

MF-tree: Extracting and Clustering the Structural Features from Music Object in MusicXML¹

Yu-Chih Shen, Jia-Lien Hsu² and Shuk-Chun Chung
Department of Computer Science and Information Engineering,
Fu Jen Catholic University, Taiwan, R.O.C.
alien@csie.fju.edu.tw

Abstract—In the music information retrieval field, the most important topic is to extract the feature which represents the content from the music objects. The content feature is useful for music analysis, music retrieval, and other services.

In this paper, our data form is MusicXML. We extract the structural feature from the classical music object and it is based on music theory and music form. According to the hierarchical rule of music form, we construct a tree which is called MF-tree, it is also to be provided with music theory feature and structure in the MF-tree. We also propose the dissimilarity function of MF-tree to calculate the dissimilarity of music objects. Finally, we adapt the hierarchical clustering and system implementation to show the clustering.

I. INTRODUCTION

IN recent years, the music information retrieval is the fast developed investigation field. By extracting music feature and application, the system can provide more services for user, e.g., recommendation and querying by humming. It is one of the most important topics in feature extraction.

A. Motivation

In early stages, most of investigations often transform the melody of music to string in music information retrieval field. In general, there are two types of data sources. One is the symbolic form, for example: midi. The other one is wav form such as mp3, CD audio. But midi is designed for the connecting various devices, and for the storage of the basic music string like melody string, rhythm string. It doesn't store up the structure of music. So we adapted the MusicXML as the source of our music data form.

The remarkable traits of the structural feature of music are repeating rule and hierarchical rule. Both can present the structure of music. Hsu[4] finds the repeating pattern of the music object to display the music structure. However, there are no investigations that use the hierarchical rule to display the music structure. We propose a novel idea to present the music feature with the concepts of structure. By the hierarchical rule of music form, we extract the new music feature and construct a tree structure which is called MF-tree that consists of music theory feature and structure.

The next is the applications about the music system. For example, (1) Feature extraction for classification: Classify the music by extracting the music feature. (2) Find the repeating

structure pattern: Find the repeating structure pattern in the music information. (3) Music search: Support the design of music inquire system, and promote the inquire speed. (4) Music recommendation: promote the recommend of the music system. (5) Music summary: According to the music structure, find the important structure pattern to be the music summery. (6) Music analysis: Regard to the specific music composer, and analysis the music works. (7) Copy detection: Find the copy works which structure and melodies are similar by the comparison and calculation of the music structure.

B. Music Features

According to the characteristics of the music objects and Hsu[4], music features can be classified into four categories:

1) **Static music information.** It refers to the keyword-based description of music objects, such as key, beat, and tempo, e.g., the static music information of Beethoven's Symphony No. 5, Op. 67 is C minor, 4/4.

2) **Acoustical feature.** It include the sound characteristics, such as loudness, pitch, duration, bandwidth, and brightness, which can be derived from the raw data of music objects, e.g., loudness of a music object is the root mean square value in decibels of its audio signals[21].

3) **Thematic feature.** It is melodies, rhythms, and chords which can be derived from the score information of music objects. It can be represented in a string form, e.g., "G-G-G-E" is a melody string of a music object, and "C-Am-Dm-G7" is a chord string.

4) **Structural feature.** According to the investigation of musical form, the music form is constructed by the specific rules. The basic rules are hierarchical rule and repetition rule [6][7][13].

The hierarchical rule says music objects are formed hierarchically, i.e., a music object consists of *movements*, a movement consists of *sentences*, a sentence consists of *phrases*, and a phrase consists of *figures*, e.g., Figure 1. [4]

The repetition rule says that some sequences of notes, known as motives, repeatedly appear in a movement., We can also find the repetition rule in other music types, e.g., the refrain is an instance of repetition rule in popular music.

¹ The research was partially supported by the Republic of China, National Science Council under Contract No. NSC-94-2213-E-030-014

² To whom all correspondence should be sent.

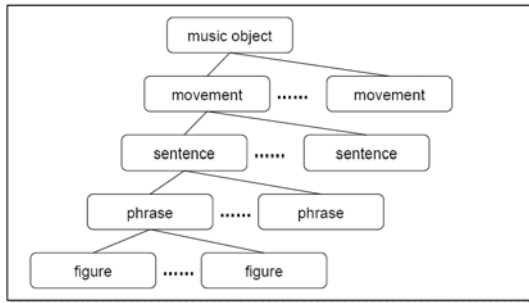


Fig. 1. The hierarchical rule of a music object.

