition to be fully met. However, our practical experience with

[5]In this work, we only use homogenous Markov chains, i.e. chains, for which the transition probabilities do not change over time. In Section 6 we briefly discuss how and why a more general approach equivalent to using inhomogeneous chains might be advantageous.

[6]"Hänschen Klein" is a popupar German childrens song

$$t_{\text{pitch}} = \begin{array}{c|ccccc} & c & d & e & f & g \\ \hline c & 0 & 0.5 & 0.5 & 0 & 0 \\ d & 0.4 & 0.4 & 0.2 & 0 & 0 \\ e & 0 & 0 & 0.33 & 0.5 & 0.16 \\ f & 0 & 0.66 & 0 & 0 & 0.33 \\ g & 0.14 & 0 & 0.29 & 0 & 0.57 \end{array}$$



$$t_{\text{rhythm}} = \begin{array}{c|ccc} & \frac{1}{4} & \frac{1}{2} & \frac{1}{1} \\ \hline \frac{1}{4} & 0.67 & 0.28 & 0.06 \\ \frac{1}{2} & 1 & 0 & 0 \\ \frac{1}{1} & 0 & 0 & 0 \end{array}$$

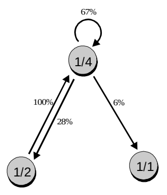


Figure 3: Transition probability matrices and Finite State Machines for absolute pitch and absolute rhythm

the experimental system described here indicates that even when using this simplistic probabilistic model, the correlation between probability scores and pieces containing exact matches is sufficient to be used as a basis for a music information retrieval mechanism.

Obviously, the transition probability matrices corresponding to related features, such as absolute pitches and intervals, are not independent, and in fact two matrices (one for absolute pitches, one for absolute durations) are sufficient as a basis for handling any type of query. However, in practice, there is a trade-off between the amount of pre-computed statistical data (transition probabilities), and the time required for matching a given query against a probabilistic model that might not be explicitly available.

*Note:* The techniques presented here do not directly support the efficient search of matches *within* a given piece (which might have been selected based on a high probability score for a given query). To efficiently search matches within a piece, conventional techniques, such as suffix trees (see, e.g., [20]) can be used. Alternatively, pieces can be segmented (manually, or automatically, using any suitable segmentation algorithm; see, e.g., [22]), and probabilistic modelling and matching can be applied to the segments individually.

## Hierarchical Clustering

The probabilistic matching technique described before can help to reduce search effort by eliminating some of the pieces that do not match a given query, and more importantly, by identifying promising candidate pieces based on their transition probability matrices only. However, a naive search for good candidates based on probability scores would still require to evaluate the query against the probabilistic models for all pieces in the database. For very large databases, or when short response times are required, this might be too time-consuming.

One way of addressing this problem is to organise the database in form of a tree, where each leaf corresponds to one element (i.e., piece) of the musical database. For a given query, we could now start at the root and follow the paths leading to the leaves which contain pieces which match the query. This would allow us to retrieve matches in time proportional to the height of the tree, i.e., logarithmic in the number of leaves for a balanced tree. In order to do this, we need a mechanism that at each node of the tree allows us to identify the subtree that is most likely to contain a match.

As a first approximation to such a mechanism, we use combined probabilistic models which summarise the properties of all sequences in a given subtree. Note that our first-order Markov chain model can be easily generalised to sets of pieces instead of single pieces: Given two pieces $p_1, p_2$, we combine the two transition probability matrices $T(p_1), T(p_2)$ derived from their respective interval sequences into one joint matrix $T(\{p_1, p_2\})$ by computing a weighted sum such that the resulting transition probabilities are equivalent to those that would have been obtained by deriving a transition probability matrix from the concatenation $p_1 \cdot p_2$ of the two sequences:

$$\begin{aligned} t(\{p_1, p_2\})_{ab} &= t(p_1 \cdot p_2)_{ab} \\ &= \frac{|p_1| t(p_1)_{ab} + |p_2| t(p_2)_{ab}}{|p_1| + |p_2|} \end{aligned}$$

This method generalises to the case of combining the models of more than two sequences in a straightforward way. Matching for these combined probabilistic models works exactly as for single pieces, and the probability scores thus obtained can be used to guide the search for matches in a tree structured index of the database. Figure 4 shows how the tree structure of transition matrices is build; the search for a pattern begins at the root matrix and then continues at the descendant matrices as long as these match the transition probabilities of the query.

