Journal of Computer Science 1 (3): 369-375, 2005
ISSN 1549-3636

Music Information Retrieval Schemes in Peer-to-Peer Environments

Chaokun Wang, Jianzhong Li and Shengfei Shi
School of Computer Science and Technology, Harbin Institute of Technology
Harbin 150001, People's Republic of China

Abstract: Peer-to-peer systems are useful tools for music applications. Existing peer-to-peer systems support the storing and sharing of music data, however they cannot effectively and efficiently support the content-based information retrieval and the cooperation of musicians. This study focuses on the methods of content-based music information retrieval in peer-to-peer environments. Four music information retrieval schemes are evaluated in detail on communication cost, retrieval time, update complexity and robustness. Peers-peers-coordinator scheme is found to be the best one from theoretical analysis and simulated experiments. Also, an algorithm is designed for an implementation of the peers-peers-coordinator scheme and a simple but effective method is brought forward to filter out the replica in the final results. The results of simulated experiments show the efficiency of the algorithm.

Key words: Music Information Retrieval, Peer-to-Peer Systems, CBP2PMIR

INTRODUCTION

Peer-to-Peer (P2P) systems have been greatly successful in facilitating storage and exchange of huge volumes of data since their scalability, fault-tolerance and self-organizing nature. In some P2P based data systems, such as Kazaa, music data is commonly a large part of its contents. This calls for techniques that provide users the abilities to store music data safely and retrieve music data quickly and accurately.

Example 1: Imagine an Internet-scale P2P music system that consists of peers ranging from desktops behind modem lines to powerful servers connected to the Internet through high-bandwidth lines. Each peer shares some resources with other peers, such as music data and portion of storage, for the common good of everyone. Peers in music system can interchange music data and other information. A user, for example, can store and publish a piece of music in the system without specifying its exact location. Also, he or she can search the whole system for a piece of music, for example, the piece of music titled as "My Sun". This example shows the role of P2P in storing and sharing of music data.

Example 2: Suppose that you are sitting before your computer connecting to a P2P music system. You suddenly hear a song. It's the best song you have heard for a long time, but you missed the announcement and don't recognize the artist. Wouldn't it be nice if you could hum a melody you remembered, e.g. the melody showed in Fig. 1, or push a few keys on your keyboard and a few seconds later the P2P system would tell you the name of the artist and the title of the music you're listening to? Perhaps the system even sends this piece of music to your computer. This example shows that content-based information retrieval in P2P systems is very important for music applications.

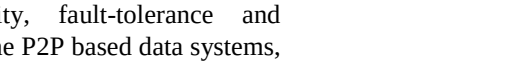

Fig. 1: A Sample Melody

Example 3: In a music community, the computers of composers, conductors, violinists, pianists, organists and so on, are connected to establish a peer-to-peer system. In this P2P system, many tasks, such as the creation of a huge opera, should be cooperated by several musicians together via the P2P network. Each musician could compose his part and communicate with others and then send his work to a certain musician who is responsible to combining all the parts into the final opera. When a composer is working on a symphony, he can use an organist's computer to tune various music instruments and use a pianist's computer to synthesize the symphony and use his own computer to test the symphony. This example shows the role of P2P in the cooperation of musicians.

From the examples above, it can be seen that P2P systems are useful tools for music applications and the P2P systems used in music applications must be have abilities of supporting storing and sharing of music data, content-based information retrieval and cooperation of musicians.

Although existing P2P systems support the storing and sharing of music data, but they can not effectively and efficiently support the content-based information retrieval and the cooperation of musicians. Thus, existing P2P systems are not suitable to music applications. The purpose of our research is to create a

P2P music system that has the three abilities above and supports most of music applications. In the rest of the study, CBP2PMIR is used to express content-based music information retrieval in P2P environments.

There has been a considerable amount of work on content-based information retrieval for music data in non-P2P environments. A variety of results were gained in symbolic music information retrieval (MIR), which is music information retrieval on music events, such as MIDI data [1, 2, 5, 6]. Other works focused on acoustic MIR, which is music information retrieval on music digital signals, such as WAV format [3, 4, 7]. Most of the previous research work on content-based music information retrieval is not related to P2P environments. Also, they cannot be directly applied to P2P systems.

Many P2P systems have been developed. They can be divided into three classes. One is centralized, such as Napster. The second one is decentralized, such as Gnutella. The third one is hybrid that combines the advantages of centralized and decentralized systems, such as Kazaa and Morpheus. All the P2P systems are identifier or keyword based rather than content based. They can hardly process music queries, such as a melody hummed.

