Distance Metrics and Indexing Strategies for a Digital Library of Popular Music

C. Francu, C. G. Nevill-Manning

Computer Science, Rutgers University
Piscataway, New Jersey 08854, USA

ABSTRACT

People identify powerfully with music: someone might say "that's my song!" but they are unlikely to say "that's my book!" or "that's my picture!" A digital library of popular music therefore has the potential to be a compelling application of information retrieval technology. Such a library requires a retrieval method that is appropriate for a non-technical audience. Experiments on "query by humming," which attempt to retrieve a tune based on sampled recording of a user singing an excerpt, have heretofore concentrated on relatively small, well-curated collections. Scaling up introduces three problems: availability of source material, an increase in false positive hits, and slower retrieval. We describe our experiments with MIDI files, propose a new, more accurate distance metric between queries and songs, and discuss possibilities for efficient indexing.

1. INTRODUCTION

Digital libraries until now could hardly be described as popular: they tend to be based on esoteric, scholarly sources close to the interests of digital library researchers themselves. We are developing a digital library containing the quintessence of popular culture: music. The principal mode of searching this library will be by sung query: the system should be able to answer the kinds of queries that shop assistants in music stores deal with every day, where a customer can sing a tune, but can't remember the title or artist.

The operation of our musical digital library is sketched in Figure 1. At the time of collection creation, MIDI files are gathered from the internet, and indexed based on their note sequences. At retrieval time, a user's sung query is transformed from a waveform to a sequence of pitch-duration events by a pitch tracker. We are using an off-the-shelf pitch tracker: pitch tracking is beyond the scope of the current project. Potential close matches are identified using the precalculated inverted index, and the most promising tunes are compared to the query using a more expensive distance function, and returned to the user ranked according to distance. The distance function must be robust in the face of pitch-tracking errors, singing errors, and differences between the query and the tune in both tempo and key.

Libraries of music in digital form have been discussed by [2], [3], [4]. and [5], but their experiments have been based on small collections in the order of a few thousand tunes. Scaling up introduces a number of new problems, including source, specificity and efficiency.

First, in order to populate a large library, we chose to use music in MIDI format, because 300,000 tunes are readily available in this format on the world wide web. MIDI describes each note as a discrete event with a pitch, start and stop time, and amplitude. The drawback of the MIDI format is that it was designed for musical performance rather than analysis, lacks metadata to describe the form of the song (e.g. which voice is the melody, where the chorus is located), and does not necessarily correspond to musical notation. Also, MIDI files are not usually of high enough quality for casual listening. We propose using MIDI files as surrogates for high quality sampled recordings, such as CD tracks or MP3 format songs. Under this scheme, the MIDI file allows us to construct an

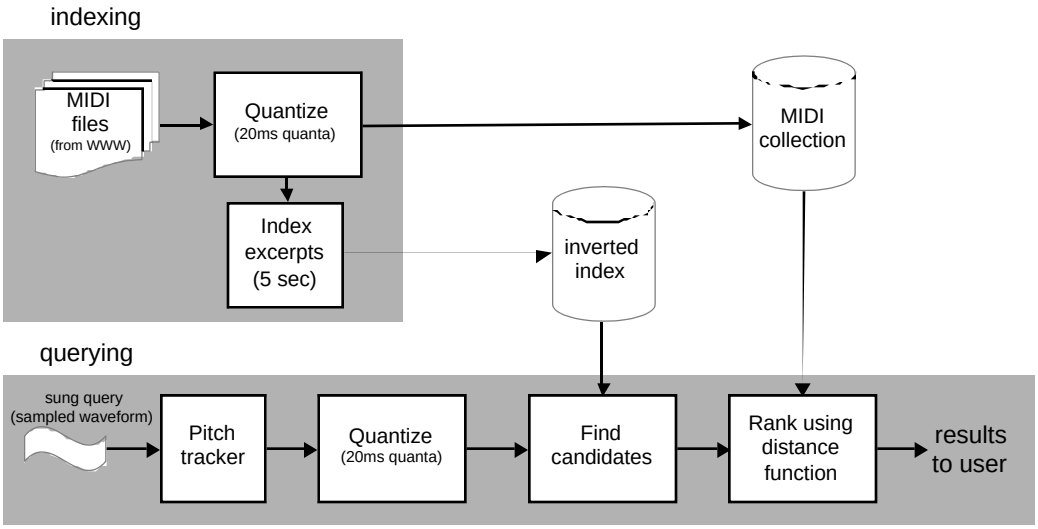

Figure 1 The digital library of popular music: index- and query-time processes

index based on its symbolic representation of the tune, but the user would be provided with high quality recordings corresponding to the matching MIDI files.

