A JXTA-based Music Information Retrieval System

Hyosook Jung and Seongbin Park*
Department of Computer Science Education
Korea University, Seoul, Korea
{est0718, psb}@comedu.korea.ac.kr

Abstract: In this paper, we present a JXTA-based system for contents-based music information retrieval. The system finds matching melodies from a set of XML documents that encode music contents. The XML documents are stored in a native XML database and XPath query language is used to extract the information about the structure of music data. The matching algorithm utilizes the geometric hashing technique [Wo97] by analyzing geometric properties of both XML documents and music contents. The system preprocesses extracted music contents and stores the information about melody patterns in a relational database. When a user sends a query about a melody, it searches melody patterns that have similar geometric properties as the query melody.

1 Introduction

In this paper, we present a contents-based music information retrieval (CBMIR) system using JXTA technology [Go01]. Given a query melody, the system searches similar melodies in XML-based music documents using the geometric hashing algorithm [Wo97]. Figure 1 shows the overall process of the CBMIR when a query melody is given. The searching process proceeds in two steps; preprocessing and recognition. The preprocessing step consists of the following steps.

1. XML-based music documents are stored in a native XML database (i.e., Xindice [Url3]).
2. The contents of music and structural property are extracted in the native XML database using XPath.
3. The content is represented by 3D coordinates and its geometric property is computed.
4. A hash table is constructed using the geometric and structural property.
5. The geometric property and the hash table are stored in a relational database.

*To whom correspondence should be addressed.

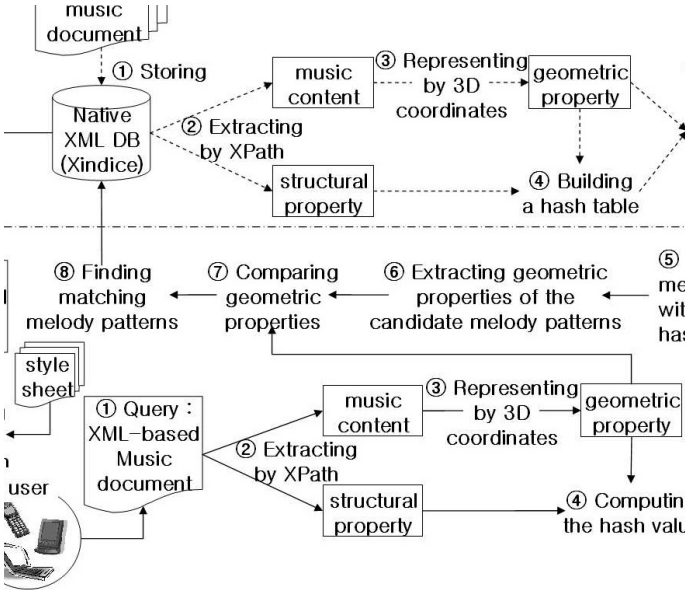

Figure 1: The overall process of the proposed CBMIR system. The dotted arrow is preprocessing step and the solid is recognition step

After processing the music document in advance, we can find melody patterns that are similar to the query melody in the database. The recognition step consists of the following steps.

1. A user sends a query that is a XML-based music document.
2. The content of music and structural property are extracted in the query document by using XPath.
3. The query content is represented by 3D coordinates and its geometric property is computed.
4. A hash value is created by its geometric and structural property.
5. The melody patterns with the hash value are searched in the relational database.
6. If there are the melody patterns, their geometric properties are extracted in the relational database.
7. The properties are compared with the geometric property of the query.
8. The music which contains the matching melody patterns is found.
9. The XML-based document of the music is found in the native XML database.

10. The result presentation is created by applying a XSLT stylesheet to the document. It can support the tailored presentation according to user's device.

This paper is organized as follows. Section 2 describes related works. Section 3 explains how the structure of music is modeled. Section 4 describes the process of searching a melody pattern in a set of XML documents. The proposed JXTA-based CBMIR system is discussed in section 5 and section 6 describes the experimental results. Finally, the paper concludes in section 7.

2 Related works

The modeling of music notation is complex and several initiatives have addressed the need for a standardized markup-based music notation [Co01]. Especially, XML-based languages can support a suitable music notation model that has multimedia capabilities and includes identifications, classification, symbolic, printing, protection, images score, image, document, etc [BN01].

