A Kind of Content-Based Music Information Retrieval Method in a Peer-to-Peer Environment

Chaokun Wang
Department of Computer Science and Engineering, #318
Harbin Institute of Technology
Heilongjiang, 150001, P.R. China
+86— (0)451—6415872
wangchaokun@0451.com

Jianzhong Li
Department of Computer Science and Engineering, #318
Harbin Institute of Technology
Heilongjiang, 150001, P.R. China
+86— (0)451—6415827
lijz@mail.banner.com.cn

Shengfei Shi
Department of Computer Science and Engineering, #318
Harbin Institute of Technology
Heilongjiang, 150001, P.R. China
+86— (0)451—6415872
shengfei@0451.com

ABSTRACT

In this paper, we propose four peer-to-peer models for content-based music information retrieval (CBMIR) and carefully evaluate them on network load, retrieval time, system update and robustness qualitatively and quantitatively. And we bring forward an algorithm to improve the speed of CBP2PMIR and a simple but effective method to filter out the replica in the final results. And we present the architecture of QUIND, a content-based peer-to-peer music information retrieval system, which can implement CBMIR. QUIND combines content-based music information retrieval technologies and peer-to-peer environments, and has strong robustness and good expansibility. Music stored and shared on each PC makes up of the whole available music resource. When a user puts forward a music request, e.g. a song or a melody, QUIND can retrieve a lot of similar music quickly and accurately according to the content of music. After the user selects his favorite ones, he can download and enjoy them.

1. INTRODUCTION

As the amount of music information increases rapidly, people raise higher demands for music retrieval. Instead of retrieving music by tag-like information (e.g., the title of music) people want to retrieve their favorite music by just providing a piece of similar music, such as singing a song or humming a melody. This kind of music retrieval is named Content-Based Music Information Retrieval (CBMIR), which means that given a piece of music, we should retrieve a lot of similar music from a depository of music only based on the content of music. In recent years, more and more researchers pay attention to CBMIR. And they have gotten a lot of results in this field [1~3].

Peer-to-peer (P2P) is another increasing research field these years, for example Gnutella [4]. As the basis of the Internet, now P2P becomes another focus of people. Differing from Client/Server, in P2P architecture there is not obvious discrimination that exists between client and server in C/S. Any device connected via P2P has the same degree and has the abilities of client and server. P2P makes PC become the center of the Internet. One can share the files saved on his hard disk, enjoy the files shared and preserved on PCs of other people and directly download these files if he likes them. For convenience we consider the node or peer of a P2P system as a personal computer (PC) in the rest of this paper and think they have the same meaning.

We believe that Content-Based Peer-to-peer Music Information Retrieval (CBP2PMIR) will be an interesting field as the combination of CBMIR technologies and P2P environments. And it will effectively utilize the gigantic music resource distributed on numerous PCs of the Internet. So far as we know, this paper is the first one about CBP2PMIR.

Many new problems arise in CBP2PMIR because there are a lot of tasks about music computation that must be finished in P2P environments. In this paper, we will research the following three questions. The first is what the appropriate model of CBP2PMIR is. We should have a proper model to work on it. The second is how to improve the retrieval speed in CBP2PMIR. There are so many songs in the P2P system that it is very important to find a good method to accelerate the retrieval. The last is how to filter the retrieval results. Perhaps many same music files exist in the original results, so we'd better find them and just retain one.

In this paper we propose four peer-to-peer models for CBP2PMIR and describe the query process in each one. And we discuss these models on network load, retrieval time, system update and robustness qualitatively and quantitatively. After that, we bring forward an algorithm to improve the retrieval speed based on several useful concepts. And we also put forward a simple but effective method to filter out the replica in the final results.

And we present the architecture of QUIND, a CBP2PMIR system, which implements CBMIR and consists of PCs and coordinators. Music stored and shared on each PC makes up of the whole available music resource. When a user puts forward a music request, for example a song or a melody, QUIND quickly and efficiently retrieves some similar music from the whole system according to the content of music represented as the features of music. After the user selects his favorite ones from the final results, he can download and enjoy them.

The rest of this paper is organized as follows. In next section we propose four different models of CBP2PMIR and describe the query process in each one. In section 3 we carefully discuss these models on network load, retrieval time, system update and robustness qualitatively and quantitatively and list the advantages and disadvantages of these models. And we present an accelerating algorithm and an effective filtering method in section 4. Then we describe the architecture of our system QUIND in section 5 and review related works in section 6. Finally we summarize our contributions in section 7.

2. MODELS

There are many new problems in CBP2PMIR. The first is what the model of CBP2PMIR is. The model of CBP2PMIR is very important because it is the basis of further research in the field.

In this section we developed four models of CBP2PMIR systems which can implement the content-based music information retrieval in peer-to-peer environments. The first two models have the centralized architecture, the third distributed, and the last hybrid. We will compare them in the next section.

2.1 PsC Model

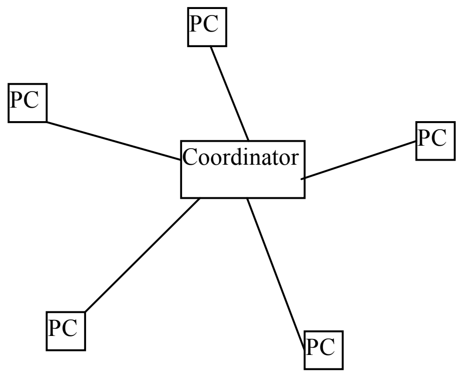

Figure 1. PsCM/PsC+M of CBP2PMIR systems

The Peers-Coordinator Model (PsCM) of CBP2PMIR can be represented as a triple (PC, Coordinator, Query).

(1) The PC consists of a network identifier, a music set and a same feature extraction method. Any element in the music set is comprised of a unique identifier in the PC, a music file and its music feature which is computed by the feature extraction method.

