Resource Description and Selection for Range Query Processing in General Metric Spaces

Daniel Blank, Andreas Henrich

Media Informatics Group
University of Bamberg
D-96045 Bamberg
{daniel.blank|andreas.henrich}@uni-bamberg.de

Abstract: Similarity search in general metric spaces is a key aspect in many application fields. Metric space indexing provides a flexible indexing paradigm and is solely based on the use of a distance metric. No assumption is made about the representation of the database objects.

Nowadays, ever-increasing data volumes require large-scale distributed retrieval architectures. Here, local and global indexing schemes are distinguished. In the local indexing approach, every resource administers a set of documents and indexes them locally. Resource descriptions providing the basis for resource selection can be disseminated to avoid all resources being contacted when answering a query. On the other hand, global indexing schemes are based on a single index which is distributed so that every resource is responsible for a certain part of the index.

For local indexing, only few exact approaches have been proposed which support general metric space indexing. In this paper, we introduce RS4MI—an exact resource selection approach for general metric space indexing. We compare RS4MI with approaches presented in literature based on a peer-to-peer scenario when searching for similar images by image content. RS4MI can outperform two exact general metric space resource selection schemes in case of range queries. Fewer resources are contacted by RS4MI with—at the same time—more space efficient resource descriptions.

1 Introduction

The efficient processing of similarity queries (e.g. range queries searching for all database objects within a given search radius from the query object) is a key aspect in many domains and application fields such as multimedia and 3D object retrieval, similarity search on business process models, data compression, pattern recognition, machine learning, bioinformatics, statistical data analysis, malware detection, and data mining [ZADB05, HCS09, KW11, BKSS07]. Hereby, many similarity search problems are modeled in general metric spaces where no assumption is made about the representation of the database/feature objects. The only assumption is that distances between feature objects can be measured by a distance metric.

Furthermore, in many search scenarios, centralized architectures are no longer sufficient and large-scale solutions are necessary. Here, as a particular technique for distributed

query processing, resource selection techniques provide a valuable solution. They are for example applicable in dynamic environments such as peer-to-peer (P2P) information retrieval (IR) systems with data sources joining and leaving frequently.

In the P2P IR domain, it can become infeasible to solely apply *global indexing schemes*, i.e. distributed indexing structures with every peer being responsible for a certain range of the feature space and peers transferring their indexing data to remote peers according to their "region(s) of interest". Peers entering the system and updating indexing data might induce a high network load the system can hardly cope with [LLOS07].

Summary-based resource selection approaches and thus *local indexing schemes* such as the ones discussed and evaluated in this work are one possibility to deal with this problem. Here, every peer indexes the data it administers and describes it in form of data summaries which are transfered to remote peers. During search, promising peers are selected based on the resource descriptions and the query is sent to them. In summary-based P2P IR systems, peers leaving the network ungracefully do not take indexing data of other peers' documents with them. Furthermore, leaving peers do not take documents with them for which indexing data is still present in the network and the documents thus still can be found. Peer autonomy is better respected compared to distributed index structures. On the other hand, many distributed index structures offer query processing with logarithmic cost [DVNV10] which is hard to guarantee for local indexing schemes.

The work in this paper focuses on space efficient resource description and corresponding selection techniques which allow for efficient distributed query processing in general metric spaces. As a proof-of-concept and application scenario being assumed, the resource description and selection techniques are designed for the use within a particular P2P IR scenario. However, they can also be applied for traditional resource selection in distributed IR and within other variants of P2P IR systems. Furthermore, there is a range of possible application fields beyond P2P IR systems, such as (visual) sensor [ERO+09] and ad-hoc networks [LLOS07], to name only a few.

As the contribution of this paper, we present RS4MI (Resource Selection for Metric Indexing), a new exact resource description and selection technique applicable for similarity search in general metric spaces. Its design is motivated by application scenarios where space efficient resource descriptions are required. As a rule of thumb, the average size of a resource description in our scenario should be below 1 kB. However, the presented techniques are by no means limited to this scenario. We review related work in the field of resource selection and identify techniques applicable in general metric spaces. The only exact technique presented in literature so far is compared against RS4MI. In addition, another baseline technique relying on local k-medoid clustering is included in the analysis.