C. Related Works

We introduce the feature extraction, music recognition, music classification, and the structure similarity of XML respectively.

For the investigation of feature extraction, Miura[12] introduces pitch spectrum of melodies, and utilize them as similarity description. Then we can determine the transposition easily in the spectrum. Wai[18] proposes a stream segregation algorithm for polyphonic music database. The pitches can be disubtrubted into few streams to show the music feature of the melody line concept. Hsu[4] defines the nontrivial repeating pattern of music feature. We utilize the *correlative matrix* and *RP-tree* to find the longest exact repeating patterns rapidly. Hsu[5] defines to use the *cut* and *pattern_join operators*, to develop a level-wise approach, and use it solve the problem of finding approximate repeating patterns.

For the recognition of music form structure, Liu[11] proposes three assumptions for instance of Johann Strauss waltz centos. We design a Melody-tree with the unit of four measures. A Melody-tree is a tree of three levels, and it is an important feature of form recognition. Seifert[15] utilizes semantic relationship and characteristic motifs to design a Leadsheet-tree as music recognition.

For the music classification, Shan[16] uses the chord to investigate the different feature representations and we investigate the mining and classification of music style to help for discovering the music style. Based on user behavior, Kuo[8] proposes a music filtering system. We utilize the multi-type melody style classification approach for the mining melody patterns and recommend the music objects. Kuo[9] proposes four melody style queries. The output of the melody style queries supply the new or not known music for the user.

For the structure similarity of XML. Wang[19] proposes a hierarchical algorithm (S-GRACE) and it can reduce the join cost for querying XML documents which is stored in relational tables. Bertino[1] proposes a matching algorithm for measuring the structural similarity between an XML document and a DTD, e.g., classification of XML documents, selective dissemination of XML documents, and the protection of XML documents. Lee[10] utilizes the structure similarity and semantics to provide XClust algorithm for integrating DTD schema.

D. The Problems

Our investigation is feature extraction and classification of music objects in MusicXML. First, we try to extract the music feature and construct a MF-tree. Then we measure the dissimilarity of MF-tree and the music classification.

The rest of this paper is organized as : in Section 2, we present our method that consists of the introduction of MusicXML document, system process, construction of MF-tree, dissimilarity of MF-tree, and hierarchical clustering. In Section 3, we introduce our implementation and system interface. Section 4 is experiment. Finally, we conclude the paper and further works in Section 5.

II. OUR METHOD

In this section, we firstly introduce the MusicXML document and the system process. Then we propose the construction of MF-tree and the dissimilarity of MF-tree. Finally we present the hierarchical clustering.

A. The Introduction of MusicXML

XML is a data form to be used in data exchanging. Based on XML we develop a new music data form called MusicXML. MusicXML is designed for music retrieval, analysis and it is stored score-like data form. It contained symbolic music data and many metadata, e.g., creator, clef, key, beat, measure, pitch, duration, melody, rhythm, chord, slur, type, grace, lyrics, etc.

We explain structure of MusicXML with an example. We select a piece of the score from the Wie Melodien Op.105 of Johannes Brahms, e.g., Figure 2. The first measure of first partition of the score relates to the first measure of the child node <part id="P1"> in MusicXML document, e.g., Figure 3. The second note of the first measure relates to the rectangle in Figure 3 that is the second child node<note>of node<measure number="1">. Its tree structure is displayed in the rectangle of Figure 4.

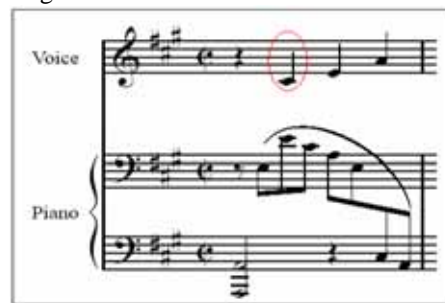


Fig. 2. The first measure of Wie Melodien, Op.105, by Johannes Brahms

```

<?xml version="1.0" standalone="no" ?>
<!DOCTYPE score-partwise (View Source for full doctype...)>
<score-partwise>
  <movement-number>Op. 105, No. 1</movement-number>
  <movement-title>Wie Melodien zieht es mir (Page 1)
  </movement-title>
  <identification>
  <part-list>
  <part id="P1">
    <measure number="1">
      <attributes>
      <direction placement="above">
      <note>
      <note>
      <pitch>
        <step>C</step>
        <alter>1</alter>
        <octave>4</octave>
      </pitch>
        <duration>2</duration>
        <voice>1</voice>
        <type>quarter</type>
        <stem>up</stem>
      <lyric number="1">
        <syllabic>single</syllabic>
        <text>Wie</text>
      </lyric>
    </note>
    <note>
    <note>
  </score-partwise>

