A WWW-based Melody-Retrieval System
- An Indexing Method for A Large Melody Database -

Tomonari Sonoda and Yoichi Muraoka

School of Science and Engineering, Waseda University

3-4-1 Ohkubo Shinjuku-ku, Tokyo 169-8555, Japan.

sonoda@muraoka.info.waseda.ac.jp, muraoka@muraoka.info.waseda.ac.jp

ABSTRACT:
This paper describes an indexing method for a WWW-based melody-retrieval system, takes a melody sung by a user as a search clue and sent over the Internet and uses it to retrieve the song's title from a large database. In previous works, it was difficult to reduce the melody-retrieval time since it was difficult to make indexes for melody sequences. We therefore propose an efficient indexing method which is based on DP (dynamic-programming) matching. Testing of our method against a database containing about 10,000 melody sequences showed that its retrieval time is about 1/1,000 that without an index and that its matching is highly accurate.

1 Introduction

A WWW-based melody-retrieval system that takes a melody sung by a user as a query requires a robust matching method that can quickly and accurately identify the desired melody in a large database.

The biggest challenge to developing such a method is creating an index of the melody sequences that reduces the retrieval time to an acceptable level. Because several notes in the input melody, including even the first note, will likely be wrong, it is difficult to apply general indexing methods, such as sorting or hashing. It was also difficult to build a WWW-based melody-retrieval service without utilizing an database index since the system can not process a lot of network accesses at a time.

Previous systems [Kageyama et al., 1993, Sonoda et al., 1998] utilized DP matching to calculate the distance between a user's input melody and each melody in a database when the input melody does not completely match a database melody. DP matching, however, requires a long time to complete its calculations.

We propose an effective indexing method which takes advantage of DP matching and reduces the retrieval time. In this method, if the distance calculated by DP matching for two long melody sequences is close, there is assumed to exist similar shorter sequences in those two sequences and the shorter sequences are used for indexing.

Testing of our method against a database containing about 10,000 melody sequences showed that its retrieval time is about 0.78 seconds, about 1/1,000 that without an index. Its matching accuracy was highly accurate, about 92% for 120 inputs from 12 different people. This level is high enough for a melody-retrieval over the Internet.

2 WWW-based Melody-Retrieval System

This section describes the overview of our system and proposes a method for indexing melody sequences.

2.1 Overview

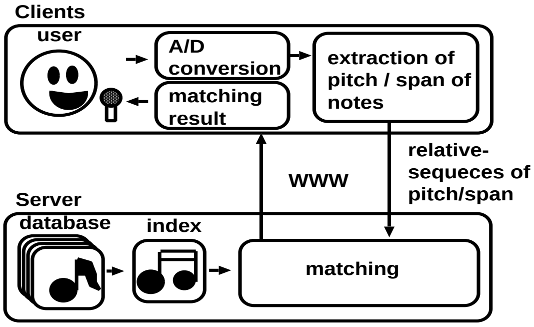

Figure 1: Melody-retrieval system with a large database.

Our current system differs from the one we proposed in 1998 [Sonoda et al., 1998]. As shown in Fig. 1, our current system consists of a melody-retrieval server and user-side client for use by a Netscape Plug-in or a JAVA applet. The server has a database consisting of standard MIDI file (SMF) data and melody data generated by singing and an index of the database.

A user inputs a melody by singing from an any part of a piece of music. The key and tempo can be arbitrarily chosen. The client system assumes that each input note begins with a voiceless consonant and ends with a vowel (e.g. ta-ta-ta).

After A/D conversion, the client extracts sequences of pitch and span of notes from the input. These sequences are converted into relative-value sequences of pitch and span to normalize the key and tempo. These relative-value sequences are then transmitted to the server.

After receiving the relative-value sequences, the server executes a matching process. It compares the input sequences with relative-value sequences for each database melody by using the database index that the server generated in advance. The titles of the corresponding melodies are then ranked based on the distance calculated by the matching process.

Finally, the list of ranked titles is transmitted to the client system.

2.2 Indexing Method

The most important process is a matching process which uses the melody database index. The results of our estimations, presented in section **6.3**, showed that it is quite difficult to reduce the retrieval time without indexes.

Traditional text-matching method such as sorting, hashing, or n-gram often use the first character of an input query to reduce the retrieval space and time. Those matching method, however, can not be easily applied to melody matching. A user's input melody consists of various errors and even the first note in the input is uncertain.

To solve this problem, we propose an effective matching for generating a melody database index. In this "short DP matching", the distance between two melody sequences is calculated by comparing short parts of each sequence with each other.

3 Normal DP Matching

This section describes a normal DP matching method and the problem in applying it for indexing melody sequences.