(2) The coordinator consists of a network identifier, a data structure to store all music features in the P2P system, and a feature matching method to compute the distance between two music features, i.e. the similarity between the two corresponding music files.

(3) The query is a request proposed by a user, which may be a piece of music uploaded, or a section of melody recorded.

In this model (Figure 1), any PC in the CBP2PMIR system connects with the coordinator. Music data are distributed over all PCs. PCs connect with each other via the coordinator, but they directly transfer music data from one to another.

The main steps of a query process in the model PsCM are the following:

0) Each PC shares some music stored on the local hard disk, and registers at the coordinator. The register information includes its network identifier, and the summary of music resource shared on itself, such as the identifiers of the music files and the features of them.

1) A user brings forward a music request on any PC of the system by providing a piece of music, singing a song or humming a melody (This PC is named the request PC). Then the feature of the music request is extracted from the music request by the feature extraction method and sent to the coordinator.

2) The coordinator receives the feature of the music request and compares it with all music features uploaded in step 0, and then sends the result to the request PC. The result contains the locations (i.e. network identifiers) of the PCs which store the music similar to the request (These PCs are named destination PCs), the identifiers of the similar music files and the matching values between these files and the music request.

3) After the request PC receives the result in step 2, the user selects his favorite music from it and then asks the destination PCs for connections.

4) If the connections are established, the user can download and enjoy his selection.

In the model PsCM, the system provides the directory service and feature matching is processed on the coordinator. We can get the model PsC+M, another realization of the centralized P2P system, if we put the feature matching process on each PC.

2.2 PsC+ Model

The Peers-Coordinator+ Model (PsC+M) of CBP2PMIR can also be represented as a triple (PC, Coordinator, Query).

(1) The PC consists of a network identifier, a music set, a same feature extraction method and a same feature matching method. Any element in the music set is comprised of a unique identifier in the PC, a music file and its music feature which is computed by the feature extraction method. The music feature matching method is used to compute the distance between two music features, i.e. the similarity between the two corresponding pieces of music.

(2) The coordinator consists of a network identifier and a data structure which stores the network identifiers of all PCs.

(3) The query is a request proposed by a user, which may be a piece of music uploaded, or a section of melody recorded.

In this model (Figure 1), the main steps of a query process are the following:

0) Each PC shares some music stored on the local hard disk, and registers at the coordinator. The register information is the location of this PC, i.e. its network identifier. The information of music resource shared on this PC does not be uploaded.

1) A user brings forward a music request on any PC of the system by providing a piece of music, singing a song or humming a melody (This PC is named the request PC). Then the feature of the music request is extracted from the music request by the feature extraction algorithm and sent to the coordinator.

2) The coordinator receives the feature of the request music and sends it to all PCs which have registered. Each PC compares it with the features of the local shared music and sends the local result to the coordinator. The local result includes the network identifier of this PC, the identifiers of some similar music files, and the matching values between these files and the music request. The coordinator gathers all the local results and sorts them according to the matching values, then sends the final result to the request PC. The final result contains the network identifiers of PCs which store the music similar to the music request (These PCs are named destination PCs), the identifiers of the similar music files and the matching values between these files and the music request.

3) After the request PC receives the final result in step 2, the user selects his favorite music from it and then asks the destination PCs for connections.

4) If the connections are established, the user can download and enjoy his selection through the request PC.

2.3 PsPs Model

In a distributed CBP2PMIR system, if one PC can send the same query to at most *m* PCs, *m* is said the width of the system. And if one query can be sent through at most *n* hops, *n* is said the depth of the system.

In the PsPs model, the CBP2PMIR system is formed by a lot of PCs without a coordinator. One case of the model can be illustrated in the figure 2.

The Peers-Peers Model (PsPsM) of CBP2PMIR can be represented as a 2-tuple (PC, Query).

(1) The PC consists of a network identifier, a music set, a same feature extraction method, a same feature matching method and a data structure that stores the network identifiers of PCs with which this PC connects (these PCs are named its neighbors). Any

element in the music set is comprised of a unique identifier in this PC, a music file and its music feature which is computed by the feature extraction method. The music feature matching method is used to compute the distance between two music features, i.e. the similarity between the two corresponding pieces of music.

(2) The query is a request proposed by a user, which may be a piece of music uploaded, or a section of melody recorded.

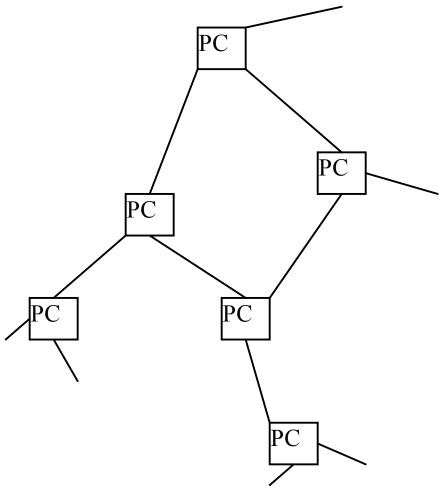

Figure 2. PsPsM of CBP2PMIR systems

Supposing that the width of the system is *m* and the depth of the system is *n*, the main steps of a query process in PsPsM are the following:

0) Each PC shares some music stored on the local hard disk, and gets the network identifiers of its neighbors by broadcast or other algorithms. The number of the neighbors of any PC is assumed more than *m*.

1) A user brings forward a music request on any PC of the system by providing a piece of music, singing a song or humming a melody (This PC is named the request PC). The feature of the music request is extracted from the music request by the feature extraction algorithm. Then the request PC compares it with the features of all shared music files on this PC, and sends it to *m* PCs randomly selected from the neighbors of this PC.