```

Fig.3. The corresponding MusicXML document of Fig.2

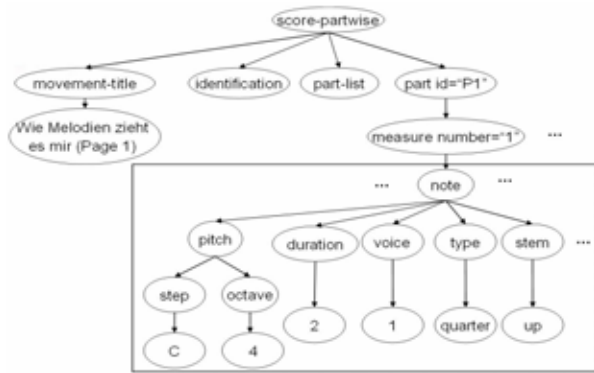


Figure 2. The structure of MusicXML document.

We have basic understand that MusicXML document . We can know one thing, the structure of MusicXML document is difference from XML document structure. We can get a lot of XML documents form the Internet, but its structure is different. There are many methods to compute the similarity of XML documents by structure [1] [2] [10] [20]. However, the MusicXML documents have similar height and level, e.g., in Figure 4, the score-partwise contains movement-title, identification, part-list, part id="P1". The part id="P1" contains measure number="1". The measure number="1" contains note. The note contains pitch, duration, voice, type, stem, etc. In this case, we cannot get the similarity of music object by the similarity method of XML document. So we propose a dissimilarity function for tree structure to compute the dissimilarity of music object in later of this section.

B. System Process

Figure 5 shows our system process. Based on the hierarchical rule of music form we get the music feature for classical music objects in MusicXML and construct a MF-tree. We propose the dissimilarity of MF-tree and cluster the classical music objects by the hierarchical clustering.

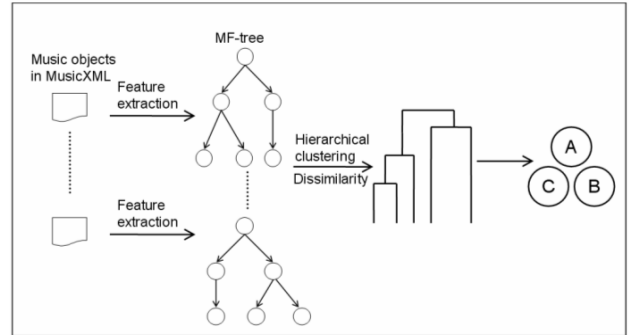


Fig. 5. The system process.

C. The Construction of MF-Tree

MusicXML is musical data form. But it contains a lot of non-related music information such as creator, clef, key. So we extract the music feature based on the hierarchical rule of music form for classical music objects in MusicXML and construct MF-tree that consists of music theory feature and structure. It consists of music feature that can represent the music structure and without losing the content of music, e.g., melodies, rhythms.

A music object is a classical music and it constructs a MF-tree. A classical music consists of one or more movements. A classical symphony can divide more movements, e.g., the C Major fifth symphony of Beethoven consists of four movements. If a classical music does not consist of the movement, and we conclude itself as a movement like the Wie Melodien Op. 105 of Brahms is a classical music, and it is a movement.

We extract the music object feature, movement feature, segment feature, and sub-segment feature for a music object of classical music by the hierarchical rule of music form. Then we construct MF-tree in Figure 6. It consists of music object nodes, movement nodes, segment nodes, and sub-segment nodes etc. The MF-tree structure is similar to a document. It just likes a document consists of paragraph, sentence, and word.