The remainder of this paper is organized as follows. In Sect. 2, we briefly recapitulate main concepts of similarity search in metric space. Sect. 3 discusses further related work by presenting existing solutions to the resource description and selection problem in general metric spaces. RS4MI and the two competing approaches are in detail outlined and evaluated in Sect. 4. The paper concludes with an outlook on future work in Sect. 5.

2 Metric Space Indexing

Multi-dimensional (spatial) access methods (SAMs; for an overview cf. [Sam06]) are designed for vector spaces whereas metric access methods (MAMs) can be applied in any metric space. An overview on MAMs is for example given in [CNBYM01, ZADB05]. A metric space $\mathcal{M}$ is defined as a pair $\mathcal{M} = (\mathbb{D}, d)$. $\mathbb{D}$ represents the domain of objects $o \in O$ with $O \subset \mathbb{D}$ and $d : \mathbb{D} \times \mathbb{D} \to \mathbb{R}$ corresponds to a metric distance function which satisfies the metric postulates $\forall x, y, z \in \mathbb{D}$ [ZADB05]:

$$
\begin{aligned}
&d(x,y) = 0 \iff x = y && \textit{identity}\\
&d(x,y) > 0 \iff x \neq y && \textit{non-negativity}\\
&d(x,y) = d(y,x) && \textit{symmetry}\\
&d(x,y) + d(y,z) \geq d(x,z) && \textit{triangle inequality}
\end{aligned}
$$

Many MAMs rely on a set $C = \{c_i | 1 \leq i \leq n\}$ of reference objects (also called pivots or centers) in order to structure the feature space. There are different ways of how to partition the feature space. Within *ball partitioning* methods [ZADB05], the feature space is partitioned by often multiple hyper-spheres. In contrast, many structures relying on *hyperplane partitioning* conceptually rely on a list L_o, ordering the pivot IDs i by increasing $d(c_i, o)$. In case of *generalized hyperplane partitioning* [ZADB05], o is assigned to the cluster (i.e. a region of the feature space induced by the space partitioning) with ID $L_o[1]$ of the closest reference object $c^* = \arg\min_{c_i \in C} d(c_i, o)$. In other cases, the list L_o truncated after position l with $l \in \{2, \ldots, n\}$ identifies the cluster where o lies in (cf. [NBZ11]).

The distance between feature objects is frequently used to model the similarity between them. Usually, it is assumed that the smaller the distance the higher the similarity. In this context, range queries are a popular type of similarity queries [Sko06, p. 4].

A *range query* $R(q, r)$ with query object $q \in \mathbb{D}$ and search radius $r \in \mathbb{R}^+$ retrieves all database objects from $O \subset \mathbb{D}$ which are within distance r from q, i.e. $\{o \in O \,|\, d(q, o) \leq r\}$. The subspace $\mathbb{V} \subset \mathbb{D}$ for which $\forall v \in \mathbb{V} : d(q, v) \leq r$ and $\forall v' \in \mathbb{D} \backslash \mathbb{V} : d(q, v') > r$ is called the *query ball* [SB11].

For the space partitioning methods outlined above as well as hybrid combinations, various pruning criteria can be applied. They are in the following described in the context of range queries following the notation of [ZADB05].

Pruning criteria in metric spaces

When only *per-cluster* information (in contrast to *per-object* information) is stored in the resource descriptions, range query processing can be summarized as follows. The data descriptions of the resources are iteratively analyzed. If all populated database clusters of a resource can be pruned, i.e. no populated cluster intersects the query ball, the very resource can be discarded from search. Remaining resources have to be contacted. Criteria capable of cluster and hence resource pruning are outlined in the following, similarly to [NBZ11].