2) After receiving the feature of the request music, each of the *m* PCs sends it to *m* PCs randomly selected from its neighbors, and compares it with the features of the local shared music. After that, this PC sends the local result to the request PC. The local result includes the network identifier of this PC, the identifiers of some similar music files, and the matching values between these files and the request.

3) Each PC which receives the feature of the music request repeats step 2 until the hop number is equal to *n*.

4) The request PC gathers all the local results and sorts them according to the matching values. After the user selects his favorite music from the final result, the request PC asks the destination PCs for connections.

5) If the connections are established, the user can download and enjoy his selection through the request PC.

2.4 PsPsC Model

In the PsPsC model (Figure 3), there is one coordinator which each PC registers at. This coordinator collects and manages the statistical data from all PCs, then improves the retrieval speed based on these data. And the centralized architecture can be used to implement the distributed system according to this hybrid model.

The Peers-Peers-Coordinator Model (PsPsCM) of CBP2PMIR can be represented as a triple (PC, Coordinator, Query),

(1) Besides the 5 parts of the PC in PsPsM, the PC in PsPsCM has other two parts. The first is the PC feature and the second is the same PC feature extraction method.

(2) The coordinator consists of a network identifier, a data structure to store the network identifiers of all PCs, a data structure to store the PC features of all PCs, and an accelerating structure which can utilize the PC features to locate some proper PCs for faster CBMIR.

(3) The query is a request proposed by a user, which may be a piece of music uploaded, or a section of melody recorded.

The PC feature is used to characterize a PC, i.e. the music set of the PC. The accelerating structure is usually some rules between the network identifiers of all PCs and the PC features of them. Different PC features or accelerating structures will obtain different accelerating algorithms. A brief PC feature, an effective accelerating structure, and an accelerating algorithm based on them are presented in section 4.

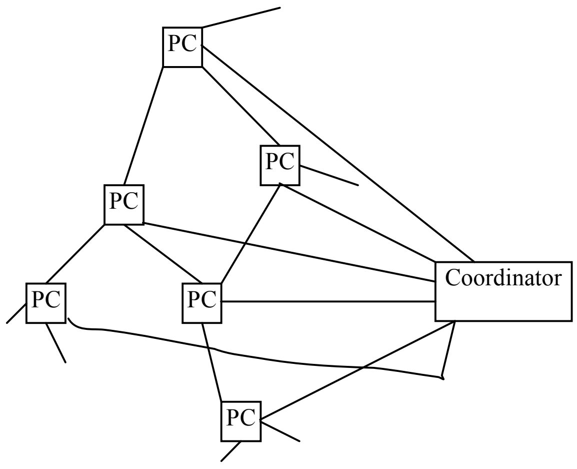

Figure 3. PsPsCM of CBP2PMIR systems

Given that the width of the system is *m* and the depth of the system is *n*, the main steps of a query process in the model PsPsCM are the following:

0) Each PC shares some music stored on the local hard disk, and registers at the coordinator. The register information includes its network identifier and its PC feature.

1) In the accelerating structure, the coordinator connects the network identifiers of all PCs in the system with the PC features of them under some rules.

2) A user brings forward a music request on any PC of the system by providing a piece of music, singing a song or humming a melody (This PC is named the request PC). The PC feature of the music request, i.e. the PC feature of the music set which includes only the music request, can be extracted from the music request by the PC feature extraction method and sent to the coordinator.

3) Based on the PC feature of the music request and the accelerating algorithm, the coordinator selects some network identifiers of PCs which should have more similar music to the request (These PCs are named the likely-destination PCs), and sends them to the request PC. Then the request PC sends the feature of the music request to these likely-destination PCs. And each likely-destination PC searches its shared music files and returns the local result to the request PC. The local result includes

the network identifier of this PC, the identifiers of some similar music files, and the matching values between these files and the request.

4) The request PC receives all results and sorts them according to the matching values. After the user selects his favorite music from the final result, the request PC asks the destination PCs for connections.

5) If the connections are established, the user can download and enjoy his selection through the request PC.

There is system update in these models. For example, when a PC takes part in a P2P system of PsCM, PsC+M or PsPsCM, its register information will be saved in the coordinator. And the information will be deleted if the PC exits from the system.

Table 1. Parameters table

Parameter name	**Description**	**Models used**
q	the size of the music feature extracted from a user's first query	all
re	the size of one record satisfying the first query	all
t	the number of the records satisfying the first query	PsCM, PsC+M
n	the final number of the music files the user selected	all
q'	the size of a user's second query for downloading	all
m	the average size of the music files	all
W	the number of PCs in a P2P system	all
w	the width of the system	PsPsM, PsPsCM
d	the depth of the system	PsPsM, PsPsCM
t'	the number of the records satisfying the first query	PsPsM
W'	the number of the PCs satisfying the first query	PsPsCM
t''	the number of the records satisfying the first query	PsPsCM

3. EVALUATION

The four models in section 2 have some common features, which are also the features of P2P systems. Firstly, the music resource is dispersed over the whole system. Secondly, the music data is transferred directly from one PC to another. Finally, the system can extend easily and quickly, that is any PC can take part in the P2P system handily.

However, there are still some differences among these models, such as network load, system robustness and so on. A best model should be selected from them because a proper one is needed for more effective working. So we will evaluate these four models qualitatively and quantitatively in this section.

3.1 The Description of CBP2PMIR

We can describe CBP2PMIR as the following optimization problem.

$$\begin{array}{ll} \text{min} & \text{COOR, LOAD, TIME} \\ \text{sub to.} & \text{RTN} \geq n \end{array}$$

Where COOR is the number of coordinators, LOAD is the whole network load, TIME is the total time used to retrieve similar music to the music request, and RTN is the number of music files as the final results. Because there is not a coordinator in PsPsM, and COOR has the same effect in the other three models, so we can simplify the above problem to the following form:

$$\begin{array}{ll} \text{min} & \text{LOAD, TIME} \\ \text{sub to.} & \text{RTN} \geq n \end{array}$$

We will discuss LOAD and TIME separately in the following two subsections. The size of the PC feature of the music request and that of likely-destination PCs are all small, and the time used to compute the likely-destination PCs is also short, so the process of finding the likely-destination PCs in PsPsCM is not considered during the evaluation.