Second, as the collection grows, the distance between a query and a tune in the collection must be more accurately computed in order to maintain the same rate of false positive hits. Section 2 describes a new approach that matches pitch-duration contours of the query against excerpts from tunes in the collection at a variety of transpositions[1] and tempos.

Third, the time to compute the distance between a query and each of the tunes in the collection scales linearly with the size of the collection. This is unacceptable as collection sizes grow by orders of magnitude, and an indexing scheme is necessary to keep retrieval time close to constant. A preliminary indexing scheme is described in Section 3.

2. DISTANCE METRICS

The techniques described [4] and [5] treat the query and tune as sequences of pitch-duration tuples, and use dynamic programming to match a query to a tune. That is, the match is described as the minimum number of deletions, insertions or replacements of notes required to transform the query into the tune excerpt. This is analogous to matching notes in a musical score. The disadvantage of this scheme is that it fails to take into account the time-based nature of music. By mapping a quarter note to an eighth note, one tune is out of time by an eighth note. The inherent synchronization of musical sequences distinguishes it from applications where the edit distance has been successfully used, notably in computational biology where biochemical processes routinely introduce insertions, deletions and mutations into DNA sequences. [7] takes a different approach, where tunes are represented by a graph of pitch versus time, so called *contours*. Ó Maidín compares two tunes by computing the area between two contours, as shown in Figure 2. This work has a number of drawbacks: it considers only simple contours where each pitch lasts for a multiple of an eighth note, it assumes that both tunes have the same tempo, and it does not consider silence or deal with polyphony.

Our approach computes the distance between a query and an excerpt from a tune by finding a key and tempo for the query that minimizes the average difference in pitch between them. That is, the query is transposed and scaled in such a way as to minimize the area between the two pitch-duration contours. First, the contour is quantized into a sequence of notes of equal duration. Whereas Ó Maidín could express any note as a multiple of an eighth note, the note durations in a MIDI file usually have a greatest common divisor of one millisecond (ms). Most notes last a few hundred ms, and can be as short as 8ms. If multiple notes start within 10ms of each other, they are perceived as simultaneous. Taking these factors into account we quantize the query and tunes into periods of 20ms.

To explain the procedure, we assume initially that the query is sung in the same tempo as the song, the song contains one monophonic sequence of notes, and the song and query contain no silent intervals. Under these assumptions the

[1] We mean transposition in the musical sense of changing the key of a tune, rather than in the mathematical sense.

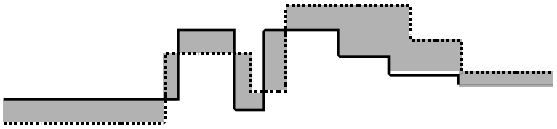

Figure 2 Two pitch-duration contours and the area between them

query and tune contours can be expressed as sequences of numerical note pitches:

$q = q_1, q_2, ..., q_m$

$t = t_1, t_2, ..., t_n$

The distance between a query q and a tune t for a given scaling is defined as:

$$d(q,t) = \frac{1}{m}\min_{j=0}^{n-m}\left(\min_{\Delta}\sum_{i=1}^{m}\left|q_i - t_{i+j} + \Delta\right|\right)$$

The distance is the minimum average difference in pitch between the query and the song when the query is both shifted along the song by j quanta and transposed by Δ semitones. This distance is a metric.

In order to allow different tempos between query and tune we rescale the query with factors varying between 0.5 and 2 in coarse steps of 0.1. After finding the best coarse scale, a similar search is done at three progressively finer granularities to arrive at a final scale.

A complicating factor with MIDI files is that tunes usually consist of multiple channels. Each channel usually corresponds to a different instrument. Furthermore, there may be polyphony (several notes played simultaneously) within a single channel. We deal with multiple channels by computing the distance from the query to each channel and reporting the minimum distance. Polyphonic channels are made monophonic by deleting in each time interval every note but the one with the highest energy. The energy of a note is computed as the energy of the acoustic wave front and is directly proportional to the square of the product of the amplitude and the frequency of the note.

