On the Selectivity of Multidimensional Routing Indices

Christos Doulkeridis[1], Akrivi Vlachou[1], Kjetil Nørvåg[1], Yannis Kotidis[2], Michalis Vazirgiannis[2]
[1]Department of Computer Science, NTNU, Trondheim, Norway
[2]Department of Informatics, AUEB, Athens, Greece
{cdoulk,vlachou,noervaag}@idi.ntnu.no, {kotidis,mvazirg}@aueb.gr

ABSTRACT

Recently, the problem of efficiently supporting advanced query operators, such as nearest neighbor or range queries, over multidimensional data in widely distributed environments has attracted much attention. In unstructured peer-to-peer (P2P) networks, peers store data in an autonomous manner, thus *multidimensional routing indices* (MRI) are required, in order to route user queries efficiently to only those peers that may contribute to the query result set. Focusing on a hybrid unstructured P2P network, in this paper, we analyze the parameters for building MRI of high selectivity. In the case where similar data are located at different parts of the network, MRI exhibit extremely poor performance, which renders them ineffective. We present algorithms that boost the query routing performance by detecting similar peers and reassigning these peers to other parts of the hybrid network in a distributed and scalable way. The resulting MRI are able to eagerly discard routing paths during query processing. We demonstrate the advantages of our approach experimentally and show that our framework enhances a state-of-the-art approach for similarity search in terms of reduced network traffic and number of contacted peers.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems—*Query processing*

General Terms

Algorithms, Experimentation, Performance

Keywords

Multidimensional routing indices, P2P query processing

1. INTRODUCTION

Routing indices [5] have been proposed to improve the performance of search in unstructured peer-to-peer (P2P) networks. The purpose of routing indices is to direct queries to peers in an intentional manner, by discarding network paths. Traditional routing indices in P2P systems are mainly designed for document retrieval applications, thus maintaining aggregated one-dimensional values, representing the number of documents that can be obtained in a certain direction in the network. However, many applications handle multidimensional data and require more advanced query types. Recently, there exists a trend towards supporting advanced query processing in P2P networks as well [7, 19, 31], using multidimensional summary information for directing queries in the network in a deliberate way. Similar to centralized multidimensional indices, data stored on peers is described by representative (multidimensional) data descriptors, realized as minimum bounding regions (MBRs) that enclose all data points on a peer.

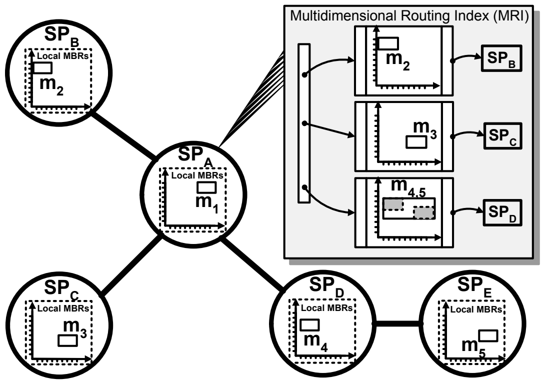


Figure 1: MRI of SP_A in a super-peer network.

The main disadvantage of using multidimensional indexing techniques in unstructured P2P networks, is that the query processing performance quickly deteriorates when the number of participating peers increases. Therefore, in this paper, we assume a super-peer overlay that provides the necessary stability in order to provide efficient query processing. Nowadays, the advantages of using a super-peer architecture [32] have been recognized by most of the existing P2P file-sharing networks (eMule, KaZaA) that rely on a super-peer architecture. Beyond file-sharing, there are several important applications, such as distributed image search [12] and query processing over distributed collaborative scientific databases [22, 24], which can benefit from a super-peer architecture. For example, consider scientists (peers) that collect massive datasets of astronomical observations [24] or biological data [22] and upload their descriptions to dedicated servers (super-peers), in order to share it with other scientists. Unlike plain file-sharing platforms, these distributed applications handle multidimensional data and require more advanced query types, such as nearest neighbor or range queries, in order to provide the desired functionality.

Multidimensional routing indices (MRI) [7, 10, 19] are composed of different local centralized multidimensional indices, each stored at a super-peer. The data objects stored by the MRI at a super-peer are MBRs that summarize the data available through each neighboring super-peer. The simplest form of a MRI at a super-peer is a list of one or more MBRs for each neighboring super-peer. In the example of Figure 1, the MRI of SP_A is depicted, which consists of three entries, one for each neighbor $\{SP_B, SP_C, SP_D\}$. The MRI are constructed in a distributed manner. SP_D is informed about SP_E's content by receiving a set of MBRs (i.e., m_5), representing the data that can be retrieved if a query is forwarded in that direction. This information is then aggregated ($m_{4,5}$) with its own MBRs (i.e., m_4) and forwarded to SP_A. Thus, each super-peer (in this case SP_A) builds and maintains its own MRI and utilizes it in order to efficiently route a query to its neighbors.

There exists a distinct difference between a MRI and a distributed multidimensional index, such as [11]. In the latter case, the nodes of a single index are themselves distributed on different servers, whereas MRI consist of multiple centralized indices at super-peers that describe the data available through each neighboring super-peer. Also notice the difference between approaches that rely on a structured [28] (i.e., DHT-based) or tree-based [21] P2P network, where each peer is (a priori) assigned and becomes responsible for a part of the data space. These approaches follow a space partitioning approach and rely on deliberate data placement on peers, whereas MRI adopt a data partitioning approach where each peer stores its data autonomously. Even though multidimensional indexing techniques based on data partitioning have been extensively studied in centralized settings [13], the advantages of using data partitioning techniques in P2P systems have been only partially explored.

Assuming that each super-peer stores its own MRI, a super-peer is able to deliberately route the query to some of the neighboring super-peers during query processing. For example, a range query is forwarded to all neighboring super-peers for which there is an MBR in the MRI that overlaps with the query. Thus, the performance of query processing depends on the *selectivity* of routing indices, which is the ability to discard neighboring super-peers (or routing paths) during query routing. An important parameter that affects the selectivity of the MRI is the similarity of the data maintained by neighboring super-peers. The aggregation of the MBRs during the MRI construction causes the enlargement of the MBRs. In turn, this leads to overlapping MBRs and affects the selectivity of the MRI, similar to multidimensional access methods [13] in the centralized case. In our running example in Figure 1, the aggregation of m_4 and m_5 leads to an enlarged MBR, namely $m_{4,5}$. Notice that SP_A stores locally only $m_{4,5}$ and the enclosed MBRs are depicted only for sake of clarity. In addition, since each super-peer summarizes the data stored by its peers by a set of MBRs, an important factor is the underlying distribution of data to super-peers. Although data on peers may be clustered into a few thematic areas that reflect the user's interests, when peers join the network by connecting to a randomly chosen super-peer, the super-peers end-up indexing data spread all over the data space. Again in this case the selectivity of the MRI is poor.

Focusing on a super-peer network, in this paper, we propose an efficient approach for constructing MRI of high selectivity that boosts the query processing performance, since fewer super-peers need to be contacted. We present algorithms for the identification of peers with similar content and reassign them to the same super-peer, by establishing new additional overlay connections. Our approach is self-organizing, in that there is no prior assignment of space partitions or data to each super-peer, but the data distribution is dynamically captured. The peer MBRs reassignment takes into account the super-peer topology, so that neighboring super-peers index peers with similar content.

The individual contributions of this paper are:

- We discuss the concept of multidimensional routing indices and analyze important performance parameters that determine their routing ability (Section 3).
- We present novel self-organizing algorithms for detecting peers with similar content and, subsequently, grouping such peers to the same super-peer (Section 4).
- We study the problem of assigning groups of peers to super-peers, by taking into account the network topology, in order to maximize the similarity of neighboring super-peers. The proposed mapping creates a clustered overlay topology leading to improved selectivity of the resulting multidimensional routing indices (Section 5).
- We describe efficient maintenance techniques of MRI in the presence of data updates, peer joins and failures (Section 6).
- We demonstrate the advantages of our approach using large-scale simulations and show that our techniques support efficient query processing in terms of reduced network traffic and contacted peers (Section 7).