Firstly we do some assumptions. We don't count the addition in size during the packaging process. The distribution of the music files satisfying the music request is not singular, i.e. the music files satisfying the user's query don't just exist in a finite area. And we don't think the P2P system has a limitation on the number of original results from the coordinator or other PCs, i.e. if a piece of music satisfies the matching condition, its identifier and the matching value between it and the music request will be sent to the coordinator or to the request PC and will not be lost. Along with that, if there is no file meeting the query on a PC, this PC will return nothing. And the user always selects n music files as the final choice and these files will be downloaded correctly.

3.2 Network load

Table 2. LOAD of each model

model **part**	**PsCM**	**PsC+M**	**PsPsM**	**PsPsCM**
1	q	$q\times W$	$(w+w^2+\cdots+w^d)\times q$	$W'\times q$
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	$n\times m$	$n\times m$	$n\times m$	$n\times m$
LOAD	$q + t\times re + n\times q' + n\times m$	$q\times W + t\times re + n\times q' + n\times m$	$(w+w^2+\cdots+w^d)\times q + t'\times re + n\times q' + n\times m$	$W'\times q + t''\times re + n\times q' + n\times m$

In each model, the network load consists of four parts. The first part is the music feature of the user's first query. The second is the original results from the coordinator or other PCs. The third is the user's second query for downloading. The last is the size of music files downloaded.

Convenient for understanding, the parameters used in the evaluation are listed in the table 1, and we compute the network load of each model in the table 2.

It can be verified that the LOAD of PsPsCM is the least when the number of PCs in the system is sufficiently large. We should note that commonly $t' \le t$, and $(w+w^2+\cdots+w^d) \le W$ when the number of PCs in the system is sufficiently large.

Firstly, it follows from the table 2 that the LOAD of PsCM is less than that of PsC+M.

Secondly, the LOAD of PsPsM is less than that of PsCM when the number of PCs in the system is sufficiently large. Since the number of PCs is sufficiently large, i.e. $W \to \infty$, $t \to \infty$. Note that w, d, q and re are all constants, $w+w^2+\cdots+w^d-1$ is a constant, and there is an upper bound for t'. Then $[(w+w^2+\cdots+w^d-1)\times q] / (re \times t) \to 0$, $1-[(w+w^2+\cdots+w^d-1)\times q] / (re \times t) \to 1$, and thus $t'/t \le 1-[(w+w^2+\cdots+w^d-1)\times q] / (re \times t)$. So $(w+w^2+\cdots+w^d) \times q + t' \times re + n \times q' + n \times m \le t \times re + q + n \times q' + n \times m$. It follows that the LOAD of PsPsM is less than that of PsCM when the number of PCs in the system is sufficiently large.

Thirdly, the LOAD of PsPsCM is not more than that of PsPsM. It suffices to show that when the final results are the same in PsPsCM and PsPsM, the LOAD of PsPsCM is not more than that of PsPsM. To simplify this case, it is assumed that the user selects all files from the final results. That means t' — the number of the records satisfying the first query in PsPsM, is equals to t'' — that in PsPsCM when the final results in the two models are the same.

According to the definition of PsPsCM, the accelerating structure in PsPsCM can accelerate CBMIR by locating proper PCs on which usually more similar music to the user's query are stored. So if Hit is used to denote the ratio of the number of records satisfying the user's query to the number of PCs which receive the user's query and search their local disks, $\text{Hit}_{\text{PsPsCM}}$ is higher than $\text{Hit}_{\text{PsPsM}}$. That means $t''/W' \ge t'/(w+w^2+\cdots+w^d)$. So $W' \le (w+w^2+\cdots+w^d)$, and thus $W'\times q + t''\times re + n\times q' + n\times m \le (w+w^2+\cdots+w^d)\times q + t'\times re + n\times q' + n\times m$. It follows that the LOAD of PsPsCM is not more than that of PsPsM.

Finally, based to the above discussion, when the number of PCs is sufficiently large, $\text{LOAD}_{\text{PsCM}} < \text{LOAD}_{\text{PsC+M}}$, $\text{LOAD}_{\text{PsPsM}} < \text{LOAD}_{\text{PsCM}}$, $\text{LOAD}_{\text{PsPsCM}} \le \text{LOAD}_{\text{PsPsM}}$. It follows that the LOAD of PsPsCM is the least when the number of PCs in the system is sufficiently large.

3.3 Retrieval time

The retrieval time of each model is composed of 3 parts. The first part is the time for computation, i.e. the time used for feature matching. The second is the time used to merge the local results from the PCs and sort the final results. The last is the time used for transmission, which could be measured by network load and has been discussed in the previous subsection. So the time used for computation, merging and sorting is discussed in this subsection.

Before the discussion of retrieval time, we assume that each PC sorts its local result and then sends them. As in the previous subsection, we compute the retrieval time of each model in the table 3, where A_i is the feature set of music stored on the i-th PC, and A is the feature set of all music stored in the P2P system. $\text{T}(A)$ is the time used to compute and sort the distances between the feature of the request and each element of A, and $\text{T}(A_i)$ the time used to compute and sort the distances between the feature of the request and each element of A_i. Merge$\{t\}$ is the time used to merge the result of t records. Merge$\{t'\}$ and Merge$\{t''\}$ have the similar meaning.