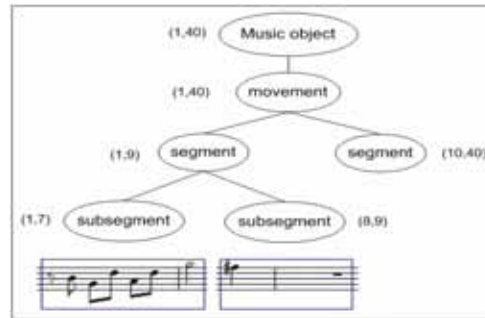


Fig. 6. An example of MF-tree

We introduce the definition of MF-tree, and the construction method of MF-tree.

Definition 1: The MF-tree is a four level of tree that has three properties:

- 1) **Property 1.** From the root node computing, the first level is music object node, the second level is movement node, the third level is segment node, and the fourth level is subsegment node.
- 2) **Property 2.** Each node responds to a music sequence (b_1, b_2) , which express from beat b_1 to the beat b_2 in a music object.
- 3) **Property 3.** For each node of the MF-tree, the music sequence of each child node comes into being a partition with its parent node.

For the construction of MF-tree, we propose the method by heuristic, and refer to Figure 6.

- 1) **Music object node.** A classical music object represents a music object node. A music object is being stored with one or more MusicXML documents.
- 2) **Movement node.** A movement represents a movement node and each is being stored with one MusicXML document.
- 3) **Segment node.** We cut the segment features from each movement node based on the duration of the rest. We start to compute the first beat of music sequence from the movement node. If it appears a note that is the rest and the duration is longer than one beat or more. Then we obtain a music sequence and consider that it is a segment feature. However, if we compute the note which is the last, we still consider as a segment feature.
- 4) **Subsegment node.** We cut the sub-segment features from each segment node based on the duration of the note. We start to compute the first beat of music sequence from the segment node. If it appears a note and its duration is longer than two beats or more. Then we say it is a sub-segment feature. However, if we compute the note which is the last, we still conclude it is a sub-segment feature.

D. The Dissimilarity Function of MF-Tree

In this section, we introduce the dissimilarity function of MF-tree. The similarities properties of the nodes pair are decided based on whether overlapped or not. If they overlap, we continue to compute the dissimilarity, otherwise we say it is not related. The dissimilarity of each internal node is obtained by the dissimilarity of child nodes. We first normalize the music sequence of the node between zero and one, and determine whether the nodes pair overlap or not by Function $over(u,v)$.

Definition 2: Function $over(u,v)$ overlapping of nodes

Given two nodes $u : (a_1, a_2)$ and $v : (b_1, b_2)$, if $over(u,v)=1$, it is overlapped:

$$over(u,v) = \begin{cases} 1, & \text{if } \frac{b_1}{N} \leq \frac{a_1}{M} < \frac{b_2}{N}, \text{ or } \frac{a_1}{M} \leq \frac{b_1}{N} < \frac{a_2}{M} \\ 0, & \text{otherwise} \end{cases}$$

where M is the number of the music sequence of u , N is the number of the music sequence of v .

Example 4:

Given two sub-tree of MF-tree t_1 and t_2 , we normalize the music sequence of each node between zero and one in Figure10. The music sequence $(1,11)$ of the segment node a normalizes to $(0,1)$. The music sequence $(1,5)$ of the sub-segment node a_1 normalizes to $(0,0.45)$. In the same measure, the music sequence $(1,9)$ of the segment node b normalizes to $(0,1)$. The music sequence $(5,8)$ of the sub-segment node b_2 normalizes to $(0.44,1)$. At the result we find the segment nodes a and b , the sub-segment nodes a_1 and b_1 , a_1 and b_2 , a_2 and b_2 , a_3 and b_2 are overlapped.

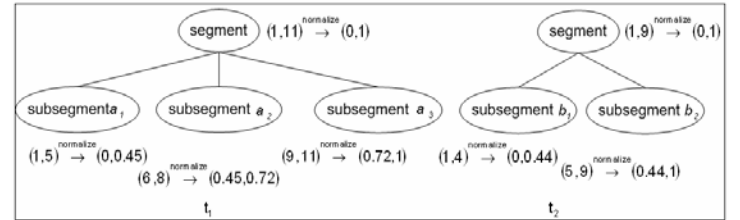


Fig. 7. The two subtrees of MF-tree.

We mentioned at the above, if the nodes are overlapped, we compute the dissimilarity of the nodes pair. Now we use the Function $DS_{sub}(a,b)$ to compute the dissimilarity of the sub-segment nodes. Then we can use the Function $DS_{seg}(a,b)$ and Function $DS_{mov}(a,b)$ to compute the dissimilarities of the segment nodes and movement nodes in a similar way. Finally, we can find the dissimilarity of the music object nodes by using Function $DS_{obj}(a,b)$.

