A POLYPHONIC MUSIC RETRIEVAL SYSTEM USING N-GRAMS

Shyamala Doraisamy
Fac. of Comp. Sc. and IT
University Putra Malaysia

Stefan Rüger
Department of Computing
Imperial College London

ABSTRACT

This paper describes the development of a polyphonic music retrieval system with the n-gram approach. Musical n-grams are constructed from polyphonic musical performances in MIDI using the pitch and rhythm dimensions of music. These are encoded using text characters enabling the musical words generated to be indexed with existing text search engines. The Lemur Toolkit was adapted for the development of a demonstrator system on a collection of around 10,000 polyphonic MIDI performances. The indexing, search and retrieval with musical n-grams and this toolkit have been extensively evaluated through a series of experimental work over the past three years, published elsewhere. We discuss how the system works internally and describe our proposal for enhancements to Lemur towards the indexing of 'overlaying' as opposed to indexing a 'bag of terms'. This includes enhancements to the parser for a 'polyphonic musical word indexer' to incorporate within document position information when indexing adjacent and concurrent musical words. For retrieval of these 'overlaying' musical words, a new proximity-based operator and a ranking function is proposed.

1. INTRODUCTION

N-grams have been widely used in text retrieval, where a sequence of symbols is divided into overlapping constant-length subsequences. A character string formed from n adjacent characters within a given text is called an n-gram. N-gramming has recently been adopted as an approach for indexing sound-related data. An experimental system by [4] which used a database of folksongs, allowed indexing of the entire musical work. Using this approach for full music indexing of monophonic data, each folksong of the database was converted into a sequence of pitch intervals ignoring individual durations. Using a gliding window, this sequence was fragmented into overlapping length-n subsections. These n-grams were then encoded as 'text' words or 'musical words', a term coined by [4] that we have continued to adopt. These are basically a string of characters with no semantic content. As a consequence, each folksong could be represented as a 'text document', and regular text search engines can be used.

Several studies have investigated the use of n-grams and information retrieval (IR) approaches for music information retrieval (MIR) [4, 9, 7]. However, the construction of n-grams has been confined to monophonic sequences. We introduced a method to obtain n-grams from polyphonic music using the pitch and rhythm dimensions of music, and through a series of experimentation, the robustness of musical n-grams with polyphonic music retrieval was shown [2].

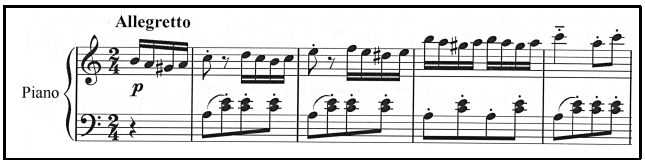
The IR process can be briefly described based on a general IR model. Information items in a collection are preprocessed and indexed. During information retrieval, a user's query is processed and formulated to the format requirements of the indexed collection. Information items that are most similar to the query would be retrieved and presented to the user based on a similarity computation. Developments on this basic model has resulted in rather diverse IR models, but with the common general emphasis of retrieving information relevant to a request, rather than direct specification of a document. Modern IR systems include sophisticated indexing, searching, retrieving technologies, similarity computation algorithms and user-interfaces. We utilise these for our MIR system.

This paper is structured as follows: The approach to constructing n-grams from polyphonic music data using the pitch and rhythm dimension of music is presented in Section 2. Section 3 presents our polyphonic music retrieval system design. The user interface of our system demonstrator is shown in Section 4.

2. MUSICAL N-GRAMS

2.1. Pattern extraction

With polyphonic music data, the approach described in Section 1 for generating n-grams from monophonic sequences would not be applicable, since more than one note may be sounded at one point in time. The approach proposed by us for n-gram construction from polyphonic data is discussed briefly in this section. Polyphonic music pieces are encoded as an ordered pair of onset time (in milliseconds) and pitch (in MIDI semitone numbers) and these are sorted based on the onset times. There may possibly be a few different pitches corresponding to one particular onset time with polyphonic music data. Pitches with the same or similar onset time together as musical



0	71	window 1
150	69	window 2
300	68	window 3
450	69	
600	57	window 4
	72	
900	64	
	60	

Figure 1. Excerpt from Mozart's 'Alla Turca' and the first few events with onset times and pitches

events are grouped together. Using the gliding window approach as illustrated in Figure 1, this sequence of events is divided into overlapping subsequences of n different adjacent events, each characterised by a unique onset time. For each window, we extract all possible monophonic pitch sequences and construct the corresponding musical words. These words would be used for indexing, searching and retrieving from a polyphonic music collection.