It is evident that $\text{TIME}_{\text{PsCM}}$ is more than the other three. Because $W' \le W$, merge$\{t''\} \le$ merge$\{t\}$, $\text{TIME}_{\text{PsPsCM}}$ is less than $\text{TIME}_{\text{PsC+M}}$. And because $W' \le (w+w^2+\cdots+w^d)$ when $t'' = t'$, the real time of computation and merging in PsPsM is more than $\max\{\text{T}(A_i)\}+$ merge$\{t''\}$, i.e. $\text{TIME}_{\text{PsPsCM}}$ is less than $\text{TIME}_{\text{PsPsM}}$. So $\text{TIME}_{\text{PsPsCM}}$ is the least.

3.4 Others

There is system update in P2P systems. When one PC connects into a P2P system, it registers at the coordinator or sends location message to other PCs for finding it. In PsCM and PsC+M, system update is processed by the coordinator, but in PsPsM, it is processed by each PC. Differing from them, system update can be processed by the coordinator or PCs themselves in PsPsCM.

When there is something wrong in the coordinator, PsCM and PsC+M will stop working, but PsPsCM will work continuously via the neighborship relation between PCs. Usually there isn't global network paralysis in PsPsM. So PsPsM and PsPsCM have stronger robustness.

Table 3. TIME of each model

model / part	PsCM	PsC+M	PsPsM	PsPsCM
computa-tion	$\text{T}(A)$	$\max\{\text{T}(A_i)\}$	$\max\{\text{T}(A_i)\}$	$\max\{\text{T}(A_i)\}$
merge	____	merge$\{t\}$	merge$\{t'\}$	merge$\{t''\}$
TIME	$\text{T}(A)$	$\max\{\text{T}(A_i)\}+$ merge$\{t\}$	$\max\{\text{T}(A_i)\}+$ merge$\{t'\}$	$\max\{\text{T}(A_i)\}+$ merge$\{t''\}$
remark	____	$i = 1, \cdots, W$	$i = 1, \cdots, w+w^2+\cdots+w^d$	$i = 1, \cdots, W'$

3.5 Summary

All in all, there are many differences in the four models. In centralized CBP2PMIR systems, i.e. PsCM and PsC+M, the coordinator is easily overloaded and becomes the bottleneck of the whole system. Then this kind of system has poor stability. For example, if there is something wrong with the coordinator, the system can not work correctly. Differing from centralized CBP2PMIR, the stability of distributed CBP2PMIR systems, e.g. PsPsM, is better. However the number of messages that are sent by PCs of distributed CBP2PMIR systems is numerous when the scale of the system expands. The cost of this kind of system is too expensive. Fortunately, from the above estimation, the balance of stability and cost can be obtained if we select the last model PsPsCM. And we can select the proper number of coordinators to make it better. Thereby we decide to develop our CBP2PMIR system in the light of this model, and call PsPsCM the model of CBP2PMIR systems in the rest of this paper.

4. ACCELERATING ALGORITHM

In this section we will deal with two important problems in CBP2PMIR. The first is how to accelerate the content-based music information retrieval in peer-to-peer environments. In P2P environments there are many applications to retrieve music by meta-data. And we can easily utilize the existing CBMIR algorithms, e.g. feature extraction and feature matching, in P2P environments. But there are so much music dispensed over a P2P system and we need a fast and effective method to retrieve music information based on the content of music. In this section we propose such an accelerating algorithm to implement this function. The intuitive justification of the accelerating algorithm is searching less PCs but getting more results.

The second problem in CBP2PMIR we put forward is how to filter out the repeated music files. In P2P systems each PC is self-existent. It is very common that many copies of a piece of music stored in different PCs. So it is very important to filter out the replica in the results returned to users. We bring forward a simple but effective method to filter out the same music in the last of this section.

4.1 The description of the accelerating problem

The definition of the CBP2PMIR model is described in subsection 2.4. We can describe the problem of accelerating retrieval as follows:

In CBP2PMIR system (PC, Coordinator, Query), the accelerating problem is how to find PNI_{opt}, i.e. a subset of the set of all PCs, and there is a lot of similar music to the Query in any PC which is a member of PNI_{opt}.

4.2 Several concepts

Interval is the difference in pitch between two tones of music. The unit of interval used in this paper is semi-tone. A piece of music can be considered as a sequence of intervals [5].

Given a constant positive integer *d*, intervals which absolute values are not less than *d* are named big intervals. For a piece of music, the ratio of the number of big intervals to the number of all intervals is named the ratio of *d*-interval of this piece of music. We assume that a proper *d* is predefined in this paper.

For convenience of the description of the accelerating method, in the rest of this paper two pieces of music are said similar each other if their ratios of *d*-interval are close enough.

For a set of music, if random variable *R* represents the ratio of *d*-interval of all pieces of music in this set, <*mea, std*> is said the feature of *d*-interval of this set, where *mea* is the mean of *R* and *std* is the standard deviation of *R*. It is evident that *mea* and *std* are between 0 and 1. And for a music set which has only one piece of music, the feature of *d*-interval of this set is just <the ratio of *d*-interval of the piece of music, 0>. All features of *d*-interval make up of the space of *d*-interval features *S*. Each element of *S* is the feature of *d*-interval of a certain music set.

Given $0 \le a_m, b_n \le 1$, $m,n = 0,1,2,3,\cdots,v$, $a_0 = 0 \le a_1 \le a_2 \le a_3 \le \cdots \le a_v = 1$, and $b_0 = 0 \le b_1 \le b_2 \le b_3 \le \cdots \le b_v = 1$, S is compartmentalized by these points into v^2 subspaces marked as S_{ij}, where

(1) $S_{ij} \ne \Phi$, $1 \le i, j \le v$,

$S_{ij} = \{\langle a,b \rangle \mid a_{i-1} \le a < a_i, b_{i-1} \le b < b_i\}, 1 \le i,j \le (v-1)$;

$S_{ij} = \{\langle a,b \rangle \mid a_{i-1} \le a \le a_i, b_{i-1} \le b < b_i\}$, $i = v$, $1 \le j \le (v-1)$;

$S_{ij} = \{\langle a,b \rangle \mid a_{i-1} \le a < a_i, b_{i-1} \le b \le b_i\}$, $1 \le i \le (v-1)$, $j = v$;

$S_{ij} = \{\langle a,b \rangle \mid a_{i-1} \le a \le a_i, b_{i-1} \le b \le b_i\}$, $i = j = v$;

(2) $S_{i1\,j1} \cap S_{i2\,j2} = \Phi$, $1 \le i1, i2, j1, j2 \le v$, $i1 \ne i2$ or $j1 \ne j2$;

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

We say $\{S_{ij} \mid i,j = 1,\cdots,v\}$ is the partition of S and $(a_0, a_1, \cdots, a_v, b_0, b_1, \cdots, b_v)$ the partitioning sequence (Figure 4.(a)).

