HIT-DML: A Novel Digital Music Library

Chaokun Wang, Jianzhong Li, and Shengfei Shi

Department of Computer Science and Engineering
P.O.Box 318, Harbin Institute of Technology,
150001, Harbin, Heilongjiang, China
chaokun@hit.edu.cn
lijz@mail.banner.com.cn
shengfei@0451.com

Abstract. The design and implementation of Harbin Institute of Technology-Digital Music Library (HIT-DML) is presented in this paper. HIT-DML adopts a novel framework which is inherently based on database systems. In this framework, musical data is structurally stored in the database, some algorithms of musical computation are implemented as algebraic mirco-operations in the database management system, and thus database technologies and multimedia technologies are combined seamlessly. A musical feature-matching algorithm and the appropriate dynamic index are also applied in HIT-DML. HIT-DML can retrieve musical information based on content, especially against different kinds of musical instruments.

Keywords: Digital Music Library, Content-Based Music Information Retrieval

1 Introduction

Digital libraries have been a phenomenal success in terms of facilitating organization and processing of large volume of music data. Current music digital libraries, however, are either unscalable or unable to provide content-based retrieval and play based on musical instruments, e.g. just search "do-re-mi" on a certain musical instrument, or just play a combination of several musical instruments of a piece of music. Usually they are based on one of the following techniques: storing a piece of music as a file in a directory or as one binary large object in a database.

Besides the performance inefficiency, a common problem with most existing music digital libraries is that they usually ignore the transaction processing algorithms and other advanced features devised by the Database community through decades of refinement and evaluation, and thereby rely on ordinary file systems. For example, the replication mechanism in distributed database systems can be used to backup data between two remote libraries.

In this paper, a novel framework for digital music library and the prototype Harbin Institute of Technology-Digital Music Library (HIT-DML) are introduced. This framework is inherently based on database systems. HIT-DML is an instance of the framework. As a key part of Infinite Digital Library Project (IDLP) in Harbin Institute of Technology, HIT-DML is planned to realize the functions of acquisition, conversion, storage, management, computation, retrieval and consumption of musical data. It is also the platform for our music information research.

T.M.T. Sembok et al. (Eds.): ICADL 2003, LNCS 2911, pp. 265–274, 2003.

HIT-DML not only has the basic functions to navigate, query and play musical data, but also has the functions to acquire and load musical data, to extract features from them and compare musical features. Different from VARIATIONS [1], NZDL [2], and AVPROT [3], HIT-DML is inherently based on database systems. HIT-DML can retrieve musical information based on content, further it centers on database systems. It structurally stores musical data in a database, and implements some music operations, e.g. feature-extracting, in the database. It can utilize the advantages of database systems, and combine multimedia technologies and database technologies seamlessly. Besides that, HIT-DML can query musical data according to various musical instruments. As far as we know, this function is not implemented by other current digital music libraries.

The rest of this paper is organized as follows. Section 2 provides a review of related work. Section 3 introduces and describes the framework of digital music library proposed in this paper. This framework is the basis of HIT-DML. It is not only a blueprint for digital music library, but also a reference to digital libraries of other media. Section 4 reports the implementation of the current HIT-DML prototype, and presents some experimental results. Section 5 concludes the paper and highlights some future research directions.

2 Related Work

There is much work on digital music libraries. Almost all of this work has concentrated on music meta-data instead of raw data. In [4], Fingerhut describes the transition of the IRCAM Music Library from a traditional setup to one tightly integrating digital technologies. Self-describing objects and referencing object parts are introduced in this paper. The IRCAM concert recordings are transferred to self-describing hybrid compact discs, and URLs are used to reference multimedia objects and their parts.

NZDL is one of the successful digital music libraries. A comprehensive suite of tools built for NZDL are described in [5] [6]. These tools gather musical material, convert between many of these representations, allow searching based on combined musical and textual criteria, and help present the results of searching and browsing.

The VARIATIONS/VARIATIONS2 digital library project [7] [8] at Indiana University currently is one of the first practical applications of digital music libraries. It delivers online access to music in a variety of formats - sound recordings, scanned musical scores, computer score notation files, and video - and is designed to support research and learning in the field of music.