Interval representation of pitches, i.e., the difference of adjacent pitch values, are used rather than the pitch values themselves, owing to their transposition-invariance. For a sequence of n pitches, we define a sequence of $n-1$ intervals by

$$\text{Interval}_i = \text{Pitch}_{i+1} - \text{Pitch}_i. \tag{1}$$

Figure 1 illustrates the pattern extraction mechanism for polyphonic music: The performance data of the first few notes of a performance of Mozart's 'Alla Turca' was extracted from a MIDI file and converted into a text format, as shown at the bottom of Figure 1. The left column contains the onset times sorted in ascending order, and the corresponding notes (MIDI semitone numbers) are in the right column. When using a window size of 3 onset times, we get one interval sequence for the first window $[-2\ -1]$, one for the second window $[-1\ 1]$ and two for the third window, $[1\ -12]$ and $[1\ 3]$. The polyphonic data in the fourth window gives rise to 4 monophonic pitch sequences within this window.

$$\text{Ratio}_i = \frac{\text{Onset}_{i+2} - \text{Onset}_{i+1}}{\text{Onset}_{i+1} - \text{Onset}_i}. \tag{2}$$

Although MIDI files encode the duration of notes, we do *not* take the actual or perceived duration of notes into consideration, as this is nearly impossible to determine from actual performances or raw audio sources. By contrast, onset times can be identified more readily with signal processing techniques.

For a sequence of n onset times we obtain $n-2$ ratios using Eqn 2 and $n-1$ interval values using Eqn 1. An n-gram representation which incorporates both pitch and rhythm information using intervals (I) and ratios (R) would be constructed in the form of

$$[I_1\ R_1\ \ldots\ I_{n-2}\ R_{n-2}\ I_{n-1}]. \tag{3}$$

Using the example of Figure 1, the combined interval and ratio sequences from the first 3 windows of length 3 are $[-2\ 1\ -1]$, $[-1\ 1\ 1]$, $[1\ 1\ -12]$ and $[1\ 1\ 3]$. Note that the first and last number of each tuple are intervals while the middle number is a ratio.

2.2. Pattern encoding

In order to be able to use text search engines, the n-gram patterns have to be encoded with text characters. One challenge that arises is to find an encoding mechanism that reflects the patterns we find in musical data. With large numbers of possible interval values and ratios to be encoded, and a limited number of possible text representations, classes of intervals and ratios that clearly represent a particular range of intervals and ratios without ambiguity had to be identified. For this, the frequency distribution for the directions and distances of pitch intervals and ratios of onset time differences that occur within the data set were obtained. The frequency distribution of all occurring intervals (in units of semitones) of 3096 polyphonic MIDI files was analysed. According to the distribution, the vast bulk of pitch changes occurs within one octave (i.e., with semitone differences between -12 and $+12$), and a good encoding should be more sensitive in this area than outside it. We chose the code to be the integral part of a differentiable continuously changing function, the derivative of which closely matches the empirical distribution of intervals.

$$C(I) = \text{int}\left(X \tanh(I/Y)\right), \tag{4}$$

where X and Y are constants and $C(I)$ is the code assigned to the interval I. The function int returns the integer portion of its argument. X has the effect of limiting the number of codes, and with 26 letters (a-z) adopted for the study, it is accordingly set to 27 in our experiments. Y is set to 24 for this achieves a 1:1 mapping of semitone differences in the range $\{-13, -12, \ldots, 13\}$. In accordance with the empirical frequency distribution of intervals in this data-set, less frequent semitone differences (which are bigger in size) are squashed and have to share codes. The codes obtained are then mapped to the ASCII character values for letters. In encoding the interval direction, positive intervals are encoded as uppercase letters A$-$Z and negative differences are encoded with lowercase letters a$-$z, the code for no difference being the numeric character 0.

The frequency of the logarithm of all occurring ratios of the data collection in the sense of Eqn 2 was analysed. Peaks clearly discriminated ratios that are frequent. Mid-points between these peak ratios were then used as the bin boundaries which provide appropriate quantisation ranges. Ratio 1 has the highest peak, as expected,

and other peaks occur in a symmetrical fashion where, for every peak ratio r, there is a symmetrical peak value of $1/r$. From our data analysis, the peaks identified as ratios greater than 1 are 6/5, 5/4, 4/3, 3/2, 5/3, 2, 5/2, 3, 4 and 5. The ratio 1 is encoded as Z. The bins for ratios above 1, as listed above, are encoded with uppercase letters A−I and any ratio above 4.5 is encoded as Y. The corresponding bins for ratios smaller than 1 as listed above are encoded with lowercase letters a−i and y, respectively.