In addition, we provide the definitions in Section 2, we review related work in Section 8, and we conclude in Section 9.

2. OVERVIEW AND DEFINITIONS

In this paper, we assume a super-peer network [7, 32] that consists of N_{sp} super-peers, each connected to a limited set of at most DEG_{sp} other super-peers. Such networks include many simple peers and few enhanced super-peers, in terms of processing power, storage capacity or network connectivity. Each super-peer SP_i is responsible for DEG_p simple peers, which connect to SP_i directly. The initial assignment of peers to super-peers is random with respect to peers' contents [7, 32]. Each peer P_i, $i \in [1, N_p]$, maintains its own dataset O_i that consists of n_i d-dimensional points. Hence, the complete dataset O is the union of all N_p datasets O_i ($O = \cup O_i$, $i \in [1, N_p]$), which is the case of horizontal data distribution, and moreover the size of the complete set of points is $n = \sum_{i=1}^{N_p} n_i$.

We are interested in supporting *exact query processing* on top of the super-peer network, which means that we would like to process a query in a distributed manner, but the answer set should be the same as if the query had been executed on the dataset O in a centralized setting. In the rest of the paper, we describe how range queries are processed, merely as a showcase example, although other query types can be supported as well.

DEFINITION 1. ***Range query*** $R(q, r)$. *Given a query object q and a radius r, a point $p \in O$ belongs to the result set of the range query if $dist(q, p) \leq r$, where $dist()$ denotes the distance function.*

Multidimensional routing indices are built at super-peer level, in order to route user queries more efficiently to only few super-peers that can contribute to the query result set. We now provide a more concrete description of a multidimensional routing index at super-peer SP_i. The routing index MRI can be considered as a set of up to DEG_{sp} entries $MRI = \{S_1, \ldots, S_{DEG_{sp}}\}$, one for each neighboring super-peer. Each entry S_j consists of k_j MBRs and is associated with a super-peer SP_j that is a neighbor of SP_i. Any data point stored by a peer that is accessible through super-peer

SP_j is enclosed by an MBR that belongs to the S_j entry. Each super-peer SP_i can use any centralized technique to store and query efficiently these sets of MBRs, such as an R-Tree [16].

In the following, we quantify the similarity between MBRs by defining appropriate metrics, analogously to construction methods of multidimensional access methods. We refer to *dead space* as the space that is covered by the aggregated MBR, but not by the MBRs that are enclosed by it.

The similarity of a set of MBRs can be expressed by the volume of the MBR that encloses all MBRs that belong to the set.

DEFINITION 2. ***Enclosed Volume.*** *Given a set of n MBRs $\{m_1,\ldots,m_n\}$, the enclosed volume $\hat{V}$ is defined as the volume of the MBR m' that encloses $m_1,\ldots,m_n$:*

$$\hat{V}(m_1,...,m_n) = Volume(m') \tag{1}$$

When $\hat{V}$ is large, the enclosed MBRs are dissimilar and the probability of enclosed dead space is high. Small values of $\hat{V}$ indicate that the enclosed MBRs are probably overlapping or lying nearby in the data space.

The most important factor that determines the quality of an individual MBR is the volume of dead space that it encloses. In order to quantify the dead space, we define a quality measure named *compactness*.

DEFINITION 3. ***Compactness.*** *Given an MBR m_i which encloses a set of k MBRs m_j, $1 \leq j \leq k$, the compactness of m_i is defined as:*

$$CO(m_i) = \frac{\sum_{j=1}^{j=k} Volume(m_j)}{Volume(m_i)} \tag{2}$$

A small value of compactness (much smaller than 1) means that there is a large volume of dead space inside m_i.

The overlap of two MBRs also conveys some notion of similarity, therefore we define the relative overlap of two MBRs.

DEFINITION 4. ***Relative Overlap.*** *Given m_i and m_j, their overlap from the perspective of m_i is defined as relative overlap:*

$$RO(m_i, m_j) = \frac{Volume(m_i \cap m_j)}{Volume(m_i)} \tag{3}$$

By definition relative overlap is not symmetric $RO(m_i,m_j) \neq RO(m_j,m_i)$, when $Volume(m_i) \neq Volume(m_j)$, and its values vary between 0 and 1. When RO is high, close to 1, with respect to the volume of m_i, then m_i is almost covered by m_j. In the case that RO is low, the volume of the intersection is much smaller than the volume of m_i, even though the overlap may be significant.

3. MULTIDIMENSIONAL ROUTING INDICES

In this section, we first describe the construction of MRI and then we describe how query processing is performed. Finally, we analyze the factors that affect the selectivity of MRI.

3.1 Construction Phase

Prior to MRI construction, *local indexing* of peer MBRs at the responsible super-peer is performed. Each super-peer aggregates and maintains a set of MBRs that summarize the available data stored on its peers. Assume that each peer's data is described by a list of k_i MBRs, which enclose all data objects that exist on the peer. Each super-peer gathers the MBRs of its DEG_p associated peers and stores them locally using any available centralized multidimensional indexing data structure, such as an R-Tree. In addition, each super-peer creates a list of *aggregated MBRs*[1], based on the collected MBRs of its peers. Then, given a peer P_j connected to a super-peer SP_i, every MBR of P_j is enclosed by at least one MBR of the aggregated MBRs of SP_i. In the case of an R-Tree, the aggregated MBRs could be the root MBRs.

The MRI are constructed in a distributed manner, by having each super-peer receive, aggregate and propagate the MBRs of its neighbors. Assuming an acyclic network topology, each super-peer informs its neighbors about the data that it indexes, by broadcasting its aggregated MBRs. A super-peer SP_i that receives a set of MBRs by a neighbor SP_N performs two operations. First, SP_i updates the set S_N with SP_N's MBRs, in order to accurately describe the data that can be obtained, if a query is forwarded to SP_N. Then, for each neighbor SP_j ($j \neq N$), SP_i aggregates its local MBRs with all sets of MBRs present in its routing index (except for the MBRs in entry S_j), and sends the aggregated MBRs to SP_j. In this way, any neighbor super-peer SP_j has an accurate view of the data that can be obtained through SP_i. Following this construction protocol, all super-peers have eventually built their routing indices and have sufficient information in the form of MBRs that describe what data can be retrieved by routing the query to each of their neighbors. Figure 1 serves as a showcase of MRI construction, from the aspect of super-peer SP_A.

In order to handle cycles, one simple solution is to assume a spanning tree over the super-peer network. Then, MBRs are propagated and aggregated using only the super-peer connections in the spanning tree. An alternative solution is that each super-peer propagates the MBRs that it has received from its neighbors, without aggregating them, so duplicate elimination can be performed. Then, each super-peer that receives some MBRs from a neighboring super-peer, can exclude already received MBRs, i.e., from super-peers that are accessible through a different neighbor.

3.2 Query Processing

Consider a range query $R(q,r)$ initiated at a *querying super-peer*. At query time, any super-peer SP_i is able to decide which of the neighboring super-peers may contribute to the query result set based on the MBRs in its MRI. The query is forwarded to all neighbors SP_j, for which at least one routing entry S_j overlaps with the query, while all other neighbors are pruned. This constitutes the *query routing* mechanism.