Blandford [9] et al. have evaluated four web-accessible music libraries, focusing particularly on features that are particular to music libraries, such as music retrieval mechanisms. The main challenges identified relate to the details of melody matching and to simplifying the choices of file format.

There is some work on MIR in peer-to-peer (P2P) networks. It can be used to establish digital music libraries based on P2P environments. Four models for content-based music information retrieval are proposed in [10]. A prototype and an accelerating algorithm are also given. One transaction mechanism for music files exchange in a P2P system is described in [11], but it can not protect copyrights.

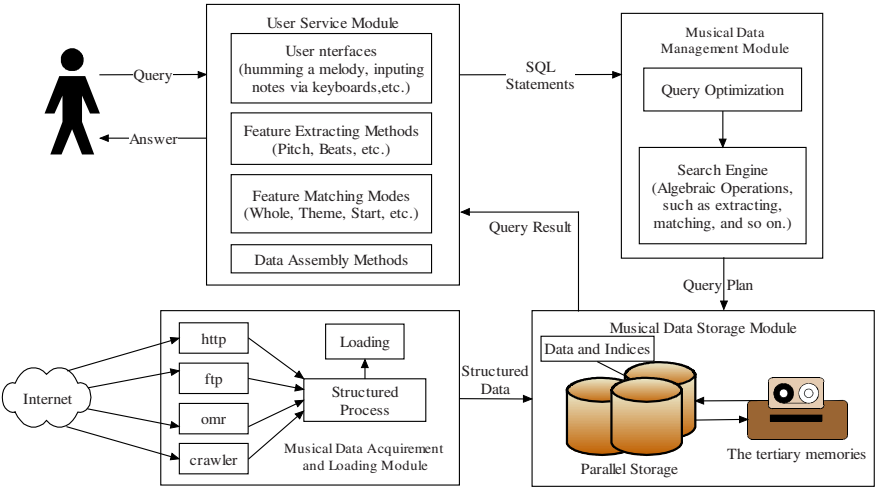

Fig. 1. The Framework of Digital Music Library

3 The Framework Design of Digital Music Library

The framework of digital music library is composed of four modules (Fig.1). This framework has two characteristics. One is that a novel data model [12] is adopted by it, and musical data instead of only musical features are structurally stored in databases. The other is that some music computing algorithms, such as feature-extracting and feature-matching, are realized as algebraic micro-operations of the database management system. In this framework, database technologies and multimedia technologies are combined organically. Not only meta-data based retrieval but also content based retrieval can be carried out. For example, search musical data similar to a piece of melody, especially to that in a certain musical instrument. Each module is described in detail as follows.

3.1 Musical Data Acquisition and Loading Module

There are a lot of musical data in a digital music library. It is the first step in the construction of a digital library to acquire these musical data. In order to more efficiently support applications of a digital library, musical data stored inside the library should be processed before loading.

There are various methods to acquire musical data. Using OMR (Optical Music Recognition) technique, music books can be scanned to pictures, from which digital musical information can be picked up. Through an AD converter, analog musical signals stored in tapes can be transformed into digital musical signals. Especially, with the increasing popularization of computer networks, there are very large musical data dispersed over the Internet. They can be gotten from FTP sites or Web sites, even by crawlers.

Musical data perhaps need be processed before loading of them into a database. For example, a musical file may be converted from its original format to a popular and better format, or be watermarked.

The loading of musical data is the process within which musical data is segmented and stored into the appropriate tables of a database based on the inherent structure of the data. In detail, there are two methods for data loading in a digital music library. The first method is to make a loading program, and use it to partition musical files outside the database, and then store the musical segments into the database. The second is to deposit musical files in a appointed directory, and then inside the database system use proper APIs of the database management system to read these files and process them, e.g. partition them, finally store musical segments into the appropriate database tables. Obviously the latter implements more operations inside the database system, and better combines multimedia technologies and database technologies.