Musical words obtained from encoding the n-grams generated from polyphonic music pieces with text letters are used in the construction of index files. Queries, either monophonic or polyphonic are processed similarly. The query n-grams are used as search words in a text search engine.

2.3. N-gramming strategies

Several problems had been identified in the use of n-grams with polyphonic music retrieval. These problems and solutions were tested. We carried out evaluations with several possible indexing mechanisms [2, 3]. In this subsection we summarise our conclusions from the evaluations that informed in our system design.

2.3.1. Path restrictions

When the window size n is large or when too many notes could be sounded simultaneously, the number of all monophonic combinations within a window becomes large. Consider, for example, the case of $n = 5$, with ten different notes played at each of the 5 onset times. As a result there would be $10^5 = 100,000$ different monophonic paths through this window: this appears to be an impractical way of indexing a tiny bit of music! In this case, we suggest restricting the possible combinations to variations of upper and lower *envelopes* of the window, i.e., we only allow monophonic sequences which run through the highest two notes per event (variation of the upper envelope) or which run through the lowest two notes per event (variation of the lower envelope). In the above example there would only be $2 \cdot 2^5 = 64$ different monophonic paths through this highly polyphonic passage of music.

2.3.2. Position Information

Proximity-based retrieval has been widely used with text. In general, proximity information can be quite effective at improving precision of searches [1]. Adopting its use with music data has been very limited — a preliminary study performed by [7] using monophonic musical sequences. Apart from just storing the document id, the location for the occurrence of a term or within document position(s) can be added to the index data. With exact positions where a word appears in a text, single-word queries can be extended as a phrase. A more relaxed version of the phrase query is the proximity query. In this case, a sequence of single words or phrases is given, together with a maximum allowed distance between them. The words and phrases may or may not be required to appear in the same order as the query. The first word's location is identified and if all terms are found within a particular distance, the term frequency is incremented for the given phrase query [1].

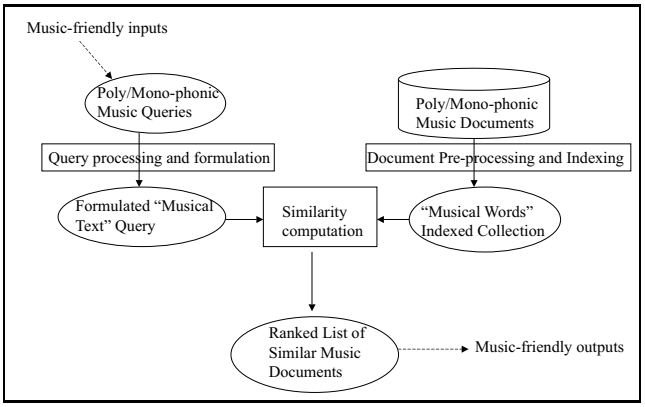


Figure 2. Polyphonic Music Retrieval System Overview

In considering within document position and adjacency of terms for polyphonic music, not only the 'listening order'/'playing order' based on the timeline that has to considered but also the concurrency of this 'order'. The only sequentially that has been preserved with the n-gram approach for musical words generation when indexing has been n contiguous notes [7]. Polyphonic music would require a new approach towards indexing position information using 'overlaying' positions of polyphonic musical words. This would take into consideration the time-dependent aspect of polyphonic musical words compared to indexing a 'bag of terms'. In using the n-gram approach towards indexing polyphonic music, apart from the adjacent musical words generated based on a time line, words may be generated concurrently at a particular point in time. Preliminary investigation performed [3] emphasizes the need for a 'polyphonic musical word position indexer'.

3. POLYPHONIC MUSIC RETRIEVAL SYSTEM

The scope of our MIR system is to retrieve all similar pieces given either a monophonic or polyphonic musical query excerpt. For music similarity assumptions, evaluation of the approaches have been based on the relevance assumption used by [9]. The polyphonic music retrieval system design is shown in Figure 2 and the following subsections describe the processes shown.

3.1. Document Preprocessing and Indexing

The document collection used contained almost 10,000 polyphonic MIDI performances. These were mostly classical music performances which had been obtained from the Internet [http://www.classicalarchives.com]. Files that converted to text formats with warning messages on the validity of the MIDI file such 'no matching offset' for a particular onset, by the midi-to-text conversion utility [6], were not considered for the test collection.