Geometric hashing was developed in computer vision for model-based recognition of objects[Wo97]. The object information is transformed in preprocessing step and saved in a hash table. By using geometric information, we can access the hash table. The information is an invariant property of the object and computed at the accessing stage. During recognition step, the approach searches the constructed hash table after indexing geometric properties extracted from the stage for candidate objects. Searching is needed at each stage but geometric hashing prevents from searching all objects and their features[Wo97]. The geometric hashing has been used successfully in the field of computational molecular biology that includes the problems of protein structure comparisons and molecular docking [PA94, FCG03].

[JP05a] present a music retrieval system that compares the geometric structure of a melody by using the geometric hashing technique. It represents a melody as a melodic contour and describes the notes by 2D coordinates (x, y), that is, pitch and duration. A melody pattern is a set of five notes and its hash value is computed with three cosine values and a main chord. It constructs a segment that connects a note to its next note and produces a pattern with four continuous segments.

[JP05b] propose a music information retrieval system that exploits various types of contextual information based on JXTA technology and supports the personalized retrieval of music information at the mobile environment. [FGS03] present a scalable P2P system that supports a large set of music retrieval methods. It extracts musical features from a music file and creates music file description that is a set of attribute-value pairs. The features can be either manually described by a user or they can be automatically extracted from the musical content. [WLS02] propose four types of P2P models for content-based music information and QUIND that is an architecture and consists of PC and coordinator.

3 Content-based music structure modeling

Melody as a sequence of notes is the horizontal dimension in music, a succession of organized pitches having a definite rhythm, where the vertical dimension arises from the harmony. Each bar contains a number of notes and/or rests whose total time value is given by the time signature. Both the relations between pitches and between durations are commonly defined by numbers and ratios and the order of each note is decided [Url2].

Figure 2 shows how a melody pattern is modeled in three successive bars. We describe notes into each bar by 3D coordinates(x, y, z). We select two notes after sorting the notes into each bar by the value of the pitch and duration. One is the first ranked note and the other is the last ranked note. The X axis is the order of appearance of a note into each bar. The Y axis is the pitch which is a MIDI number. The Z axis is the duration which is measured in 16th notes.

Then, we describe three segments by connecting the first and the last ranked note in each bar. A group of three segments is a melody pattern. We compute the length of each segment and the three cosines of the dihedral angle between the segments. The length of each segment is used to match candidate melody patterns against a query melody and the cosines is used to build a hash table.

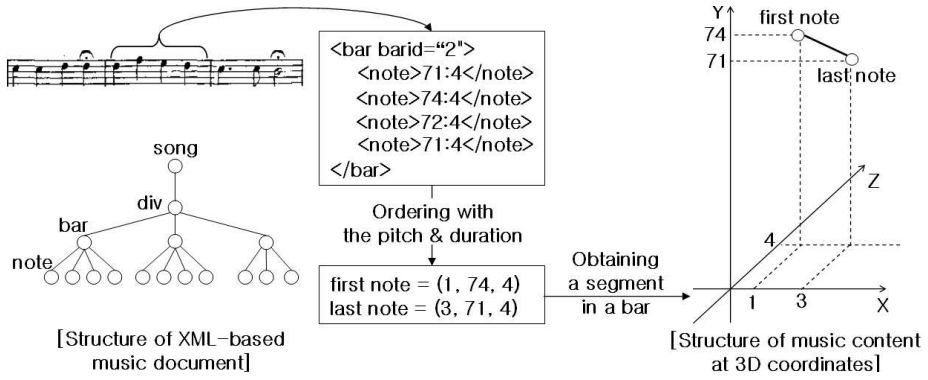

Figure 2: Structure of XML based music document and music data at 3D coordinates

We store the MML-based music documents [Url1] into Xindice. The MML focuses on the structure of music and music related processes. For the matching process, we extract only necessary contents from MML-based document using the following rules. (See figure 3.)

First, we extract a note that has the highest pitch out of notes that are played simultaneously. Typically, the highest-pitch notes would be classed as the melody notes unless they are monotonous because listeners usually hear the highest pitch notes as melody notes. In MML, simultaneous events are grouped together inside the square brackets [and]. For example, [A C E] means that the notes A, C, and E from a chord group played simultaneously and E is the highest pitch note.