3.2 Musical Data Storage Module

Musical Data Storage Module (MDSM) is an important component in a digital music library. This module is responsible for efficient storage and access of musical data to answer users' queries quickly. The digital library framework proposed in this paper supports not only meta-data-based query, e.g. titles of music, names of singers, but also content-based query, e.g. a piece of melody. These functions need the effective support from storage structures and access methods of musical data.

For storage structures, the structured storage schemas in a digital music library are designed in accordance with the data model proposed in [12]. In this model, musical data is stored in a database as certain structures instead of a BLOB (Binary Large OBject). This design supports more effective storage and retrieval of musical data. For instance, if a user only wants to listen the beginning of a piece of music, or the combination of several musical instruments of the piece of music, just the corresponding data is processed. It reduces the data transmitted, and increases the retrieval speed.

With the increase of musical data in the library, tertiary memories can be used to store the considerable volume of data. Parallel storage mechanism can also be used to accelerate the retrieval.

There are many kinds of meta-data and features for music. Features, e.g. frequency, hardness, loudness, pitch, harmony, energy spectrum, and Mel-Frequency Cepstral Coefficients (MFCC), are in common use with acoustic music. For access methods, many indices can be established based on various meta-data and features of music. These indices will improve the retrieval speed of the digital music library.

3.3 Musical Data Management Module

Musical Data Management Module (MDMM) is designed for query processing. After receiving the query extracted from a user's request, MDMM scans it and analyzes its grammar, and then carries out the optimization process based on certain rules to generate concrete query plan, finally executes the query codes of the plan to generate query result. With the introduction of new algebraic operations, some computing

methods in music processing field, e.g. feature-extracting and feature-matching, are fused into database management systems. Thus musical data management module can directly manipulate musical data in databases instead of simply access them. It is a speciality that the proposed digital music library framework has. Of course, this needs the support from given storage structure discussed in the previous subsection.

Music feature-extracting means to extract features from musical data by a certain algorithm. As far as we know, music feature-extracting in other digital music libraries is finished outside databases, i.e. features are extracted before loading of musical data into databases. In the framework proposed in this paper, feature-extracting of musical data is implemented inside the database management system.

Likewise, in the proposed framework, feature-matching of musical data is also implemented inside the database management system. Besides that, music feature-matching is an approximate process because of the error between the ideal input and the melody hummed by a user, which is introduced by the imprecision of the user' memory and noise of the input devices.

The musical data management module has the ability of query processing, and the ability of music computing, e.g. feature-extracting and feature-matching of musical data, inside the database system. It can realize the musical computation, while exert the merit of database systems, e.g. separation of programs from data, integrity of information, universal functions, control of data redundancies, data share, various views of data, and so on. So it can be used to increase the efficiency of digital music libraries.

3.4 User Service Module

User Service Module (USM) of a library is the window for interaction with users. It usually provides navigation and retrieval services on musical data. Under the constraint of certain QoS (Quality of Service), USM takes in access requests posed by users, and returns the results.

The DML framework proposed in this paper offers services for users via Web pages. From the point view of user management, Web services can be divided into two kinds: anonymous user service and registered user service. For anonymous users, the library offers LFU (least frequently used) mechanism to cache most frequently used musical data. When a request is posed by a user, the library firstly searches in the cache. If the result is found, it will be returned to the user, otherwise the request will be sent to the database system for a new search. Besides that, the framework also offers active sending service for a registered user. For instance, a user can register the musical style he likes in the library, and then the library will send appropriate information to him when music of this style is added.

From the point view of library functions, Web services can be divided into two kinds: navigation service and query service. Navigation service means that a user can navigate musical data stored in the library according to some metrics, such as musical styles, years of creation, and so on. Query service can be classified as the following four levels:

A) A user poses a request based on meta-data, then the system returns the result, where meta-data means fields used in the conventional music information retrieval, e.g. titles of music, names of composers, and so on. Also, the request may be a combination of meta-data.

B) A user writes a piece of melody, USM extracts useful features from it as a query, and then the library processes this query.

C) A user uploads a piece of music, the library processes the query extracted from it.