Index	Pos.	Pitch	Rhythm	n	Y	#R.Bins
PPR4	yes	yes	yes	4	24	21
PPR4ENV	yes	yes	yes	4	24	21
PR3	no	yes	yes	3	24	21
PR4	no	yes	yes	4	24	21
PR4ENV	no	yes	yes	4	24	21
PR5ENV	no	yes	yes	5	24	21

Table 1. Musical word format and index file variants

Documents are preprocessed using our n-gram approach with several variants of indexes are developed based on the strategies described in Subsection 2.3. Queries are usually subjected to the same kind of processing. Index mechanisms that combine various elements were evaluated and following are those recommended from our investigation:

PR3, PR4: The pitch and rhythm dimensions are used for the n-gram construction, as described in Subsection 2.1. $n = 3$ or $n = 4$ were values of n adopted. For interval encoding, the value of Y in Eqn 4 is set to 24. For the ratio encoding, all 21 bin ranges that had been identified as significant, as listed in Subsection 2.2, were used.

ENV as suffix: The generation of n-grams is restricted to the variations of the upper and lower envelopes of the music, as discussed in Subsection 2.3.1.

PPR4: Incorporation of position information to PR4 as discussed in Section 2.3.4.

Table 1 shows a summary of the used word and index file formats listed in alphabetical order (with Pos. indicating if position information is included with the index data).

3.2. Query Processing and Formulation

Both monophonic and polyphonic queries can be made to the indexed polyphonic collection. The musical words obtained from the query document can be queried as a bag of terms against the collection indexed in the same way. What is currently being investigated are queries formulated as structured queries to be queried against the collection indexed with the includion of position information.

3.2.1. Bag of terms

Queries are processed in the same approach to the indexed collection. The musical word format variants are listed in Table 1. Adjacancy and concurrency information of the musical words are not considered.

3.2.2. Internal Structured Query Format

The use of the various proximity-based and structured query operators available within [5] are currently being investigated for the inclusion to the system. Query languages allow a user to combine the specification of strings (or patterns) with the specification of structural components of the document. Apart from the classic IR models, IR models that combine information on text content with information on the document structure are called structured text retrieval models [1] [1]. Following are operators within Lemur that were tested:

Sum Operator: #sum ($T_1 \ldots T_n$) The terms or nodes contained in the sum operator are treated as having equal influence on the final result. The belief values provided by the arguments of the sum are averaged to produce the belief value of the #sum node.

Ordered Distance Operator: #odN ($T_1 \ldots T_n$) The terms within an ODN operator must be found in any order within a window of N words of each other in the text in order to contribute to the document's belief value.

A few initial tests using the known item search with the ODN operator showed, as expected, a poor performance. Retrieval using more complex query formulations were then looked into [3]. A monophonic theme extracted from Figure 1 was encoded as:
bZaZA aZAZC AZCIB CIBib BibZa bZaZA aZAZD AZDIA DIAia AiaZa aZaZA aZAZG AZGZb GzbZa bZaZA aZAZB AZBZb BzbZa bZaZA aZAZC
The query was then reformulated as (This reformulation would be done automatically by the system internally and would be transparent to the user):

```
#SUM(   #ODN3(bZaZA aZAZC)
        #ODN3(AZCIB CIBib)
        ...
        #ODN3(bZaZA aZAZC))
```

Based on the analysis of our retrieval results [3], a more specific operator for music retrieval is required and the introduction of MODN (Musical Ordered Distance Operator) is discussed in the following subsection.

3.3. Similarity Computation

The main aim of the system is to retrieve all similar pieces, given a monophonic or polyphonic musical excerpt as a query and using the relevance assumption adopted by [9]. Currently the system retrieves based on the vector-space IR model. Ongoing work in adapting the structured retrieval model for proximity-based retrieval is discussed in the second part of this subsection.

3.3.1. Vector-space model

To index, search and retrieve, the Lemur Toolkit was selected as the research tool as it supported a number of IR

[1] In using the Boolean model, a classic IR model, for a query 'white house' to appear near the term 'president', it would be expressed as ['white house' and 'president']. Classic IR models include the Boolean, vector-space and probabilistic models [1]. A richer expression such as 'same-page(near('white house','president')))' would require a structured query language.