If a query lies in the cell of center c^* (i.e. reference object c^* is the closest center out of the set C of all available reference objects according to a given query object q), by exploiting the triangle inequality, any cluster $[c_i]$ can be pruned if $d(c_i, q) - d(c^*, q) > 2r$, where r corresponds to the search radius (*double-pivot distance constraint*).

If a maximum cluster radius r_i^{max} for a cluster $[c_i]$ is given, i.e. the maximum distance of any object o in the cluster from its center c_i, the very cluster can be pruned if $d(c_i, q) - r > r_i^{max}$ (*range-pivot distance constraint*). A similar condition can be applied according to the minimum cluster radius r_i^{min}, i.e. the minimum distance of any object o within the cluster from its center c_i. Cluster $[c_i]$ can be pruned if $d(c_i, q) + r < r_i^{min}$.

The range-pivot distance constraint can also be used in an inter-cluster way. To this end, two matrices *MAX* and *MIN* are applied to store maximum and minimum cluster radii $r_{i,j}^{max}$ and $r_{i,j}^{min}$ respectively for $i, j \in \{1, \ldots, n\}$, where $r_{i,j}^{max}$ represents the maximum distance of any object from cluster $[c_i]$ to cluster center c_j, and $r_{i,j}^{min}$ represents the minimum distance of any object from cluster $[c_i]$ to cluster center c_j. Elements $r_{i,i}^{max}$ and $r_{i,i}^{min}$ on the diagonal of the matrices *MAX* and *MIN* thus capture the maximum cluster radius r_i^{max} and the minimum cluster radius r_i^{min} of cluster $[c_i]$, respectively, as described above. Cluster $[c_i]$ can be pruned if there exists a cluster $[c_j]$ for which $d(c_j, q) + r < r_{i,j}^{min}$ or $d(c_j, q) - r > r_{i,j}^{max}$ [Woj02].

Fig. 1 visualizes a search situation in case of a range query with search radius r where cluster $[c_1]$ can be pruned successfully. By solely using the double-pivot distance constraint, cluster $[c_1]$ cannot be pruned, since the query ball $\mathbb{V}$ intersects cluster $[c_1]$. If, for every cluster, we administer only the minimum and the maximum cluster radius of objects in the cluster (shown by the hyper-ring $\mathbb{H}_{1,1}$ around cluster center c_1 in Fig. 1), cluster $[c_1]$ can still not be pruned. The matrices *MIN* and *MAX* are thus necessary to successfully prune cluster $[c_1]$. If we also apply the radii $r_{1,2}^{min}$ and $r_{1,2}^{max}$, i.e. the minimum and the maximum distance of feature objects in cluster $[c_1]$ from c_2, it can be determined that there are no relevant feature objects in the intersection area of the query ball $\mathbb{V}$ and the hyper-ring $\mathbb{H}_{1,1}$. The region of possible feature objects is limited to the two dark gray shaded intersection areas of $\mathbb{H}_{1,1}$ and $\mathbb{H}_{1,2}$, and since the query ball $\mathbb{V}$ does not intersect any of these regions, cluster $[c_1]$ does not contain any database objects relevant to the query.

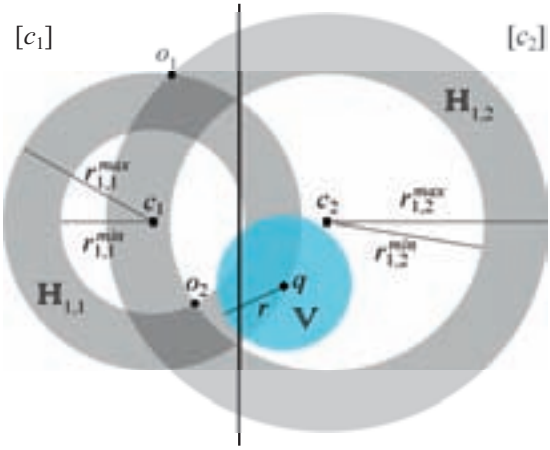

Figure 1: Cluster pruning example.