Second, we divide a group of notes into individual note into a bar. In MML, <bar>

element contains all notes played simultaneously and sequentially into a bar. We divide the highest notes in a bar individual note by adding sub-elements that is <note> element. For example, <bar> [A C E] < /bar> is represented as <bar><note>E< /note>< /bar>.

Third, we represent the pitch as a MIDI number (60 = C4, 61 = C#4, etc.) and the duration as a number measured in 16th notes (sixteenth note = 1, quarter note = 4, etc.).

Finally, we compute the number of <note> element in a <bar> element. The number is used to build a hash table as a structural feature of an XML document.

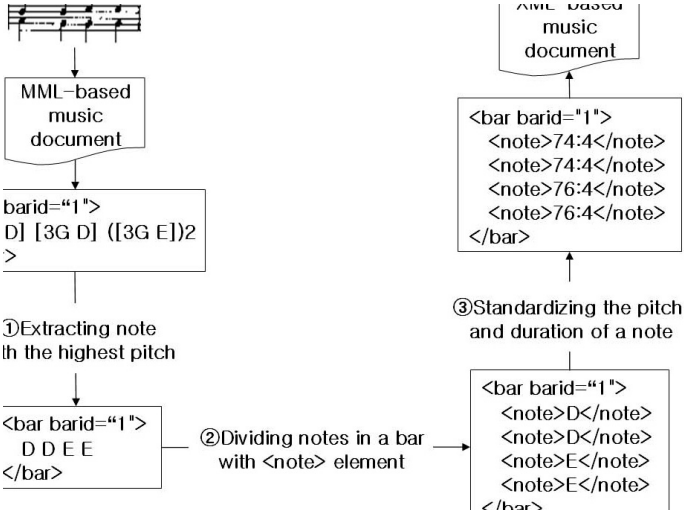

Figure 3: Transforming a MML-based document into another XML-based document

4 Searching for a melody pattern

4.1 Preprocessing

In general, indexing techniques can use different types of invariant properties extracted based on various features of data for obtaining indices for a hash table when storing the data in a redundant way [FCG03]. If a hash value of a query melody is equal to an index of slot in the hash table, the information of the slot will contain a melody which is similar to the query melody.

To define an invariant property, we consider dihedral angles between each pair of segments which connect two notes. We analyze invariant properties of a melody pattern to compute its hash value. A melody pattern S is a triplet of segments of music (S_i, S_j, S_k). Two segments S_i and S_j can form the dihedral angle α_{ij}. The three cosines (cosα_{ij}, cosα_{jk}, cosα_{ki}) are used as invariant properties for the hash table. Also we select another invariant

property which is the number of <note> elements in a bar by parsing XML-based music document, which is a sum of all notes in three bars.

The three cosines and the number of notes of a melody pattern are used to generate slot indices for a hash table. An index is mapped into a certain slot of the hash table. Each slot corresponds to a record of fields in a database. The record contains information of each melody pattern such as a pattern id, original music id and the lengths of each segment.

Note that the difference of the length between segments determines the similarity between a given query melody and melodies of music. An index of the hash table is used for finding the position of melody patterns in music and for selecting the information of each melody pattern at the database because the index corresponds to a record which is a set of fields.

Figure 4 shows the process of the preprocessing step.