Obviously, the topology of the tree as well as the decision how pieces and sets of pieces are grouped together can have a large impact on the efficiency of the proposed search mechanism. One potentially very fruitful approach for deriving tree structures is the use of hierarchical clustering techniques [10]. However, it is presently not clear whether similar pieces should be clustered together or whether clustering dissimilar pieces together would be more beneficial; the former approach might make it easier to identify larger sets of promising candidates for matches early in the

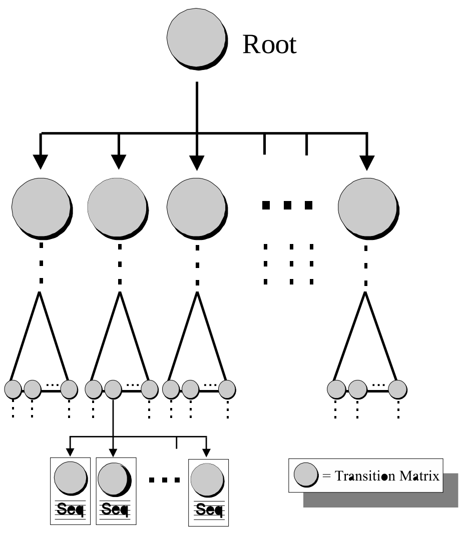


Figure 4: Tree structured index of the database

search, while the latter should facilitate selecting the most probable match from a set of pieces.

The issues arising in this context are rather complex and require thorough empirical analyses; we plan to further investigate and discuss these elsewhere. For our present prototype, we use a simple and rather arbitrary hierarchical clustering resulting in a balanced tree where each node has up to 32 children.[7] Furthermore, to speed up the search within this tree, for each node we store three bit matrices whose entries indicate whether the transition probabilities in the probabilistic model for the cluster corresponding to that node exceeds thresholds of 0, 0.15, and 0.3, respectively. Those threshold matrices are used for rapidly selecting the most promising subcluster at each internal node of the cluster tree that is visited during the search. (For details of this mechanism, see [13].)

Once again, it should be noted that the mechanisms introduced here serve a double purpose: They can potentially prune large parts of the search, for which exact matches cannot be encountered (based on the occurrence of transition probabilities with value zero), and they also heuristically guide the search such that promising candidate pieces are identified early in the search.

## Approximate Matching and Error-tolerant Search

Often, queries are inaccurate or may contain errors, and relevant matches cannot expected to be perfect matches. In other cases, a user querying a musical database system might be interested in "almost matches", which might indicate interesting musical similarities. One way of addressing this situation is to use exact matching in combination with "fuzzy" queries that support features such as melodic or rhythmic trends or interval classes. But this is not always the most appropriate approach, and many music approximate retrieval mechanisms instead or additionally support true error-tolerant search, which allows (penalised) mismatches when matching queries against pieces from the given musical database[8].

Our retrieval mechanism based on probabilistic models, although primarily developed for exact matching, quite naturally extends to a certain type of error-tolerant search. To that end, both the search for good candidate sequences, as well as the search within the sequences need to be modified. While we cannot discuss the technical details of these extensions here, we will outline the general ideas and provide a detailed description elsewhere.

To localise candidate sequences in an error-tolerant way, we could modify the probabilistic models associated with the individual pieces in the database with prior information by factoring pseudo-observations into all transition probabilities (this is a standard method in machine learning, which is applied frequently when probabilistically modelling sequence data, see, e.g., [4]). Intuitively, this would reflect a certain degree of uncertainty about the pieces in the database. The hierarchical clustering of the probabilistic models and the search process based on scoring the query sequence using these probabilistic models remains unchanged, but the whole process now supports imperfect matches, which are still penalised, but no longer ruled out.

The same effect can achieved by factoring the prior information into the query; this corresponds to allowing for errors or inaccuracies in the query. The mechanism is exactly the same, only now the prior is associated with the query and gets dynamically factored into the probabilistic scoring process rather than folded statically into the transition matrices stored in the database. Under this view, it is possible to allow the uncertainty associated with particular aspects of the query to be explicitly specified. For example, a user might be absolutely certain about the first and the second pitch of a melodic fragment used as a query, but less certain about the third pitch, and very uncertain about a fourth one.[9] We devised an extension of GUIDO Music Notation that allows to express such local uncertainties in a simple and intuitive way by using a the symbols "?" and "!". An instance of the example given above could thus be specified as `[g1! e1! e1? f1??]`. We are currently working on extending this concept to all melodic and rhythmic features supported by our MIR Engine.

[7]The number 32 is chosen in order to allow bit-parallel operations to be used on this data, see also [21].

[8]See [21] for an overview of MIR system and their approximate matching types.