Definition 3: Function $DS_{sub}(a,b)$

Given two subsegment nodes a and b , the dissimilarity of a and b :

$$DS_{sub}(a,b) = \begin{cases} \left(1 - \frac{v_i \cdot v_j}{\|v_i\| \|v_j\|} \right), & v_i \text{ is the pitch histogram of } a \\ & v_j \text{ is the pitch histogram of } b, \text{ if } over(a,b) = 1, \\ \text{N/A} & \text{, otherwise.} \end{cases} \quad (a)$$

$$\begin{cases} \left(1 - \frac{v_i \cdot v_j}{\|v_i\| \|v_j\|} \right), & v_i \text{ is the duration histogram of } a \\ & v_j \text{ is the duration histogram of } b, \text{ if } over(a,b) = 1, \\ \text{N/A} & \text{, otherwise.} \end{cases} \quad (b)$$

We now propose two computing methods for Function $DS_{sub}(a,b)$. The first method is utilizing the pitch histogram based on the type of pitch (c,d,e,f,g,a,b,r) to compute the dissimilarity of the two nodes. The second method is

utilizing the duration histogram based on the type of duration (full, half, quarter, eighth, sixteenth, thirty-second, sixty-fourth, rest) to compute the dissimilarity of the two nodes.

Definition 4: Function $DS_{seg}(a,b)$

Given two segment nodes a and b , the dissimilarity of a and b :

$$DS_{seg}(a,b) = \begin{cases} \frac{\sum_{1 \leq i \leq m, 1 \leq j \leq n} DS_{sub}(a_i, b_j)}{k} & , \text{if } over(a,b) = 1 \text{ and } over(a_i, b_j) = 1 \\ \text{N/A} & , \text{otherwise.} \end{cases}$$

a_i is the child of a , $1 \leq i \leq m$, m is the number of sub-segment nodes of a . b_j is the child of b , $1 \leq j \leq n$, n is the number of sub-segment nodes of b , $\sum_{1 \leq i \leq m, 1 \leq j \leq n} over(a_i, b_j) = k$.

We use the similar concept to compute the dissimilarity of the movement and the music object nodes.

Definition 5: Function $DS_{mov}(a,b)$

Given two movements nodes a and b , the dissimilarity of a and b :

$$DS_{mov}(a,b) = \begin{cases} \frac{\sum_{1 \leq i \leq m, 1 \leq j \leq n} DS_{seg}(a_i, b_j)}{k} & , \text{if } over(a,b) = 1 \text{ and } over(a_i, b_j) = 1 \\ \text{N/A} & , \text{otherwise.} \end{cases}$$

a_i is the child of a , $1 \leq i \leq m$, m is the number of segment nodes of a . b_j is the child of b , $1 \leq j \leq n$, n is the number of segment nodes of b , $\sum_{1 \leq i \leq m, 1 \leq j \leq n} over(a_i, b_j) = k$.

Definition 6: Function $DS_{obj}(a,b)$

Given two music object nodes a and b , the dissimilarity of a and b :

$$DS_{obj}(a,b) = \begin{cases} \frac{\sum_{1 \leq i \leq m, 1 \leq j \leq n} DS_{mov}(a_i, b_j)}{k} & , \text{if } over(a,b) = 1 \text{ and } over(a_i, b_j) = 1 \\ \text{N/A} & , \text{otherwise.} \end{cases}$$

a_i is the child of a , $1 \leq i \leq m$, m is the number of movement nodes of a . b_j is the child of b , $1 \leq j \leq n$, n is the number of movement nodes of b , $\sum_{1 \leq i \leq m, 1 \leq j \leq n} over(a_i, b_j) = k$.

We utilize the dissimilarity function of MF-tree to compute the dissimilarity of two MF-trees and obtain the result of music object classification by hierarchical clustering.

E. Hierarchical Clustering

After introducing the dissimilarity of MF-tree, we adopt the hierarchical clustering [18] to class the music objects. Initialize we assume each music object as clusters. For the all clusters, we utilize the single-link method to find the distance that is smallest between the clusters, and merge into one cluster. Repeating the all clusters merge into one cluster. We adopt the advantage of hierarchical clustering that is not need to set the number of clustering in advance and has the result of visualization.

