Music Information Retrieval in P2P networks[i]

Ioannis Karydis Yannis Manolopoulos
Data Engineering Lab., Department of Informatics
Aristotle University of Thessaloniki, 54124, Thessaloniki, GREECE
Tel:++302310991912, Fax: ++302310991913
E-mail: {karydis,manolopo}@ csd.auth.gr

Music Information Retrieval in P2P networks

Keywords: Data Retrieval, Data Sharing, Indexing, Multimedia Database, Content-based Retrieval

Abstract
In this chapter we present the most significant trends in recent research in the field of content-based music information retrieval in peer-to-peer networks. Despite the diminished attention the area has received in general terms, the relatively close area of metadata MIR in P2P is by far new. As metadata prove to be inefficient for the purposes of MIR as well as the peculiarities of music in comparison to text and image data, developing dedicated solutions for CBMIR in P2P networks becomes a necessity while the challenges faced therein, unique. Depending on the type of P2P network, a number of prominent research works are presented and compared in this chapter.

INTRODUCTION

The World Wide Web (WWW) is being used for commercial, entertainment or educational purposes and has become the primary means for information dissemination. One popular type of data that is being disseminated over WWW is digitised music. Recently, the new opportunities that emerge from this activity have been recognised and led to the development of systems like iTune (www.apple.com/itunes), iMusic (www.imusic.com) and Napster (www.napster.com). Although abundantly used, even nowadays, traditional metadata (title, composer, performer, genre, date, etc.) of a music object give rather minimal information regarding the actual content of the music object itself. On the other hand, research efforts in the field of Music Information Retrieval (MIR) have developed efficient methods for searching music data collections by ***content***. For instance, queries based on humming (using a microphone) or on a small piece of musical file, are far more natural an approach to MIR. This type of queries lies within the Content-Based MIR (CBMIR). In CBMIR, an actual music piece is required in order to compare its content with the content of the music pieces already available in a database.
As with regard to the infrastructure for exchanging music data, peer-to-peer networks over the WWW have gain significant popularity. A peer-to-peer (P2P) network is a distributed system in which peers employ resources that are distributed in the network of peers in order to perform a function in a decentralised manner. Nodes in P2P networks normally hold equivalent roles, thus, also called peers. Within the advantageous qualities of the P2P networks lie the increased size of the overall database offered by a P2P network, its fault tolerance support to peer failure by other peers and the workload distribution over a network of available CPUs, since CBMIR is computationally highly intensive, the absence of the requirement of special administration or financial arrangements and their self-organisation capability and adaptability. Additionally, P2P networks offer the ability to harness the collaborative efforts of users to provide various semantic tags aiming at musical content description. Nonetheless, the very advantages of the P2P network are the same parameters that make P2P information retrieval much more complex than in the traditional server-client model. That is, the lack of a central repository for the documents to be retrieved, the large number of documents available and the dynamic character of the network, introduce an increased degree of difficulty in the retrieval process. Accordingly, as collections become larger, CBMIR in P2P networks presents new and challenging requirements the highlights of which are:

- Richer set of search semantics that can support efficient CBMIR
- Appropriate P2P models that ensure scalability
- Distribution of the workload over a network of available CPUs, as CBMIR is computationally intensive

Though, despite the previously mentioned advantages of P2P networks, the trend of musical data dissemination over P2P networks became obscure by the illegal exchange of copyrighted material. One of the key advantages of P2P networks, as previously discussed, is the lack of necessity for an administration, which acted as a loophole. Accordingly, numerous approaches (Chu et al., 2006; Praveen, Sridhar, Sridhar, & Gadh, 2005; Kalker, Epema, Hartel, Lagendijk, & Steen, 2004) for the protection and reproduction of intellectual property have been proposed, while the field is still developing (Dubosson-

Torbay, Pigneur, & Usunier, 2004; Dhamija & Wallenberg, 2003; Sastry, 2005; Schyndel, 2005), both in terms of technology as well as ethics. CBMIR applications in P2P networks can, and must, adopt any such developments.

The field of music information retrieval has received increased attention during the last decade. Numerous surveys examine the state of the art developments in the area (Byrd, & Crawford, 2002; Karydis, Nanopoulos, & Manolopoulos, 2006; Orio, 2006; Typke, Wiering, & Veltkamp, 2005) while a litany of works spawns rapidly in all directions of MIR.

Although outside the scope of this chapter, work in multimedia (other than music) information retrieval in P2P networks (Lew, Sebe, Djeraba, & Jain, 2006) shows a wealth of research issues that still remain open. Moreover, multimedia IR in P2P faces similar problems (such as indexing high-dimensionality data) appearing to MIR in P2P, though the difference of the nature of the data, such as interfaces, data representation as well as data volume issues, requires, in many cases, a completely differentiated approach. Additionally, multimedia IR research is naturally complemented by streaming in P2P networks research as many of the retrieved multimedia documents are videos. Therein lie more open research issues (Liu, Kundur, Merabti, & Yu, 2006).

In what follows this chapter, we summarise existing work on music content-based information retrieval in peer-to-peer networks. First, necessary background information on P2P networks such as classification and searching methods are provided in Section "Networks". Next, in Section "CBMIR in Wired P2P Networks" we examine the task of CBMIR in wired P2P networks, which has attracted significant attention in research related to CBMIR. A number of methods as well as systems that have been proposed are presented therein, classified according to their degree of centralisation and structure. In the section to follow, we study methods for CBMIR in wireless ad-hoc networks and present the challenges this new field introduces. In the final section, we conclude this chapter and present the perspective of P2P CBMIR.

P2P NETWORKS

This section of the chapter offers a concise presentation of the common classification of P2P systems based on their attributes, as well as some of the prominent searching methods for both wired and wireless such systems.

P2P Classification

P2P networks can be classified based on the control over data location and network topology in *unstructured*, *loosely structured* and *highly structured* (X. Li & Wu, 2004). Unstructured P2P networks follow no rule in where data is stored while the network topology is arbitrary (Gnutella). Loosely structured P2P networks have both data location and network architecture non-precisely determined (Freenet). Finally, in highly structured networks data storage and network topology are explicitly defined (Chord). What is more, P2P networks can also be classified according to the number of central directories of document locations in *centralised*, *hybrid* and *decentralised*. Centralised networks maintain a central directory in a single location (Napster), hybrid networks maintain more than one directories in super-peers (Kazaa) while for the decentralised (Chord) no central directory is kept. P2P networks can also be classified into hierarchical and non-hierarchical based on whether their overlay structure is a hierarchy or not. It is common for decentralised systems to have no hierarchy, while hybrid and most centralised systems ordinarily incorporate some degree of hierarchy. Hierarchical systems provide increased scalability, ease in exploiting peer heterogeneity and high routing efficiency. On the other hand systems with no hierarchy offer load-balance and increased resilience.

Searching Methods In Unstructured P2P Networks

In unstructured P2P systems, no rule exists that strictly defines where data is stored and which nodes are neighbours of each other. Many alternative schemes (X. Li and Wu (2004) offer a survey of searching techniques in P2P networks) have been proposed to address the problems of the original flooding search. These works include iterative deepening, k-walker random walk, modified random Breadth First Search

(BFS), two-level k-walker random walk, directed BFS, intelligent search, local indices based search, routing indices based search, attenuated bloom filter based search, adaptive probabilistic search and dominating set based search. Following are some prominent searching methods for decentralised unstructured P2P networks.

Early works such as the original Gnutella used flooding algorithms, which is the BFS of the overlay network graph with depth limit D. In the BFS, a query peer *Q* propagates the query *q* to all its neighbour peers. Each peer *P* receiving the *q* initially searches its local repository for any documents matching *q* and then passes on *q* to all its neighbours. In case a *P* has a match in its local repository then a *QueryMatch* message is created containing information about the match. The *QueryMatch* messages are then transmitted back, using reversely the path *q* travelled, to *Q*. Finally, since more than one *QueryMatch* messages may have been received by *Q*, it can select the peer with best connectivity attributes for direct transfer of the match. It is obvious that the BFS sacrifices performance and network traffic for simplicity and high-hit rates. In order to reduce network traffic, the TTL parameter is used, which is the number of peers a query should be forwarded to. In a modified version of this algorithm, the Random BFS (RBFS) (Kalogeraki, Gunopulos, & Zeinalipour-Yazti, 2002), the query peer *Q* propagates the query *q* not to all but a fraction of its neighbour peers.

In an attempt to rectify the inability of the RBFS to select a path of the network leading to large network segments, the *>RES* algorithm was developed (B. Yang & Garcial-Molina, 2002). In this approach, the query peer *Q* propagates the query *q* to a subset of its neighbour peers based on an aggregated statistic. That is, *Q* propagates the *q* to *k* neighbouring peers, all of which returned the most results during the last *m* queries, with *k* and *m* being configurable parameters. *>RES* is a significant amelioration in comparison to the RBFS algorithm, however its attitude is rather quantitative than qualitative, since it does not select the neighbours to propagate the query *q* based on the similarity of the content of *q* with the previous queries.

To overcome this quantitative behaviour of the *>RES* approach, the *ISM* approach emerged (Kalogeraki et al., 2002). In *ISM*, for each query, a peer propagates the query *q* to the peers that are more likely to reply the query based on the following two parameters; a profile mechanism and a relevance rank. The profile is built and maintained by each peer for each of its neighbouring peers. The information included in this profile consists of the *t* most recent queries with matches and their matches as well as the number of matches the neighbouring peer reported. The relevance rank (*RR*) function is computed by comparison of the query *q* to all the queries for which there is a match in each profile. Thus, for a querying peer P_Q the *RR* function is calculated by the following formula:

$$RR_Q(P_i, q) = Qsim(q_j, q)^a \times S(P_i, q_j)$$

where Qsim is the similarity function used between queries and $S(P_i, q_j)$ is the number of the results returned by P_i for query q_j. The *ISM* allows for higher ranking of the neighbouring peers that return more results by adjustment of the α parameter. Obviously, the strong point of the *ISM* approach reveals in environments that show increased degree of document locality.

Searching Methods In Structured P2P Networks

In strictly structured P2P systems, the neighbour relationship between peers and data locations is explicitly defined. Thus, the particular network architecture of such systems determines the searching process. Of the alternative strictly structured systems available, some implement the Distributed Hash Table (DHT), others have flat overlay structures while some have hierarchical overlay structures.