3.1 Overview

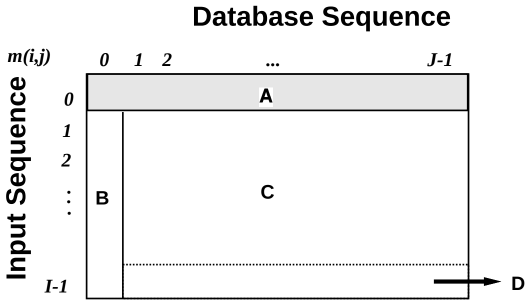

Figure 2: DP matching.

In normal DP matching, a DP matching matrix, $m(i,j)$ like that illustrated in Fig. 2, is used to obtain distance D between sequence Q, with length I, and sequence S, with length J. Elements Q and S are assumed to take the integer set represented by C, and the penalty between elements Q and S is represented by d.

$$\begin{array}{ll} Q & = q_0, q_1, q_2, \ldots, q_{I-1}\ , \quad q_i \in C \\ S & = s_0, s_1, s_2, \ldots, s_{J-1}\ , \quad s_j \in C \\ d(q_i, s_j) & = |q_i - s_j| \\ & : \text{penalty between } q_i \text{ and } s_j.\ q_i, s_j \in C \end{array}$$

Under these assumptions, D is calculated as follows.

- **Step A**

  $m(0,j) = d(q_0, s_j)$, where $0 \leq j < J$.

- **Step B**

  $m(i,0) = \infty$, where $0 < i < I$.

- **Step C**

  $m(i,j) = d(q_i, s_j) + min(m(i-1,j), m(i-1,j-1), m(i,j-1))$, where $1 \leq i < I$ *and* $1 \leq j < J$.

- **Step D**

  $D(Q,S) = \min\{m(I-1,j)\ |1 \leq j < J\}$

3.2 Problems

As shown in the Fig. 2, DP matching takes $O(I * J)$ calculation time for a database melody and a lot of time is needed for the entire database melodies. To solve this problem, several approaches that utilize DP matching can be thought as follows.

1. An approach which assumes that the user sings the melody from one of only a few points in the piece of music. This method omits retrieval space by calculating only limited parts of a melody in using DP matching [Kageyama et al., 1993].
2. An approach which executes DP matching with all input patterns against the database in advance. This approach uses the matching results when a user inputs a query and quickly obtains the list of ranked music titles without calculation.

These methods, however, respectively has the following problems.

1. It is difficult to set the adequate points on each database melody because the limiting points are hard to identify.
2. DP matching for a large database took enormous retrieval time even for one time retrieval and it was difficult to calculate matching distances for all input patterns.

As explained in the next section, our short DP matching method overcomes these problems.

4 Short DP Matching

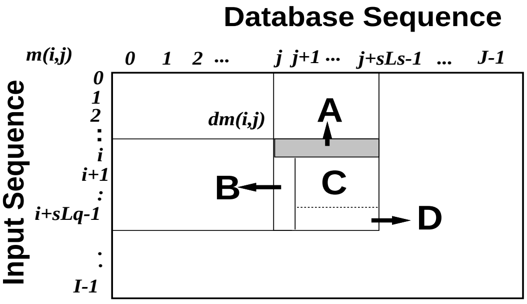

Figure 3: Short DP Matching.

As illustrated in Fig. 3, short DP matching calculates distance sD between sequences Q and S (described in **3.1**) as follows.

First, it extracts short sequences $sQ(i)$ in Q and $sS(j)$ in S:

$$\begin{aligned} sQ(i) = &\quad q_i, q_{i+1}, q_{i+2}, \ldots, q_{i+sL_q-1}\ , \quad q_i \in C \\ sS(j) = &\quad s_j, s_{j+1}, s_{j+2}, \ldots, s_{j+sL_s-1}\ , \quad s_i \in C \\ &\quad where \quad 0 \leq i \leq I - DL \ and\ 0 \leq j \leq J - DL. \end{aligned}$$

Distance $sD1$ between sQ and S is defined using normal DP matching distance D:

$$sD1(sQ(i), S) = \min\{D(sQ(i), sS(j))\ |0 \leq j < J - sL_s\};$$

$sD1(sQ(i), S)$ is almost equal to $D(sQ(i), S)$ if sL_s is sufficiently longer than sL_q.

Finally, short DP matching distance sD is defined as

$$sD(Q, S) = \Sigma_i\ sD1(sQ(i), S).$$

According to this definition, distance $sD(Q, S)$ increases with the number of input errors. If Q completely matches S, distance sD equals zero because sD is defined by using normal DP matching.

5 Indexing and Retrieving

This section describes an indexing method which utilizes proposed short DP matching and shows the retrieving method by using the index.

5.1 Indexing

Our system uses short DP matching distances as index data. The indexing method is as follows.