III. SYSTEM IMPLEMENTATION

Regarding our system implementation, we use program software which is *Borland C++ Builder 6.0* and the program language is C++. Our equipment is the Pentium 2.00 GHz, 768 MB memory, and the OS system which is Windows XP sp1. In our system implementation the parsing module of MusicXML document comes from the Delphi K. Top [3].

F. System Architecture

We design the system architecture, e.g., Figure 11, that is consists of module below:

- 1) **MusicXML document parser:** We parse the MusicXML document, and present the tree structure of MusicXML document.
- 2) **MF-tree module:** By the feature extraction, we construct the music object out the MF-tree, and provide the presenting the related position of music sequence of node.
- 3) **Clustering module:** We obtain the dissimilarity of MF-tree by the dissimilarity function of MF-tree, and derive the number of clustering by the hierarchical clustering.
- 4) **Interface module:** We implement a system and display the graph of interface.

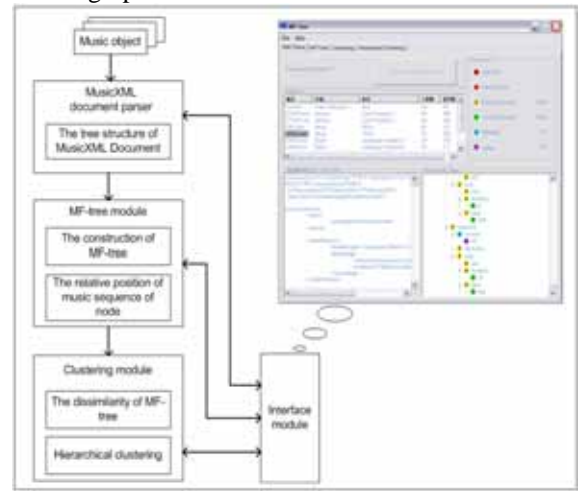


Fig. 8. System architecture.

G. System Interface

In system implementation, we divide four processes between system interface that is MusicXML document Parser, MF-tree, Clustering, and Hierarchical Clustering.

The first process is the parsing of MusicXML document in Figure 9. It has four parts. Part A shows out the input documents, filename, composer, song name, and measures of each document. Part B displays content of the document which is be chosen. Part C displays the details of document that is be chosen such as the number of elements. Part D shows the tree structure of document that is be chosen.

The second process is about the feature extraction, MF-tree, and the related position of segment nodes in Figure 10. It has four parts. Button feature extraction can retrieve

music features at Part A. Part B shows the document that is being chosen and its related information such as the number of nodes. Part C shows out the MF-tree. In addition we assign the number to the nodes, e.g., if the number of segment nodes of MF-tree is eighteen, then the first node will be assign as segment-1. The first sub-segment node of thirteen segment node will be assign as sub-segment 13-1. Part D shows the related partition of the node that is being chosen.

The third process is calculation of the dissimilarity of MF-tree and result of clustering demonstration in Figure 11. It has four parts. Choosing one of the two methods in Part A to compute the dissimilarity of MF-tree. Choosing the cluster number and executing the hierarchical clustering in Part B. Part C displays the clustering result that by the hierarchical clustering.

The fourth process is showing the tree structure of hierarchical clustering in Figure12. The distance-axis shows dissimilarity of the two MF-trees.

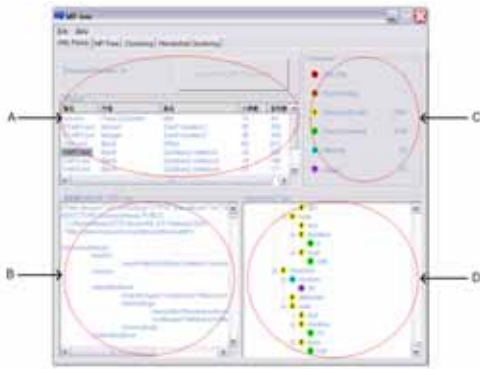


Fig. 9. The parsing of MusicXML document.