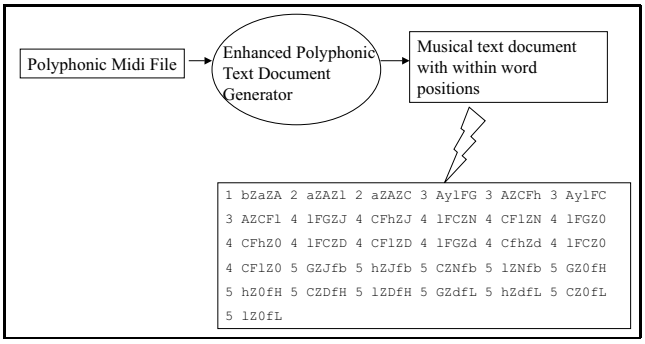


Figure 3. Musical text document with within document word positions

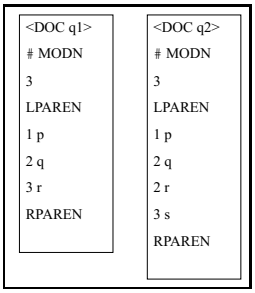


Figure 4. Query Documents

models which we investigated for MIR. Models supported include the use of language models (based on a probabilistic approach) and the vector-space model. Similar to the findings from the study by [7], the language modelling approach did not perform as well with musical n-grams and teh vector-space model was adopted. Details of adapting this model in Lemur are discussed in detail in [10].

3.3.2. Structured retrieval model and polyphonic position indexing

For proximity-based retrieval, the structured query language provided by Lemur was investigated and in this section we discuss the enhancements being tested for polyphonic music retrieval.

Using the first five onsets of the music excerpt in Figure 1, the document generated from our polyphonic text document generator would be similar as the bag of terms shown in Subsection 3.2.2. The polyphonic text document generator is the tool we developed (modules added to utilities by GN MIDI Solutions [6]) for the conversion of a polyphonic MIDI file into a polyphonic musical text document. This was enhanced to output 'overlaying' positions. The parser in [5] had to be modified to parse these new position information format. 'Overlaying' positions as shown in Figure 3 would need to be recorded by the index.

The existing proximity-based operator ODN retrieves only documents that contain *all* query terms in the similar order within a given proximity distance are retrieved. Intercepting onsets from a polyphonic document would generate n-grams that are dissimilar to its corresponding monophonic query, resulting in non-retrieval of relevant documents. Erroneos queries would also generate dissimilar n-grams from the relevant document. A 'musical ordered distance operator' (MODN) should enable this difference between n-grams generated from the query and the relevant polyphonic document to be reflected by a similarity measure [3], i.e., retrieval that partially satisfies query conditions would be needed. Ranked retrieval should be based on *the number of query terms found within a given proximity distance and not the condition that all terms must be found within a given proximity distance*. The requirement of MODN therefore would be to retrieve documents based on a rank whereby documents that match the query with the *The Highest* number of query n-grams within a given proximity distance would be retrieved with the highest rank, i.e., rank 1.

We are currently investigating a simple scoring function based on the notion of 'fuzzy match points' for MODN. Therefore in formulating a scoring function for MODN, we start at the very beginning and look at the notion of match points for term or phrase frequencies instead [1]. The term match point defined by [1] refers to the position in the text of a sequence of words which matches (or satisfies) the user query. If a string appears in three positions in the text of a document d_j, we say that the document d_j contains three match points. With MODN, we continue to look at match points but the match points for polyphonic musical documents would be the position in the text document that matches the first term available of the query sequence. Apart from assigning a match point only based on the first term of the query sequence, any of the following terms in the query sequence can assume the first position if any of the prior terms do not exist in that particular sequence. For the score calculation, 1 point is given as a score for the first term from the query sequence that is found in a text document and 1 point for each of the following term that is found within the given proximity.

This score calculation is shown using the following example. Two sample queries, q1 (a monophonic query) and q2 (a polyphonic query) are given in Figure 4. The query documents are shown in the format required by Lemur's parser. For an example of four relevant documents, D1: 1 p 2 q 2 r 3 r 4 s 4 t, D2: 1 p 2 q 3 x 4 y, D3: 1 p 2 q 3 p 4 q 5 r and D4: 1 p 2 q 3 a 4 b 5 c 6 d 7 r 8 p 9 q 10 r, the scores for each of these documents for each of the queries would be as follows:

The relevance scores of the documents for q1 are: D1 = 5 (from sequences (p q r) and (p r)), D2 = 2 (from sequence (p q)), D3 = 6 (from two sequences (p q r)), and D4 = 5 (from sequences (p q) and (p q r)).