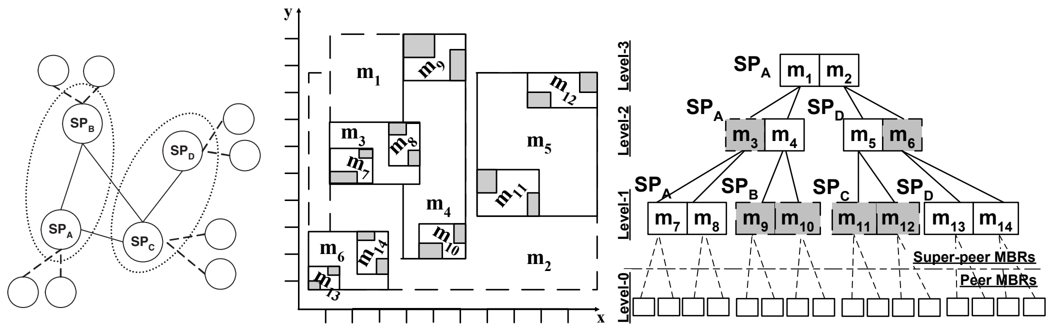
Furthermore, by storing its peers' MBRs, SP_i is able to determine during query processing the peers that may contribute to the result set of the query. Thus, SP_i contacts only peers responsible for MBRs that overlap with the query and this process is called *local query processing* at SP_i. Then, the queried peers process the query based on their local data, and return their results to SP_i.

The query is processed by all super-peers that receive the query in a similar way. Each recipient super-peer first performs query routing, then performs local query processing and waits to gather the results of its neighbors. Finally, the combined result set is sent back through the reverse query path to the querying super-peer.

3.3 Selectivity of MRI

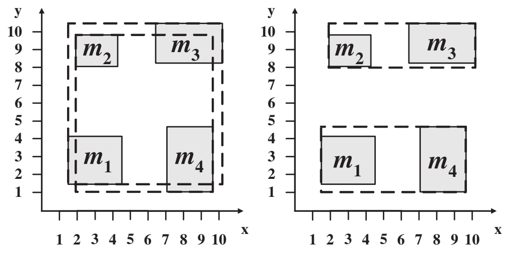
During the construction of the routing indices, MBRs are aggregated at intermediate super-peers. As it is well-known from centralized multidimensional access methods, the aggregation of MBRs causes the enlargement of the MBRs and worsens the per-

[1]Any technique proposed in centralized settings for creating MBRs that enclose multidimensional data may be used, such as packing algorithms [23, 29] or spatial clustering algorithms [17].



(a) Super-peer network. (b) Data space. (c) MBR hierarchy.

Figure 3: Super-peer network, data space and MBR hierarchy.



(a) Poor. (b) Good.

Figure 2: Example of MBR aggregation.

formance of the indices during query processing. Similarly in the case of MRI, the aggregation of the MBRs affects their selectivity, since their enlargement may lead to forwarding the query to more super-peers than necessary. Obviously, the enclosed dead space in an aggregated MBR leads to its enlargement. The aggregation of the MBRs may lead to MRI of low selectivity due to high overlap and dead space.

EXAMPLE 1. *Figure 2 depicts two alternative aggregations of four MBRs in two groups. The aggregation in Figure 2(a) is obviously inappropriate, as the aggregated MBRs have much dead space and overlap, whereas the aggregated MBRs in Figure 2(b) are non-overlapping with much less dead space. Consider now the case that these MBRs represent peer data, and as such they are distributed over a super-peer network. Let us assume that m_1 and m_3 belong to super-peer SP_A, while m_2 and m_4 belong to super-peer SP_B. Then the aggregated MBRs of these super-peers are similar to Figure 2(a), leading to a high probability that a query must be sent to both super-peers. Therefore, a third super-peer that indexes the aggregated MBRs of SP_A and SP_B forwards most queries to both directions.*

High volumes of dead space of the aggregated MBRs increase the probability that a query is forwarded to a direction that retrieves no results, while high overlap results in more traversed paths. The combination of both is even worse, since the query is forwarded to multiple directions with low probability of finding any result.

The problems of MBR aggregation are also reflected in the selectivity of MRI, due to: (1) the initial peer to super-peer assignment, and (2) the MBR propagation and aggregation required by MRI construction. First, even if the peers' MBRs cover a small part of the data space, when peers join the network and connect to a super-peer that is chosen at random, super-peers end up indexing MBRs spread all over the data space. In this case, the MBRs describing the super-peer's data cover a large part of the data space. This causes the degradation of query processing performance, since a query may – in worst case – reach all super-peers, even though some of them may not return any result. Secondly, during MBR propagation, if the MBRs of a super-peer's neighbors are not similar to its own MBRs, the aggregated MBRs will span over the entire data space.

To address these problems, we describe two techniques that are applied *prior to MRI construction*, aiming to improve the selectivity of MRI.

- Discovering and reassigning peers with similar content to the same super-peer, in order to improve the quality of MBRs indexed a each super-peer (Section 4). Then, queries can be directed to specific super-peers only, thus improving the selectivity of MRI during query routing.

- Maximizing the similarity of MBRs at neighboring super-peers, in order to improve the performance of MRI even further (Section 5).

In this way, the initial random peer to super-peer assignment changes in an intentional manner. By establishing new additional overlay connections, super-peers are assigned with peers with similar content in a self-organizing way, which is beneficial for the quality of the subsequently constructed MRI.

4. DISCOVERING SIMILAR PEERS

In order to identify similar peers across the entire network, we use a hierarchical overlay that aggregates in a distributed way the MBRs of all peers into N_R MBRs. These MBRs describe the data available in the entire network. In the next step, the N_R MBRs are dynamically decomposed to N_{sp} groups[2], essentially one group for each super-peer. After N_{sp} groups are created, the peers whose MBRs belong to the same group are assigned to a super-peer. We emphasize that the hierarchical overlay is only used to assign similar peers to super-peers. Thereafter, MRI are constructed at super-peer level (i.e., only one level is used), as described in Section 3.1, to ensure fault-tolerance and load-balancing during query processing.

4.1 Hierarchical Aggregation of MBRs

In order to acquire an overview of the data distribution over the entire network, a super-peer hierarchy is required. Although any hierarchical overlay can be applied, we employ the DESENT hierarchy [6] to ensure scalability. The hierarchy is formed in an unsupervised, bottom-up manner. The bottom level consists of the individual peers, while the next level consists of all super-peers. In

[2]In the following instead of referring to the set of the MBRs that belong to a group, we refer to the group.

Figure 3(a), the first level of the hierarchy consists of 8 peers, while the next level of 4 super-peers.

At the same time, a *hierarchy of MBRs* is created, as depicted in Figure 3(c). Initially, each super-peer aggregates the MBRs of its peers by using at most N_R MBRs. For the sake of simplicity, in our example we set $N_R = 2$ and Figure 3(b) shows the corresponding data space. The aggregated MBRs of SP_A at level-1 are represented by m_7 and m_8. As the super-peer hierarchy is created, each super-peer assembles the MBRs of its children and creates a new aggregated set of MBRs that correspond to all the contents of the tree rooted at that super-peer. For example, SP_A maintains the aggregated MBRs m_3 and m_4 at level-2. Eventually, at the top-level super-peer, a set of N_R MBRs that span the contents of the entire network is created. In the example, SP_A becomes the super-peer of the last level, collects SP_D's MBRs and aggregates them with its own, and produces MBRs m_1 and m_2 that describe all data. Notice that the information about which MBRs of the previous level form an MBR m_i is maintained only at the super-peer responsible for m_i, and it is not necessary to be disseminated to the rest of the network.

The aggregation process results in a hierarchy of MBRs, where the top-level consists of N_R MBRs enclosing all the available data. Notice that the MBR hierarchy is distributed over different super-peers and each level of the hierarchy describes the data in the entire network at a different *level of detail*.