A DHT is a hash table whose table entries are distributed among different peers located in arbitrary locations. In such systems each node is assigned with a region in a virtual address space, while each shared document is associated with a value of this address space. Thus, locating a document requires only a key lookup of the node responsible for the key. Numerous different flat data structures can be used to implement the DHT, including ring, mesh, hypercube and other special graphs such as de Bruijn graph.

Hierarchical DHT P2P systems organise peers into different groups or clusters, where each group forms

its own overlay. The entire hierarchical overlay is formed by the entirety of all groups. Typically, the overlay hierarchies are two-tier or three-tier. Their key difference refers on the number of groups each tier has, the structure of the overlay each group forms and whether super-peers do exist. The existence of such super-peers offers increased computing resources, stability and effectiveness in routing.
In an attempt to solve the deficiencies of DHT, the non-DHT P2Ps avoid hashing. Hashing ignores data locality and cannot support range queries, a serious two disadvantages. SkipNet, SkipGraph and TerraDir are three of the most representative such systems (X. Li & Wu, 2004). SkipNet stores data close to users, SkipGraph offers support to range queries, while TerraDir is targeted for hierarchical name searches. Searching in these systems is achieved based on the specified neighbouring relationships between nodes.
In loosely structured P2Ps, the overlay structure is not strictly specified. It is either formed based on hints or formed probabilistically. Some systems develop their structure based on hints or preferences, while in others the overlay is constructed probabilistically. Searching in loosely structured P2P systems depends on the overlay structure and how the data is stored. Approaches do exist, where searching is application dependent, based on hints or done by reducing the numerical distance from the querying source to the node that stores the desired data.

Information Discovery/Provision In Wireless Mobile Ad-hoc Networks

A Mobile Ad-hoc NETwork (MANET) is a collection of wireless Mobile Hosts (MH) forming a temporary network without the aid of any centralised administration or standard support services regularly available on the wide area network to which the hosts may normally be connected. When a source node desires to send a message to some destination node and does not already have a valid route to that node, it initiates a path discovery process to locate the destination. It broadcasts a route request to its neighbours, which then forward the request to their neighbours and so on, until the destination or an intermediate node with a route to the destination is located. Nodes are identified by their IP address and maintain a broadcast ID, which is incremented after every route request they initiate. The broadcast ID together with the node’s IP address, uniquely identify a route request. In the same manner, the transmitted data requests can be identified.
Routing algorithms for MANETs are radically different from the traditional routing (e.g., Open Shortest Path First) and information search protocols (e.g., Distributed Hash Table) used in hardwired networks, due to the absence of “fixed” infrastructure (servers, access points, routers and cables) in a MANET as well as the mobility of the nodes. For wireless ad-hoc networks, there have been proposed various routing/discovery protocols, which roughly fall into the following categories (Agrawal & Zeng, 2003): a) table-driven routing protocols, b) source-initiated on-demand routing protocols and c) hybrid routing protocols. Apart from the former, which require consistent, up-to-date routing information from each node to every other node in the network and thus are practically unfeasible for large-scale and dynamic MANETs, the remaining two families of information (node) discovery protocols rely on some form of *broadcasting*; broadcasting is best suited in cases where information packets are transmitted to multiple hosts in the network. *Flooding* is the simplest broadcasting approach, where every node in the network forwards the packet exactly once; flooding ensures full coverage of the MANET provided that there are no network partitions. Flooding, though, generates too many redundant transmissions, causing the *broadcast storm problem* (Ni, Tseng, Chen, & Sheu, 1999).
Numerous algorithms have been proposed to address this problem (Lou & Wu, 2004). They can be classified as follows: a) *probabilistic approaches* (counter-based, distance-based, location-based) and b) *deterministic approaches* (global, quasi-global, quasi-local, local). The former methods do not guarantee full coverage of the network, whereas the latter do provide coverage guarantee, and thus are preferable.
The *deterministic approaches* provide full coverage of the network for a broadcast operation, by selecting only a subset of nodes to forward the broadcast packet (*forward nodes*), and the remaining nodes are adjacent to the nodes that forward the packet. The selection of nodes is done by exploiting the “state” information, i.e., network topology and broadcast state (e.g., next selected node to forward the packet, recently visited nodes and their neighbour sets). All the categories of the deterministic algorithms, apart from the *local algorithms*, require (full or partial) global state information, thus being impractical. The local or *neighbour-designating* algorithms maintain some local state information, i.e., 1-hop neighbourhood information by periodic exchange of ‘HELLO’ messages, which is feasible and not

costly. In the neighbour-designating methods, the forwarding status of each node is determined by its neighbours. As a matter of fact, the source node selects a subset of its 1-hop neighbours as forward nodes to cover its 2-hop neighbours. This forward node list is piggybacked in the broadcast packet. Each forward node in turn designates its own forward node list.

Remotely related to the topic of this chapter is the issue of multicasting streaming media (audio/video) to MANETs (Dutta, Chennikara, Chen, Altintas, & Schulzrinne, 2003) or unicasting audio to 3G UMTS devices (Roccetti, Salomoni, Ghini, & Ferretti, 2005). These works though assume the existence of a central server (supplier), which provisions the mobile clients with multimedia data.

CBMIR IN WIRED P2P NETWORKS

The field of combined CBMIR and P2P networks is definitely very young as the inaugural research paper dates back in 2002 (Wang, Li, & Shi, 2002).

In this first attempt, the authors of (Wang et al., 2002) present four P2P models for CBMIR (Sections "Generalised", "PsPs Model" & "Hybrid"). The four models include all centralised, decentralised and hybrid categories. Another research based on a hybrid configuration is presented in (Tzanetakis, Gao, & Steenkiste, 2004) (Section "Decentralised Structured"). Therein the authors propose a scalable and load balanced P2P system with an underlying DHT-based system. C. Yang (2003) proposed the utilisation of the feature selection and extraction process that is described in (C. Yang, 2002) for CBMIR in a decentralised unstructured P2P system (Section "MACSIS-P2P"). Finally, (Karydis, Nanopoulos, & Manolopoulos, 2005) (Section "Sampling Framework") focus on similarity searching for similar acoustic data over unstructured decentralised P2P networks, proposing a framework, which takes advantage of unstructured P2P networks and minimises the required traffic for all operations with the use of a sampling scheme.

The following sections contain an overall description of the systems that have been proposed for the purposes of CBMIR in P2P networks, classified by their degree of centralisation and structure.

Centralised

Wang et al. (2002) introduced CBMIR in peer-to-peer environments by developing four systems based on different P2P models. Two of these models, PsCM & PsC+M, have a centralised architecture.

PsC Model. In the Peers-Coordinator Model (PsCM) peers contain a music set as well as sharing a common feature extraction method. A coordinator maintains 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. Queries are content requests proposed by a user. In this model, any query PC in the system connects with the coordinator, while peers connect with each other via the coordinator, though they directly transfer music data from one to another. During the query process the feature of the music request is extracted from the music request by an extraction method and sent to the coordinator. The coordinator receives the feature, compares it with all music features uploaded during peers registration and sends the result to the request PC. Results contain the locations of the peers which store the music similar to the requested.

PsC+ Model. A slightly differentiated implementation of a centralised system, PsC+M, assigns the feature matching process on each peer. In the Peers-Coordinator+ Model (PsC+M) each peer has additionally to the PsC model a commonly shared feature matching method. The coordinator consists of a data structure which stores the network identifiers of all peers. The main steps of a query process are the following: Each peer shares some music stored on the local hard disk and registers its location at the coordinator. A query by any peer sends the coordinator the extracted feature of the music request. The coordinator forwards the feature of the query to all registered peers, each peer compares it with the features of the local shared music and sends the local result to the coordinator. The coordinator in turn passes on the results to the querier.

In both the previously mentioned models, the role of the coordinator can significantly ease the process of retrieval due to its supervising character in terms of route to peers discovery and knowledge of the available content on other peers (PsCM). Though, centralised P2P networks are subject to the same drawbacks for which the traditional server-client model was originally not used (network failures due to central peer failure, impaired scalability, joining/leaving of peers not easily handled and possible undesirable dominion of coordinator controllers). In other words the coordinator is easily overloaded and becomes the bottleneck of the whole system, leading to poor stability. As for these two models (PsCM & PsC+M) Wang et al. (2002) conclude that the load for a given query produced by PsCM is less than that of PsC+M. Though comparing time requirements for the completion of the same query PsC+M is faster than PsCM.

Decentralised Structured

In the system proposed by Tzanetakis et al. (2004) each node in the network stores music files for sharing as well as information about the location of other music files in the network. Shared files in the system are described by a Music File Description (MFD) which is essentially a set of AV pairs (artist = "Jennifer Warnes", album = "Dirty Dancing", song = "The Time Of My Life", ..., specCentroid = 0.65, mfcc2 = 0.85, ...). A Music Feature Extraction Engine (MFEE) calculates the features from the audio files, while others are manually appointed. Two operations are supported by the system, the registration of a music file based on its associated MFD and the search where the user query is converted into an appropriate MFD that is then used to locate nodes containing files matching the search criteria. The MFD of either registration or search query is passed to the content discovery system (CDS), which runs on top of a DHT-based P2P system (see Section "Searching Methods in Structured P2P Networks").
The MFEE component takes as input an audio file and produces a feature vector of AV pairs that characterises the particular musical content of the file as proposed by Tzanetakis and Cook (2002). Features based on the Short Time Fourier Transform as well as Mel-Frequency Cepstral Coefficients are used to represent sound texture, and features based on Beat and Pitch Histograms are used to represent rhythm and pitch content.
Scalable Content Discovery. Unlike centralised systems in which files are registered at a single place, or broadcast-based (decentralised unstructured) systems where a query may potentially be sent to all peers in the system, CDS uses a scalable approach based on Rendez-vous Points (RPs) for registration and query resolution. Essentially, the P2P network is structured to efficiently represent the space of AV pairs for the tasks of searching and retrieving music.
MFD Registration. To register an MFD, the CDS applies a uniform hash function such as SHA-1 to each AV pair in the MFD to obtain n node IDs. The MFD is then sent to each of these n peers, while this set of peers is known as the Rendez-vous Point (RP) set for this MFD. Upon receiving an MFD, the peer inserts it into its database. Hence, each peer is responsible for the AV pairs that are mapped onto it. Obviously, the load on each node depends on the distribution of specific AV pairs, addressed by a load-balancing scheme.
MFD Searching. Searching has been divided into two categories: exact search and similarity search. In an exact search, the user is looking for MFDs that match simultaneously all the AV pairs of the query, while extra AV pairs that may be in the MFDs but not in the query are ignored. For a query Q with m AV pairs, the MFDs that match Q are registered at RP peers N_1 through N_m, while the CDS chooses the peer that has the smallest MFD database, to fully resolve the query. Once a query is received, the peer conducts a pair-wise comparison between the query and all the entries in its database to find the matching MFDs.
In a similarity search, the user is trying to find music files that have a similar feature vector to what is specified in the query. Thus, a pair of feature-value selected from the extracted feature vector of the query is sent to a peer designated by the hash function. The receiving peer computes the "distance" between the query vector and each MFD in its database, using a distance function (Manhattan). The distances are then ranked, and the k top MFDs that have the smallest distance are returned to the user. For the discovery of MFDs that slightly differ from the query, in a selected pair of feature-value, but are similar or identical regarding other features, the query is also sent either by the peer that received the query or querier to peers corresponding to values near the originally selected pair of feature-value.
Load Balancing. The use of RPs, ensures the absence of network-wide message flooding at both registration and query. Though, some AV pairs may be much more common or popular in MFDs than others, causing a few peers to be overloaded by registrations or queries, while the majority of peers in the