Four schemes of CBP2PMIR are proposed in our previous conference studies [9] and the query processing methods in each one are also presented in it. The first two schemes are centralized. The third one is distributed. And the last one is hybrid.

The proposed four schemes have some common features. First, music resource is dispersed over the whole system. Second, music data is transferred directly from one peer to another. Finally, the systems behave in ad-hoc manner, that is, any peer can moves in to and moves out of the P2P system freely. The performance of the schemes is different in term of communication cost, robustness and information retrieval time. Also, it should be evaluated so that the best scheme can be selected.

Assumptions and Parameters: The key problem of CBP2PMIR is an optimization problem, that is, developing a music retrieval algorithm so that *COMM* and *TIME* are minimized under the constrain condition $RTN \geq n$, where *COMM* is the communication cost and *TIME* is the time of processing a music query *Q* and *RTN* is the number of music files in query result.

Parameters used in the following evaluation are defined in Table 1. In the rest of the study, *R* and *N* are used to denote the set of nonnegative real numbers and the set of positive integers respectively.

Communication Cost: In each scheme, the communication cost consists of four parts. The first part is the music feature *Qf* transferred from the querying peer to the coordinator or other peers. The second is the results, $\{(Pid^i, Mid^i_j, Cf(Qf, Mf^i_j))\}$, transferred from the coordinator or other peers to the querying peer, where Pid^i is the network identifier of the i^{th} peer, Mid^i_j is the identifier of the j^{th} music file in the i^{th} peer, Mf^i_j is the feature of the j^{th} music file in the i^{th} peer and *Cf* is a music feature-matching function, that is, $Cf: Mf \times Mf \rightarrow RANK$, where *Mf* is the set of music features and $RANK \subseteq [0, 1]$ is a set of real numbers.. The third is the set of the user's downloading requests, {(the network identifier of the querying peer, the network identifier of a destination peer, the identifier of a selected music file)}, transferred from querying peers to destination peers. The last is the music files downloaded.

From the description of the querying processes in the four music information retrieval schemes in [9] and the parameters in Table 1, we can easily derive the communication cost of each scheme. Due to the limitation of the paper length, we ignore the deriving process here and only give the results in Table 2.

Please note that $t' \leq t$, and $(w + w^2 + \ldots + w^d) \leq W$ when the number of peers in a system is sufficiently large.

Lemma 1: The *COMM* of PsC is less than that of PsC$^+$.

Proof: The result follows from Table 2.

Lemma 2: When the number of peers in a system is sufficiently large, the *COMM* of PsPs is less than that of PsC.

Proof: Since the number of peers is sufficiently large ($W \rightarrow \infty$), $t \rightarrow \infty$. Because of *w*, *d*, *q* and *re* being all constants, $w+w^2+\ldots+w^d-1$ is also a constant and there is an upper bound for *t'*. Thus, $[(w+w^2+\ldots+w^d-1)\times q]/(re\times t) \rightarrow 0$, $1-[(w+w^2+\ldots+w^d-1)\times q]/(re\times t) \rightarrow 1$ and furthermore $t'/t \leq 1-[(w+w^2+\ldots+w^d-1)\times q]/(re\times t)$. Finally, we have $(w+w^2+\ldots+w^d)\times q + t'\times re + n\times q' + n\times m \leq t \times re + q + n \times q' + n \times m$, that is, the *COMM* of PsPs is less than that of PsC.

Lemma 3: The *COMM* of PsPsC is not more than that of PsPs.

Proof: It only needs to show that when the query results are the same in PsPsC and PsPs schemes and user selects all files from the merged results, the *COMM* of PsPsC is not more than that of PsPs. Since user selects all files from the merged results in both schemes, the number of music files satisfying the user's query in PsPs is equal to that in PsPsC when the final results in the two schemes are the same, i.e. $t' = t''$.