4.2 Dynamic MBR Decomposition Algorithm

We propose an algorithm that dynamically decomposes the MBR hierarchy to form N_{sp} groups of similar MBRs. Each group contains one or more MBRs of potentially different level of detail. The overall objective is to assign similar MBRs to the same group, while at the same time make the different groups as dissimilar as possible. This is accomplished by dynamically identifying the level of detail of MBRs that results in groups with high intra-group and low inter-group similarity. Intuitively, this can be thought of as a *cut* in the MBR hierarchy, depicted in Figure 3(c) by the grey-colored MBRs. In this example, an appropriate group assignment is: $\{m_3\}$, $\{m_9, m_{12}\}$, $\{m_{10}, m_{11}\}$ and $\{m_6\}$.

The dynamic decomposition of the N_R MBRs to N_{sp} groups is performed by the top level super-peer in the hierarchy. Notice that the lack of global knowledge of lower level MBRs makes the task of MBR assignment particularly challenging. Moreover, each time an MBR of lower level of detail is required, communication is necessary between the top level super-peer and the responsible super-peer.

Initially, the N_R top-level MBRs are given as input to our algorithm that dynamically explores the hierarchy by expanding some MBRs, until the appropriate level of detail is obtained. By expansion, we mean that an MBR m_r is replaced by the MBRs that are enclosed by m_r in the previous level. For example, if m_1 is expanded in (Figure 3(c)), then it is replaced by m_3 and m_4. There exist two reasons that require the expansion of an MBR. First, a group may be empty and all MBRs have already been assigned to other groups. Then, an MBR must be expanded, in order to ensure that all groups have at least one MBR assigned. Second, and more important, our algorithm aims to determine an appropriate cut in the MBR hierarchy, which creates high quality groups. Consider for example Figure 3(b). Let us assume that m_1 and m_2 have been expanded, so that m_3, m_4, m_5 and m_6 have been assigned to the 4 groups. Even though there exists no empty group, the quality of the groups is not sufficient. For example, by expanding m_4 and m_5 and replacing them with more detailed MBRs, groups of higher quality can be created, as demonstrated earlier. Our algorithm uses

Algorithm 1 Dynamic MBR decomposition.

```
1: Input: Top-level MBRs $\{m_1, ..., m_{N_R}\}$
2: Output: Groups $\{G_1, ..., G_{N_{sp}}\}$ of MBRs
3: $G_i \leftarrow \emptyset, \forall i \in [1...N_{sp}]$ // group initialization
4: $list \leftarrow \{m_1, ..., m_{N_R}\}$ // $list$ initialization
5: while ($list \neq \emptyset$) do
6:    if ($\exists G_i = \emptyset$) then
7:       $m_r \leftarrow list.getMostDissimilarMBR()$
8:       $G_i.add(m_r)$
9:    else
10:      $(m_r, j) \leftarrow list.getMostSimilarMBR()$
11:      $G_j.add(m_r)$
12:   end if
13:   if (($list = \emptyset$) and ($\exists G_i = \emptyset$)) then
14:      $(m_r, j) \leftarrow getMaxVolMBR()$
15:      $G_j.remove(m_r)$
16:      $list.add(expand(m_r))$
17:   end if
18:   if (($list = \emptyset$) and ($\nexists G_i = \emptyset$)) then
19:      for ($\forall m_r \in G_j, 1 \leq j \leq N_{sp}$) do
20:         if ($m_r.getCO() < m_r.getRO()$) then
21:            $G_j.remove(m_r)$
22:            $list.add(expand(m_r))$
23:         end if
24:      end for
25:   end if
26: end while
```

a heuristic quality test that ensures that when it terminates, there exist N_{sp} groups of sufficient quality, each of them containing at least one MBR, without exploring the whole hierarchy.

The pseudocode of our algorithm is presented in Algorithm 1. A $list$ is used to keep the MBRs that have not been assigned to any group yet. Initially the $list$ maintains only the N_R top-level MBRs (line 4). In each iteration, if there exists an empty group, we assign to it the MBR that is unassigned and most dissimilar to all already assigned MBRs (lines 6-9). This MBR m_r is determined by function *getMostDissimilarMBR()*, which returns the MBR with maximum enclosed volume. In the case where no empty group exists, but there exist unassigned MBRs in the $list$, these MBRs are assigned to the most suitable group (lines 9-12) based on similarity with already assigned MBRs. Function *getMostSimilarMBR()* identifies for each MBR m_r in the list, the most appropriate group based on minimizing the enclosed volume. In case the $list$ becomes empty and there exists an empty group (line 13), then it is necessary to expand an MBR (m_r) that is already assigned to a group G_j (lines 13-17). m_r is removed from G_j (line 15) and all enclosed MBRs are inserted in the $list$ (line 16). Lines 18-23 describe our quality test for the created groups, which is invoked only when the $list$ is empty and all groups have been assigned with MBRs. Finally, the algorithm terminates, when the $list$ is empty at a new repetition (line 5). This occurs if all groups have been assigned with MBRs of acceptable quality. In the following we describe the basic functionality in more detail.

Expansion of MBR, if an empty group exists (lines 13-17). This is achieved by the function *expand()* (line 16). The MBR with the largest volume is selected from all assigned MBRs, using *getMaxVolMBR()*, since this MBR is more probable to lead to overlapping groups and its expansion may reduce the enclosed dead space. Notice that replacing an MBR m_r by its enclosed MBRs requires communication between the top level super-peer and the super-peer SP that generated m_r. This communication is efficiently accomplished using the super-peer hierarchy. The cost for such a communication is bound by a number of messages equal to the height of the hierarchy ($logN_p$), which are required to contact SP.

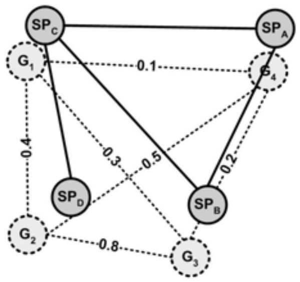


Figure 4: Graph matching example.

Expansion of MBR, if no empty group exists (lines 18-23). In this case, our algorithm employs the quality test to determine if a cut of sufficient quality in the MBR hierarchy has been found, otherwise some MBRs have to be expanded further. An MBR m_r is expanded, if its compactness is smaller than the relative overlap of m_r with m_j, where m_j any MBR that belongs to a different group. The intuition is that when this condition holds, the volume of the overlap with m_j is larger than the volume of the enclosed MBRs of m_r. Notice that the compactness provides a lower bound of the dead space in the MBR, while the relative overlap provides an estimate of the overlapping volume of two MBRs. The compactness of a bounding region is computed by function *getCO()*, whereas the overlap is computed using *getRO()* (line 20).

5. MAPPING OF MBRs TO SUPER-PEERS

Given the generated N_{sp} groups of MBRs, the remaining challenge is to assign each group to a super-peers. A naive way is to pick for each group a super-peer randomly. However, this approach can result in a situation where neighboring super-peers in the super-peer topology index dissimilar MBRs of peers. This influences the aggregated MBRs of neighboring super-peers, and hence, the selectivity of the MRI. Therefore, the aim is to assign groups to super-peers, in such a way that neighboring super-peers index peers with similar content.

In the following, we first show that the problem of group assignment to super-peers can be mapped to a *weighted graph matching problem* [30] and then we present our algorithm for mapping the MBRs to super-peers.

5.1 Problem Formulation

The problem of group assignment to super-peers can be modeled in the following way. The super-peer network topology is an undirected graph *Graph*$_1$ that consists of N_{sp} vertices. We also represent the groups of MBRs and their similarity as a graph *Graph*$_2$, also mentioned as *similarity graph*. The similarity graph is an undirected fully-connected graph with N_{sp} vertices. Each vertex corresponds to a group of MBRs and the weights on edges represent the similarity between the corresponding vertices, i.e., groups. We use as a similarity measure the enclosed volume of the MBRs that belong to the two groups. The weights are normalized in the interval $[0, 1]$ and smaller weight values indicate higher similarity.