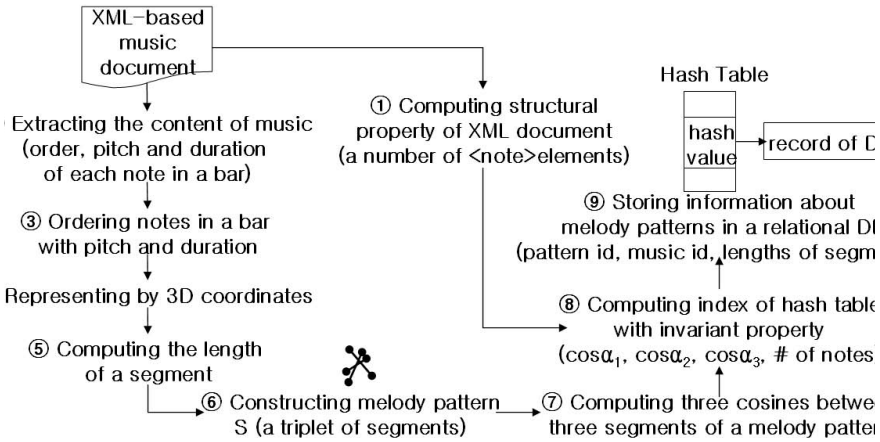

Figure 4: The preprocessing step

1. The structural property of the XML-based music document that is a number of <note> is computed.
2. The contents of the music such as order, pitch, and duration are extracted.
3. The notes in each bar with the pitch and duration are ordered.
4. The notes are represented by 3D coordinates.
5. The length of segments in each bar is computed.
6. A melody pattern that is a triplet of segments is construted.
7. Three cosines between a triplet of segments which are geometric properties are computed.
8. Hash values that are indices of the hash table with the invariant properties are computed.

9. The information about all melody patterns of the music is stored in a relational database.

4.2 Recognition

When a user sends a query melody, we first confirm whether the query melody consists of at least three bars in XML document for generating an index of the hash table. The steps to search melody pattern which is similar to a given query melody are as follows: (See figure 5.)

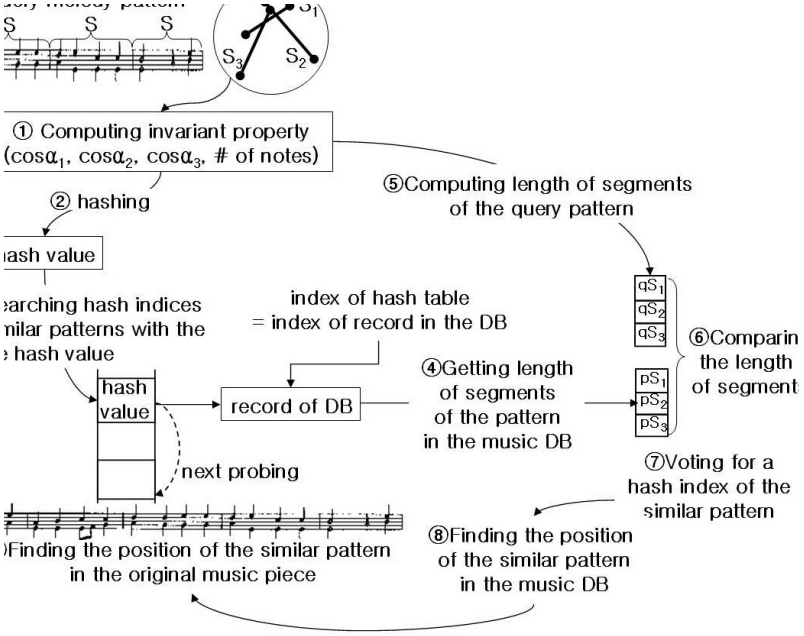

Figure 5: The recognition step

1. The invariant properties of the query melody are computed.
2. The invariant properties are hashed and a hash value of the query is produced.
3. The indices of melody patterns that have the hash value which is equal to the query's are searched.
4. The length of the segments of the melody patters in the music database is found by using the searched indices.
5. The length of the segments of the query pattern is computed.

6. The lengths of the segments between the query and candidate patterns are compared.
7. A vote is cast to melody patterns below the threshold after checking whether each difference of the length is below the threshold or not.
8. The position of the similar melody pattern is found in the music database.
9. The matching position is found in its original music piece.

5 A JXTA-based music retrieval system

Figure 6 depicts the architecture of the JXTA-based CBMIR system. It consists of five types of peers such as user, rendezvous, manager, database, and computing peer and the peers collaborates for distributed music retrieval.