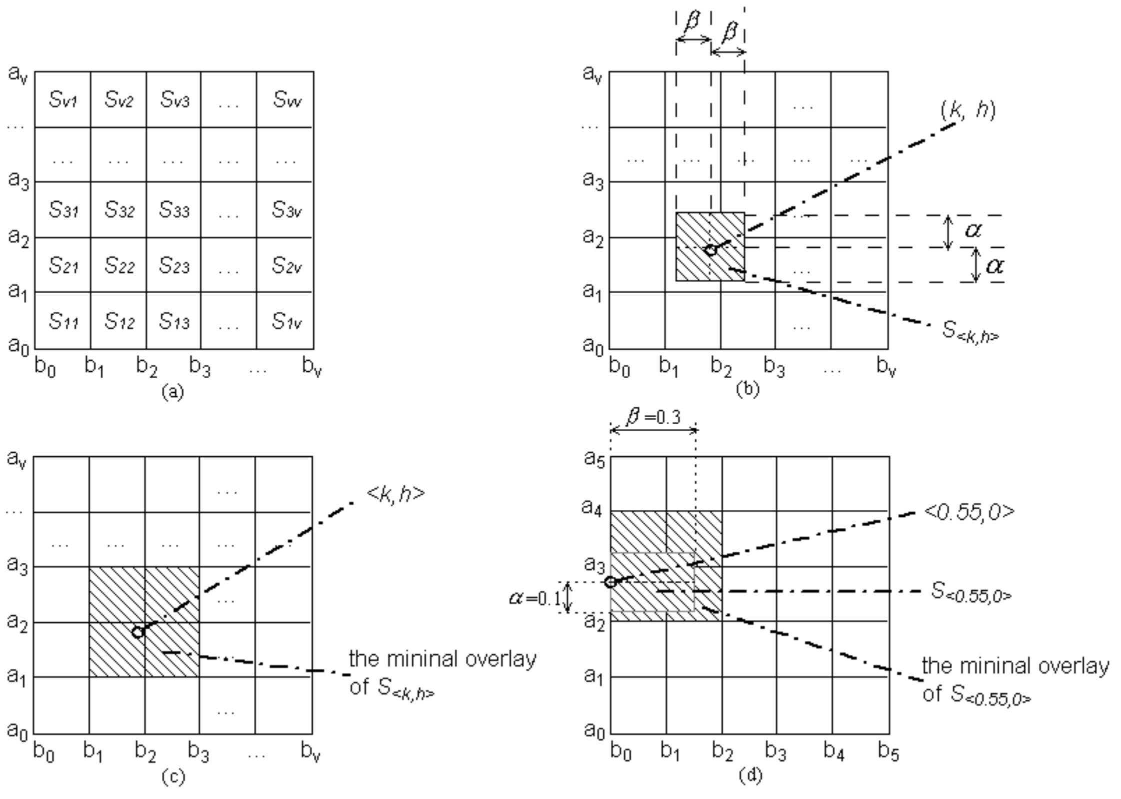

Figure 4. The accelerating algorithm

(a) The partition of S ; (b) The expanding set of <k, h> ;

(c) The minimal overlay of $S_{<k,h>}$; (d) The illustration of the example

On the supposition that <*k*, *h*> is the feature of *d*-interval of a certain music set, $S_{<k,h>}$ is named the expanding set of <*k*, *h*> , *α* the expanding factor in the mean direction, *β* the expanding factor in the standard deviation direction, where *α* and *β* are between 0 and 1 (Figure 4. (b)). $S_{<k,h>}$ is a subset of *S*. For any element of $S_{<k,h>}$, e.g. <*a*, *b*>, *a* is between max(*k*-*α*, 0) and min(*k*+*α*, 1), and *b* is between max(*k*-*β*, 0) and min(*k*+*β*, 1).

Let *E* be a subset of *S* and *G* a subset of the partition of *S*, *G* is said the minimal overlay of *E* if (1) *E* is a subset of the union of all elements in *G*, (2) there isn't a proper subset of *G* the union of whose elements is a superset of *E*. The minimal overlay of $S_{<k,h>}$ is illustrated in the figure 4. (c).

Each PC is mapped into one subspace of *S* according to the *d*-interval feature of the music set preserved and shared on it. This mapping is named the partition mapping. In the rest of this paper, we use $ff(S_{mn})$ to represent the set of PCs which are mapped into the subspace S_{mn} (*m*, *n*=1,…,*v*).

4.3 The accelerating algorithm

If the constant *v*, the partitioning sequence, the expanding factors α and β are defined, based on the concepts in subsection 4.2, the accelerating algorithm can be described as follows:

Input: A query *Q* which is a piece of music uploaded, a song sung, or a melody hummed by a user;

Output: PNI_{opt}, i.e. a subset of the set of all PCs, and there is a lot of similar music to *Q* in any PC which is a member of PNI_{opt}.