A further pruning constraint can be applied on an object level rather than a cluster level. The application of this constraint in a resource selection scenario requires per-object in-

formation to be stored in the resource descriptions—either solely or in addition to storing per-cluster information. Contacting a resource can be avoided by storing distance values $d(c_i, o)$ in the summaries. If $|d(c_i, q) - d(c_i, o)| > r$, object o can be pruned without computing $d(q, o)$. This is called the *object-pivot distance constraint*.

In order to enhance the pruning power for complete resources, $d(c_i, o)$ values can be stored for multiple cluster centers c_i. Hence, contacting a resource is not necessary if $\max_{c_i} |d(c_i, q) - d(c_i, o)| > r$ is fulfilled for all database objects of a resource. This so called *pivot filtering* is a direct application of the object-pivot distance constraint. Although appearing impracticable at first glance due to the space requirements, per-object information might be useful in hybrid approaches for peers with few objects. This will be considered in Sect. 4.4.

3 Related Work on Resource Selection in General Metric Spaces

There is plenty of work on the description and selection of text databases in distributed IR (cf. [SS11]). In addition, some resource description and selection schemes have emerged in the context of content-based multimedia IR such as in content-based image retrieval (CBIR). Our work addresses *local indexing approaches* and in particular the ones which consider the resource selection task as a *geometric problem*. Here, certain properties of the feature space or distance information are used in order to prune resources which cannot contribute database objects to the search result. The remaining resources can be ranked by the "proximity" of their feature objects and the query object (which can be beneficial when e.g. performing k-nearest neighbor (k-NN) queries). We will discuss these approaches in the following. Probabilistic (cf. e.g. [NF03, EBMH08]) as well as geometric resource selection techniques only applicable in vector spaces (cf. e.g. [KLC02]) are out of the scope of our present work. We also do not consider database selection approaches based on one-dimensional numeric values (e.g. [YSMQ01]).

In the following, two approaches applicable in general metric spaces are described. The approach by Berretti et al. [BDP04] is the only approach which represents an exact resource selection scheme, the latter approach is an approximate technique. However, it is presented here, because RS4MI can be considered as an extension of this approach w.r.t. exact query processing.

Berretti et al. [BDP04] applies a special form of hierarchical clustering based on the M-tree [CPZ97] to a resource's set of feature objects to generate a resource description. A cluster radius threshold θ is used for determining the cluster centers which are included in the resource description. Every path in the clustering tree built for the local collection is descended as long as the cluster radius of a node is bigger than the predefined threshold θ. The centers of the nodes where the search stops are included in the resource description. In addition, per cluster, the maximum cluster radius, i.e. the maximum distance of a database object from its cluster center as well as the number of objects within the cluster are stored in the resource description (the latter may be beneficial for ranking peers when performing k-NN queries). By varying θ, the granularity and size of the resource descriptions can be

adjusted. It is suggested in [BDP04] to set θ to the maximum possible distance value if the distance metric has an upper bound. The block size of the M-tree nodes is the second tuning parameter of this approach. When it comes to resource selection, a resource cannot be pruned from search if the query ball intersects any cluster ball of the resource.