system stay underutilised. The system's throughput is improved by deployment of a distributed dynamic load-balancing mechanism where multiple peers are used as RP points to share the heavy load incurred by popular AV pairs.
The work of Tzanetakis et al. (2004) addresses a number of important issues concerning both MIR as well as P2P networking factors. On one hand, the use of both metadata and content-based features confronts a number of real life scenarios where a user may query based on a sample-file or humming as well as by the singer's name or song title. On the other hand, the disadvantages of the use of metadata, such as the fact that there are not always available, the inconsistencies introduced by their manual appointment, the fact that their sole use in MIR requires knowledge for the query not provided by listening and the fact that their description lacks customisation since it relies on predefined descriptors are compromised by the co-existence of the feature extraction process. On the networking side of their proposal, the use of an decentralised system avoids scalability issues as well as potential coordinator problems. Additionally, given the assumption that the proposed system is structured, the use of an underlying DHT-like scheme alleviates problems concerning networking searching in order to identify the host that may have a match to a query. Though, such an advantage comes at the cost of requiring the peers to host foreign data. Finally, the use of the load-balancing scheme ensures that no peer is over exploited due to the use of the underlying DHT-based system, at the cost of redundant data replication and increased processing requirements for the distributed dynamic load-balancing mechanism to work efficiently.

Decentralised Unstructured

PsPs Model

Wang et al. (2002) presented a distributed CBMIR P2P model, where the system is formed by a numerous of peers without a coordinator, while the width of the system is *m* and the depth is *n*. Peers consist of a music set, a same feature extraction method, a common feature matching method and a data structure that stores the network identifiers of neighbouring peers. The main steps of a query process in this system, PsPsM, are the following: Each peer shares some music and gets the network identifiers of its neighbours by broadcast or other algorithms. Upon a query by any peer, the feature of the query is extracted and sent to *m* peers, randomly selected from the neighbouring set. After receiving the feature of a query, each peer sends it to *m* random neighbour peers, as long as a maximum allowed hop number is less than *n*, examines the local content and returns the results to the querying peer.
This approach is identical to the Random Breadth First Search (RBFS) as also presented by Kalogeraki et al. (2002), aside from the domain of MIR, where the querying peer *Q* propagates the query *q* not to all but at a fraction of its neighbour peers.
In comparison to the centralized P2P CBMIR systems, the stability of their distributed version, such as the PsPsM, is far better since no coordinator exists that can easily overload and become the bottleneck of the whole system. However the number of messages that are sent by PCs of distributed P2P CBMIR systems increases when the scale of the system expands. Accordingly, the load produced for a given query by PsPsM is smaller than that of either PsCM or PsC+M, but by no means smaller than that of the hybrid approach PsPsCM, as it will be seen in section "Hybrid" .

MACSIS-P2P

C. Yang (2003) proposed a peer-to-peer model for music retrieval, where nodes are interconnected as an undirected graph and each node stores a collection of music files. Following the MACSIS system, analysis of raw audio and conversion to characteristic sequences are done locally at each node, for both the database and queries. While building the database, characteristic sequences for each music file are stored in multiple LSH hashing instances (each with its own set of randomised parameters). The same music file may appear in many different nodes and indexed under different sets of hashing instances. At any time, every node maintains a variable "CPU Availability" which indicates how much CPU power it can offer to run peers' search tasks. When the node is busy with other tasks or when the owner wants to start using the node, CPU-Availability is set to zero.

Feature extraction. The music indexing framework (MACSIS) (C. Yang, 2002) consists of three phases: (1) Raw audio goes through a series of transformations, including Fourier Transformation, leading to a stream of characteristic sequences, which are high-dimensional vectors representing a short segment of music data. (2) Characteristic sequences for the audio database are then indexed in a high-dimensional indexing scheme known as Locality-Sensitive Hashing, or LSH (Indyk & Motwani, 1998). (3) During retrieval, Phase 2 finds a list of matches on characteristic sequences, representing short segments of music, which require joining together to determine which song is the best "global" match.

Searching phase. A music query issued by one node (querier) in the network, is processed by a set of other nodes, which may include the querier itself. Due to the symmetric nature of P2P networks, any node can become a querier and a node may be simultaneously a querier and response node for different queries. A query is sent as a stream of characteristic sequences, obtained by analysing user-input audio at the querier. Each query has two required parameters: expiration-time and search depth, indicating the time at which the query expires, and the maximum number of links through which the query can be passed in the P2P network graph. The query is sent from the querier to its neighbours, which may in turn pass it on to other neighbours of their own (while decrementing search-depth by 1) as long as search depth is greater than zero. The processing of all query vectors is not obligatory, since one way to speed up retrieval is to process only a fraction of such query vectors.

A common method between peers for hashing binary files to a signature value is assumed, so that identical files must have identical signatures and the chance of different files having identical signatures to be very low.

The work considers both a replicated database and a general P2P scenario, while special attention is given on the control of the workload produced at queried peers during query time. Each query is divided into two phases, the first of which includes only a subpart of the actual query vectors, in order to distinguish high probability response peers. Accordingly, a peer ranking occurs and the full query vectors are sent to all peers. Given that a peer has free CPU resources, it decides whether to process a query or not based on the ranking that the specific query received, among other factors.

The work proposed by C. Yang (2003), is based on music indexing framework (MACSIS), which reports (C. Yang, 2002) high accuracy levels in most of the five levels of musical similarity (i. identical digital copies; ii. same analog source, different digital copies, possibly with noise; iii. same instrumental performance, different vocal components; iv. same score different performances, possibly at different tempo; v. same underlying melody, different otherwise, with possible transposition) used therein. As the MACSIS framework is highly computationally intensive, its proposed utilisation in a P2P environment is safeguarded by a the "CPU-availability" variable, allowing peers to select their engagement to the retrieval process. Moreover, the use of different sampling levels of the query vectors as well as a two staged search phase, in the later of which higher probability response candidates may perform more demanding tasks, allows a further level of security in terms peer CPU-availability as well as network utilisation.

Sampling framework

The work of Karydis, Nanopoulos, and Manolopoulos (2005) examines the problem of searching for similar acoustic data over unstructured decentralised P2P networks using Dynamic Time Warping (DTW) as a distance measure.

DTW can express similarity between two time series even if they are out of phase in the time axis, or they do not have the same length. The DTW distance $D_{DTW}(S,T)$ between time series *S* and *T* is essentially a way to map *S* to *T* and vice-versa. The most important disadvantage of the DTW method is that it does not satisfy the triangular inequality, which is a desirable property for constructing efficient indexing schemes and pruning the search space. Moreover, the calculation of $D_{DTW}(S,T)$ is significantly more CPU intensive than the calculation of $D_{\mathrm{Euclidean}}(S,T)$. Therefore, a performance improvement is the definition of a lower bound, that would take advantage of indexing schemes and avoid the computation of DTW when there is a guarantee that the two time series are not similar. One such lower bound, as proposed in (Keogh & Ratanamahatana, 2005), is termed *LB_Keogh* and defined as follows:

$$\text{LB\_Keogh}\quad (\text{S, T}) = \begin{cases} \sqrt{\sum_{i=1}^{n} (T[i]-U[i])^2} & T[i] \geq U[i] \\ \sqrt{\sum_{i=1}^{n} (T[i]-L[i])^2} & T[i] < L[i] \\ 0 & otherwise \end{cases} \qquad (1)$$

where U and L is the upper and lower bound respectively for the time series *S*. Essentially, for each *i*, the upper bound guarantees that $U[i] \geq S[i]$ and the lower bound guarantees that $L[i] \leq S[i]$. In (Keogh & Ratanamahatana, 2005) it has been proven that $\text{LB\_Keogh} \leq \text{D}_{\text{DTW}}(\text{S,T})$, and therefore the distance measure *LB_Keogh(S,T)* can be effectively used for pruning, resulting in considerably less number of D_{DTW} computations.

Due to the selected similarity model, the information that is propagated between nodes comprises the *U* and *L* sequences of the query sequence. A node receiving these sequences computes the LB value between its documents and envelope. When a LB value is smaller than the user-specified similarity threshold, then the query sequence is propagated to this node and the actual DTW distance is computed between the query and corresponding document.

As acoustic data tend to be very large, the number of elements in a phrase of even few seconds can be several hundred thousands. The length of the *U* and *L* sequences is equal to the length of the query sequence, meaning that a straightforward approach, which directly propagates *U* and *L* sequences between nodes, will result into an extremely large traffic over the P2P network. Moreover, the computation of LB in each node can become rather costly, violating the need of a P2P network to burden the participating nodes as little as possible.