When the query and the song contain periods of silence, the minimal area is computed between matching non-silence intervals. The area is then divided by the number of note-note pairs instead of the total length of the query. We require that the query be at most 66% silence, and that the number of note-note pairs be at least 75% of the non-silence in the query.

The distance between a query and a monophonic note sequence can be computed in $O(mn)$, where m and n are the lengths of the query and the sequence. We do not have the space to describe the algorithm in detail, but it consists of shifting the query along the song, computing the optimal transposition of the query as the median of the differences between corresponding pitches, and computing the area of the shift after transposing the query optimally.

The average-case complexity can be reduced to $O(m+n)$ by evaluating how promising the distance is at each shift. We first construct a histogram of the pitches in the query and song fragment. Only if the histograms are similar is the actual

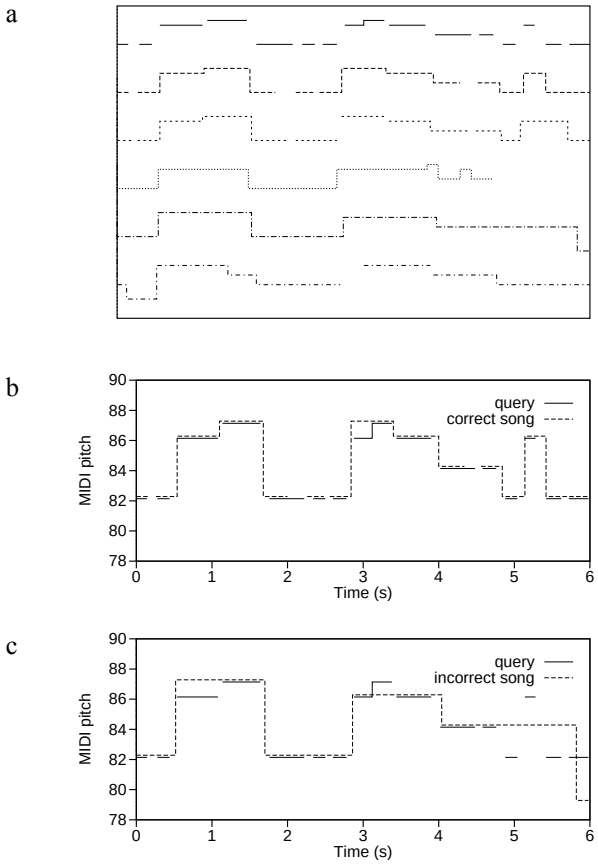

Figure 3 Melodic contours of query and matches to "Every breath you take" by The Police (a) query (topmost) and close matches (b) query and correct match (c) query and incorrect match "Losing my religion" by REM. MIDI pitch is in semitones, where 60 is middle C.

distance computed. If the mean is substituted for the median in calculating the optimal transposition, then the histogram can usually be updated incrementally each time the query is shifted. Therefore most of the shifts will have O(1) computing time, giving an amortized linear time.

The algorithm can be summarized as follows:

1. Transform polyphonic channels to monophonic channels (voices) by choosing the note of higher energy whenever two notes or more overlap.
2. For each voice in the song do
sample voice at 20 ms
at finer and finer scales do
rescale the query to current scale
sample the query at 20msec intervals
for each alignment of the query with the tune,
Find the transposition of the query that minimizes the area between the two pitch contours
3. Report the minimum area found

3. INVERTED INDEX

The problem of indexing music is similar to that of indexing protein sequences. Neither musical nor protein sequences naturally split into words as text does, and similar sequences can differ at any location. For this reason, we take inspiration from BLAST [1], the preeminent DNA and protein retrieval system. BLAST operates in two phases: it first weeds out the vast majority of sequences in the database that clearly cannot match, then computes a more expensive distance on the remaining sequences.

Our current weeding function creates an inverted index of two second tune segments where each segment is quantized into intervals of 200ms. This produces many different intervals, so extreme intervals, such as those that someone is unlikely to sing, are discarded, and the remaining keys are clustered. That is, intervals that are closer than 0.2 from each other are merged, with a single interval chosen as a representative for the merged cluster. This reduces the number of keys to 210,000 for 10,000 tunes. A subset of promising matches for a query can be quickly retrieved based on this coarse quantization, and the more expensive distance used to rank them.