Eisenhardt et al. [EMH$^+$06] extends the cluster histogram technique initially proposed in Müller et al. [MEH05a]. To compute a cluster histogram as resource description, a set with a moderate number of reference objects c_i is applied: $C = \{c_i | 1 \leq i \leq n\}$ with e.g. $n = 256$. Every feature object of a resource's collection is assigned to the closest reference object and a histogram captures how many objects have been assigned to a certain reference object. Eisenhardt et al. [EMH$^+$06] shows that a random selection of reference objects might replace distributed clustering. Resource selection performance slightly decreases, but network load can be reduced because distributed clustering becomes obsolete. For performing k-NN queries, during peer ranking a list L_q of reference object IDs i is sorted in ascending order according to $d(q, c_i)$, i.e. the distance from the query object q to a cluster center c_i. The first element of L_q corresponds to the ID of the cluster center being closest to q. A peer with more documents in the corresponding cluster—indicated by the summary—is ranked higher than a peer with fewer documents in the very cluster. If two peers p_a and p_b administer the same amount of documents in the analyzed cluster, the next element out of L_q is chosen and—based on the indicated number of documents within the very cluster—it is tried to rank peer p_a before peer p_b or vice versa. When the end of the list L_q is reached, a random decision is made. The resource descriptions of this approach are further improved in [BEMH07, BH10]. They are binarized and the number of used reference objects is increased to e.g. $n = 8192$ or even more. Compression techniques are applied to prevent a huge increase in average summary sizes.

4 Exact Resource Selection Approaches for Metric Space Indexing

In the following, we describe and compare three different resource description and selection schemes for metric space indexing. The experimental setup is outlined in Sect. 4.1. In Sect. 4.2, the technique introduced in Berretti et al. [BDP04] (cf. Sect. 3) is analyzed. Another approach—based on k-medoid clustering and used as second comparison baseline for RS4MI—is presented in Sect. 4.3. RS4MI is explained and analyzed in Sect. 4.4. Finally, Sect. 4.5 subsumes the main results of the experimental comparison.

4.1 Experimental setup

We analyze a scenario where every peer knows the resource description of every other peer. Of course, such an approach would not scale. However, this scenario is for example typical in a subnet of a scalable Rumorama-based P2P IR network. Rumorama [MEH05b] can cope with multiple subnets and thus scale to much higher workloads than the ones analyzed in this work.

As underlying data collection, 233827 images crawled from Flickr are used (cf. [BH10]). They are assigned to peers based on the Flickr user ID in order to reflect a realistic scenario, i.e. distribution to resources. Hence, we assume that every Flickr user operates a peer of its own. In this way, the images are mapped to 10601 peers/users. Fig. 2 shows the distribution of peer sizes, i.e. the number of images which are maintained per peer. The general characteristic is typical for P2P file sharing applications, with few peers managing large amounts of the images and many peers administering only few images [SKG02].

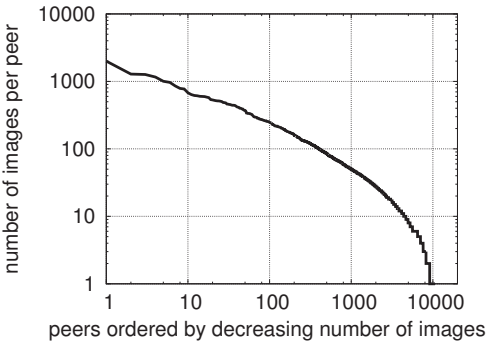

Figure 2: Distribution of peer sizes, i.e. the number of images per peer.

Pivots for summary creation and peer ranking in case of RS4MI are randomly chosen from a secondary data collection consisting of 45931 Flickr images. This reflects a scenario where the reference objects are transfered to the resources with updates of the P2P software in order to reduce network load. All resources administer the same set of pivots. The external (secondary) collection is disjoint from the underlying collection according to the unique Flickr image and user IDs. However, there is some minor natural overlap amongst collections according to image content; 24 of the 233827 images also appear in the external collection because some images are uploaded independently by multiple users on Flickr.

In the experiments, query objects are randomly chosen from the underlying data collection. This seems reasonable in case of range queries relying on the query-by-example paradigm. Retrieval performance is measured by analyzing peer selectivity, i.e. the fraction of peers which must be contacted to retrieve all images with feature objects lying within distance r from q. In addition to search efficiency, the size of the resource descriptions is analyzed. If not mentioned otherwise, summaries are compressed with gzip[1].