According to definition of PsPsC scheme, *Cas* in PsPsC, a data structure in the coordinator for accelerating the information retrieval process, can accelerate music information retrieving by locating the retrieving to some but not all peers on which there are more music files matching user's query. Let *Hit* be the ratio of the number

Table 1: Definitions of Parameters

Parameter Name	Definition	Scope of Application
$q=2spid+sqf$	*Spid* is size, in bytes, of peer id, *sqf* is size, in bytes, of the music feature extracted from a user's query Q	all schemes
$re=2spid+smid+r$	*Spid* is size, in bytes, of peer id, *smid* is size, in bytes, of a music file's id and r is size, in bytes, of the matching value between this file and Q	all schemes
t	Number of music files satisfying Q	PsC, PsC$^+$
n	Number of music files a user selects	all schemes
$q'=2spid+mid$	*Spid* is size, in bytes, of peer id and *smid* is size, in bytes, of a music file's id	all schemes
m	Average size, in bytes, of music files	all schemes
W	Number of peers in P2P system	all schemes
w	Width of the system	PsPs, PsPsC
d	Depth of the system	PsPs, PsPsC
t'	Number of files satisfying Q	PsPs
W'	Number of peers satisfying Q	PsPsC
t''	Number of files satisfying Q	PsPsC

Table 2: Communication Cost of Each Scheme

Part	PsC	PsC$^+$	PsPs	PsPsC
1	q	$W \times q$	$(w + w^2 + \ldots + w^d) \times q$	$W' \times q$
2	$t \times re$	$t \times re$	$t' \times re$	$t'' \times re$
3	$n \times q'$	$n \times q'$	$n \times q'$	$n \times q'$
4	$n \times m$	$n \times m$	$n \times m$	$n \times m$
COMM	$q + t \times re+$ $n \times q' + n \times m$	$q \times W + t \times re+$ $n \times q' + n \times m$	$(w + w^2 + \ldots + w^d) \times q+$ $t' \times re + n \times q' + n \times m$	$W' \times q + t'' \times re+$ $n \times q' + n \times m$

of files satisfying user's query to the number of peers involved in the processing of user's query. *Hit* in PsPsC scheme is higher than *Hit* in PsPs scheme. That means $t''/W' \geq t'/(w + w^2 + \ldots + w^d)$. Thus $W' \leq (w + w^2 + \ldots + w^d)$ and $W' \times q + t'' \times re+ n \times q' + n \times m \leq (w + w^2 + \ldots + w^d) \times q + t' \times re + n \times q' + n \times m$, that is, the *COMM* of PsPsC is not more than that of PsPs.

Theorem 1: When the number of peers in the system is sufficiently large, the *COMM* of PsPsC is the least.

Proof: The result follows from Lemmas 1, 2, and 3.

Retrieval Time: The retrieval time of each scheme is composed by three parts. The first part is the time for computation, i.e. the time used for feature-matching. The second part is the time used to merge the local results from the peers and to sort the final results. The last part is the time used for communication that has been discussed in the above text. Thus we will consider the time for computation, merging and sorting here.

The retrieval time of each scheme is listed in Table 3, where $Ai = \{(Mf^i_j, Cf(Qf, Mf^i_j)) \mid j \in N\}$ is the set of features of music files stored on the i^{th} peer, $A = \{(Mf^i_j, Cf(Qf, Mf^i_j)) \mid i, j \in N\} = \sum i \in N\, A_i$ is the set of features of all music files stored in a whole P2P system, $T(A_i)$ is the time for computing and sorting $Cf(Qf, Mf^i_j)$ of A_i, $T(A)$ is the time for computing and sorting $Cf(Qf, Mf^i_j)$ of A, $M\{t\}$ is the time for merging the result of t files and $M\{t'\}$ and $M\{t''\}$ are similar to $M\{t\}$.

It is obvious that the *TIME* of PsC is more than others. Because $W' \leq W$, $M\{t''\} \leq M\{t\}$, the *TIME* of PsPsC is less than the *TIME* of PsC+. Because $W' \leq (w + w^2 + \ldots + w^d)$ when $t'' = t'$ from Lemma 3, the time for computing and merging in PsPs is more than $max\{T(A_i)\} + M\{t''\}$, that is, the *TIME* of PsPsC is less than the *TIME* of PsPs. Thus the *TIME* of PsPsC is the least.

Update Complexity and Robustness: The update in a P2P system includes music file update on a peer and peer moving in to or moving out of the P2P system.

In PsC scheme, a peer sends its network identifier and the features of its shared music pieces to the coordinator when it moves in to the P2P system. The information of the peer should be deleted from the coordinator when it moves out of the P2P system. The information of a document should be added into or deleted from the coordinator when it is added into or deleted from the shared music set of a peer.