A two-fold optimisation scheme is used for the reduction of the traffic over the P2P network when querying music documents by content. The scheme works as follows:

- It reduces the length of the envelope's sequences by sampling them. However, plain sampling can be ineffective, since it leads to underestimation of LB, thus a sampling method to reduce the length of the sequences without significantly affecting the computation of LB or introducing false-negatives is provided.
- The scheme uses (whenever possible) a compact representation of the sampled sequences of the envelope. The representation comprises a form of compression for the sequences, but does not burden the nodes with the cost of decompression.

Sampling and representation methods. Let the considered phrase length be equal to *N*. The length of each query *Q*, and therefore of its upper (*U*) and lower (*L*) sequences, will also be equal to *N*. The goal is to sample *U* and *L*, so as to obtain two sequences U' and L', each of length $M << N$. Initially, it is assumed that uniform sampling is performed. In this case, each time the $(i \times N / M)$-th element of *U* and *L* is selected, where $1 \leq i \leq M$. When the LB Keogh between the query sequence *Q* and a data sequence is computed, each phrase *C* of length *N* in *Q* is considered. Thus, each phrase has to be sampled in the same way as *U* and *L*. This leads to a sampled phrase C'. Therefore, the lower-bound measure LB', is given as:

$$\text{LB}' = \sqrt{\sum_{i=1}^{M} \begin{cases} (C_i' - U_i')^2 & if \quad C_i' > U_i' \\ (C_i' - L_i')^2 & if \quad C_i' < L_i' \\ 0 & otherwise \end{cases}} \qquad (2)$$

In the aforementioned equation, the third case (i.e., when $L_i' \leq C_i \leq U_i'$) does not contribute in the computation of LB'. The problem of uniform sampling is that, as it selects elements without following any particular criterion, it tends to select many elements from *U* and *L* that result to this third case.

Therefore, LB' may become a significantly bad underestimation of LB that would have been computed if sampling was not used. The underestimation of the lower-bound value will result to an increase in false-alarms, which will in turn incur high traffic.

To overcome this problem, an alternative sampling method is therein proposed. U and L are sampled separately. Initially, the elements of U are stored in ascending order. In U' the first M elements of this ordering are selected. Respectively, L is sorted in descending order and the first M elements in L' are selected. The intuition is that the selection of the smallest M values of U, helps in increasing the number of occurrences of the first case (i.e., when $C_i > U_i'$), since the smallest the value of U_i' is, the more expected is to have a C_i' larger than it. Accordingly, it is easy to prove (Karydis, Nanopoulos, & Manolopoulos, 2005) that the sampling of U and L does not produce any false negatives.

The separate sampling of U and L presents the requirement of having to store the positions from which elements are being selected in U' and L'. If the positions are stored explicitly, then this doubles the amount of information kept (2M numbers for storing U' and L' and additionally 2M numbers for storing the positions of selected elements). Since this information is propagated during querying, traffic is increased. For this reason an alternative representation is proposed. To represent U', a bitmap of length N (the phrase length) is used. Each bit corresponds to an element in U. If an element is selected in the sample U', then its bit is set to 1, otherwise it is set to 0. Therefore, the combination of the bitmap and the M values that are selected in U' are used to represent U'. The same applies for L'. This representation proves to be efficient: the space required for U' is $M + \lceil N/8 \rceil$ bytes[ii].

The plain representation requires 5M bytes (since it requires only one integer, i.e., 4 bytes, to store the position of each selected element). Thus, the proposed method is advantageous when N<32M, i.e., for sample larger than about 3% (experimentation showed that samples with size 10% are the best choice).

In the following, existing algorithms for searching in P2P networks that can be used in a modified version within the framework are presented.

The BFSS algorithm. The simplest similarity searching algorithm is on the basis of breadth-first-search over the nodes of the P2P network. The adapted algorithm, which uses the proposed sampling and representation methods, is denoted as BFSS (breadth-first-search with sampling). Each time, the current node n is considered. A TTL (time-to-leave or Maxhop) value denotes how many valid hops are remaining for n, whereas T_s is the user-defined similarity threshold. It is assumed that sequences U' and L' carry also the associated bitmaps.

Evidently, the movement of the actual query sequence from the querier to the currently visited node, increases the traffic (not being sampled, the query sequence has rather large length). For this reason, it is important not to have a large number of false-alarms.

The algorithm that does not use sampling (denoted as BFS) may produce less false-alarms. However, between each pair of peers it has to propagate U and L sequences, with length equal to the one of the query sequence. Therefore, it is clear that there is a trade-off between the number of additional false-alarms produced due to sampling and the gains in traffic from propagating sampled (i.e., smaller) envelopes.

The >RESS algorithm. The >RES algorithm tries to reduce the number of paths that are pursued during searching. Instead of selecting, at random, a subset of the peers of the currently visited node, it maintains a profile for each such peer and bases its decision on this profile. In particular, each node maintains for each of its peers the number of positive answers that it has replied. Then, it selects the k peers that provided the most answers during the previous m queries. Both k and m are user specified.

It is clear that >RES algorithm can be easily adapted in the framework. The query sequence is sampled and represented according to the method previously mentioned. This does not affect the profile that is maintained by >RES. The resulting method is denoted as >RESS (>RES with sampling).

Since only a subset of peers is actually visited, >RESS tries to reduce traffic without missing a large number of answers. However, compared to BFSS, >RESS is proved experimentally to produce fewer answers.

The ISMS algorithm. The ISM algorithm shares the same objective with >RES, i.e., it tries to reduce the number of examined paths. However, the profile maintained for each peer is different. ISM does not base its decision only on the number of answers to previous queries, but also examines the similarity between the previously answered queries and the current one. Therefore, for each peer, a node maintains the t most recent queries that were answered by the peer. When a new query *q* arrives in the node, then it computes the similarity Qsim between *q* and all queries that are maintained in the profile of each node. A relative-ranking measure is given to each peer P_i, using the following formula:

$$RR_Q(P_i, q) = Qsim(q_j, q)^a \times S(P_i, q_j)$$

where $S(P_i, q_j)$ is the number of the results returned by P_i for query q_j. Thus, ISM ranks higher the neighbouring peers that return more results by adjustment of the α parameter. The comparison is made more clear when α is set equal to 1, therefore focusing only on the criterion of similarity. We also have to notice that ISM may become biased towards the nodes that have answered somewhat similar queries in the past and may not give the chance to new nodes to be explored. For this reason, the following heuristic is used in (Kalogeraki et al., 2002): besides the peers selected with the aforementioned criterion, ISM also selects at random an additional very small subset of peers (e.g., one node). In total, *k* peers are selected, where *k* is user-defined. The length of each profile (the number of queries stored in it) is also user-defined.

In order to adapt ISM to the sampling framework, one has to consider how to maintain the previously answered queries. In this framework, query sequences are represented by their samples. Therefore, the similarity between the current query's sample and the samples of previously answered queries is measured. For this reason the samples of the answered queries are maintained in the profiles of the peers. To save time during the computation of the ranking, instead of measuring the actual similarity (through the DTW measure), the *LB_Keogh* value is computed. The resulting algorithm is denoted as ISMS (ISM with sampling).

ISMS is expected to have slightly larger traffic than >RES, since it propagates the sample of an answered query to all nodes involved in the search (in order to update their profiles). However, by testing the content of the queries, it tries to reduce the number of missed answers.

The work proposed by Karydis et al. (2005) focuses mainly on developing a framework for similarity searching algorithms using a sampling method, while musical pieces are treated as sequences and their similarity based on DTW. Accordingly, three algorithms (BFS, >RES and ISM) used for searching similar text documents in P2P networks are examined and adapted for CBMIR. The use of DTW proves too expensive and thus a lower bound is used therein. Though, the cheap lower bound calculation advantage is alleviated whenever a real match is available, by requiring the full DTW measurement. Nevertheless, DTW's characteristic being able to withstand distortion of the comparing series in the time axis has had great appeal in the MIR research community.

Hybrid

The fourth model proposed by Wang et al. (2002), PsPsC, is a hybrid approach that combines characteristics from the centralised models PsC & PsC+ (Section "Centralised") and the decentralised unstructured PsPs (Section "PsPs Model"), as previously described.

PsPsC Model. In the PsPsC model there is one coordinator peers registers at, which collects and manages their statistical data. The architecture of the system is similar to the PsPs model (see Section "PsPs Model") with the addition of a peer characterisation feature and a common method to obtain such features. The coordinator consists of a data structure to store the network identifiers of all peers, a data structure to store the PC features of all peers and an accelerating structure which can utilise the peer features to locate proper peers for faster CBMIR. The query process steps are the following: Each peer shares musical content and registers at the coordinator information including its network identifier and PC feature. Upon a query the peer feature of the querier, i.e. the peer 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. Based on the PC feature of the music request and the accelerating algorithm, the coordinator selects a high probability subset of the peers which may have similar music to the query and the querier directly addresses this subset. In this case, these peers act as some short of super-peers.

In this approach, the balance of stability and cost can be obtained by selecting the proper number of high probability peers. Moreover, Wang et al. (2002) report that PsPsCM has the best load and time attitude in comparison to the family of presented models, thus supporting the further development of the accelerating algorithm solely on this model.

CBMIR IN WIRELESS P2P NETWORKS

As already discussed in Section "Introduction" of this chapter, the sovereign of the traditional music distribution has undergone a significant alteration under the auspices of new technologies like MP3 and the penetration of the World Wide Web. Brand new opportunities for music delivery are additionally introduced by the widespread penetration of the wireless networks (wireless LANs, GPRS, UMTS as described by DeVriendt, C., Lerouge, and Xu (2002)) such as the pioneering applications (Roccetti et al., 2005) supporting the distribution of MP3-based songs to 3G UMTS devices. These applications rely on the existence of a central server, which receives requests from and delivers audio files to the mobile clients. Sadly enough, CBMIR research in this type of infrastructures has not yet received attention and is currently considering only metadata MIR. Though, aside from these single-hop infrastructure wireless networks, music delivery can also unfold over the emerging wireless ad-hoc networks. The wireless ad-hoc networks are peer-to-peer, multi-hop, mobile wireless networks, where information packets are transmitted in a store-and-forward fashion from source to destination, via intermediate nodes. The salient characteristics of these networks, i.e., dynamic topology, bandwidth-constrained communication links and energy-constraint operation, introduce significant design challenges.
This section of the chapter focuses on CBMIR in wireless ad-hoc networks. Consider a number of mobile hosts that participate in a wireless ad-hoc network, where each host may store several audio musical pieces. Assume a user that wants to search in the wireless network, to find audio pieces that are similar to a given one. For instance, the user can provide an audio snippet (e.g., a song excerpt) and query the network to find the peers that store similar pieces. As will be described in the following, the definition of similarity can be based on several features that have been developed for Content-Based Music Information Retrieval. It is important to notice that the querying host is assumed to have no prior knowledge of both the qualifying music pieces and the hosts' locations that contain them. This is the key differentiation from existing researches that are interested in just identifying the hosts, in a wireless ad-hoc network, that contain a known datum. Moreover, the issues discussed in this section are complementary to the problem of delivering streaming media, such as audio and video, as considered by Baochun and Wang (2003) in wireless ad-hoc networks, since the latter does not involve any searching for similar audio pieces, and just focuses on transferring data from one host to another.