The relevance scores of the documents for q2 are based on two monophonic sequences – (p q s) and (p r s): D1 = 9 (from sequences (p q s), (p r s) and (p r s)) , D2 = 2 (sequence (p,q)), D3 = 4 (2 sequences (p q)), and D4 = 6 (sequences (p q), (p q) and (p r)).

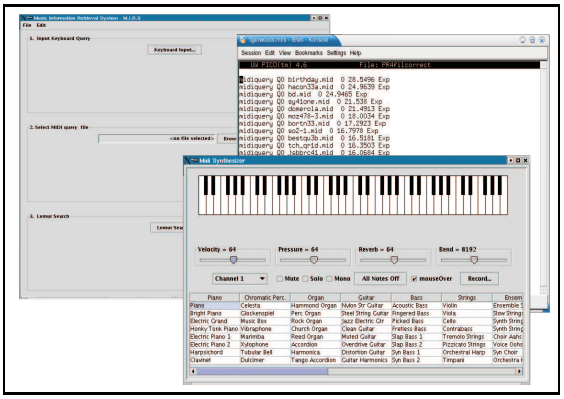


Figure 5. User interface

Figure 6. QBH System Interface

4. SYSTEM DEMONSTRATOR

The screen shots of our early polyphonic music retrieval system demonstrator is shown in Figure 5. Apart from selecting a poly/monophonic MIDI file from the dialog box provided, a query input can also be made via a 'graphical keyboard' enabling monophonic performance inputs only. MIDI files generated will be processed by the musical document generator discussed Subsection 3.3.2. generating query terms in the same format as the indexed documents as shown in Table 1. The screen shot shows retrieval results as a text file with a ranked list of documents. For the retrieval shown, the musical text documents were indexed with the PR4ENV format with Lemur version 1.9. The system is currently being tested with Lemur Version 2.2 being enhanced for the inclusion of the new musical words parser, MODN and its scoring module. Future work includes evaluation of this.

This development work is part of the Multimedia Knowledgement Research Group's framework for content-based information retrieval. Figure 6 shows the QBH interface developed where one can hum a query which would be transcribed by a pitch tracker written by [8] and converted to PR4ENV. Future work also includes integrating this and other interfaces to music inputs that have been developed include a text contour input and polyphonic audio file input [http://km.doc.ic.ac.uk] to the current early prototype.

5. CONCLUSION

We have outlined a polyphonic music retrieval system design and described its development in detail. These include details of the document preprocessing and indexing, query processing and formulation and the similarity computation using musical n-grams from polyphonic music data. An early prototype has been shown and we are currently investigating enhancements for incorporating proximity-based retrieval.

6. ACKNOWLEDGEMENTS

This work is partially supported by the EPSRC, UK.

7. REFERENCES

[1] Baeza-Yates, R and Ribeiro-Neto, B. *Modern Information Retrieval*. ACM Press Addison Wesley, 1999.

[2] Doraisamy, S and Rüger, S. "Robust Polyphonic Music Retrieval with N-grams", *Journal of Intelligent Information Systems*, 21(1), 53–70.

[3] Doraisamy, S and Rüger, S. "Position Indexing of Adjacent and Concurrent N-Grams for Polyphonic Music Retrieval", *Proceedings of the Fourth International Conference on Music Information Retrieval, ISMIR 2003*, Baltimore, USA, 2003, pp 227–228.

[4] Downie, S. "Evaluating a Simple Approach to Music Information Retrieval: Conceiving Melodic N-grams as Text". *PhD Thesis*, University of Western Ontario, 1999.

[5] Lemur Toolkit. http://www-2.cs.cmu.edu/lemur.

[6] Nagler, G. GN MIDI Solutions. http://www2.iicm.edu/Cpub.

[7] Pickens, J. "A Comparison of Language Modeling and Probabilistic Text Information Retrieval", *1st Annual International Symposium on Music Information Retrieval, ISMIR 2000*, Plymouth, USA, 2000.

[8] Tarter, A. "Query by Humming", Project Report, Imperial College London, 2003.

[9] Uitdenbogerd, A and Zobel, J. "Melodic Matching Techniques for Large Databases", *Proccedings of ACM Multimedia '99*, pp 57-66.

[10] Zhai, C. "Notes on the Lemur TF-IDF model", http://www-2.cs.cmu.edu/lemir/1.9/tfidf.ps, Unpublished report.