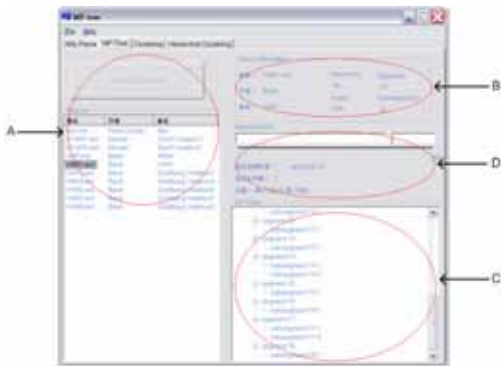


Fig. 30. The construction of MF-tree



Fig. 41. The illustration of clustering result

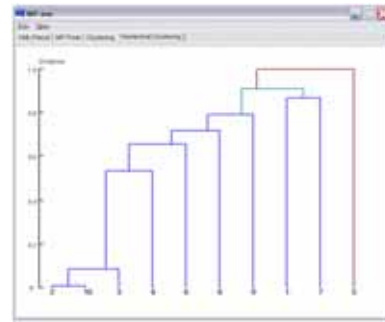


Fig. 52. The hierarchical cluster tree.

IV. EXPERIMENT

Our aim is to construct a tree structure to present the structure of music object which is based on the hierarchical rule of music form. We use the dissimilarity of MF-tree to improve the result of music classification. Also we adapt the structure concept to increase the accuracy of dissimilarity and allow the clustering more efficiency.

A. Experiment Set

Due to we do not have the benchmark of similarity of music objects. We consider the music fragments are similar if we cut from the same variations. Although music fragments are very similar to the listeners, we are still choose the pitch histogram and duration histogram from the music fragment. Actually, its pitch histogram and duration histogram of the music fragments are not similar. We cut the music fragment by the partition of variation from the Zwolf Variation KV 265 of Mozart. We also cut the music fragment from the Goldberg Variation BWV 988 of J.S. Bach. In addition, we add a music fragment of the Mut of Winterreise of Franz Schubert. We consider ten music fragments as ten music objects for our experiment. Table 1 shows the beats of each data. We utilize the tool of TaBazar[17] to transform the nine music fragments of Zwolg Variation and Goldberg Variation to the MusicXML documents. The rest document is from the Recordare[14] site. Each document only stores the first part of the score. We class the data into three classes by the composer. The two music fragments are cut from the Zwolg Variation are considered as one class. In the same way, music fragments are cut from the Goldberg Variation are considered as one class. The rest music fragment is considered as one class.

Table 1. Experiment data

composer	title	Object size (beats)
Mozart	Zwolf Variation 1	350
	Zwolf Variation 7	348
Bach	Boldberg Variation Theme	612
	Boldberg Variation 1	280
	Boldberg Variation 3	188
	Boldberg Variation 4	171
	Boldberg Variation 6	219
	Boldberg Variation 7	215
Schubert	Boldberg Variation 8	367
	Mut	44

B. Experiment Result

We extract the pitch and duration histogram from each music objects, and use it to compute the dissimilarity of music objects by the vector space model. Then we utilize the F-measure to evaluate the result of clustering. We assign (Naïve-PH) and (Naïve-DH) of Naïve as the names for the pitch histogram, and duration histogram respectively. Finally, we also compare the result from Naïve with our method.

Based on the Table 1, we set the number of clustering as 3. Figure 13 shows the clustering result of our method and Naïve method. The experiment shows our method is better than Naïve. Because we concern on the overlapping problems of nodes pair and use it to compute the dissimilarity of music objects. We do not choose the feature of whole music object and use the tree structure to present the feature of music object. It raise the similarity precision of music object and the efficiency is improved. In addition, the value of F-measure that is obtained by Naïve method is same but it is not by our method. Due to the different music properties of data, the Naïve method cannot discriminate the difference, e.g. pitch histogram of the data is similarity but duration histogram of the data is not. Although the music fragments of variation causes similar auditory sensation to the listeners. But in our method, we can discriminate the difference because of we utilize the structure concept.

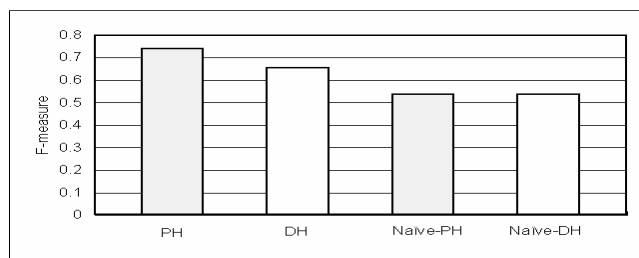


Fig. 63. The classification result of MF-tree vs. Naïve.