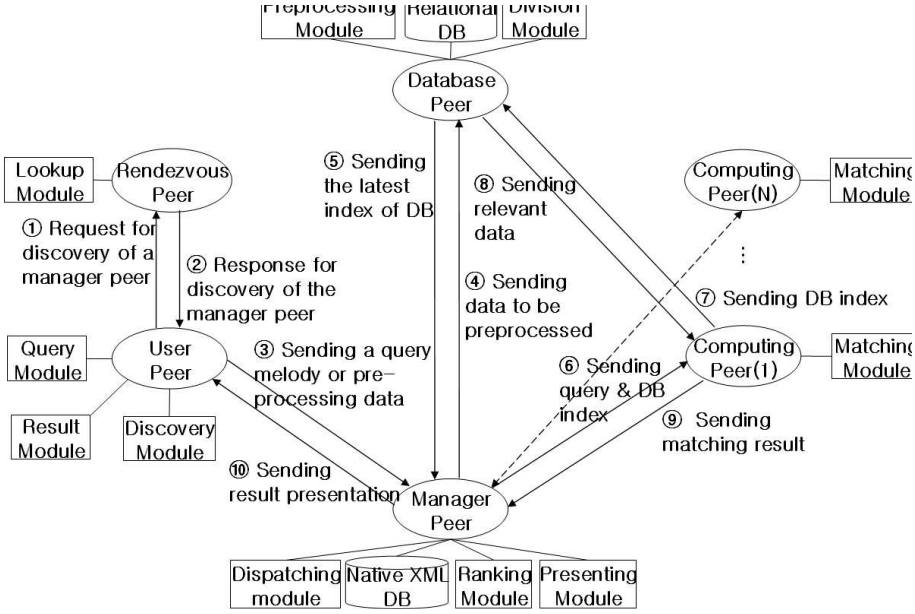

Figure 6: The architecture of the CBMIR system

1. A user peer sends a query melody and receives music retrieval results. The discovery module sends a discovery request for a manager peer to a rendezvous peer and receives information about the manager peer. The query module sends user's query melody or preprocessing XML documents to a manager peer and result module receives the searching results.

2. A rendezvous peer maintains a cache of advertisements and forwards discovery requests to help other peers discover resources. The lookup module checks whether the requested advertisement is stored at the local cache. If so, it forwards the advertisement to user peer. Otherwise, it gains the advertisement from other rendezvous peer. Then, it stores the advertisement in the local cache and forwards the requested advertisement to user peer.

3. A manager peer dispatches jobs for melody structure comparison to computing peers, collects the searching results, and sends them to user peer. The dispatching module decomposes preprocessed data of music stored in the music database into several parts and assigns the pars to connected computing peers by sending DB index as searching scope. The ranking module collects local results from each computing peer, computes the total ranking, and sends the final result to user peer. The presenting module applies a style sheet to retrieved XML-based music document in a native XML database and creates a tailored presentation based on user's device.

4. A database peer preprocesses music contents and sends a part of data in the music database to the computing peer. The preprocessing module extracts invariant properties in music contents, computes indices of slot at the hash table, and stores preprocessed data of melody patterns in the music database. The division module sends assigned music data to each computing peer when it informs searching DB index.

5. A computing peer preprocesses a given query melody and searches melodies which are similar to the query melody. It works in parallel with other computing peers. The matching module preprocesses the query melody, that is it computes the invariant properties and the hash value of the query melody. Then it finds candidate melody patterns which are equal to the hash value of the query melody within the assigned music data. It casts a vote at exact matching melody patterns by computing difference of length of three segments between melody pattern and the query melody. Then, it sends local results to the manager peer.

6 Experimental results

We tested how the system would find music according to the change of a threshold value. A threshold value is the difference between the total length of the three segments of the query and the corresponding length of a candidate melody pattern. Each of the three differences must be below a given threshold.

We preprocessed twenty music XML documents that are Bach chorales and stored the results in a music database. We made a query XML document that consisted of five bars. The query was a part of the No.19 music from its third to seventh bar. As we increased the threshold value, we input the same query and checked how many the result contained the valid patterns. We define that the valid pattern is a retrieved melody pattern that two out of three bars in the retrieved melody pattern at least matches against the query and the rest is

a neighboring bar with the query that is eighth or ninth bar. The recall was computed the ratio of melody patterns which must be retrieved to the real retrieved valid patterns. The precision was computed the ratio of all retrieved melody patterns to the real retrieved valid patterns.