The problem of assigning each group to a super-peer, is equivalent to matching the vertices of the network topology graph (*Graph*$_1$) to the vertices of the similarity graph (*Graph*$_2$), in such a way that the edges of *Graph*$_2$ that correspond to an edge in *Graph*$_1$ have a weight as small as possible. Consider for example Figure 4, where a super-peer topology is matched to a similarity graph. By mapping groups to super-peers as depicted in Figure 4, we ensure that similar groups are assigned to neighboring super-peers, because the sum of weights is minimized.

More formally, we further assume that *Graph*$_1$ is also weighted,

Algorithm 2 Mapping of groups to super-peers.

```
1: Input: $N_{sp}$ groups $\{G_1, ..., G_{N_{sp}}\}$
2: Output: Assignment
3: $parent \leftarrow \emptyset, nei \leftarrow \emptyset$
4: $list \leftarrow \{G_1, ..., G_{N_{sp}}\}$
5: $SP_{next} \leftarrow max_{DEG_{sp}} SP$
6: $parent.add(SP_{next})$
7: $G_i \leftarrow min_{\forall G_j \in list}(score_{deg(SP_{next})}(G_j))$
8: assign $G_i$ to $SP_{next}$
9: $list \leftarrow list - \{G_i\}$
10: while ($parents \neq \emptyset$) do
11:     $SP_i \leftarrow parents.pop()$
12:     $nei \leftarrow SP_i.getUnassignedNeighbors()$
13:     $best \leftarrow top_{nei.size()}(G_i)$
14:     for ($\forall SP_j \in nei$) do
15:         $G_j \leftarrow best.pop()$
16:         assign $G_j$ to $SP_j$
17:         $list \leftarrow list - \{G_j\}$
18:         $parent.add(SP_j)$
19:     end for
20: end while
```

where edges that correspond to the links between super-peers have a weight equal to 0, while other edges have a weight of 1. Then, the problem of group assignment to super-peers is mapped to the weighted graph matching problem.

PROBLEM 1. ***Weighted Graph Matching:*** *Let $Graph_1(V_1, E_1)$ and $Graph_2(V_2, E_2)$ be weighted undirected graphs with $|V_1| = |V_2| = n$ nodes. Also, let $w_1 : E_1 \rightarrow \Re^+$ and $w_2 : E_2 \rightarrow \Re^+$ be the weighting functions of graphs $Graph_1$ and $Graph_2$ respectively. The weighted graph matching problem is finding a correspondence f between $V_1 = \{v_1, v_2, ..., v_n\}$ and $V_2 = \{v'_1, v'_2, ..., v'_n\}$, which minimizes the difference:*

$$\sum_{i=1}^{n}\sum_{j=1}^{n}(w_1(v_i, v_j) - w_2(f(v_i), f(v_j)))^2 \qquad (4)$$

The aim is to find a 1-1 mapping between the vertices of the two graphs, while keeping the weights of the edges as similar as possible, by minimizing the sum of their squared difference. Intuitively, this conveys that similar groups are assigned to neighboring super-peers, since groups with small values of enclosed volume are assigned to super-peers with network-edges that exist (weight 0).

5.2 Mapping Algorithm

The algorithm that solves the problem of weighted graph matching has a combinatorial complexity with respect to the number of vertices (N_{sp}), which becomes prohibitively expensive for large networks. Therefore, we present an approximate algorithm of linear complexity with respect to N_{sp}. Our algorithm dynamically creates a spanning tree over the network topology and assigns the most appropriate group to each super-peer in breadth-first manner. As a root of the spanning tree, the super-peer with the maximum connectivity degree is chosen. We assume that the super-peer with highest connectivity degree is more important and the reason is twofold. First, a super-peer with many connections influences the performance of the MRI more, since it propagates aggregated MBRs to more super-peers. Secondly, it is less probable to find a group that is similar to many other groups, and it gets more difficult if some groups are already assigned, as they impose constraints to new assignments. After assigning an appropriate group to the first super-peer, the next subset of super-peers is its neighbors; priority is based on super-peer connectivity degree, with well-connected super-peers being assigned with groups first. This procedure is repeated until all groups are assigned to a different super-peer.

Group selection is based on the similarity to the group already assigned to the neighboring super-peer. If G_i denotes the already assigned group to SP_i, we assign to a neighboring super-peer SP_j the group G_j that minimizes the enclosed volume $\hat{V}(G_i, G_j)$. Given a group G_i, a ranking of a set of groups $\{G_j\}$ can be defined based on the enclosed volume with respect to G_i. We denote as $top_k(G_i)$ the k groups G_j that have the minimum values $\hat{V}(G_i, G_j)$. Then, the following formula defines a score for each group G_i based on its k most similar groups.

$$score_k(G_i) = \sum_{\forall G_j \in top_k} \hat{V}(G_i, G_j) \qquad (5)$$

The pseudocode describing the algorithm is presented as Algorithm 2. First, the super-peer SP_{next} with the highest connectivity degree $deg(SP_{next})$ is selected (line 5). Since there is no group assigned to SP_{next}, we pick the group G_i that has the highest similarity with k=$deg(SP_{next})$ unassigned groups, i.e., minimum $score_k(G_j)$ value (line 7). Notice that we take into account only the k=$deg(SP_{next})$ most similar groups, because each group of MBRs has high similarity to some groups and low similarity to the remaining groups. Then, SP_{next} is added to the $parent$ list (line 6) which stores the super-peers for which their unassigned neighboring super-peers will be processed in the next iterations. Then, until the parents list is empty (line 10), the first element is examined (line 11) and a group is assigned to each neighbor (lines 14-19). The most similar groups to SP_i are retrieved (line 13) and assigned to the neighbors (lines 15-16). Each neighbor is added to the parent list (line 18). Thus, the algorithm assigns groups to super-peers by visiting them in a breadth-first way, starting from SP_{next}. In the end, each of the N_{sp} groups has been assigned to a super-peer.

Notice that the algorithm requires knowledge of the super-peer topology only at runtime. As super-peers are usually a couple orders of magnitude fewer than peers (i.e., the size of the P2P network), this assumption is tolerated. Moreover, notice that peer to super-peer reassignment may increase their real geographic distance. This is the basic idea behind overlay networks, which create overlay links between peers based on content similarity. It is generally established that the performance of query processing improves by using such overlay links (as for example in CAN [28] or VBI-tree [21]), compared to the case of links reflecting geographic distance but no useful information about content location.

6. MAINTENANCE

Data and MBR Updates. Updates, insertions and deletions of data stored at a peer may alter the set of MBRs that encloses the peer's data. As long as the data distribution of the peer's data does not change, the MBRs will not change by each single data update. Only if some of the peer's MBRs changes, the corresponding super-peer has to be informed. Then, this super-peer has to update its locally stored peer MBRs and the set of aggregated MBRs that enclose the data of all peers. Notice that this does not necessary lead to an update of the MRI. Assuming that a peer's MBR changes slightly, it is quite probable that the super-peer's MBR that summarizes the peer data does not change, since other peers may store similar data. Only when the set of aggregated MBRs of a super-peer changes, this leads to an *MBR update*[3], which influences the MRI. Thus, MBR updates occur less frequently than data updates.

The case of an MBR update at a super-peer is the most generic maintenance operation, as it can be triggered by peer data updates, peer joins and failures. When an MBR increases in size, this change must be immediately communicated to other super-peers, otherwise a query intersecting only with the increased region will not be routed to the updated super-peer. When an MBR decreases in size, this change can be communicated with some delay, as in the meantime the updated super-peer may be queried in vain, but the correctness of the result is still guaranteed. In a similar way, if a super-peer SP_N receives an updated set of neighboring MBRs, then SP_N updates its local MRI and informs its neighboring super-peers about the MBR update.