V. CONCLUSION

Based on the hierarchical rule, we construct MF-tree that consists of music theory feature and structure. In the system implementation, we use the linear method to normalize the music sequence. We use the system implementation to present our result.

In the future work, we will study more music theories to find the new music features, and we will consider the non-linear method to normalize the sequence. Finally, we will provide more experiment data to allocate our methods in the system.

REFERENCES

- [1] Bertino, Elisa, Giovanna Guerrini, and Marco Mesoti, "A Matching Algorithm for Measuring the Structural Similarity between An XML Document and A DTD and its Applications," *Information Systems*, Vol. 29, No. 1, March 2004.
- [2] Canfora, Gerardo, Luigi Cerulo, and Rita Scognamiglio, "Measuring XML Document Similarity: A Case Study for Evaluating Information Extraction Systems," in *Proc. of 10th IEEE International Symposium on Software Metrics*, 2004.
- [3] <http://delphi.ktop.com.tw/loadfile.asp?TOPICID=13896274&CC=310786>

- [4] Hsu, Jia-Lien, Chih-Chin Liu, and Arbee L. P. Chen, "Discovering Nontrivial Repeating Patterns in Music Data," *IEEE Transactions on Multimedia*, Vol. 3, No. 3, September 2001.
- [5] Hsu, Jia-Lien, Arbee L.P. Chen, and Hung-Chen Chen, "Finding Approximate Repeating Patterns from Sequence Data," in *Proc. of 5th International Conference on Music Information Retrieval*, 2004.
- [6] Jones, G. T., *Music Theory*, Harper & Row, Publishers, New York, 1974.
- [7] Krumhansl, C. L., *Cognitive Foundations of Musical Pitch*, Oxford University Press, New York, 1990.
- [8] Kuo, Fang-Fei, and Man-Kwan Shan, "A Personalized Music Filtering System Based on Melody Style Classification," in *Proc. of IEEE International Conference on Data Mining (ICDM)*, 2002.
- [9] Kuo, Fang-Fei, and Man-Kwan Shan, "Looking for New, Not Known Music Only: Music Retrieval by Melody Style," in *Proc. of the 4th ACM/IEEE-CS Joint Conference on Digital Libraries*, 2004.
- [10] Lee, Mong Li, Liang Huai Yang, Wynne Hsu, and Xia Yang, "XClust: Clustering XML Schemas for Effective Integration," in *Proc. of ACM International Conference on Information and Knowledge Management (CIKM)*, 2002.
- [11] Liu, Dan, Nai-Yao Zhang, and Han-Cheng Zhu, "Form Recognition for Johann Strauss's Waltz Centos Based on Music Features," in *Proc. of IEEE International Conference on Machine Learning and Cybernetics*, 2002.
- [12] Miura, Takao, and Isamu Shioya, "Similarity among Melodies for Music Information Retrieval," in *Proc. of ACM International Conference on Information and Knowledge Management (CIKM)*, 2003.
- [13] Narmour, E., *The Analysis and Cognition of Basic Melodic Structures*, The University of Chicago Press, Chicago, 1990.
- [14] <http://www.recordare.com/default.asp>
- [15] Seifert, Frank, and Wolfgang Benn, "Semantic Relationship and Identification of Music," in *Proc. of IEEE International Conference WEB Delivering of Music*, 2003.
- [16] Shan, Man-Kwan, Fang-Fei Kuo, and Mao-Fu Chen, "Music Style Mining and Classification by Melody," in *Proc. of IEEE International Conference on Multimedia and Expo. (ICME)*, 2002.
- [17] http://www.tabazar.de/frame_e.htm
- [18] Theodoridis, S., and Koutroubas, K., *Pattern Recognition*, Academic Press, 1999.
- [19] Wai, Man Szeto, and Man Hon Wong, "A Stream Segregation Algorithm for Polyphonic Music Databases," in *Proc. of 7th IEEE International Database Engineering and Applications Symposium (IDEAS)*, 2003.
- [20] Wang, Lian, David Wai-lok Cheung, Nikos Mamoulis, and Siu-Ming Yiu., "An Efficient and Scalable Algorithm for Clustering XML Documents by Structure," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 16, No. 1, January 2004.
- [21] Wold, E. T. Blum, D. Keislar, and J. Wheaton, "Content-based Classification, Search, and Retrieval of Audio," *IEEE Multimedia Magazine*, Vol. 3, No. 3, Fall 1996.