As feature descriptor, we use the unquantized version of the CEDD descriptor[2] (144-dimensional vector of 4 byte floats and thus in total 576 byte per descriptor). CEDD has the potential to outperform the MPEG-7 features for CBIR [CZBP10]. The Hellinger metric $d(q,o) = d_H(q,o) = (2 \cdot d_{SC}(q,o))^{\frac{1}{2}} = (2 \cdot \sum_i (\sqrt{q[i]} - \sqrt{o[i]})^2)^{\frac{1}{2}}$ (cf. [DD09]) is applied converting the non-metric squared chord distance d_{SC} into a metric. It is shown in [LSR$^+$08] that d_{SC} provides good retrieval results in case of CBIR. Internal studies with two collections of groundtruth images reveal that the Hellinger metric in combination with CEDD features offers promising retrieval results, outperforming many other distance measures. However, our analysis does not focus on search effectiveness in CBIR and thus the choice of an effective feature descriptor in combination with a distance metric is not the

[1]The time requirements for building the resource descriptions are not analyzed in this work. This task is parallelized in a real-world scenario with every peer computing its resource description and hereby all promising approaches subsumed in Sect. 4.5 are suitably fast.

[2]Features were extracted using the Lire library obtained from *http://www.semanticmetadata.net/lire/*.

	min	q25	median	mean	q75	max
database objects	1	4	21.5	126.7	94.3	2028
peers	1	3	18	72.0	76.3	654

Table 1: Statistics of the number of relevant *database objects* and the number of *peers* administering relevant documents for the 200 range queries with search radius $r = 0.5$.

main focus of our work. Our general setting offers an intrinsic dimensionality (as defined in [CNBYM01, p. 303]) of almost 10 and thus represents a rather hard indexing task.

We evaluate 200 range queries with search radius $r = 0.5$ for every parameter setting. Tab. 1 shows statistics of the number of database objects lying within the search radius. Relevant documents are on average found at 72 peers. An optimal resource selection would thus on average only contact $\frac{72}{10601} \approx 0.7\%$ of the peers to retrieve the relevant documents.

4.2 M-tree based local clustering

To our knowledge, the approach by Berretti et al. [BDP04] is the only exact approach which has so far been proposed for general metric space indexing. Thus, we apply this technique as a comparison baseline for RS4MI. In order to do so, we use revision 27 of the M-tree library from *http://mufin.fi.muni.cz/trac/mtree/* (last visit of all URLs in this paper on 27.09.12) and acknowledge its contributors. The approach mainly depends on two parameters. A cluster radius threshold θ and the block/node size of the M-tree are the keys for trading-off the granularity of the resource descriptions (cf. Sect. 3) versus their selectivity. The influences and interactions when varying these two parameters are evaluated in the following.

The insertion of all database objects of a resource into an M-tree and the threshold-based search algorithm for generating the resource description leads to a partitioning of the feature space based on multiple hyper-spheres. A reference object m_i together with the cluster radius r_i^{max}, both maintained in a node entry of the M-tree, has to be stored in the resource description for every cluster (i.e. hyper-sphere) to be able to perform exact range queries. With this information, the range-pivot distance constraint (cf. Sect. 2) testing the overlap of the query ball with any cluster ball can then be applied during search in order to prune irrelevant peers, i.e. peers with no relevant documents.

Analysis of M-tree based local clustering

In the upper left quadrant of Fig. 3, the selectivity of the summaries of the M-tree based local clustering approach is shown. The lower left quadrant depicts the corresponding summary sizes. To understand this figure, the following aspects have to be considered: (1) A block size of 576 byte corresponds to leaf nodes containing one object each. In this case, the M-tree implementation assures that inner nodes (including the root node) are bigger and the degree of each inner node is two. In general, a block size of s_b means that