D) A user hums and records a piece of melody by a microphone, then uploads it. The library begins a query on it.

The last three cases are queries based on content of musical data, i.e. content-based music information retrieval. The last one is most natural. After receiving the query result, the user has several choices. He can download some pieces of music, play a piece of music integrally, just listen to the theme of a piece of music, or enjoy the combination of several selected musical instruments in a piece of music, and so on. The user can play, pause, rapidly play, and rapidly back during his enjoyment.

4 The Implementation of HIT-DML

In this section, the implementation of HIT-DML is presented. The feature-extracting and feature-matching algorithms used in HIT-DML are described firstly.

According to the music theory and MIDI specification [13], the range of notes can be extended to C_3-G^6, which can be denoted by integers in [0, 127]. The set of notes C_3...G^6 is named the note set, that is {C_3, $^{\#}C_3$, D_3, ..., F^6, $^{\#}F^6$,G^6}. An interval is the difference in pitch between the current note and the previous note. The unit of interval used in this paper is semitone. Based on interval, an algorithm for musical feature-extracting is implemented in HIT-DML. This algorithm is applied widely, and the idea of it can be found in [15]. An example is as follows.

Example1. Let the feature character set be *C = {R,U,W,D,B}*. *R* means the latter note is same to the former note. *U* means the latter is higher than the former by at more 3 semi-tones. *W* means the latter is higher than the former by at least 4 semi-tones. *D* means the latter is lower than the former by at more 3 semi-tones. *B* means the latter is lower than the former by at least 4 semi-tones. Then a segment of melody $C^1C^1D^1AE^1{}^{\#}F^1F^1$ is mapped to the feature string *RUBWUD*.

The music feature-matching used in current implementation of HIT-DML is based on an edit distance. The distance between two feature characters is as follows.

$$d = (d_{ij}) = \begin{bmatrix} d_{WW} & d_{WU} & d_{WR} & d_{WD} & d_{WB} \\ d_{UW} & d_{UU} & d_{UR} & d_{UD} & d_{UB} \\ d_{RW} & d_{RU} & d_{RR} & d_{RD} & d_{RB} \\ d_{DW} & d_{DU} & d_{DR} & d_{DD} & d_{DB} \\ d_{BW} & d_{BU} & d_{BR} & d_{BD} & d_{BB} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 & 0 \end{bmatrix}$$

For example, the distance between W and U is 1.

The distance between two feature strings with the same length is the sum of the distances of corresponding feature characters. Given a positive integer *n*, called the standard feature string length, the distance between two feature strings, *q* and *p*, with different length can be computed as follows. Firstly, compute the distances between each substring of *p* and each substring of *q* with length *n*. Second, select the minimum of the distances of the substrings of *p* and *q* as the distance of *p* and *q*. Please note that the length of *p* and the length of *q* are all more than *n*.

Example2. Suppose the feature character set C is that used in Example 1, feature string p is *RUWUWWDRRRU*, q is *WRBRRWDBBU*. $|p| = 11 > |q| = 10$, the standard feature string length is selected as $n = min\{|p|, |q|\} = 10$. The substrings of p with the length n are p_1= *RUWUWWDRRR*, p_2= *UWUWWDRRRU*. The distance $Dis(p_1, q) = 15$, $Dis(p_2,q) = 14$, and then $Dis(p, q)= min\{\varphi(p1,q), \varphi(p2,q)\} = 14$.

Using the digital music library framework proposed in Section 3 and the previous feature-extracting and feature-matching algorithms, HIT-DML prototype has been developed. It stores music data structurally in databases, and implements feature-extracting and feature-matching algorithms in algebraic micro-operations in the database system to combine database technologies and multimedia technologies organically. HIT-DML shows that the correctness of the proposed DML framework and algorithms.

To simplify the implementation, the system can be developed upon any existing database system. The open source database system PostgreSQL is adopted in HIT-DML, which has the database server-Web server-Web client structure. The database server is constructed on Redhat Linux 6.22, and the Web server is IIS 5.0 on Windows 2000.