[9]Note that this information need not necessarily be explicitly entered by the user — it could theoretically be added automatically based on a learned model of typical errors made by (particular) users, e.g., in the context of a Query-by-Humming approach.

The second stage of error-tolerant retrieval, locating approximate matches within candidate pieces, can be handled in many different ways, including standard methods based on edit-distances as well as techniques closely related to the one we discussed for finding candidate pieces in an error-tolerant way. The latter approach appears to be conceptually more elegant; we are currently developing a unified approximate retrieval mechanism based on this idea, which will be discussed in detail elsewhere. (The current implementation of our experimental Retrieval Engine contains a more ad hoc method for error-tolerant search, which we intend to replace with the theoretically more solid approach outlined above.)

## 5 Related Work

Over the last few years, a substantial amount of work on music database and retrieval systems has been published. While we cannot nearly cover all relevant approaches, we will outline and discuss similarities and differences between the key ideas of our approach and some recent and earlier work in the field.

Lemström and Laine recognised early that music representations which are more expressive than MIDI provide a better basis for certain retrieval tasks (see [20]; this paper also contains a nice overview of earlier work in music information retrieval). Recently, a number of musical database and retrieval systems have been developed in which music representations other (and more expressive) than MIDI are used (see, e.g., [5]); however, we believe that our use of GUIDO goes one step further than most of these in using a uniform formalism for representing pieces in the database and for formulating queries which is powerful enough to capture basically any aspect of a musical score. Although our present system only supports queries based on primary melodic and rhythmic features, we feel that the ability to extend this in a natural way to other musical concepts, such as key, metre, or barline information, is an important advantage of our approach.

Recently, a number of XML-based music representations have been proposed (see, e.g., [14; 26; 6]. While these offer some advantages by allowing the use of standard XML tools and certainly have the potential to represent arbitrary aspects of score-level music, we are not aware of any existing musical database and retrieval system based on an XML-representation. As discussed in Section 2 of this paper, XML-based representations share many desirable features with GUIDO. Aside from tool support, we cannot see any features which would make XML intrinsically suitable for content-based music information retrieval. While XML-based representations are typically much too verbose and syntactically complex to be used directly for musical queries, many aspects of our work (particularly our retrieval technique) are independent from the use of GUIDO as the underlying music representation, and can be easily applied to a broad range of other formats.

Sonoda et al. have been developing a WWW-based system for retrieving musical information from an online musical database based on the “Query-by-Humming” approach [19; 28]. Their system is based on MIDI as the underlying music representation, and their indexing and retrieval method, which uses dynamic programming for matching, has recently been optimised for efficient retrieval from large musical databases [29]. Similar to their approach, we follow the “Query-by-Example” paradigm (using GUIDO instead of MIDI) and acknowledge that matching against large databases, using dynamic programming or similar techniques, can be prohibitively inefficient, particularly in the context of an on-line system. Our probabilistic matching technique is fundamentally different from their “Short Dynamic Programming”. Their technique requires very large indeces (compared to the size of the database), while our probabilistic models are relatively compact. Their retrieval technique is a rather efficient stand-alone method for finding matches in the given database.[10] In contrast, we mainly focus on a technique for identifying promising candidate pieces in the database, which can be combined with various methods for identifying matches within a given piece (e.g., dynamic programming). Another difference between their approach and ours is the fact that they focus on melodic information alone, while we support queries that can combine various melodic and rhythmic features. Evidence for the importance of supporting such combined queries is given in [8], who use a fixed time-grid for rhythmical structure (in contrast to our more flexible rhythmical query types) and a retrieval method based on inverted file indexing.

An interesting approach to music information retrieval which has recently gained some popularity is the use of text-retrieval methods on suitably encoded music representations [23]. Although our system uses a text-based music representation, our approach to music information retrieval is radically different, and actually more related to techniques for biomolecular sequence analysis and genomic information retrieval (see, e.g., [11; 4]). It is our belief that musical information is in many ways inherently different from text, and that specific properties of musical data should be exploited for music information retrieval. To that end, sequence retrieval methods developed for text data can potentially provide a valuable starting point (as has been the case for biomolecular sequence analysis), but ultimately will have to be complemented and augmented by techniques specifically developed for musical data. The probabilistic matching approach we propose provides a basis for such techniques, and the overall design of our system facilitates such extensions. Furthermore, text-based methods can be used in the context of our approach for locating matches within candidate pieces identified by our probabilistic matching technique.

[10]Since the only evaluation of their approach we are aware of is based on a database of mainly random pieces, we feel that the accuracy of the method in practice is hard to assess.