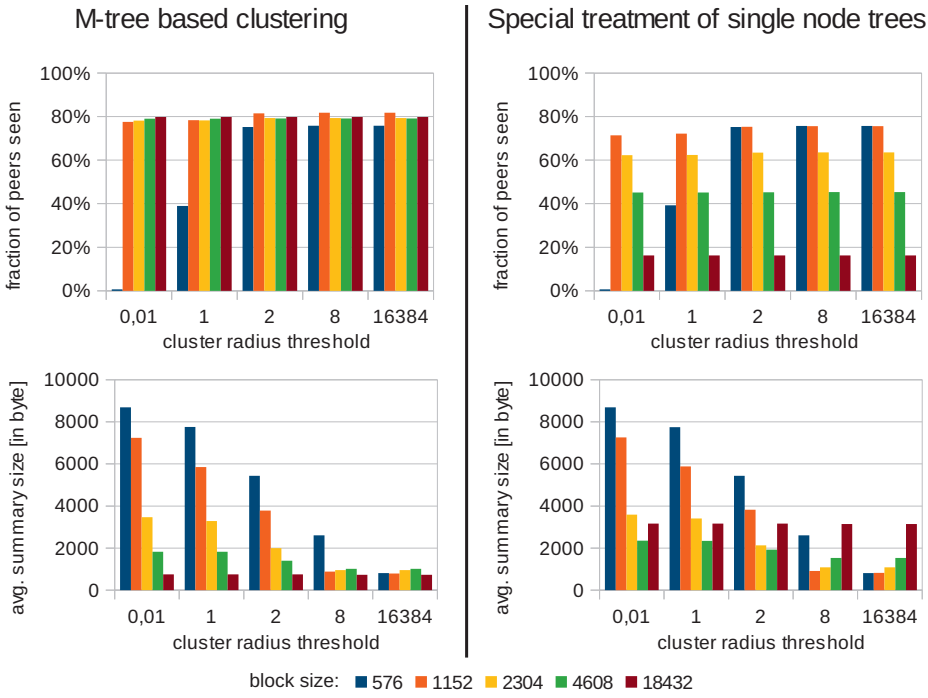

Figure 3: M-tree based local clustering (left) with special treatment of single node trees (right).

a leaf node contains at most $s_b/576$ objects. Hence, e.g. a node size of 18432 corresponds to leaf nodes containing at most 32 objects. (2) A cluster radius threshold of 0.01 has the consequence that the summary roughly contains clusters describing exactly the leaf nodes. At the other extreme, a cluster radius threshold of 16384 would yield a summary containing only one cluster representing the root node of the M-tree and in consequence the complete set of objects on the peer[3].

With the above information in mind, we can interpret the left side of Fig. 3. If we consider the average summary size in dependence of the cluster radius threshold (lower left quadrant) it becomes obvious that the summary sizes decrease for higher threshold values. The reason is that for higher threshold values the clusters for the summaries are taken from higher levels of the M-tree. Obviously, this effect is only given for small block sizes (blue, orange and yellow bars), because for higher block sizes (e.g. dark red bars) the height of the M-trees is extremely low anyway.

The upper left quadrant of Fig. 3 shows the selectivity of the summaries measured by the fraction of peers seen. Let us first consider the fraction of peers seen in dependence of the cluster radius threshold. As a special case, the block size of 576 together with a cluster radius threshold of 0.01 has to be considered. In this situation, each leaf node contains

[3]Please note that all object distances are at most 2. However, due to heuristic upper bound approximation of the cluster radii in the inner nodes of the M-tree, values bigger than 2 exist in the tree.

only a single item and because of the low threshold value, the clusters describing the leaf nodes are included in the summaries. Consequently, the summaries exactly represent the objects on each peer. Based on this information, a querying peer can exactly determine the peers containing objects in the query ball and therefore, the fraction of peers seen corresponds to the theoretical optimum of 0.7%. However, this result is achieved by a complete replication of all objects within the network on all peers. Unfortunately, these parameter settings are not realistic for huge networks. Neither a threshold value yielding only leaf nodes nor a node size storing only one object per node are practical.