In PsC$^+$ scheme, the network identifier of a peer is saved in the coordinator when it moves in to the P2P system. The information should be deleted from the coordinator when it moves out of the P2P system. In PsPs scheme, the network identifier of a peer is saved in its neighbor peers when the peer moves in to the P2P system. When the peer moves out of the P2P system, its information should be deleted from its neighbor peers. Document update of a peer in these two schemes has no effect on other peers.

Table 3: Retrieval Time of Each Scheme

Part	PsC	PsC^+	PsPs	PsPsC
Computation time	$T(A)$	$max\{T(A_i)\}$	$max\{T(A_i)\}$	$max\{T(A_i)\}$
Merge time	—	$M\{t\}$	$M\{t'\}$	$M\{t''\}$
TIME	$T(A)$	$max\{T(A_i)\}+M\{t\}$	$max\{T(A_i)\}+M\{t'\}$	$max\{T(A_i)\}+M\{t''\}$
Remark	—	$i = 1, \ldots, W$	$i = 1, \ldots, (w + w^2 + \ldots + w^d)$	$i = 1, \ldots, W'$

In PsPsC scheme, the network identifier of a peer is saved in the coordinator and its neighbor peers when it moves in to a P2P system. Also, the statistic of the shared music files of the peer is saved in the coordinator. When the peer moves out of the P2P system, the information is deleted from the coordinator and its neighbor peers. After a piece of music is added into or deleted from the shared music set of a peer, the corresponding statistic saved in the coordinator should be updated if the difference between the new statistic and the old one is more than a value specified by the system.

In conclusion, when a peer moves in to or moves out of a P2P system, the update cost of PsC^+ scheme is smallest, the update cost of PsPs scheme is smaller than the update costs of PsPsC scheme and PsC scheme. When music files are updated on a peer, the update costs of PsC^+ and PsPs scheme are all smallest, the update cost of PsC scheme is largest and the update cost of PsPsC scheme is between them.

In PsC and PsC^+ schemes, the coordinator is easily overloaded and becomes the bottleneck of the whole system. P2P systems constructed by these schemes have weak robustness. For example, if the coordinator is attacked by denial of service from a malicious peer, the system can be failed. Inversely, PsPs scheme has strong robustness because it is fully distributed. However the number of messages sent by peers in the scheme is numerous. The communication cost of PsPs scheme is high. PsPsC is a hybrid system that takes the advantages of PsC, PsC^+ and PsPs schemes. It can continue working via neighbor peers when there is something wrong with the coordinator. PsPsC scheme has strong robustness.

Performance Evaluation of Schemes: From the above comparison, PsPsC scheme is better than others in terms of communication cost, retrieval time and robustness except its update performance being lower in some cases. Thereby our CBP2PMIR system is developed in the light of this scheme.

In order to compare the performance of the four schemes, a simulator, which simulates P2P systems of 10,000 PIII/600 personal computers, is first created. Then, four sets of music files with sizes of 1,268,870, 3,169,806, 5,073,459 and 6,341,780 are respectively used. Each set is generated by all the peers each of which randomly generates one of its subsets based on different mean and standard deviation between 0 and 1 under the normal distribution. Finally we run 15 queries on the four sets in four music information retrieval schemes mentioned previously. The features of the queries are 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85 and 0.9. The parameters *spid*=4bytes, *sqf*=8bytes, *smid*=4bytes, *r*=8bytes, so *q*=16bytes and *re*=20bytes. The experimental results are shown in Table 4. In the table, each retrieval time is the average value of retrieval times of the 15 queries. The retrieval time of a query is the elapsed time from the time of the query being submitted by a user to the time of the query result being sent to the user.

The retrieval time in PsC scheme is very long because the comparison of all documents with the query and the sorting of results are performed in the coordinator. When the computation is dispersed over the peers, retrieval time comes down as in PsC^+ scheme. If there are some indices in the coordinator, the retrieval time used in PsC and PsC^+ schemes will be shorter. The retrieval time in PsPs scheme is short, but the query result may be lost because that some peers may not be searched during the query processing. Time used in PsPsC scheme is the smallest because the query is processed approximately and computation is performed in distributed manner.

If a user only needs approximate answer, PsPsC scheme is the best selection. In this scheme, if the coordinator failed, the system can still run in PsPs scheme whose performance is still very high. If a user needs exact answer, PsC^+ is a good selection.