Although data and MBR updates can be efficiently handled using the aforementioned protocols, it is possible that high rates of churn may result in MRI of poor quality. However, using the metrics defined and used throughout this paper, a super-peer can individually decide when the quality of its local MBRs has degraded significantly. For example, by observing the average compactness of its local MBRs, a super-peer can decide that their quality is no longer acceptable. Therefore, when most of the super-peers identify such a situation, the routing indices construction is triggered anew, in order to generate MRI of higher selectivity. Even though the construction could run periodically regardless of churn rate, the proposed approach is more appropriate, because it is on demand and it can self-adapt to the peer dynamics in the system.

Peer joins. When a new peer P_J joins the system, P_J provides a summary description of its data by means of a set of k MBRs. Obviously the value of k is not fixed and it varies for different peers. Initially, P_J connects to a super-peer SP_J at random, using the basic bootstrapping protocol. Then, SP_J undertakes the task to discover the most relevant super-peers that P_J should connect to. In order to achieve this goal, the MRI are exploited.

For each m_i ($1 \leq i \leq k$) of P_J's MBRs, SP_J executes the following procedure. SP_J computes the minimum enlargement of volume of its aggregated MBRs caused by m_i. This value serves as a measure of goodness for accommodating m_i. Then, SP_J uses its MRI to compute the value of enlargement for each neighboring super-peer, and forwards m_i only to those neighbors that have a smaller value of enlargement based on the MRI. Any recipient super-peer SP_R computes the value of enlargement and checks if m_i needs to be forwarded further. Eventually, any recipient super-peer SP_R sends back to SP_J its locally computed value of enlargement and SP_J assigns m_i to the SP_R with smallest value. The objective is to assign m_i to that super-peer, which can accommodate m_i with minor changes to its MBRs. Using this peer joining protocol, P_J establishes up to a maximum k intentional connections to other super-peers in the network. For each super-peer SP_R that is assigned with P_J due to one of its MBRs, it is possible that SP_R needs to update its MBRs, in order to enclose also the data of P_J.

Fault-tolerance. Regarding peer failures, each super-peer SP_i can easily and timely detect the failure of a peer P_F, by sending periodically ping messages to all peers connected to SP_i. Since P_F was connected to SP_i, we assume without loss of generality that there exists (at least) one MBR of SP_i that encloses an MBR of P_F. When the failure is detected, SP_i deletes the peer's MBRs from its local index, and updates its list of aggregated MBRs that contained P_F's MBR. Finally, SP_i checks if an MBR update occurred and if necessary the neighbors are informed as already described.

Churn of super-peers is infrequent, however it can be efficiently handled capitalizing on the methods for MBR updates. When a super-peer failure occurs, peers have to reconnect to the network using the bootstrapping protocol. Whenever neighboring super-peers detect a super-peer failure, the corresponding MBRs are deleted from the MRI and their neighbors are informed. When a super-peer joins the network, it connects to DEG_{sp} other super-peers, and

[3]The term *MBR update* refers to an update of an aggregated MBR.

peers connect to this super-peer through the peer join procedure. In this way, the super-peer builds its aggregated MBRs and exchanges MBRs with its neighbors.

7. EXPERIMENTAL STUDY

We evaluate the selectivity of MRI using large-scale simulations, with a simulator prototype implemented in Java. We used the GT-ITM topology generator[4] to create well-connected random graphs of N_{sp} (200-600) super-peers with average connectivity DEG_{sp} (4-7). Unless mentioned explicitly, we use DEG_p=10 for the initial P2P topology, thus obtaining the number of peers N_p (2000-6000). We conduct experiments varying the dimensionality d (2-6) and the cardinality n (1-3M) of the dataset. We keep the number of objects per peer (n/N_p) constant and equal to 500.

In order to evaluate the scalability of our approach, we experimented with synthetic clustered data collections, partitioned evenly among the peers. For the clustered dataset generation, we randomly pick N_{sp} d-dimensional points and each peer obtains k distinct centroids from them at random. Obviously, two peers may share the same centroid. Thereafter, each peers' object is generated inside a radius r from one of the peer's centroids. The radius r is selected in a way that the total volume of the data capture $v\%$ of the data space. Although we use as default value v=10%, we study the effect of varying v values in our experiments. Peer data are represented by spherical MBRs, and they are determined using the k-means clustering algorithm. Notice that the choice of the clustering algorithm is orthogonal to our framework and other techniques can be employed. MBR aggregation at any super-peer is also performed using the k-means algorithm on the MBRs centroids. Naturally, we study the performance of MRI for varying values of k.

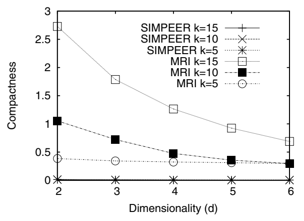
To study the performance of query processing on top of the MRI, we employ the SIMPEER [7] framework, which is the state-of-the-art approach for multidimensional query processing over a super-peer architecture. For comparative purposes, we compare against SIMPEER that assumes random peer to super-peer assignment. The comparison against SIMPEER helps to quantify the improvement of the MRI in terms of selectivity achieved by our approach. Furthermore, we evaluate the performance of routing indices constructed by our approach (MRI) to indices constructed by an OPTIMAL peer to super-peer assignment, which provides the best performance that any algorithm can achieve.

7.1 Selectivity of MRI

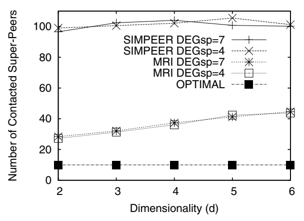
Our first objective is to evaluate the selectivity of MRI, which directly depends on the quality of the groups (or equivalently MBRs) that have been assigned to the super-peers, as their dead space and overlap determine the selectivity of the MRI. Our default setup consists of a network of 2000 peers, 200 super-peers, 1M data objects, and we set the k-means parameter to k=10. All experiments were repeated using 10 different synthetically generated datasets and the average number of the measurements is depicted in all cases.

In Figure 5(a), we study the compactness with respect to the data dimensionality, in particular by increasing the dimensionality from d=2 to d=6. We compare the compactness of the groups assigned to super-peers using MRI to SIMPEER, for varying number k of MBRs. Recall that a value of compactness smaller than 1 means that definitely dead space is enclosed, while for values larger than 1 the probability of existing dead space decreases. Clearly, the groups created by SIMPEER have compactness close to zero for all tested values of k. In contrast, MRI manages to create groups of high quality, especially with increasing k. This is because higher

[4]Available at: http://www.cc.gatech.edu/projects/gtitm/



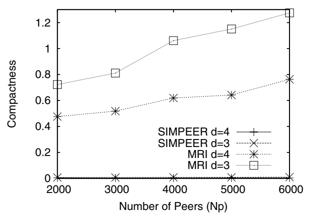
(a) Compactness vs. d.



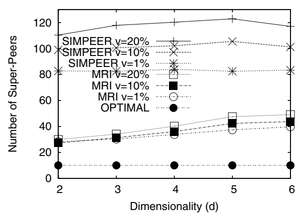
(b) Comparison to OPTIMAL.

Figure 5: Selectivity of MRI for varying d.

number of MBRs enables more detailed description of data, hence MBRs are aggregated in a more efficient way. Notice that the decrease of compactness with increasing dimensionality is expected, however its values remain high and 1 to 2 orders of magnitude better than SIMPEER in all cases.



(a) Compactness vs. N_p.



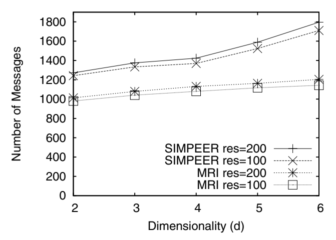
(b) Effect of dataset sparseness.