Requirements Set By The Wireless Medium

This section focuses on methods for searching audio music by content in wireless ad-hoc networks, where the querier receives music excerpts matching to a posed query. The actual searching procedure can benefit from the latest approaches for CBMIR in wired P2P networks (see Section "CBMIR in Wired Networks"). Nevertheless, the combination of the characteristics of the wireless medium and the audio-music data pose challenging requirements:

1. CBMIR methods for wired P2P networks do not consider the continuous alteration of the network topology, which is inherent in wireless ad-hoc networks, since Mobile Hosts (MHs, the terms MH and peer are similar in this context and thus interchangeable) are moving and become in and out of range of the others continuously. One impact of this mobility is that selective propagation of the query among MHs, e.g., by using data indexing like DHT as proposed by Tzanetakis et al. (2004) or caching past queries (Kalogeraki et al. (2002) for text documents and Karydis, Nanopoulos, Papadopoulos, and Manolopoulos (2005b) for music), is not feasible. Additionally, the recall of the searching procedure is affected by the possibility of unsuccessful routing of the query, as well as the answers, over the changing network topology.
2. The need to reduce traffic, which results from the size of audio-music data (approx. 8 MBytes for a 3 minute query). This is can be achieved by replacing the original query with a representation that utilises appropriate transcoding schemes. Although traffic concerns CBMIR in wired P2P networks too, the requirement of traffic reduction is much more compelling in wireless ad-hoc networks, where the communication bandwidth ability is usually limited to approximately 1 MBps. It is worth noticing, that the reduction of traffic also reduces the involvement of other MHs, due to constraints in their

processing power and autonomy.

3 In CBMIR over wired P2P networks, should a matching music excerpt be found, it can immediately be returned to the querying node, since the querier is directly accessible (through its IP address). In contrast, in wireless ad-hoc networks the answers to a query have to be propagated back to the querier via the network (the querier is not directly accessible), burdening further traffic.

The aforementioned issues can be addressed, to a certain extent, by algorithms proposed for the problem of routing in wireless ad-hoc networks, though, these approaches consider neither the peculiarities of searching for CBMIR purposes nor the size of the transferred data, since music data are considerably larger than routing packets.

Research related to the application of CBMIR in wireless ad-hoc P2P networks is so young, that to our best knowledge the work by Karydis, Nanopoulos, Papadopoulos, Katsaros, and Manolopoulos (2006) is the only one to examine the issue of CBMIR in ad-hoc wireless networks.

Accordingly, to address the requirements posed by the wireless medium, Karydis et al. (2006) proposed the following techniques:

1 To fulfil the first requirement, breadth-first searching is performed over the wireless ad-hoc network using knowledge about neighbouring MHs (obtained by probing neighbourhood at specific time points). This approach can cope with mobility, maintain increased final recall and constraint the drawbacks of flooding, e.g., excessive traffic due to multiple broadcasts (as already explained in Section "Information Discovery/Provision In Wireless Mobile Ad-hoc Networks").

2 The second requirement is addressed by a technique that uses a concise, feature-based representation of the query with reducing length. The reducing-length representation (a.k.a transcoding) drastically degrades traffic, while reducing the computation performed at each MH as well.

3 The additional traffic produced by the third requirement is addressed by a twofold technique: (i) Policies to constraint the number of MHs involved for the propagation of the answers, by exploiting any MHs that were involved during the propagation of the query. (ii) By allowing such MHs to prune the propagation of answers, based on a property of the previously described representation.

Outline Of The Searching Procedure

The problem of finding similar music sequences in a MANET requires a searching procedure, which will detect MHs in the MANET that have similar sequences, find those sequences in the MHs and return them back to the querier. The already described requirements of the wireless framework formulate the examined searching procedure in the following way:

1. There is no prior knowledge of the data MHs store; that is, the querier has no knowledge of the location of the required data.
2. MHs that have qualifying sequences have to be reached in a way that addresses their mobility and minimises traffic. Due to their relative positions and the preferred tolerance to traffic (see below), all such nodes may not be possible to reach.
3. At each reached MH, the qualifying sequences have to be detected by detaining the MHs, in terms of CPU cost, as little as possible.
4. Each qualifying sequence has to reach the querier in a way that reduces traffic. Notice that the answers may have to be routed back to the querier following paths different from those through which the MHs with qualifying sequences were reached, since intermediate MHs may have changed their position and therefore be out of range. Due to this, every detected answer may not be possible to reach the querier.

The searching procedure is initiated at the querying MH, aiming at detecting sequences in other MHs, which contain excerpts whose similarity from the query sequence Q is within user-defined boundaries, a threshold ε. The definition of the distance measure is detailed in Section "Features and Indexing". Just for now, one can intuitively think of the distance as a measure of how dissimilar two music sequences are. The length of detected excerpts is equal to the length of the query sequence Q.

To address traffic minimisation, Q has to be transformed to a representation form, denoted as R, through which qualifying sequences are detected.

1. User poses a query Q
2. Q is transformed to a representation form R
3. R is broadcast to all peers in range
4. Qualifying sequences (true- and false-positives) detected at each peer comprise an answer set
5. Each answer set is broadcast back to the querier
6. Resolution of false-positives (possible places are: at answer providing peers, the querier or intermediate peers)
7. Return of actual matches to user/application

(a) searching process

Event name	Involved steps
query initialization	1,2,3
reception of R	4,5
reception of an answer set	5,6
answer set reaching the querier	7

(b) events

Figure 1. Searching process and basic events.

Due to this transformation, it is possible that false-positive results may appear. A false positive result is a result that appears to be a true result when comparing with the transformed representation, though, under the non-transformed query is not a real result. Moreover, *R* must present no false-negatives (real results that were missed due to the transformation). However, its particular implementation determines whether false-positives may be produced or if they will be completely avoided. Based on all the aforementioned issues, an abstract scheme to describe the entire searching procedure consists of the steps depicted in Figure 1(a). These steps are summarised in four events according to Figure 1(b).

To avoid duplicate effort, the procedure tags *R* with an ID (see Section "Information Discovery/Provision In Wireless Mobile Ad-hoc Networks"). This way, MHs that have already received it will perform no further action. Additionally, the propagation of *R* to the neighbouring MHs is controlled by a parameter called *h*, which acts as a counter that is decreased at each receiving MH (denotes the available number of hops). Its initial value, at the querier, is equal to *MaxHop*. This value corresponds to the preferred tolerance to traffic and network reach/coverage. The propagation of answer sets (resulting from step 5) is handled similarly.

As already mentioned, the searching process consists of a forward and a backward phase. During the former, *R* is propagated and during the latter answers are routed back to the querier. The two phases are interleaved, since during the propagation of *R* by some MHs, other MHs are returning answers to the querier. The backward phase's volume mainly depends on the existence of answers and the number of false-positives, while the forward phase depends on the size of *R*, the user defined coverage willingness as well as the network reachability. In general, the volume of information transferred during the backward phase is larger than that of the forward phase.

Having outlined the searching procedure, the following sections detail its parts, starting with the features selected for the formation of *R*. Next, a method for the acceleration of similarity searching within each MH, using indexing, is presented. Based on these, follow two searching algorithms, which rely on different choices with respect to the formation of *R*, while, finally, methods to improve the backward phase are described.

Features And Indexing

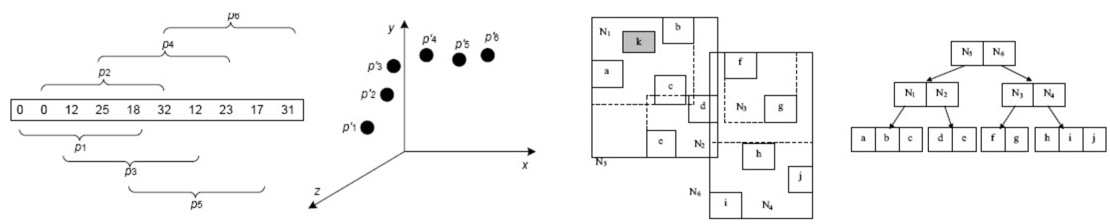
The most typically encountered features for the acoustic representation are produced by time analysis (Papaodysseus, Roussopoulos, Fragoulis, Panagopoulos, & Alexiou, 2001; Paraskevas & Mourjopoulos, 1996), spectral analysis (Papaodysseus et al., 2001; Paraskevas & Mourjopoulos, 1996; Kostek & Wieczorkowska, 1997) and wavelet analysis (Wieczorkowska, 2001).

Karydis et al. (2006) do not concentrate on devising new features, while their interest remains in the searching procedure and their methodology is able to embrace any high performance feature extraction procedure. Accordingly, a feature extraction process based on the wavelet transform is utilised. Wavelet transforms provide a simple but yet efficient representation of audio by taking into consideration both non-uniform frequency resolution and impulsive characteristics, as shown by (Roads, Pope, Piccialli, & Poli, 1997; T. Li, Li, Zhu, & Ogihara, 2002; T. Li, Ogihara, & Li, 2003).