Steps:

1. Compute the feature of *d*-interval of the music set which consists of *Q*, say $\langle a_Q, b_Q \rangle$, obviously $b_Q = 0$;

2. Compute $S_{\langle a_Q, b_Q \rangle}$ — the expanding set of $\langle a_Q, b_Q \rangle$;

3. Compute the minimal overlay of $S_{\langle a_Q, b_Q \rangle}$;

4. Accumulate all $ff(S_{mn})$s to get PNI_{opt} where S_{mn} is an element of $S_{\langle a_Q, b_Q \rangle}$;

5. Return PNI_{opt}.

We can refine the size of PNI_{opt} , the accuracy and speed of retrieval by altering the partitioning sequence, v , α and β .

4.4 An example

Supposing that $v = 5$, $a_i = b_i = 0.2 \times i$, $i = 0,1,\cdots,5$, $\alpha = 0.1$, $\beta = 0.3$. *Q* is a query that a user inputs, and the ratio of *d*-interval of *Q* is 0.55. Then the feature of *d*-interval of the music set {*Q*} is <0.55, 0> (Figure 4. (d)).

The partition of S is $\{S_{i,j} \mid i,j = 1,\cdots,5\}$, and

$$S_{ij} = \{\langle a,b \rangle \mid a_{i-1} \le a < a_i, b_{i-1} \le b < b_i\}, 1 \le i,j \le 4 ;$$

$$S_{5j} = \{\langle a,b \rangle \mid 0.8 \le a \le 1, b_{i-1} \le b < b_i\} , \ i = 5 , 1 \le j \le 4 ;$$

$$S_{i5} = \{\langle a,b \rangle \mid a_{i-1} \le a < a_i, 0.8 \le b \le 1\} , \ 1 \le i \le 4 , j = 5 ;$$

$$S_{55} = \{\langle a,b \rangle \mid 0.8 \le a \le 1, 0.8 \le b \le 1\} , i = j = 5 ;$$

The expanding set of <0.55, 0> is

$$S_{\langle 0.55,0 \rangle} = \{\langle a,b \rangle | a \in [\max(0.55 - 0.1, 0), \min(0.55 + 0.1, 1)],$$

$$b \in [\max(0 - 0.3, 0), \min(0 + 0.3, 1)]\}$$

$$= \{\langle a,b \rangle | a \in [0.45, 0.65], b \in [0, 0.3]\} .$$

The minimal overlay of $S_{\langle 0.55,0 \rangle}$ is $\{S_{31}, S_{32}, S_{41}, S_{42}\}$.

The return is $PNI_{opt} = ff(S_{31}) \cup ff(S_{32}) \cup ff(S_{41}) \cup ff(S_{42})$, where *f* is the partition mapping.

□

4.5 The filtering method

In this subsection we propose a simple and effective method to filter out the repetition in the results. Because the music files with the same content must have the same size, and usually have the same date-time, we can compare the size or the date-time of music files with equal rank-values to judge whether they are repeated.

During the process of merging and sorting the results from other PCs, if the system finds some music files with same rank-values, it compares their size and date-time, and then deletes the replica when they are same. Usually the size of file is enough.

5. SYSTEM ARCHITECTURE

In this section we propose the architecture of QUIND (QUick fIND) — a CBP2PMIR system based on PsPsCM, which can retrieve music information according to the content of music in a P2P environment. QUIND consists of two kinds of parts. One is the PC, and the other is the coordinator.

5.1 PC

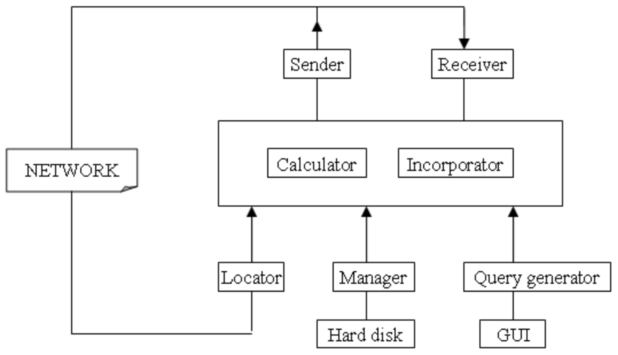

Figure 5. The structure of the PC in QUIND

The PC in QUIND consists of 7 components (Figure 5). The manager, short for the local music manager, is used to manage all music stored on the local hard disk and shared for the P2P system. On the one hand the manager processes the local music in advance to achieve the music features which will be used for feature matching, and produce the PC feature sent to the coordinator. On the other hand it stores the music features along with the meta-data of the local music, for example, the name of the music and the nationality of the author and so forth. Of course the information will be stored under a certain format and utilized in CBMIR.

The locator of a PC is a component that stores the network identifier of this PC. And it can identify and store the location information of the PCs in the P2P environment, with which it can connect. The information is just the neighbors in PsPsCM and acquired as the accumulation of previous $PNI_{opt}s$.

When a user puts forward a music request, for example singing a song or humming a melody, the query generator receives the user's input, i.e. the query in PsPsCM, and extracts music features from it to generate relevant music queries according to appropriate feature extraction algorithms and query-constructing rules. The sender component in a PC sends the generated music queries to the coordinator or likely-destination PCs, the logon and logoff information to the coordinator, the local retrieval result and the selected music on the hard disk to the request PC, and the download request to the destination PCs, etc. at different steps in the system. The receiver receives information sent by the sender of the coordinator or other PCs at different steps in the system.

On each PC of QUIND, the calculator component computes the matching values, i.e. the distances between the music request and the music files stored in the manager, and then ranks the results according to the resulted distances. The incorporator merges, sorts and filters all local results from itself and likely-destination PCs to form the final result, which will be sent to the user.