Implementation of PSPSC Scheme: In a P2P environment, there are usually a lot of music files similar to a query submitted by a user. However, the user often wants a portion of the whole similar music files. Therefore, approximate query processing is very common for music information retrieval in P2P environments and then PsPsC scheme is very important. In this study, an implementation of PsPsC scheme is also proposed. Two key problems are considered. The first problem is how to find the set of destination-like peers. In a P2P data system, a peer is called a destination-like peer of a query if there are perhaps lots of music files similar to the query on the peer. The second problem is how to filter out the repeated music files. In the following discussion, *PNIopt* represents the set of destination-like peers, each element in which is the network identifier of a destination-like peer.

In music theory, an interval is the difference in pitch between the current note and the previous note. The unit of interval used in this study is semitone. For example, the interval between "do" and "re" is 2 semitones, while

Table 4: Average Retrieval Time in Each Scheme in a Simulator (Seconds)

Number of docs	PsC	PsC$^+$	PsPs	PsPsC
1,268,870	19.1769	1.0280	0.0973	0.0256
3,169,806	47.9910	2.6715	0.1670	0.0319
5,073,459	78.3206	3.9243	0.2304	0.0461
6,341,780	99.1541	6.5202	0.2831	0.0526

the interval between "re" and "mi" is 1 semitone. A sequence of intervals can be used to represent a melody. For example, the sequence of interval of the melody in Fig. 1 is (5 1 2 -2 -1 -2 2). Also, a sequence of intervals can be used to represent a piece of music because a monophonic melody can be extracted from a piece of music [8].

Definition 1: Let d be a positive integer. For a piece of music whose sequence of intervals is $Seq = (i_1 i_2 \ldots i_j \ldots i_{Nsum})$, $1 \le j \le N_{sum}$, i_j is an interval value, N_{sum} is the number of all intervals in Seq. Let N_d be the number of intervals whose absolute values are not less than d, $Rat^d = N_d/N_{sum}$ is called the ratio of d interval of the piece of music.

Let d be 2, the ratio of d interval of the melody in Fig. 1 is 0.71. When d is given in a P2P system, Rat^d can be abbreviated to Rat. For two pieces of music m_1 and m_2, $Ratm_1$ and $Ratm_2$ are respectively the ratios of d interval of m_1 and m_2. Let *offset* be a small positive number, m_1 is called similar to m_2 if $|Ratm_1\text{-}Ratm_2|<offset$, that is, $Ratm_1$ is close enough to $Ratm_2$.

Definition 2: Let d be a given positive integer. For a set of music files, if a variable R^d is used to denote the ratio of d interval of each piece of music in the set, (a, b) is called the feature of d interval of this set, where (1) a is the mean of R^d; (2) b is the standard deviation of R^d. It is obvious that $0 \le a, b \le 1$.

Definition 3: Let d be a given positive integer. $S = \{(a, b) \mid 0 \le a, b \le 1\}$ is called the space of d interval features, where $(a, b) \in S$ if and only if (a, b) is the feature of d interval of a certain music set.

Definition 4: Given $0 \le a_m, b_n \le 1$ $(m, n = 0, 1, 2, \ldots, v)$, $a_0 = 0 < a_1 < a_2 < \ldots < a_{v-1} < a_v = 1$ and $b_0 = 0 < b_1 < b_2 < \ldots < b_{v-1} < b_v = 1$, S is partitioned by these points into v^2 subspaces that are denoted by S_{ij}, where

(1). $S_{ij} \ne \Phi$, $1 \le i, j \le v$,
$S_{ij} = \{(a, b) \mid a_{i-1} \le a < a_i, b_{j-1} \le b < b_j\}$, $1 \le i, j \le (v - 1)$;
$S_{ij} = \{(a, b) \mid a_{i-1} \le a \le a_i, b_{j-1} \le b < b_j\}$, $i = v$, $1 \le j \le (v\text{-}1)$;
$S_{ij} = \{(a, b) \mid a_{i-1} \le a < a_i, b_{j-1} \le b \le b_j\}$, $1 \le i \le (v - 1)$, $j = v$;
$S_{ij} = \{(a, b) \mid a_{i-1} \le a \le a_i, b_{j-1} \le b \le b_j\}$, $i = j = v$;

(2). $S_{i_1 j_1} \cap S_{i_2 j_2} = \Phi$, $1 \le i_1, i_2, j_1, j_2 \le v$, $i_1 \ne i_2$ *or* $j_1 \ne j_2$;

(3). $\bigcup_{i,j=1}^{v} S_{ij} = S$.