1. First, sequences $sS(j)$ $(j = 0, 1, \ldots, J - sL_s)$ are extracted from each melody sequence, S, of a database.

2. Short DP matching between each $sS(j)$ and each short input sequence sQ of all possibilities, whose lengths are sL_q, are executed N^{sL_q} times, where N is the number of the integer set that notes in sQ can take.

3. Processes 1 and 2 are repeated for all database melodies.

4. Finally, $sS(j)$ of all melodies are ranked for each sQ in order of their short distances and the short sequence IDs and their distances are stored as index data.

In step 2, if the number of integer set, N, is too large to calculate all distances, it can be reduced by using appropriate thresholds or by categorizing [Kageyama et al., 1993, Ghias et al., 1995, Kjell et al., 1998, Sonoda et al., 1998]. In step 4, if the number of database melodies, M, is too large for all index data items to be stored, r $(r < M)$ data items from the top of the ranked sequences are stored.

5.2 Retrieving

Retrieving using the described index is done as follows.

1. Q is assumed to be an input sequence with length I. Short sequences $sQ(i)$ $(i = 0, 1, \ldots, I - sL_q)$ are extracted from Q. Each length of $sQ(i)$ is sL_q.

2. For each $sQ(i)$, the stored lists of IDs and distances of database short sequences, sS, are obtained from the index.

3. The sum of distances between $sQ(i)$ and each indexed short sequences is calculated for each database melody. This sum represents the distance for short DP matching.

4. All database melodies are ranked by their sum.

Our retrieval server makes the list of music titles by using the ranked melodies and returns it to the client as a matching result.

6 Experiments and Results

We evaluate the proposed method and verify its effectiveness in this section. We then evaluate the overall performance of our system.

6.1 Conditions

To test our retrieval server, we ran it against a database containing 10,206 melodies, consisting of 10,000 randomly generated pieces of music and 206 pieces of music from various genres, such as rock, pop, and folk music.

We tested our system on a PC with a 300-MHz $Celeron^{TM}$ processor and 128 MB of RAM. We chose the short sequence lengths $sL_q = sL_s = 3$ for short DP matching. We set r, the number of data items stored, to 100, and we reduced the number of integer sets, N, to 9 by using the thresholds proposed in a previous paper [Sonoda et al., 1998]. To obtain better results when using the index, we used normal DP matching for music ranked in top 100 after we obtain the matching result by using our index.

We used 120 input queries from 12 people (8 men and 4 women). Each query was limited to a maximum of eight seconds, and the sampling rate was 8 kHz. Fast Fourier transformation with 512 points to a frame (16 msec) was used to convert the acoustic data into melody data.

6.2 Index Volume

Table 1: Calculation time and required disk volume for indexing.

number of melodies	time	volume
10,206	1376 minutes	531,441 KB

To evaluate our indexing method for practical use, we measured the calculation time and required disk volume for indexing. As shown in Table 1, the calculation time was less than a day (1,376 minutes), and about 532 MB of disk space was needed. These values are short and small enough to make our method practical.

6.3 Retrieval Time

Table 2: Retrieval time with/without index.

	with	without
average time	0.78 sec	875.4 sec

average time for five time retrievals.

To test the effectiveness of the index, we compared retrieval times with and without our index. Table 2 shows the average retrieval times for five time retrievals of the same input melody (note length of 12). Retrieval using our index reduced the retrieval time to 1/1,120 of that without using it.

6.4 Matching Accuracy

The matching accuracy of our retrieval system was 92.5 % (111/120 inputs) under the conditions described in **6.1**. It was measured by counting the cases that a subject's desired music title was ranked at the top of the retrieval results. This shows that our system achieved high matching accuracy using the index.

7 Conclusion

We have proposed an indexing method for a WWW-based melody-retrieval system. Using this method, we achieved a matching accuracy of 92.5% for a database of 10,206 pieces of music and a retrieval time 1/1,120 that not using an index. This method is thus practical enough to be used on the Internet.

Our current system utilizes only standard MIDI file data and sung melody data for building a database. It would be more useful if we could use other formats. Therefore, we plan to build databases by using various music sources, such as MP3 files or music CDs, so that many users can easily build melody-retrieval sites on the Internet.

References

[Kageyama et al., 1993] T.Kageyama, K.Mochizuki, and Y.Takashima : *Melody Retrieval with Humming*, ICMC Proc., pp.349-351, 1993.

[Ghias et al., 1995] Asif Ghias and Jonathan Logan : *Query By Humming – Musical Information Retrieval in an Audio Database*, ACM Multimedia 95, Electronic Proc., 1995.

[Kjell et al., 1998] Kjell Lemström and Pauli Laine: *Musical Information Retrieval Using Musical Parameters*, pp.341-pp.348, ICMC'98 Proc., 1998.

[Sonoda et al., 1998] Tomonari Sonoda, Masataka Goto, Yoichi Muraoka: *A WWW-based Melody Retrieval System*, pp.349-pp.352, ICMC'98 Proc., 1998.