4. RESULTS

We used a corpus of 10000 songs downloaded randomly from the World Wide Web. Twenty queries were matched against the corpus. The queries were between 4 and 22 seconds long. For each query, at least one matching tune existed in the corpus. The system successfully identified several versions of each song, and for any one query all versions of the correct song were among the first 16 matches. For most of the queries the ranked list of best matches including all matching versions contained less than three false matches. The system was able to identify modified versions of the songs, e.g. different tempo, changes in the rhythm during the song, syncopated versions, slight changes in the tune, etc. The system performed best on medium size queries (about 6 to 10 seconds). Short queries matched false fragments, while long queries tended to misalign, due to the fact that subjects do not maintain a consistent tempo over long periods. Figure 3 provides an example of matching pitch contours. Figure 3a shows a query (top), and beneath it the five closest contours in order of similarity from top to bottom. The query has many small periods of silence, a side-effect of the pitch-tracker, and thus has few vertical lines between consecutive pitches. The two closest contours are the correct tune, "Every breath you take" by the Police, and the remaining three are incorrect matches. Distances in average pitch difference are 0.05, 0.22, 0.29, 0.56 and 0.56 respectively. Figure 3b shows the best alignment of the query and the closest contour. Note that the largest mismatch at 3 seconds is a pitch-tracking error. Figure 3c shows the best alignment of a close incorrect match, "Losing my religion" by REM.

In order to calibrate the performance of the automated system, six human subjects were asked to identify six songs based on a sung queries. The subjects had no time constraints and were allowed to replay the queries. A subject was judged to have recognized a song if they could name it or sing another fragment of it. Subjects could often sing a song but not recall

the title. None of the subjects were trained musicians, but were familiar with all of the songs in the experiment.

On average, subjects recognized 1.5 out of the 6 songs, and nobody recognized more than three. One subject failed to recognize a single song. In contrast, the system recognized four songs, outperforming all the subjects. No human subject recognized either of the two songs that the computer system missed.

5. CONCLUSION

A digital library of popular music will be a compelling demonstration of information retrieval technology if queries can be satisfied efficiently, and results have high relevance. We have described initial experiments in constructing accurate, computationally tractable distance measures for musical queries, and suggested some possibilities for indexes that will allow near-constant-time retrieval. Future work will focus on making these techniques robust and testing on a wider range of subjects with varying musical ability. We expect that dealing with diverse query quality will present significant challenges. For example, how well do people keep time over the duration of a ten-second query? How likely are people to change key midway through a query? We hope that segmenting the query will help with both these problems, at the cost of additional computation. An interesting application of this technology is in mobile devices, where screen real estate is scarce, and a sung query for selecting songs on an audio device could prove invaluable. A wireless MP3 player could download tunes on demand based on a short hummed excerpt. We don't just want to build a digital library of popular music: our aim is to produce a truly popular digital library of music.

6. REFERENCES

[1] Altschul, S.F., Gish, W., Miller, W., Myers, E.W. & Lipman, D.J. "Basic local alignment search tool." *J. Mol. Biol.* 215:403-410, 1990.

[2] Bainbridge, D., Nevill-Manning, C.G., Witten, I.H., Smith, L.A., & McNab, R.J. (1999) "Towards a Digital Library of Popular Music" *Proceedings of Digital Libraries,* 1999, Fox, E.A. and Rowe, N. (Eds.) pages 161–169.

[3] Downie, J.S. "Evaluating a simple approach to music information retrieval: conceiving melodic n-grams as text," *Ph.D. Thesis* London, Ontario: University of Western Ontario.

[4] Ghias, A, Logan, J, Chamberlin, D. and Smith, B.C. (1995), "Query by humming: musical information retrieval in an audio database," Proc. Third International Conference on Multimedia '95, San Francisco, CA, 231–236

[5] McNab, R.J., Smith, L.A., Witten, I.H., Henderson, C.L., & Cunningham, S.J. "Toward the digital music library: tune retrieval from acoustic input". *Proceedings of Digital Libraries*, 1996, pages 11–18.

[6] Mongeau, M. & Sankoff, D. "Comparison of Musical Sequence". *Computers and the Humanities* 24, 161–175, 1990

[7] Ó Maidín, D. "A geometrical algorithm for melodic difference". *Computing in Musicology* 11, 65–72, 1998