$\{S_{ij} \mid i, j = 1, \ldots, v\}$ is called a partition of S and $(a_0, a_1, \ldots, a_v, b_0, b_1, \ldots, b_v)$ is called the partitioning sequence.

Definition 5: $S_{(k,h)}$ is called the expanding set of (k, h), if (k, h) is the feature of d interval of a certain music set, $S_{(k,h)}=\{(a,b) \mid a \in [max(k\text{-}\alpha, 0), min(k+\alpha, 1)], b \in [max(k\text{-}\beta, 0), min(k+\beta, 1)]\} \subseteq S$, *where* $\alpha, \beta \in [0, 1]$. Then α is called the expanding factor on mean, β is called the expanding factor on standard deviation.

Definition 6: Supposing that $E \subseteq S$, $\{S_{ij} \mid i, j = 1, \ldots, v\}$ is a partition of S, $G \subseteq \{S_{ij} \mid i, j = 1, \ldots, v\}$, G is called the minimal overlay of E if (1) $E \subseteq \bigcup_{S' \in G} S'$; (2)$G' \subseteq G$, $E \subseteq \bigcup_{S' \in G'} S' \Leftrightarrow G' = G$.

Definition 7: $S = \{(a, b) \mid 0 \le a, b \le 1\}$ is the space of d interval features, $\{S_{mn} \mid m, n = 1, \ldots, v\}$ is a partition of S. In the PsPsC scheme, the mapping $f:\{Pid^i\} \rightarrow \{S_{mn}\}$ is called a partition mapping, if $f(Pid^i)=S_{mn} \Leftrightarrow (a_i, b_i) \in S_{mn}$, where (a_i, b_i) is the feature of d interval of the shared music set on the i^{th} peer.

Please note that the shared music files on a peer may have more than one feature of d interval. For example, the shared music files on a peer can be clustered into several sets and then one feature of d interval can be extracted from each set. To simplify the discussion of the implementation, a peer only corresponds to one feature of d interval in this section, that is, a peer only corresponds to one set of music files. But it can be extended to be suitable for other environments.

In the rest of this study, $ff(S_{mn})$ is used to denote the set

$\{Pid^i \mid f(Pid^i)=S_{mn}, i \in N\}$. An implementation of *Cas*, the data structure in the coordinator, is $\{ff(S_{mn}) \mid m, n = 1, \ldots, v\}$.

The Algorithm to Find *PNIopt*: Given parameters d, partitioning sequence, expanding factors α and β, the algorithm to find *PNIopt* can be described as follows.

Input: a music query Q, which is a piece of music, a song sung, or a melody hummed.

Output: the set of destination-like peers *PNIopt*.
Steps:

* Extract the sequence of intervals of Q;
* Compute Rat_Q - the ratio of d interval of Q. Please note that Rat_Q is considered as Qf in this implementation of PsPsC scheme, that is, $Pf(Q)=Qf=Rat_Q$, where Pf denotes the operation of computing the ratio of d interval of Q from the sequence of intervals of Q;
* Compute $Qps=Stati(\{Q\})=(a_Q, b_Q)$, where $Stati(\ )$ is the operation of computing the feature of d interval of a set of music files. Obviously $a_Q = Rat_Q$, $b_Q = 0$;
* Compute $S(a_Q,b_Q)$, - the expanding set of Qps;
* Compute G - the minimal overlay of $S(a_Q,b_Q)$, $G \subseteq \{S_{mn} \mid m, n = 1, \ldots, v\}$;
* Compute PNI_{opt}
  $= \bigcup_{S' \in G} ff(S') = \bigcup_{S' \in G} \{Pid^i \mid f(Pid^i) = S', i \in N\}$
  $= \{Pid^i \mid f(Pid^i) \in G, i \in N\}$;
* Return PNI_{opt}.

PNIopt can be refined by tuning partitioning sequence, v, α and β.

Filtering Method: In the implementation of PsPsC scheme, the coordinator sends *PNIopt* to the querying peer. The querying peer sends Rat_Q, the feature of the music query, to these destination-like peers. Each destination-like peer, say the k^{th} peer, returns the local result, that is, Pid^k and $\{(Mf^k_j, Cf(Rat_Q, Mf^k_j)) \mid Cf(Rat_Q, Mf^k_j) < offset, j \in N\}$, to the querying peer, where *offset* is the matching condition given by the user. The querying peer receives all results, sorts them and exhibits them to the user. Repeated copies of a version of a piece of music stored in different peers may be returned. Thus, it is important to filter out the redundant music files in the result shown to the user.