Currently, only music data in MIDI (Musical Instrument Digital Interface) format is processed in HIT-DML. There are 1069 midi music pieces stored in HIT-DML. The music data is acquired from some Web sites and ftp sites on the Internet. Some meta-data, e.g. nationalities of composers, cannot be gotten from MIDI music, so they have to be added by hand. This information can be collected into a text file, which will be loaded into the database by a tool developed by ourselves, or by tools of PostgreSQL. During the loading of music data, the second method discussed in Section 3.1 is used.

MIDI data is structurally stored in HIT-DML according to TRACKs. The database comprises 34 tables, which are used to store meta-data, raw data, and feature data of music. PL/pgSQL functions in PostgreSQL are used to simulate the algebraic micro-operations during the implementation of MDMM. In detail, 10 PL/pgSQL functions are developed to implement the feature-extracting and feature-matching operations on MIDI music. Web service is developed by ASP technique, and VB script is used. Anonymous user service is provided in current HIT-DML. The advanced query interface is shown in Fig. 2.

Fig. 2. The Advanced Query Interface of HIT-DML

In HIT-DML, a user can query music in two different methods. He can query music by meta-data of music, e.g. titles of music and styles of music. He can also query music based on content according to feature information, e.g. melody. Especially, HIT-DML can query music in accordance to different musical instruments, and play the combination of several musical instruments. In order to serve the ordinary users, a simplified user interface is provided. In this interface, letters a, b, ..., h, ..., u respectively denote bass do, bass re, . . . , medilant do, ..., alt si.

Experiments are made on the database server to test HIT-DML. A segment of feature string is randomly selected from the database, and then the returns are carefully checked. One SQL statement [12] used in this experiment is

Retrieve (id, title, rank) from midi_db query="DBWDB" top 10.

It retrieves the top 10 pieces of music whose melodies similar to "DBWDB" from the database "midi_db". The result is ordered by the degree of similarity.

Titles of Songs (chinese)	Queries	Positions in Results
忘情水	lmlkmmpommlllmlk	4
为情所困	onmmmnm	5
特别的爱给特别的你	hijloqp	1
太傻	ihfefdedd	5
天意	jjjiijh	6
零时十分	kihehjjihe	4
领悟	llmompqp	6
最浪漫的事	eefhhhhf	5
大哥你好吗	qqqpqsqqssqs	4
白天不懂夜的黑	ebcdcbab	5

Fig. 3. The Veracity Test Results in HIT-DML

In the experiments for veracity, the testing set is composed of 10 feature strings. The lengths of them are between 6 and 15, whose mean is 9. In the results illustrated in Fig 3, the correct music files are always listed in the top set of returns. 1 piece of music is located in the first position, and 3 in 4th, 4 in the 5th, 2 in the 6th. It shows that our system has higher usability, because the query submitted by a user is inherently fuzzy. It is reasonable that several pieces of music correspond to a same feature string. Besides that, the user can navigate the returns to select the music he liked.

There are some works of index structure on music data. For example, Lee and Chen [16] propose four index structures based on suffix trees for music data retrieval. In order to accelerate retrieval, a dynamic index based on inverted lists is designed in HIT-DML. This index structure can efficiently improve retrieval speed. The experimental results about feature strings whose lengths are between 4 and 12 are illustrated in Fig. 4. Each value is the average of 5 tests. During the testing, the return set is very large for the shorter testing strings, and the testing string will be partitioned for the longer strings, so the time used to process the both kinds of feature strings is more.

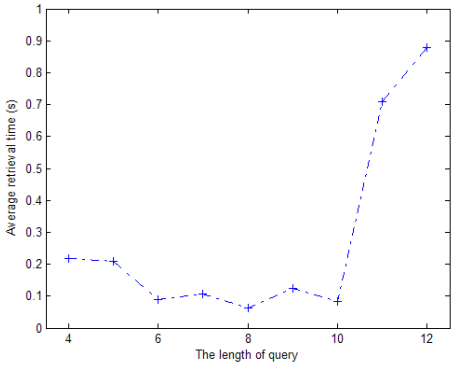

Fig. 4. The Average Retrieval Time Used in HIT-DML

5 Conclusion and Future Work

Our main contributions are: 1) A framework of digital music library is proposed which is constructed on a database system. Music data is structurally stored in a database. Many algorithms on music processing, e.g. feature-extracting and feature-matching, are implemented in the database management system. So multimedia technologies and multimedia technologies are combined seamlessly in the framework. 2) The implementation of HIT-DML, a digital music library prototype, is reported. A music feature-extracting and a music feature-matching algorithm are implemented in it. A dynamic index structure is also given. It shows the correctness of the proposed framework.