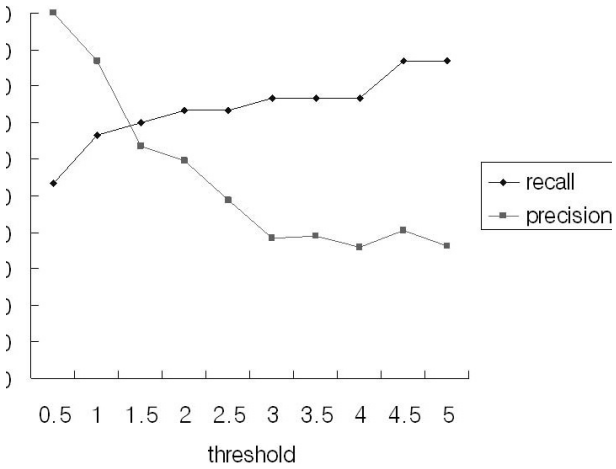

Figure 7: The recall and precision of our system

When the threshold was over 0.5, the precision was getting lower gradually because the system searched similar patterns in other music as well as No.19. When the threshold was over 1, the precision dropped suddenly. When the threshold was below 0.5, the system searched completely matched patterns. When the threshold was over 0.5, the system searched partially matched patterns. As the threshold increased, the recall increased more and more.

7 Conclusions and future directions

It is useful to search a music with just a part of the music because it is difficult to memorize and reproduce the whole music exactly. In this paper, we propose a JXTA-based CBMIR system that searches for a given query melody in a set of XML documents that encode music data using a geometric hashing algorithm. The system can be used for both exact maching and similar matching (i.e., melody patterns that are similar to a given query melody can be found.) It can be also used to detect plagiarism by entering whole music as a query meolody.

We plan to test our system with more various types music than Bach chorales. We also plan to implement modules so that mobile devices such as cell phones or PDAs can be

used as peers in the system.

References

[BN01] P. Bellini and P. Nesi, WEDELMUSIC Format : an XML Music Notation Format for Emerging Applications, Proceedings of the First International Conference on WEB Delivering for Music(WEDELMUSIC'01), 2001, IEEE press.

[Co01] R. Cover, The XML Cover Pages : XML and MUSIC, (2001). `http://xml.coverpages.org/xmlMusic.html`

[FCG03] C. Ferrari, C. Guerra, G. Zanotti : A grid-aware approach to protein structure comparison, Journal of Parallel and Distributed Computing, 63, (2003) 728-737,.

[FGS03] G. Tzanetakis, J. Gao, P. Steenkiste : A Scalable Peer-to-Peer System for Music Content and Inforamtion Retrieval, Johns Hopkins University, 2003.

[Go01] L. Gong, JXTA : A Network Programming Environment, IEEE Internet Computing, Industry Report, (2001) 88-95.

[JP05a] H. Jung and S. Park : Music Retrieval Using the Geometric Hashing Technique, The journal of Korea Association of Computer Education, 8(5), (2005) 109-118.

[JP05b] H. Jung and S. Park : Context Awareness for Music Information Retrieval using JXTA Technology, On the Move to Meaningful Internet Systems 2005: OTM Workshops: Context-Aware Mobile System, LNCS, 3762, (2005) 211-214.

[PA94] X. Pennec and N. Ayache : An $O(n^2)$ Algorithm for 3D Substructure Matching of Proteins, In Proceedings of the First International Workshop on Shape and Pattern Matching in Computational Biology, A. Califano, I. Rigoutsos, and H.J. Wolson eds, June, (1994) 25-40.

[Pl02] F. Plotkin : Classical Music 101 - A Complete Guide to Learning and Loving Classical Music, Hyperion, New York, 2002.

[St02] J. Steyn, Framework for a music markup language, MAX 2002, International Conference Musical Application Using XML, September 19 - 20, 2002, State University of Milan, Italy.

[Url1] `http://www.musicmarkup.info/index.html`

[Url2] `http://www.dolmetsch.com/musictheorydefs.htm`

[Url3] `http://xml.apache.org/xindice`

[WLS02] C. Wang, J. Li, S. Shi : A Kind of Content-Based Music Information Retrieval Method in a Peer-to-Peer Environment, IRCAM, 2002.

[Wo97] H. J. Wolfson : Geometric Hashing : An Overview, IEEE Computational Science & Engineering, (1997) 10-21.