More particularly, they consider the Haar wavelet transformation for its simple incremental computation, its capability concerning the capture of time dependant properties of data and overall multiresolution representation of signals (K.-P. Chan, Fu, & Yu, 2003) as well as for the incorporation of the previously mentioned properties. However, the approach can easily be extended to other types of wavelet transforms. Moving on to the indexing procedure within peers to facilitate the searching, they propose the following approach. In a peer, each original audio sequence is transformed to a number of multidimensional points. A sliding window of length *n* is used over the sequence and apply Discrete Wavelet Transform (DWT) to the contents of each window, producing *n* coefficients per window. An example is depicted in Figure 2a. Therefore, each audio sequence produces a set of *n*-dimensional points in the feature space. Since *n* depends on the query length and, thus, takes relatively large values (e.g., 64 K), in order to efficiently index them in the feature space, only the first *d* dimensions from each point (experiments with d = 64 are presented therein) are selected. This procedure dramatically reduces both the size of the index and the number of dimensions without affecting much the quality of the index. The reason for the latter is the merit of DWT to concentrate the energy of the sequence in the first few coefficients. However, false-positives remain a possibility and thus require resolution.

Most importantly, it has been proven by K. Chan and Fu (1999) that no false dismissals are introduced when using only the *d* first coefficients (due to Parseval's theorem). Notice that this property is proven in (K. Chan & Fu, 1999) for the Euclidean distance. Although this distance measure is simple, it is known to have several advantages, as it has been illustrated by Keogh and Kasetty (2002). Nevertheless, the methodology proposed by Karydis et al. (2006) does not decisively depend on the particular features and distance measure.

To speed-up the retrieval, for each sequence the collection of the resulting *d*-dimensional points is organised in Minimum Bounding Rectangles (MBRs), which are, then, stored in an R*-tree (Beckmann, Kriegel, & Seeger, 1990). Answering to a query, the root is initially retrieved and its entries that intersect the query are only further examined recursively until reaching a leaf. All non intersecting nodes are not included in the search. An example is given in Figure 2b. Therefore, when searching for similar subsequences, candidates from the R*-tree are first retrieved. The candidates are ranked so as to process the most promising ones first and then, those candidates are examined against the provided query representation. When the latter is reduced (as in the case of transcoding that will be explained), false-positives are still possible. Nevertheless, their number is significantly reduced. More details about indexing can be found in (Karydis, Nanopoulos, Papadopoulos, & Manolopoulos, 2005a).



(a) (b)

Figure 2. Feature extraction process.

Query initialisation. The querier assigns to *R* the entire query sequence (plus the few query coefficients) and propagates (broadcasts) it to all its neighbours.

Reception of R. Upon the reception of *R*, each MH *P* probes its indexes, resolves the false-positives, and produces a list of results (only true-positives). The answer-set is propagated back to the querier, by broadcasting it to *all* the neighbours of *P* (backward phase). Accordingly, should there be available *h*, *R* is conveyed to all P's neighbouring MHs (forward phase).

Reception of an answer-set. Each MH *P*, that is not the querier, receiving an answer-set, continues the propagation (backward phase) to *all* its neighbouring MHs as long as there is available *h*.

An answer-set reaches the querier. When an answer-set reaches the querier, then the results are immediately presented to the user.

Figure 3. The ML algorithm.

Searching Algorithms

This section presents the two algorithms that implement the searching procedure, as described in (Karydis et al., 2006). The first is based on simple choices concerning the representation *R* of the query sequence and its propagation during the forward and backward phases, while the second is based on more advanced choices with respect to the latter issues.

Algorithm Based On Maximal Query Representation

A simplistic approach for the representation *R* is to set it identical to the query sequence. The advantage is that no false-positives occur, since when a possible match has been found by index probing, it can be immediately tested against the query itself (i.e., *R*). Thus, no false-positives will be included in the answer-sets, which could negatively impact the backward-phase traffic, as they would be propagated to the querier just to discover that they are not actual matches. It should be noticed that, to be able to perform index probing (i.e., to avoid sequential searching at each MH), a small number of DWT coefficients are included in *R* as well. However, their size is negligible compared to the size of the query sequence.
The resulting algorithm is denoted as ML (full *M*aximum representation with *L*ocal resolution at MHs). ML is summarised in Figure 3 according to the actions performed for each occurring event (see Figure 1(b)).

Algorithm Based On Reduced Query Representation And Transcoding

Section "Algorithm based on Maximal Query Representation" made apparent that a trade-off exists between the forward and backward traffic. ML focuses only on the improvement of backward traffic and incurs high forward traffic. In this section a different algorithm is presented, which has a two-fold objective. The first is to produce a representation *R* that achieves a balance between the two phases and minimises the overall traffic. The second is to develop selective routing policies for the propagation of the answer-sets, leading to significant reduction of the backward traffic.
The first objective is confronted by setting *R* between the two extremes cases: (i) the minimum possible representation with only the *d* DWT coefficients that are required for the local index-searching (minimising forward traffic), and (ii) the maximum possible representation with all *n* elements in the query sequence itself (eliminating the burden of false-positives in terms of computation and backward traffic). Therefore, between the two extremes, *R* can consist of the *l* greater DWT coefficients, where

$d \le l \le n$. Notice that this type of representation generalises the two extreme cases: by setting $l = d$, R becomes identical to the first (i) case; in contrast, by setting $l = n$, R becomes identical to the second (ii) case, since the n DWT coefficients are equivalent to the n elements of the query sequence (due to the Parseval's theorem)[iii]. As described in Section "Features and Indexing", a number l of the greater DWT coefficients can effectively capture the energy of the music sequence and reduce the number of false-positives. The result is that, compared to the second (ii) case, the forward traffic is smaller, because $l \le n$. Compared to the first (i) case, the backward traffic is smaller too, due to the number of false-positives being significantly reduced, since $d \le l$.
The tuning of l, however, is difficult, as it depends on several factors, like the topology of the MANET, which are changeable. Accordingly, the following approach can be used: Initially, l takes an adequately large value and this value is monotonically reduced during the propagation of R in the forward phase. This constitutes a *transcoding* scheme, as it involves sequences with varying number of DWT coefficients that correspond to varying approximations of the initial query sequence. The transcoding scheme:

- Keeps forward traffic low, since the size of R is reducing during its propagation in the forward phase.
- Reduces backward traffic by letting the MHs involved in the forward phase to cache the transcoded representation and, during the backward phase, to use it for early resolving false-positives, before these reach the querier. The problem of caching depends on several network parameters and is independent to the theme of this section of the chapter, while effective solutions can be found in (Fang, Haas, Liang, & Lin, 2004). Experimentation in (Karydis et al., 2006), shows that by simply caching the representations for a small, fixed amount of time, adequate performance is attained.
- Reduces the processing (CPU) time at each MH, as the cost of resolving false-positives at each MH depends on the size of R.

The reduction is performed by getting l values according to an inverse sigmoid function (Figure 4b). Due to the shape of this function, the immediate neighbourhood of the querier, which can provide results faster, receives a larger R, whereas the burden posed on MHs that are far is appreciably smaller. Additionally, this way the exponential growth of traffic that results by plain broadcasting of a full-size representation is controlled. An example is depicted in Figure 4a. P_1 is the querier and P_4 is the node that starts propagating the answer-set. The MHs in the path from P_1 to P_4 are depicted gray shaded, and they are annotated with the size of R that reaches them (P_1 starts with 10K DWT coefficients). Figure 4b illustrates that these sizes are reducing, following an inverse sigmoid function. During the backward phase, starting from P_4, MHs P_3 and P_5 can be reached (depicted with dashed arrows). The cached representation in P_3 can help resolving possible false-positives in the answer-set. This is due to the fact that in P_4 the false-positives were examined against a smaller R than the one in P_3. In contrast, P_5 was not in the path, thus cannot resolve any false-positives.
Henceforth, the size of the initial query representation is given as a factor (denoted as I) of the complete query size, whereas the slope of the inverse sigmoid function is controlled by a parameter denoted as α (higher values of α produce a steeper slope).

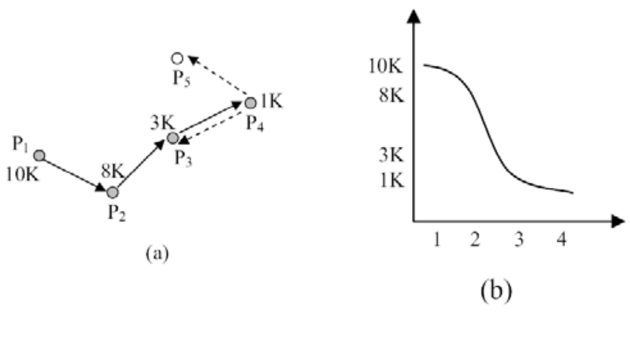


Figure 4. Searching procedure example.

Regarding the second objective, in contrast to the simplistic approach of ML, which propagates the answer-sets to all neighbours, during the forward phase, as it is typical in any dynamic source routing protocol (Johnson & Maltz, 1996), each MH that receives *R*, additionally receives the ID of all MHs in the path that were used from the querier to it. These IDs can be maintained along with *R* with minimal cost (only some bytes). When a MH starts propagating answer-sets, it selects among its current neighbours those that will propagate the answer-set (not all of them). To make this selection, it applies a policy that focuses on the neighbours that were included in the path from the querier to it. Since several such policies can be developed, Section "Routing Policies for the Backward Phase" elaborates further on them. All the policies, despite their differences, emphasise on selecting neighbouring MHs that were included in the path, due to the cached representations they maintain, which can resolve some false-positives in situ.

The algorithm that combines all the aforementioned characteristics is denoted as RT (querying by *R*educed representation with *T*ranscoding) and is illustrated in Figure 5.

Query initialisation. The querier sets *R* equal to a sample with an initial size (parameter) plus the query coefficients, and propagates (broadcasts) it to all its neighbours.

Reception of R. Upon the reception of *R*, each MH *P* probes its indexes, resolves as many false-positives as possible based on the received query sample of *R*, and produces a list of results. The answer-set is propagated back to the querier, by following the described policy for the backward phase. Accordingly, should there be available *h*, *R*'s size is reduced, and the reduced *R* is conveyed to all P's neighbouring MHs (forward phase).

Reception of an answer-set. When a MH receives a reply, it checks if it can resolve any false-positives. This is true, should it have received (if any) a representation that was larger than the one that the sequences in the answer set were examined previously (i.e., at the sending MH). After any possible pruning, as long as there is available *h*, the answer-set is routed backwards following a policy.