Generally, our probabilistic modelling approach is based on characterisations of the underlying musical data which can be potentially useful for purposes other than information retrieval, such as analysis or composition (see, e.g., [9]).[11] In this sense, our approach is related to work by Thom and Dannenberg [31; 30], who use probabilistic models and machine learning techniques for characterising melodies.

Finally, let us point out a general problem with almost any work on content-based music information retrieval we are aware of (including our own work presented here): the lack of a corpus of music for testing the efficiency and accuracy of music retrieval systems. Part of the reason for this is the lack of a commonly used and widely supported music interchange format. We believe that GUIDO Music Notation has the potential to remedy this situation, and we are currently working on translating various collections of musical material into GUIDO, in order to integrate these into our experimental musical database.

## 6 Conclusions and Future Work

In this paper we have presented the concept of a database system for structured, score-level musical information and introduced a query-by-example mechanism for retrieving information based on a variety of melodic and rhythmic search criteria. The underlying music retrieval method uses probabilistic models and a hierarchical clustering of the database for pruning and heuristically guiding the search. We also presented an extension of GUIDO Music Notation, the music representation language we use for the pieces in the database as well as for queries, which allows expressing localised uncertainty in musical queries; and we briefly described an extension of our retrieval mechanism that uses such extended queries for approximate probabilistic matching.

A first prototype of the database system and retrieval engine has been implemented and tested on a set of about 150 relatively simple musical pieces in GUIDO Notation Format. This experimental system has been equipped with a WWW interface and is available online at `http://www.salieri.org/guido/mir/`. Our experience with this small prototype suggests that the approach presented here can provide a solid foundation for larger and more complex database and information retrieval systems for structured musical data.

Conceptually as well as with respect to the implementation, this work is still in a relatively early stage, and many aspects of it will be further explored and refined in the future. On the practical side, an obvious extension of our work is to test our system and methods on larger musical databases. To that end, we have begun to include a broad range of structured musical data, including the "Essen Folksong Collection" [27] into our dataset. Finally, we hope to get access to the data coming out of Fujinaga et al.'s "Optical Music Recognition System" [7], where a large collection of American sheet music is automatically converted into GUIDO descriptions. With this additional data, we hope to be able to conduct some tests with thousands to ten-thousands of pieces in GUIDO Music Notation in the near future. We also intend to improve the integration with the experimental database/MIR system with other GUIDO tools and applications, in particular with the latest version of the GUIDO NoteServer [25] (for visualising the musical data), converters (in particular GUIDO-to-MIDI for playback), and analysis tools which are currently under development.

Another direction we would like to explore in the near future is to support queries which allow the use of GUIDO tags in addition to melodic and rhythmic information. Clearly, the information represented by tags in the GUIDO data comprising the elements of the database can be musically very meaningful, and in many contexts we consider it desirable to include such information in musical queries. This could be very useful, e.g., in order to support the specification of tonality, metre, or instrument information in a query; similarly, constraints on the metric position within bars could be expressed in queries by including barlines, and including expressive markings or dynamic information could help to make approximate queries more specific. The probabilistic matching mechanism presented here can be extended in various ways to accommodate queries including tag information, and determining a theoretically elegant and practically effective solution to this problem is a challenging problem for future research.

Even when just considering the melodic and rhythmic query types supported in our present system, it might be interesting to investigate more powerful probabilistic models as a basis for the characterisation of the musical data which is at the core of our retrieval mechanism. Obviously, higher-order Markov models could be used to capture more of the local structure, and additional statistical information which better resembles aspects of the global structure of larger pieces could be used in addition to simple Markov chains. Furthermore, larger pieces can be more appropriately handled by segmenting them into smaller fragments (using standard segmentation approaches), for which probabilistic models are then constructed individually. This way, local structure can be captured more adequately and probabilistic matching based on the fragment models will be more accurate.

Finally, we are interested in extending our approach beyond purely horizontal queries by allowing polyphonic features to be included in queries. We see two fundamental approaches for such an extension: Allowing chords and possibly tags referring to harmonic context to be included in monophonic queries, or supporting full polyphonic queries that specify simultaneous monophonic voices. We believe that our general approach should in principle be applicable to either type

[11]The simple probabilistic models used here, as well as more complex models, can be used for generating statistically similar fragments of music.

of polyphonic query, but clearly, substantial further investigation will be required to devise and implement the corresponding retrieval algorithms. Overall, we are convinced that the work presented here will provide a good basis for these and other generalised retrieval tasks.

## References

[1] `http://www.perl.com`

[2] For a definition of "Query by Example" look at `http://www.whatis.com/definition/0,289893,sid9_gci214554,00.html`

[3] Bainbridge D. The Role of Music IR in the New Zealand Digital Library project. Proceedings ISMIR 00.