5.2 Coordinator

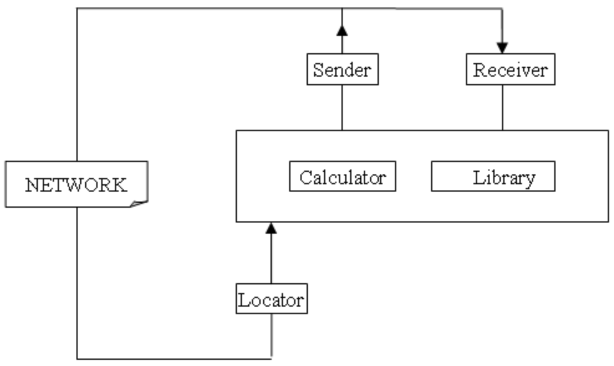

Figure 6. The structure of the coordinator in QUIND

The coordinator in QUIND is composed of 5 components (Figure 6). The locator of the coordinator stores the network identifier of the coordinator and the set of network identifiers of all PCs. The sender component in the coordinator sends the network identifiers of likely-destination PCs to the request PC. The receiver receives the PC feature of the music request that is sent by the request PC, and the logon, logoff and PC feature information of each PC in the system. The set of PC features of PCs in the system is stored in the library component. And the calculator component is the core of the coordinator because the accelerating algorithm proposed in section 4 is used in it.

QUIND has strong robustness. If there is something wrong with the coordinator, the whole system can still correctly work based on the neighbor information of each PC. The difference is just the retrieval speed in this situation becomes that of PsPsM. And we can add the number of coordinators in QUIND to improve its efficiency and robustness. It is obvious that the network identifier information in QUIND, including the set of network identifiers of all PCs at the coordinator and the network identifiers of its neighbors at each PC, should be updated regularly.

QUIND has good expansibility. The number of PCs in QUIND can change randomly. And we can add the number of coordinators with some modification. Feature extraction and feature matching are important issues in content-based music information retrieval, and we will discuss them in other papers.

6. RELATED WORK

There are many results in the fields of CBMIR and P2P. Byrd [6] reported his work on music retrieval in Conventional Music Notation (CMN) form. Chai [7] built a query-by-humming system which could find music based on a few hummed notes. Bainbridge [8] described a comprehensive suite of tools for building the music part of New Zealand Digital Library. Rowstron [9] described a kind of storage management and caching in a P2P environment. Stoica [10] presented a distributed lookup protocol to solve the locating problem in peer-to-peer applications. Napster [11] is another famous music search engine in P2P format, but it is not content-based. It can not accept a piece of music and retrieve the similar music based on content. Yang [12] developed a probabilistic model and an analytic model for “hybrid” P2P, but he didn’t consider other kinds of P2P. And his models cannot deal with content-based information retrieval.

Though there have been many prototypes about music information retrieval or P2P systems, as far as we know, this paper is the first one to consider the combination of content-based music information retrieval technologies and peer-to-peer environments.

7. CONCLUSION AND FUTURE WORK

In this paper, we propose four peer-to-peer models for content-based music information retrieval and describe the query process in each model. Then we carefully evaluate these models on network load, retrieval time, system update and robustness qualitatively and quantitatively. And we find PsPsCM is the best one of them.

After that, we bring forward two important problems in the field of CBP2PMIR. Based on some useful concepts, we design an accelerating algorithm to improve the retrieval speed. And we put forward a simple and effective method to filter out the replica in the final results.

Finally, we present the architecture of the content-based peer-to-peer music information retrieval system QUIND. It is based on PsPsCM and can implement content-based music information retrieval in a peer-to-peer environment. QUIND consists of PCs and coordinators. Music stored and shared on each PC makes up of the whole available music resource. When a user puts forward a music query, QUIND can retrieve some similar music quickly and accurately according to the content of music. After the user selects his favorite ones, he can download and enjoy them.

Now we are developing the system QUIND. And we will research other new problems about CBP2PMIR, such as the P2P protocols for music information retrieval based on content.

8. ACKNOWLEDGMENTS

This material is based on work supported in part by National Science Founds of China (69873014) and National 863 Plan of China (2001AA415410). And we thank the member of database laboratory at HIT.

9. REFERENCES

[1] S. Downie and M. Nelson. Evaluation and effective Music Information Retrieval Method. ACM SIGIR 2000, Athens, Greece, pp73~80

[2] J. L. Hsu, C. C. Liu, and A. L. P. Chen. Discovering Nontrivial Repeating Patterns in Music Data. IEEE transactions on multimedia, Vol3, No.3, September 2001, pp311~325

[3] G. Tzanetakis, G. Essl, and P. Cook. Automatic Musical Genre Classification of Audio Signals, ISMIR 2001, http://ismir2001.indiana.edu

[4] Gnutella. http://gnutella.wego.com/

[5] Stephen Handel. Listening: An Introduction to the Perception of Auditory Events. The MIT Press, 1989.

[6] D. Byrd. Music-Notation Searching and Digital Libraries. JCDL 2001, pp239~246

[7] W. Chai. Melody Retrieval On The Web, Master thesis, 2001, http://web.media.mit.edu/~chaiwei/papers.html

[8] D. Bainbridge, C. Manning, I. Witten, L. Smith, and R. McNab. Towards a Digital Library of Popular Music, ACM DL 1999, Berkeley, CA USA, pp161~169

[9] A. Rowstron and P. Druschel. Pastry: Scalable, Decentralized Object Location and Routing for Large-scale Peer-to-peer Systems, Proc. of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001). Heidelberg, Germany, November 2001.

[10] I. Stoica, R. Morris, M. Kaashoek, and H. Balakrishnan. Chord. A Scalable Peer-to-peer Lookup Service for Internet Applications, ACM SIGCOMM 2001, August 27-31, 2001, San Diego, California, USA, pp149~160

[11] Napster. http://www.napster.com/

[12] B. Yang and H. Garcia-Molina. Comparing Hybrid Peer-to-Peer Systems, VLDB 2001, September 11-14, 2001, Roma, Italy, pp561~570