An answer-set reaches the querier. When an answer-set reaches the querier, initially any remaining false positives requiring resolution are examined, and then the results are presented to the user.

Figure 5. The *RT* algorithm.

Routing Policies For The Backward Phase

Accordingly, Karydis et al. (2006) describe three policies for routing the answer-sets in the backward phase. The first two policies (global and local counter) are based on existing methods (Castaneda, Das, & Marina, 2002). As mentioned, all policies try to select nodes that were included in the path during the forward phase. Nevertheless, the backward phase cannot be based only on such nodes. Due to the mobility of MHs, it may be impossible to reach the querier unless other MHs (not included in the path) are additionally involved. The objective of all policies is to control the number of involved MHs so as to reduce backward traffic. These policies constitute a hybrid approach between probabilistic broadcasting, where the broadcasting decision is completely local to each mobile host and the deterministic broadcasting which relies on the discovery of some form of connected dominating set (Lou & Wu, 2004).

Global and local counter policies

To clarify the description of the first two policies, consider the example of Figure 6a, which depicts the path from MH P_1 to MH P_6, which was followed in the forward phase. Figure 6b depicts the routing of the answer-set from P_6 back to P_1. Comparing the two phases, several MHs have changed their location, others have switched off, and some new ones have become reachable. The MHs that are greyed are the ones that were included in the forward path too, whereas the rest are new ones that were involved only in the backward phase.

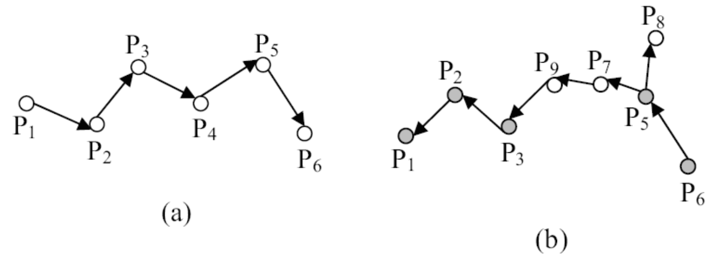


Figure 6. Propagation in a MANET: a) forward phase, b) backward phase.

With the *global-counter* (GC) policy, when selecting the MHs to route back the answer-set, GC tries to follow the MHs included in the forward path. However, to overcome problems from the alteration of the MANET (like the disappearance of P_4 in this example), it allows an amount of discrepancy by resorting to broadcasting. To control the discrepancy, and thus the backward traffic, it uses the value of *e*. Notice that with a very large *e*, GC resorts to broadcasting for a very large number of times, thus becoming equivalent to the simplistic policy used by the ML searching algorithm. In contrast, with a very small *e*, the querier may not become reachable, especially when the MANET changes very fast.

A variation of GC works as follows. After a discrepancy, when a MH from the path has been reached again, *h* is reset to its initial value. In the previous example, when P_3 is reached again, available hop is reset to 6 (initial value). Thus, *h* acts as a decreasing local counter, because it is reset independently at several MHs. For this reason this policy is denoted as *local-counter* (LC). Its objective is to increase the probability of reaching the querier, by rewarding the identification of the forward path. Nevertheless, this can increase the backward traffic.

Critical mass policy

With the *critical-mass* (CM) policy, if at least a number, denoted as *critical-mass factor* (*CMF*), of the current neighbours was in the forward path, they are selected as the only ones to propagate the answer-set. If their number is less than *CMF*, then some of the current neighbours (not in the path) are additionally randomly selected in order to have at least *CMF* MHs to propagate the answer-set. In contrast, if their number is larger than *CMF*, then they are all selected. For example, consider the case in Figure 7. Figure 7a depicts the forward phase, whereas Figure 7b presents the backward case. As shown, during the backward phase some MHs have now relocated. Let *CMF* be 2. When P_4 starts propagating the answer-set, it first selects P_3, because it belongs to the forward path. Since this is the only such MH and *CMF* is 2, it also selects P_5 at random, among the other reachable MHs.

The nodes that were selected at random in order to fulfil *CMF*, are still provided with the path of the MH that initiated the propagation of the answer-set (for the previous example, P_5 that is selected by P_4, will also know the path from P_1 to P_4). This way, due to mobility, it is possible for such nodes during the backward phase to find neighbours that appear in the forwarded path (in the same example, P_5 finds P_2 that was in the path). Therefore, the impact of such randomly selected MHs on the proposed policy may be kept at a moderate level.

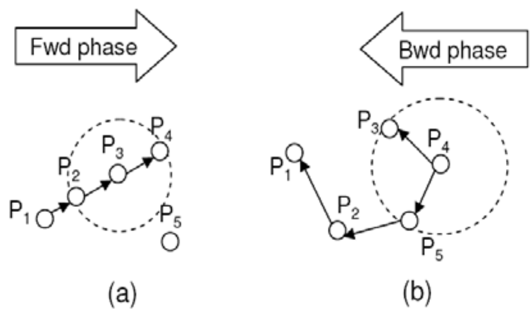
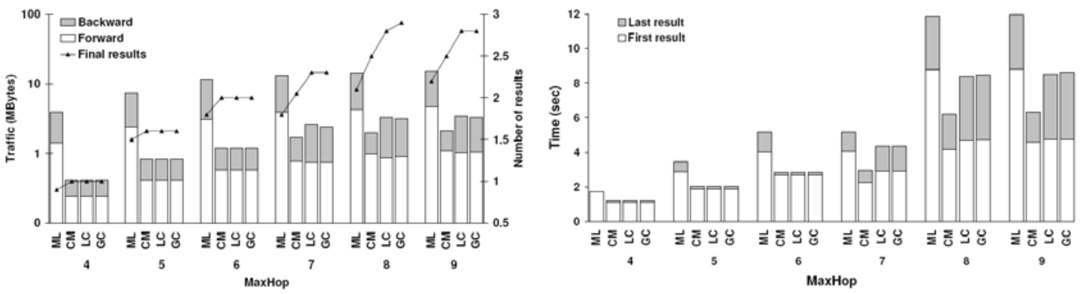


Figure 7. MHs' relative locations in forward and backward phase example.

Experimental Results

Four of the most prominent experimental results presented in (Karydis et al., 2006) are discussed in this section. The first experiment, examines the traffic against MaxHop. The results are illustrated in Figure 8a. As expected, ML produces the highest forward traffic in all cases, whereas the forward traffic of CM, LC and GC are about the same. Regarding the backward traffic, ML attains a decreased number of returning results. However, due to the absence of an efficient backward routing policy, this advantage is invalidated. The rest approaches, considerably improve backward traffic, with CM performing better for MaxHop greater than seven. From this result it becomes obvious that, although the backward phase is in general more demanding for all algorithms, due to the reduction of backward traffic attained by CM, LC and GC, the requirement for optimisation of the forward phase, is fair.



(a) (b)

Figure 8. Traffic, number and time of results vs. MaxHop.

This result can be further clarified by the results on time of the first and last results, which are depicted in Figure 8b. As expected, increase in available MaxHop produces longer times, since more MHs are examined. In all cases, the increase in time is far steeper for ML, while CM presents an advantage over LC and GC.

Next, the impact of query range ε is examined. Figure 9 shows the results for traffic with respect to ε. CM, LC, and GC perform similarly, thus only the results for the former are included. As ε increases, more results are found and, consequently, backward traffic increases too (forward traffic is unaffected). However, the increase is much more obvious for ML, whereas CM, due the effectiveness of the policy for the backward phase, has a very smooth increase.

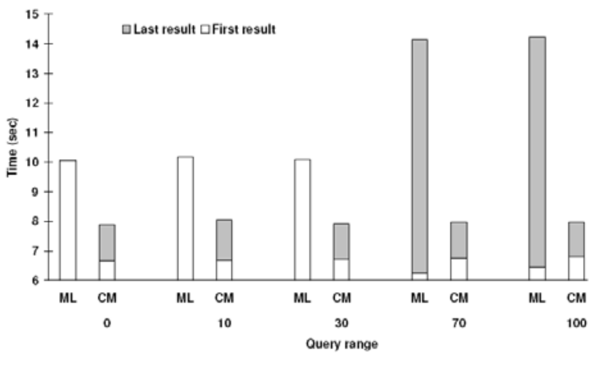


Figure 9. Traffic vs. query range (ε).

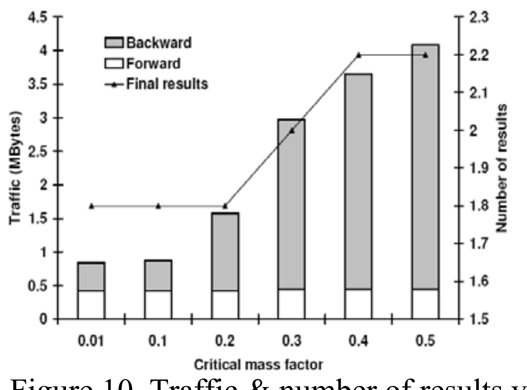


Figure 10. Traffic & number of results vs. CMF.

The last experiment, examines the sensitivity of CM against *CMF*. The traffic and number of results of CM, for varying *CMF* values, are depicted in Figure 10. When *CMF* is high, the effectiveness of the policy for the backward phase is limited, since most MHs are selected at random by this policy, resulting to high backward traffic.

SUMMARY AND PERSPECTIVES

We have presented the most significant trends in recent research in the field of content based music information retrieval in peer-to-peer networks. Despite the diminished attention the area has received in general terms, its relatively close area of metadata MIR in P2P is by far new. On the contrary, it could be argued that it was actually one of the key threads that lead to the widespread acceptance of P2P systems. Though, as metadata prove to be inefficient for the purposes of MIR as well as the peculiarities of music in comparison to text and image data, the development of dedicated solutions for CBMIR in P2P networks becomes obvious. As described, CBMIR in P2P networks presents unique challenges.

P2P networks can be classified according to numerous of their characteristics. A number of prominent research works have been presented in this chapter falling within all categories. Despite an obvious tendency towards decentralised unstructured models, solutions for other categories of P2P models do exist. Additionally, we have presented an initial attempt of CBMIR to invade the area of wireless P2P networks, as well as the challenges presented in that case.