Figure 6: Selectivity of MRI for varying N_p and v.

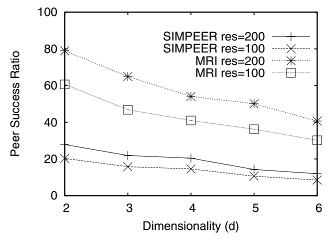
To evaluate the routing ability of the MBRs, we select randomly 20 MBRs out of the original peer MBRs, and consider them as range queries that follow the data distribution. We measure the average number of groups that contain an overlapping MBR with the queries or, equivalently, the minimum number of super-peers that have to be contacted during query evaluation. We fix the value of k to 10 and vary the super-peer network topology, from sparse (DEG_{sp}=4) to dense (DEG_{sp}=7) in Figure 5(b). We conclude that the topological characteristics only slightly affect the number of super-peers that is necessary to be contacted. This verifies the high quality and routing ability of MRI, regardless of the topology. Moreover, based on the way data is generated on each peer, we can define an OPTIMAL case where all peers that obtained a particular centroid connect to the minimum possible number of super-peers. Since the number of such peers is 100 and each super-peer indexes DEG_p=10 peers, the OPTIMAL number of super-peers is 10. We observe that MRI manages to achieve performance much closer to OPTIMAL than SIMPEER.

In Figure 6(a), we depict only dimensionality 3 and 4 and we increase the network size N_p by scaling the number of peers up to 6000. The number of super-peers N_{sp} is kept to 200. Thus, each super-peer is assigned with DEG_p=10 to 30 peers. With respect to compactness, MRI outperforms SIMPEER for all network sizes. Moreover, the compactness increases with network size, since each super-peer is initially connected to more peers, hence more MBRs are collected and their aggregation is of higher quality. In addition, we increased N_{sp} up to 600 super-peers, and we observed that MRI always outperforms SIMPEER in terms of number of contacted super-peers (chart omitted due to space limitations). More importantly this gain is maintained as the number of super-peers increases, clearly demonstrating the scalability of MRI.

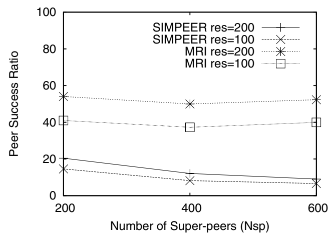
The effect of the percentage v of data space volume captured by the dataset is studied in Figure 6(b). We test v=1%, v=10% and v=20%, where increasing values resemble the more uniform data distribution. In all cases, MRI performs gracefully compared to



(a) Number of messages for search.



(b) Success ratio vs. d.



(c) Success ratio vs. N_{sp}.

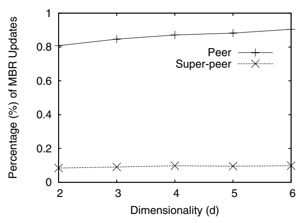
Figure 7: Study of query processing performance for multidimensional routing indices.

OPTIMAL, and MRI is more stable and less affected by increasing v values, compared to SIMPEER.

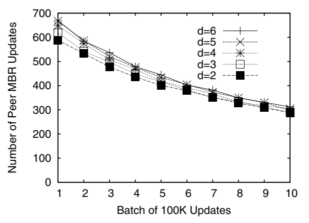
7.2 Query Processing Performance

In the following, we verify that the gains in the quality of MRI are also reflected in query processing performance. We compare the performance of routing indices constructed using our algorithms (MRI) to the indices of SIMPEER. We generate random range queries of varying selectivity (number of results) that follow the distribution of data. We measure the comparative performance of both approaches in terms of average number of messages for searching and peer success ratio, i.e., how many of the contacted peers returned results. Figure 7 reports the experimental results. Although several different setups were tested, we only show the most important findings, due to lack of space. As an example we show results of range queries with selectivity (res) equal to 100 and 200 objects.

In Figure 7(a), the average number of messages used for query processing is shown. MRI requires fewer messages than SIMPEER to retrieve the complete result set and, most importantly, this gain increases with d. This verifies the robustness and superiority of MRI, in terms of query processing performance, irrespective of dimensionality. We also measure the peer success ratio, shown in Figure 7(b). Despite the expected drop of success ratio with increased dimensionality, MRI is always better than SIMPEER and MRI achieves acceptable values even for d=6. This result demonstrates the efficiency of routing using MRI. In order to ensure the validity of this result for larger networks, we increase the number of super-peers (N_{sp}) from 200 to 600, while the number of peers also increases to $10 \times N_{sp}$. The chart in Figure 7(c) shows that MRI maintains a stable peer success ratio when the network size increases, whereas SIMPEER presents a downward tendency. Hence, MRI is a scalable solution when the network size increases.



(a) (%) of MBR updates.



(b) MBR updates over time.

Figure 8: Effect of data updates on peers.

7.3 Maintenance of MRI

In Figure 8, we study the effect of peer data updates on the MRI. We update all 500 data points at each peer (in total 1M data points are updated), modeled as deletions and insertions, and we measure the percentage of data updates that caused an update at peer and super-peer MBRs for varying dimensionality (Figure 8(a)), as well as the number of peer MBR updates as time elapses (Figure 8(b)). The updates follow the data distribution of the generated data. In Figure 8(a), the results show that fewer than 1% of data updates lead to peer MBR updates, while even fewer than 0.1% of data updates lead to super-peer MBR updates. Therefore, only 0.1% of the updates on the peers cause an update of the MRI. In Figure 8(b), we depict how many peer MBRs are updated per batch of 100K data point updates. The decreasing tendency shows that the effect of data updates on peers diminishes as time elapses.

8. RELATED WORK

Routing indices [5] have been originally proposed in the context of unstructured P2P systems, as a search-enabling technique that provides a direction to routing, instead of blind forwarding. Although the original context was best-effort document retrieval, several research papers adopt routing indices, including super-peer networks [26], semantic routing indices [25] and routing indices based on histograms [27]. Unfortunately, traditional routing indices are not applicable for multidimensional data that require more advanced query types, such as similarity search, and when the aim is retrieval of the exact and complete result set.

Distributed data summaries as routing indices. The most relevant works to ours are those that assume autonomous data storage by peers and build distributed data summaries for routing. Hose et al. use a tree-based structure, named QTree [19], for summarizing data and query routing. Recently in [20], the maintenance of routing indices is addressed and the authors discuss requirements that need to be fulfilled by such routing structures. In contrast, our approach focuses more on improving the selectivity of the MRI by dynamic peer reallocation. The use of P2P data summaries has also been proposed in [18]. SIMPEER [7, 8] supports exact query processing over multidimensional data distributed in a super-peer network. Sharing similar goals to this paper, namely efficient routing of similarity search queries, SIMPEER relies on routing indices, but ignores the factors that influence their selectivity.

Scalable distributed data structures (SDDS). Distributed multidimensional indices, such as distributed R-trees, have been proposed in the literature. SD-Rtree [11] is a distributed balanced binary spatial tree that is used to store and query spatial objects. Each node has knowledge of all other nodes in the system and this constitutes a major difference to P2P systems, where a (super-)peer is only aware of a limited set of neighboring (super-)peers.

P2P multidimensional indexing based on space partitioning. Content addressable network (CAN) [28] was the first approach for P2P multidimensional indexing. Although it is based on hashing, CAN is similar to a dynamic multidimensional grid on the data space. Space partitioning, based on distributed tree structures, has also been considered. In [14], the authors identify two primary components for multidimensional query processing, namely partitioning and routing. They propose an adaptation of the kd-tree for partitioning and use of skip pointers for routing. P-ring [4] is a

P2P index that assigns ranges of the search space to peers. VBI-Tree [21] is a framework for multidimensional indexing in P2P networks. Peers are organized in a balanced tree structure based on the space partitions assigned to them. Our techniques differ significantly from such space partitioning approaches [4, 14, 21], because each peer stores its own data autonomously, in contrast to predefined assignment of regions of the data space to peers and deliberate data placement on peers. Moreover, P2P multidimensional indexing approaches suffer from load balancing issues, which eventually leads to the additional cost of restructuring the network.