In the future, the fusion of database technologies and multimedia technologies will be further considered. Parallel distribution and query techniques on a large amount of music data will be studied.

References

1. VARIATIONS2: *Variations2* Digital Music Library Project Web Site. (http://dml.indiana. edu/)
2. NZDL: New Zealand Digital Library. (http://www.nzdl.org/)
3. AVPORT: Library of Congress. Digital Audio-Visual Preservation Prototyping Project Web Site. (http://www.loc.gov/rr/mopic/avprot/)
4. Fingerhut, M.: The IRCAM Multimedia Library: A Digital Music Library. In: Proceedings of the 6th IEEE Form on Research and Technology Advances in Digital libraries, Baltimore, MD, USA, IEEE Computer Society Press (1999) 129–140
5. Bainbridge, D., Nevill-Manning, C.G., Witten, I.H., Smith, L.A., McNab, R.J.: Towards a Digital Library of Popular Music. In: Proceedings of the 4th ACM International Conference on Digital libraries, Berkeley, California, USA, ACM Press (1999) 161–169
6. McNab, R.J., Smith, L.A., Witten, I.H., Henderson, C.L., Cunningham, S.J.: Towards the Digital Music Library: Tune Retrieval from Acoustic Input. In Fox, E.A., Marchionini, G., eds.: Proceedings of the 1st ACM International Conference on Digital libraries, Bethesda, Maryland, USA, ACM Press (1996) 11–18

7. Dunn, J.W., Mayer, C.A.: VARIATIONS: A Digital Music Library System at Indiana University. In: Proceedings of the 4th ACM International Conference on Digital libraries, Berkeley, California, USA, ACM Press (1999) 12–19
8. Minibayeva, N., Dunn, J.W.: A Digital Library Data Model for Music. In: Proceedings of the 2nd ACM/IEEE-CS Joint Conference on Digital libraries, Portland, Oregon, USA, ACM Press (2002) 154–155
9. Blandford, A., Stelmaszewska, H.: Usability of Musical Digital Libraries: A Multimodal Analysis. In: Proceedings of the 3rd International Symposium on Music Information Retrieval, Paris, France, IRCAM-Centre Pompidou (2002)
10. Wang, C., Li, J., Shi, S.: A Kind of Content-Based Music Information Retrieval Method in a Peer-to-Peer Environment. In: Proceedings of the 3rd International Symposium on Music Information Retrieval, Paris, France, IRCAM-Centre Pompidou (2002)
11. Grimm, R., Nutzel, J.: Peer-to-Peer Music-Sharing with Pro•t but Without Copy Protection. In: Proceedings of the 2nd International Conference on Web Delivering of Music, IEEE Computer Society Press (2002) 17–22
12. Wang, C., Li, J.: A Music Data Model and Its languages. Technique Report, CS, HIT (2003)
13. MIDINOTES: MIDI Notes. (http://www.argonet.co.uk/users/lenny/midi/notes.html)
14. Tang, M., Yip, C.L., Kao, B.: Selection of Melody Lines for Music Databases. In: Proceedings of the IEEE COMPSAC, Taipei (2000)
15. Dowling, W.J.: Scale and Contour: Two Components of a Theory of Memory for Melodies. Psychological Review 85(1978) 341–354
16. Lee W. and Chen. A. L. P.: Efficient Multi-Feature Index Structures for Music Data Retrieval. In Proceedings of SPIE Conference on Storage and Retrieval for Image and Video Databases (2000).