In this study a simple and effective method is presented to filter out the repetitions in the results. Music files with the same content perhaps have different names in different peers, but they have the same sizes and usually have the same timestamps (the date and time attributes of files). So the sizes or the timestamps of music files with equal matching values can be used to judge whether these files are repeated. Usually the size of file is enough for the job.

Let the format of the result be Pid^k and $\{(Mf^k_j, Fs^k_j, Cf(Rat_Q, Mf^k_j)) \mid Cf(Rat_Q, Mf^k_j) < offset, j \in N\}$, where Fs^k_j is the size of the j^{th} similar music file in the k^{th} destination-like peer. During the merging and sorting of results from destination-like peers, if the querying peer finds some music files with the same matching values, it compares their sizes and then deletes the replica when they are the same.

RESULTS

Some simulated experiments are designed to show the efficiency of the proposed algorithm. In order to measure the experimental results, the following concepts are introduced.

Definition 8: Supposing that Q is the input query, *Clist* is the set of network identifiers of all peers and $PNIopt \subseteq CList$ is the output,

$$PR = \frac{|PNI_{opt}|}{|Clist|}$$

is called *peer-ratio*, which represents the percentage of *destination-like peers* in all peers.

Definition 9: Given a value of *offset*, the following ratio is called *hit-ratio*

$$HR = \frac{\sum_{i \in PNI_{opt}} |Ph^i|}{\sum_{i \in Clist} |Ph^i|},$$

where $Ph^i = \{Mid^i_j \mid Cf(Qf, Mf^i_j) < offset, j \in N\} = \{Mid^i_j \mid |Rat_Q - RatMid^i_j| < offset, j \in N\}$. *HR* represents the percentage of the number of music files similar to Q in PNI_{opt} to the number of all music files similar to Q in the whole P2P system.

Definition 10: The ratio of *HR* to *PR* is called *accelerating-ratio*

$$\eta = \frac{HR}{PR}.$$

η represents the efficiency of the proposed algorithm.

The following experiments are made in our simulator with 6,341,780 music files. The 15 queries mentioned previously are also processed in the simulator. The parameter v is set to 5 and the partitioning sequence (a_0, $a_1, \ldots, a_v, b_0, b_1, \ldots, b_v$) is altered from $a_0 = 0 < a_1 < a_2 < \ldots < a_{v-1} < a_v = 1$, $b_0 = 0 < b_1 < b_2 < \ldots < b_{v-1} < b_v = 1$ to $0 \le a_0 < a_1 < a_2 < \ldots < a_{v-1} < a_v \le 1$, $0 \le b_0 < b_1 < b_2 < \ldots < b_{v-1} < b_v \le 1$, $0 \le a_x, b_y \le 1$, $x, y = 0, 1, 2, \ldots, v$, where a_0, a_v, b_0 and b_v are determined by the experiment data. Different expanding factors α and β are selected to test the different effects. Then the averages of *PR*, *HR* and η are computed and shown as follows.

Figure 2 shows the relationship between average *HR* and *offset* at different expanding factors. When α increases, HR increases because more peers are searched for similar music to Q. The same is true of β. When *offset* increases, *HR* decreases. It is better to keep *offset* within