Dynamic topology adaptation has been thoroughly studied in unstructured P2P systems [1, 2, 3], where the aim is for peers to create links to peers with similar content, thus improving the performance of routing. Nevertheless, in super-peer architectures, intentional peer to super-peer assignment has only recently attracted the attention of the research community. A self-organizing super-peer network architecture named SOSPNET, is presented in [15], and deals with the issue of how clients connect to a super-peer. Although relevant to our approach, the main difference is that SOSPNET organizes peers according to requests for files, while our approach is data-centric and focuses on data distribution. Moreover our approach can handle more complex data types than files and process complex queries. In [9], self-organization of peers based on content similarity has been shown to improve the performance of search, thereby motivating the appropriateness of building selective multidimensional routing indices as described in this paper.

9. CONCLUSIONS

In this paper, we addressed the challenging problem of constructing multidimensional routing indices (MRI) of high selectivity. Focusing on a super-peer architecture, we identified an important factor that affects the performance of query routing, namely the enlargement of the MBRs that describe the data available through each neighboring super-peer. Our proposed approach improves the selectivity of the MRI by assigning peers with similar content to the same super-peer in a self-organizing manner. Moreover, neighboring super-peers index similar content which improves the routing process even further. The experimental results show that MRI significantly improve a state-of-the-art approach for P2P similarity search, by boosting the performance of query routing.

10. REFERENCES

[1] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making Gnutella-like P2P systems scalable. In *Proc. of SIGCOMM*, pages 407–418, 2003.

[2] V. Cholvi, P. Felber, and E. Biersack. Efficient search in unstructured peer-to-peer networks. In *Proc. of SPAA*, pages 271–272, 2004.

[3] E. Cohen, A. Fiat, and H. Kaplan. Associative search in peer-to-peer networks: Harnessing latent semantics. *Computer Networks*, 51(8):1861–1881, 2007.

[4] A. Crainiceanu, P. Linga, A. Machanavajjhala, J. Gehrke, and J. Shanmugasundaram. P-ring: An efficient and robust P2P range index structure. In *Proc. of SIGMOD*, pages 223–234, 2007.

[5] A. Crespo and H. Garcia-Molina. Routing indices for peer-to-peer systems. In *Proc. of ICDCS*, page 23, 2002.

[6] C. Doulkeridis, K. Nørvåg, and M. Vazirgiannis. DESENT: Decentralized and distributed semantic overlay generation in P2P networks. *IEEE Journal on Selected Areas in Communications (J-SAC)*, 25(1):25–34, 2007.

[7] C. Doulkeridis, A. Vlachou, Y. Kotidis, and M. Vazirgiannis. Peer-to-peer similarity search in metric spaces. In *Proc. of VLDB*, pages 986–997, 2007.

[8] C. Doulkeridis, A. Vlachou, Y. Kotidis, and M. Vazirgiannis. Efficient range query processing in metric spaces over highly distributed data. *Distributed and Parallel Databases*, 26(2-3):155–180, 2009.

[9] C. Doulkeridis, A. Vlachou, K. Nørvåg, Y. Kotidis, and M. Vazirgiannis. Efficient search based on content similarity over self-organizing P2P networks. *Peer-to-Peer Networking and Applications*, 2009.

[10] C. Doulkeridis, A. Vlachou, K. Nørvåg, Y. Kotidis, and M. Vazirgiannis. Multidimensional routing indices for efficient distributed query processing. In *Proc. of CIKM*, pages 1489–1492, 2009.

[11] C. du Mouza, W. Litwin, and P. Rigaux. SD-Rtree: A scalable distributed R-tree. In *Proc. of ICDE*, pages 296–305, 2007.

[12] F. Falchi, C. Gennaro, F. Rabitti, and P. Zezula. Distance browsing in distributed multimedia databases. *Future Generation Comp. Syst.*, 25(1):64–76, 2009.

[13] V. Gaede and O. Günther. Multidimensional access methods. *ACM Comput. Surv.*, 30(2):170–231, 1998.

[14] P. Ganesan, B. Yang, and H. Garcia-Molina. One torus to rule them all: Multidimensional queries in P2P systems. In *Proc. of WebDB*, pages 19–24, 2004.

[15] P. Garbacki, D. H. J. Epema, and M. van Steen. Optimizing peer relationships in a super-peer network. In *Proc. of ICDCS*, page 31, 2007.

[16] A. Guttman. R-Trees: A dynamic index structure for spatial searching. In *Proc. of SIGMOD*, pages 47–57, 1984.

[17] J. Han, M. Kamber, and A. K. H. Tung. Spatial clustering methods in data mining: A survey. *Geographic Data Mining and Knowledge Discovery*, pages 1–29, 2001.

[18] R. Hayek, G. Raschia, P. Valduriez, and N. Mouaddib. Summary management in P2P systems. In *Proc. of EDBT*, pages 16–25, 2008.

[19] K. Hose, C. Lemke, and K.-U. Sattler. Processing relaxed skylines in PDMS using distributed data summaries. In *Proc. of CIKM*, pages 425–434, 2006.

[20] K. Hose, C. Lemke, and K.-U. Sattler. Maintenance strategies for routing indexes. *Distributed and Parallel Databases*, 26(2-3):231–259, 2009.

[21] H. V. Jagadish, B. C. Ooi, Q. H. Vu, R. Zhang, and A. Zhou. VBI-tree: A peer-to-peer framework for supporting multi-dimensional indexing schemes. In *Proc. of ICDE*, page 34, 2006.

[22] A. Kementsietsidis, F. Neven, D. Van de Craen, and S. Vansummeren. Scalable multi-query optimization for exploratory queries over federated scientific databases. *Proc. VLDB Endow.*, 1(1):16–27, 2008.

[23] S. T. Leutenegger, J. M. Edgington, and M. A. Lopez. STR: A simple and efficient algorithm for R-tree packing. In *Proc. of ICDE*, pages 497–506, 1997.

[24] T. Malik, A. S. Szalay, T. Budavari, and A. Thakar. SkyQuery: A web service approach to federate databases. In *Proceedings of CIDR'2003*, 2003.

[25] F. Mandreoli, R. Martoglia, S. Sassatelli, and W. Penzo. SRI: exploiting semantic information for effective query routing in a PDMS. In *Proc. of WIDM*, pages 19–26, 2006.

[26] W. Nejdl et al. Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks. In *Proc. of WWW*, pages 536–543, 2003.

[27] Y. Petrakis, G. Koloniari, and E. Pitoura. On using histograms as routing indexes in peer-to-peer systems. In *Proc. of DBISP2P*, pages 16–30, 2004.

[28] S. Ratnasamy, P. Francis, M. Handley, R. M. Karp, and S. Shenker. A scalable content-addressable network. In *Proc. of SIGCOMM*, pages 161–172, 2001.

[29] N. Roussopoulos and D. Leifker. Direct spatial search on pictorial databases using packed R-Trees. In *Proc. of SIGMOD*, pages 17–31, 1985.

[30] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(5):695–703, 1988.

[31] A. Vlachou, C. Doulkeridis, K. Nørvåg, and M. Vazirgiannis. On efficient top-k query processing in highly distributed environments. In *Proc. of SIGMOD*, pages 753–764, 2008.

[32] B. Yang and H. Garcia-Molina. Designing a super-peer network. In *Proc. of ICDE*, page 49, 2003.