[4] Baldi P.; Brunak S. Bioinformatics: the machine learning approach. MIT Press, 1998.

[5] Boehm C.; MacLellan D.; Hall C. MuTaDeD'II – A System for Music Information Retrieval of Encoded Music. Proceedings ICMC 00. 174–177.

[6] Castan G. NIFFML: An XML Implementation of the Notation Interchange File Format. *in* Computing in Musicology (12). MIT Press, 2001.

[7] Choudhury G.; DiLauro T.; Droettboom M.; Fujinaga I.; Harrington B.; MacMillan K. Optical Music Recognition System within a Large-Scale Digitization Project. Proceedings ISMIR 00.

[8] Clausen M.; Engelbrecht R.; Meyer D.; Schmitz J. PROMS: A Web-based Tool for Searching in Polyphonic Music. Proceedings ISMIR 00.

[9] Dodge, C.; Jerse T. Computer Music: Synthesis, Composition, and Performance, 2nd ed., Shirmer Books: New York, 1997. 361–368.

[10] Duda R.O.; Hart P.E. Pattern Classification and Scene Analysis. New York: Wiley, 1973.

[11] Durbin; Eddy; Krogh; Mitchison Biological sequence analysis: Probabilistic models of proteins and nucleic acids. Cambridge University Press, 1998.

[12] Ghias A.; Logan J.; Chamberlin D.; Smith B.C. Query by Humming: Musical Information Retrieval in an Audio Database. Proceedings of ACM Multimedia 95, 231–236.

[13] Görg M. GUIDO/MIR – Ein GUIDO basiertes Music Information Retrieval System. Masters Thesis, TU Darmstadt, Fachbereich Informatik, 1999.

[14] Good M. Representing Music Using XML. Proceedings ISMIR 00.

[15] Grimmett, G.R.; Stirzaker, D.R. Probability and Random Processes, 2nd ed. Clarendon Press: Oxford, 1994. Chapter 6.

[16] Hoos H.; Hamel K.; Renz K.; Kilian J. The GUIDO Notation Format – A Novel Approach for Adequately Representing Score-Level Music Proceedings ICMC 98. 451–454.

[17] Hoos H.; Hamel K.; Renz K.; Using Advanced GUIDO as a Notation Interchange Format. Proceedings ICMC 99. 395–398.

[18] Hoos H.; Hamel K.; Renz K.; Kilian J. Representing Score-Level Music Using the GUIDO Music-Notation Format. Computing in Musicology, Vol 12, Editors: W.B.Hewlett, E.Selfridge-Field; MIT Press, 2001.

[19] Kageyama T.; Mochizuki K.; Takashima Y. Melody Retrieval with Humming. Proceedings ICMC 93. 349–351.

[20] Lemström K.; Laine, P. Musical Information Retrieval using Musical Parameters. Proceedings ICMC 98, 341–348.

[21] Lemström K.; Perttu S. SEMEX – An Efficient Music Retrieval Prototype. Proceedings ISMIR 00.

[22] Melucci M.; Orio N. The use of melodic segmentation for content-based retrieval of musical data. Proceedings ICMC 99. 120–123.

[23] Pickens J. A Comparison of Language Modeling and Probabilistic Text Information Retrieval – Approaches to Monophonic Music Retrieval. Proceedings ISMIR 00

[24] Rabiner L.R.; Juang B.H. An introduction to hidden Markov models. IEEE ASSP Magazine, January 1986, 4–15.

[25] Renz K.; Hoos H. A WEB-based Approach to Music Notation using GUIDO. Proceedings ICMC 98. 455–458.

[26] Roland P. XML4MIR: Extensible Markup Language for Music Information Retrieval. Proceedings ISMIR 00.

[27] Schaffrath, H. The Essen Folksong Collection in the Humdrum Kern Format. D. Huron (ed.). Menlo Park, CA: Center for Computer Assisted Research in the Humanities, 1995.

[28] Sonoda T.; Goto M.; Muraoka Y. A WWW-based Melody Retrieval System. Proceedings ICMC 98. 349–352.

[29] Sonoda T.; Muraoka Y. A WWW-based Melody-Retrieval System – An Indexing Method for a Large Melody Database. Proceedings ICMC 00. 170–173.

[30] Thom B. Learning Models for Interactive Melodic Improvisation. Proceedings ICMC 99. 190–193.

[31] Thom B.; Dannenberg R. Predicting Chords in Jazz. Proceedings ICMC 95. 237–238.