The prospects of MIR in P2P networks, both in terms of research and applications, seem to be encouraging. As it is relatively new a research field, it contains several open research issues. Following ethics, a very important area of possible further development is the implementation of data rights management. As the previously mentioned widespread penetration of P2P systems based on the illegal exchange of copyrighted material, methods are required to enforce rights on data. Another field is music E-commerce. Attempts like iTune or iMusic changed the paradigm that music is merchandised. Moreover, applications in P2P environments (such as Napster and Kazaa) can only set the path for commercial CBMIR in P2P networks. Additionally, with the development of high CPU capability mobile computing devices, one can envisage an ubiquitous exchange of ringtones (cell phones) and music (PDAs, portable and tablet computers or even dedicated devices) while on the go.

REFERENCES

Agrawal, D.-P., & Zeng, Q.-A. (2003). *Introduction to Wireless and Mobile systems*. Thomson, Brooks/Cole.

Baochun, L., & Wang, K.-H. (2003). NonStop: Continuous multimedia streaming in wireless ad-hoc networks with node mobility. *IEEE Journal on Selected Areas in Communications*, *21*(10), 1627-1641.

Beckmann, N., Kriegel, H., & Seeger, B. (1990). The R*-tree: An efficient and robust method for points and rectangles. In *Proceedings ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)* (p. 322-331).

Byrd, D. & Crawford, T.(2002). Problems of music information retrieval in the real world. *Information Processing and Management*, 38(2), 249-272.

Castaneda, R., Das, S. R., & Marina, M. K. (2002). Query localization techniques for on-demand routing protocols in ad-hoc networks. *Wireless Networks*, *8*(2/3), 137-151.

Chan, K., & Fu, A.-C. (1999). Efficient time series matching by wavelets. In *Proceedings International Conference on Data Engineering (ICDE)* (p. 126-133).

Chan, K.-P., Fu, A.-C., & Yu, C. (2003). Haar wavelets for efficient similarity search of time-series: With and without time warping. *IEEE Transactions on Knowledge and Data Engineering*, *15*(3).

Chu, C.-C., Su, X., Prabhu, B., Gadh, R., Kurup, S., Sridhar, G., et al. (2006). Mobile drm for multimedia content commerce in p2p networks. In *Proceedings IEEE Workshop on Digital Rights Management Impact on Consumer Communications.*

DeVriendt, J., C., P. L., Lerouge, & Xu, X. (2002). Mobile network evolution: A revolution on the move. *IEEE Communications Magazine*, 104-111.

Dhamija, R., & Wallenberg, F. (2003). A framework for evaluating digital rights management proposals. In *Proceedings International Mobile IPR Workshop: Rights Management of Information Products on the Mobile Internet.*

Dubosson-Torbay, M., Pigneur, Y., & Usunier, J.-C. (2004). Business models for music distribution after the p2p revolution. In *Proceedings Wedelmusic* (p. 172-179).

Dutta, A., Chennikara, J., Chen, W., Altintas, O., & Schulzrinne, H. (2003). Multicasting streaming media to mobile users. *IEEE Communications Magazine*, 81-89.

Fang, Y., Haas, Z., Liang, B., & Lin, Y.-B. (2004). TTL prediction schemes and the effects of inter-update time distribution on wireless data access. *Wireless Networks*, *10*(5), 607-619.

Indyk, P., & Motwani, R. (1998). Approximate nearest neighbours: Towards removing the curse of dimensionality. In *Theory of computing.*

Johnson, D., & Maltz, D. (1996). Dynamic source routing in ad-hoc wireless networks. In *Mobile computing* (Vol. 353). Kluwer Academic Publishers.

Kalker, T., Epema, D., Hartel, P., Lagendijk, R., & Steen, M. van. (2004). Music2share -copyright-compliant music sharing in p2p systems. *Proceedings of the IEEE*, 961-970.

Kalogeraki, V., Gunopulos, D., & Zeinalipour-Yazti, D. (2002). A local search mechanism for peer-to-peer networks. In *Proceedings Conference on Information and Knowledge Management (CIKM)* (p. 300-307).

Karydis, I., Nanopoulos, A., & Manolopoulos, Y. (2005). Evaluation of similarity searching methods for music data in peer-to-peer networks. *International Journal of Business Intelligence and Data Mining*, *1*(2), 210-228.

Karydis, I., Nanopoulos, A. & Manolopoulos, Y. (2006). Mining in Music Databases. *Processing and Managing Complex Data for Decision Support* (p. 340-374). Idea Group Publishing.

Karydis, I., Nanopoulos, A., Papadopoulos, A., Katsaros, D., & Manolopoulos, Y. (2006). Audio music retrieval in wireless ad-hoc networks. To appear in *IEEE Transactions on Audio, Speech and Language Processing*.

Karydis, I., Nanopoulos, A., Papadopoulos, A., & Manolopoulos, Y. (2005a). Audio indexing for efficient music information retrieval. In *Proceedings Multimedia Modelling Conference (MMM)* (p. 22-29).

Karydis, I., Nanopoulos, A., Papadopoulos, A. N., & Manolopoulos, Y. (2005b). Music retrieval in p2p networks under the warping distance. In *Proceedings International Conference on Enterprise Information Systems (ICEIS)* (p. 100-107).

Keogh, E., & Kasetty, S. (2002). On the need for time series data mining benchmarks: A survey and empirical demonstration. In *Proceedings ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)* (p. 102-111).

Keogh, E., & Ratanamahatana, A. (2005). Exact indexing of dynamic time warping. *Knowledge and Information Systems*, *7*(3), 358-386.

Kostek, B., & Wieczorkowska, A. (1997). Parametric representation of musical sound. In *Archive of acoustics* (p. 3-26).

Lew, M. S., Sebe, N., Djeraba, C., & Jain, R. (2006). Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications and Applications*. 2(1), 1-19.

Li, T., Li, Q., Zhu, S., & Ogihara, M. (2002). A survey on wavelet applications in data mining. *Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD) Explorations*, *4*(2), 49-68.

Li, T., Ogihara, M., & Li, Q. (2003). A comparative study on content-based music genre classification. In *Proceedings ACM Conference on Research and Development in Information Retrieval (SIGIR)* (p. 282-289).

Li, X., & Wu, J. (2004). Searching techniques in peer-to-peer networks. In *Handbook of Theoretical and Algorithmic Aspects of Ad hoc, Sensor, and Peer-to-peer Networks.*

Liu, Z., Kundur, D., Merabti, M., & Yu, H. (2006). On peer-to-peer multimedia content access and distribution. In *IEEE International Conference on Multimedia and Expo (ICME).*

Lou, W., & Wu, J. (2004). Broadcasting in ad-hoc networks using neighbour designating. In *Handbook of Mobile Computing.* CRC Press.

Ni, S.-Y., Tseng, Y.-C., Chen, Y.-S., & Sheu, J. (1999). The broadcast storm problem in a mobile ad-hoc networks. In *Proceedings ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)* (p. 151-162).

Orio, N. (2006). Music Retrieval: A Tutorial and Review. *Foundations and Trends in Information Retrieval*, 1(1), 1-90.

Papaodysseus, C., Roussopoulos, G., Fragoulis, D., Panagopoulos, T., & Alexiou, C. (2001). A new approach to the automatic recognition of musical recordings. *Jounal of Acoustical Engineering Society*,

49(1/2), 23-35.
Paraskevas, M., & Mourjopoulos, J. (1996). A statistical study of the variability and features of audio signals. In *Audio Engineering Society.*
Praveen, K., Sridhar, G., Sridhar, V., & Gadh, R. (2005). Dmw -a middleware for digital rights management in peer-to-peer networks. In *Proceedings International Workshop on Secure and Ubiquitous Networks.*
Roads, C., Pope, S., Piccialli, A., & Poli, G. D. (Eds.). (1997). *Musical signal processing*. Royal Swets & Zeitlinger.
Roccetti, M., Salomoni, P., Ghini, V., & Ferretti, S. (2005). Bringing the wireless Internet to UMTS devices: A case study with music distribution. *Multimedia Tools and Applications*, *25*(2), 217-251.
Sastry, N. (2005). *Stealing (from) the music industry* (Tech. Rep.). Computer Science Dept., University of Berkeley.
Schyndel, R. G. van. (2005). A preliminary design for a privacy-friendly free p2p media file distribution system. *International Workshop on Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP) part of Knowledge-Based Intelligent Information and Engineering Systems (KES)*, *3*, 1018-1024.
Typke, R., Wiering, F., Veltkamp, R. C. (2005). A Survey of Music Information Retrieval Systems. *In Proceedings International Conference on Music Information Retrieval (ISMIR)* (p. 153-160).
Tzanetakis, G., & Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, *10*(5), 293-302.
Tzanetakis, G., Gao, J., & Steenkiste, P. (2004). A scalable peer-to-peer system for music information retrieval. *Computer Music Journal*, *28*(2), 24-33.
Wang, C., Li, J., & Shi, S. (2002). A kind of content-based music information retrieval method in a peer-to-peer environment. In *Proceedings International Symposium on Music Information Retrieval (ISMIR)* (p. 178-186).
Wieczorkowska, A. (2001). Musical sound classification based on wavelet analysis. *Fundamenta Informaticae*, *47*(1/2), 175-188.
Yang, B., & Garcial-Molina, H. (2002). Improving search in peer-to-peer networks. In *Proceedings Conference of Distributed Computer Systems* (p. 5-15).
Yang, C. (2002). Efficient acoustic index for music retrieval with various degrees of similarity. In *Proceedings ACM Multimedia Conference* (p. 584-591).
Yang, C. (2003). Peer-to-peer architecture for content-based music retrieval on acoustic data. In *Proceedings international World Wide Web Conference (WWW)* (p. 376-383).

---

[i] This research is supported by the ΗΡΑΚΛΕΙΤΟΣ and ΠΥΘΑΓΟΡΑΣ II national programs funded by the ΕΠΕΑΕΚ.
[ii] Each element in an acoustic sequence is in the range 0-255, thus it requires one byte.
[iii] In the case of ML, R could consist of all the n DWT coefficients. However, the n sequence elements in the time domain are selected just to avoid the computation of the inverse DWT, since the time domain presents a smaller storage requirement as the data values are in range 0-255.