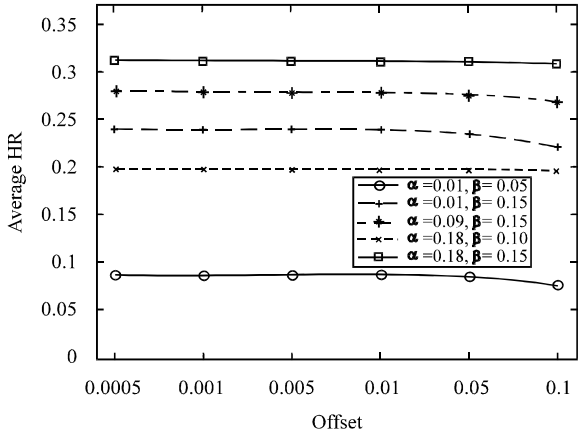

Fig. 2: The Average *HR* in the P2P System of 10000 Peers

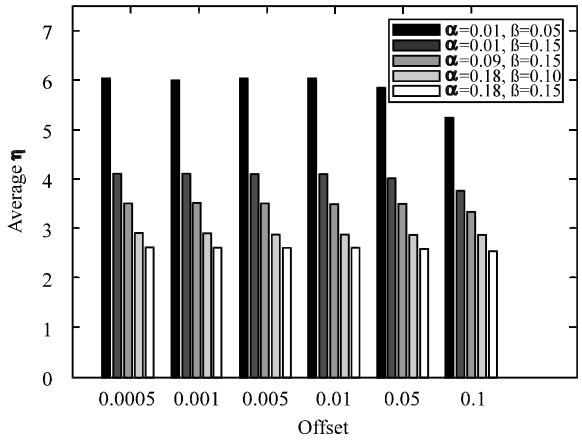

Fig. 3: The Average η in the P2P System of 10000 Peers

a proper range, such as [0.001, 0.01] in these experiments, to get the more stable *HR*.

Figure 3 shows the relationship between average η and *offset* at different expanding factors. When α increases, η decreases since peers with lower $|Ph|$ are searched. The same is true of β. When *offset* increases, η decreases because more irrelevant ingredients are involved in the retrieval process, such as more dissimilar music files to *Q*.

In conclusion, *HR* and η are always stable at certain expanding factors α and β. An expanding factor α or β can be increased to improve *HR*, but the accelerating ratio η will fall. When taking $\alpha = 0.01$, $\beta = 0.15$ and *offset* = 0.005, *HR* will be close to one fourth (23.91%) and η will be more than 4 (4.1191). It is a good tradeoff.

CONCLUSIONS

CBP2PMIR is introduced in this study. Four schemes of CBP2PMIR are evaluated in detail on communication cost, retrieval time, update complexity and robustness. PsPsC scheme is found out to be the best one for approximate queries and PsC+ is best for exact queries. After that, an implementation of PsPsC scheme is presented. Based on some useful concepts, an algorithm is designed to find the destination-like peers. A simple yet effective algorithm is also given to filter out the replica in the final results. Experiments show that these algorithms are very efficient.

AKNOWLEDGEMENT

This work was supported by the 973 Research Plan of China under Grant No. G1999032704, the NSF of China under Grant No. 60273082, the 863 Research Plan of China under Grant No. 2002AA444110 and the Army Research Plan of China under Grant No.41315.2.3.

REFERENCES

1. Downie, D. and M. Nelson, 2000. Evaluation of a Simple and Effective Music Information Retrieval Method. In Proceedings of the 23rd Intl. ACM SIGIR Conf. on Res. and Development in Information Retrieval, Athens, Greece, pp: 73-80.
2. Hsu, J.-L., C.-C. Liu and A.L.P. Chen, 2001. Discovering Nontrivial Repeating Patterns in Music Data. IEEE Transactions on Multimedia, 3: 311-325.
3. Jin, H. and H.V. Jagadish, 2002. Indexing Hidden Markov Models for Music Retrieval. In Proceedings of the 3rd Intl. Symposium on Music Information Retrieval, Service des Publications, Paris, France.
4. Liu, C.-C. and P.-J. Tsai, 2001. Content-Based Retrieval of MP3 Music Objects. In Proceedings of the 10th ACM Intl. Conf. on Information and Knowledge Management, Atlanta, Georgia, USA, pp: 506-511.
5. Melucci, M. and N. Orio, 1999. Musical Information Retrieval using Melodic Surface. In Proceedings of the 4th ACM Intl. Conf. on Digital libraries, Berkeley, California, USA, pp: 152-160.
6. Shalev-Shwartz, S., S. Dubnov, N. Friedman and Y. Singer, 2002. Robust Temporal and Spectral Modeling for Query by Melody. In Proceedings of the 25th Intl. ACM SIGIR Conf. on Res. and Development in Information Retrieval, Tampere, Finland, pp: 331-338.
7. Tzanetakis, G. and P. Cook, 2002. Musical Genre Classification of Audio Signals. IEEE Transactions on Speech and Audio Processing, 10: 293-302.
8. Uitdenbogerd, A.L. and J. Zobel, 1998. Manipulation of Music for Melody Matching. In Proceedings of the 6th ACM Intl. Conf. on Multimedia, Bristol, United Kingdom, pp: 235-240.
9. Wang, C., J. Li and S. Shi, 2002. A Kind of Content-Based Music Information Retrieval Method in a Peer-to-Peer Environment. In Proceedings of the 3rd Intl. Symposium on Music Information Retrieval, Service des Publications, Paris, France, pp: 178-186.