Despite from these special cases, the fraction of peers seen is roughly between 70% and 80%. It is also interesting to consider the effect of the block sizes e.g. for a cluster radius threshold of 8. With this threshold, only in very rare cases the clusters used in the summary are taken from lower levels of the tree. With the block size of 576 byte, peers with only one image are represented by one cluster in the summary and peers with 2 or more images are (with some exceptions) represented by two clusters, since the fan-out of the root node is 2 in this case. With the block size of 1152 byte, peers with one or two images are represented by one cluster in the summary and peers with 3 or more images are (with few exceptions) represented by two clusters. The less precise representation of peers with 2 images results in an increase of the peers which have to be considered from 75.8% to 81.8% and at the same time reduces the average summary size drastically. With the block size of 2304 byte, peers with one to four images are represented by one cluster in the summary and peers with 5 or more images are (with few exceptions again) represented by two to four clusters. Hence, the summaries of small peers become less accurate but the summaries of bigger peers become more accurate, since the root node of the M-tree now has up to 4 successors. Obviously, these considerations can be continued for bigger block sizes.

The above results achieved for the originally proposed M-tree based local clustering approach inspired us to change the approach marginally in order to exploit the long-tail distribution of images on peers (cf. Fig. 2). Over 50% of the peers contain 7 or less images. As a consequence: If the summaries of these small peers would contain the exact objects, only the peers out of these 50% which really contribute to the result of the range query must be visited. With such a technique we can easily outperform the approaches presented above which have to address 70% to 80% of the peers.

To integrate this idea into the M-tree based local clustering approach we use a special treatment for situations where the M-tree consists of only one (leaf) node—which is typical for small peers. In this case the summary now contains one cluster with radius zero for each object in this leaf node instead of one single cluster with a huge radius describing the whole node. As a consequence e.g. at a block size of 18432 byte a peer maintaining 32 objects fitting into one single leaf node is now represented by a summary containing these 32 objects as single clusters with the objects as centers and radius zero.

The effect of this variation can be seen on the right hand side of Fig. 3. Let us—again at a cluster radius threshold of 8—consider the green bars representing a node size of 4608 byte, resp., at most 8 objects. In this case 5643 (= 53%) of all peers are represented exactly in the summaries. This allows to reduce the number of peers to be contacted during query processing to 45.3%. The avg. summary size is 1531 byte (compared to 1010 byte without the special treatment of small peers).

Although, the improvements achieved with this variation are impressive, it remains a bit problematic that we have such indirect and hard to handle parameters; the threshold value θ, the block size of the M-tree and the special treatment of trees comprising only one node. According to the threshold value θ, Fig. 3 shows that the heuristic of setting $\theta = 2$, i.e. the maximum possible distance value, might not be a suitable solution in all cases. In fact, it might be much easier to use an explicit clustering approach with more intuitive parameters. This directly leads us to the k-medoid clustering.

4.3 Local k-medoid clustering

Some approximate resource selection approaches for the use in vector spaces apply k-means clustering to cluster the database objects of a peer (cf. e.g. [EBMH08]). However, k-means, due to the mean calculation, is not applicable in general metric spaces. When using k-medoid clustering instead (or any other suitable algorithm applicable in general metric spaces), an additional baseline technique for the comparison with RS4MI can be designed. In this case, each peer clusters its local data collection and stores cluster centers m_i and maximum cluster radii r_i^{max} in its resource description. This results in a similar data space partitioning and similar resource descriptions as the approach proposed in Berretti et al. [BDP04] (cf. Sect. 4.2). The resource description of a peer in case of range queries thus consists of a list of cluster center and corresponding maximum cluster radius pairs.

There are two general options for determining k, i.e. the number of clusters of a peer needed as an input parameter to k-medoid clustering. As one alternative, the maximum number of allowed clusters per peer k can be set as a global threshold being identical for all peers. Of course, peers with less than k distinct database objects directly transfer these and do not apply clustering. On the other hand, algorithms which automatically detect an appropriate number of clusters can be used. Multiple of these algorithms are presented in literature (for references see e.g. [TWH01]). Our choice of algorithms in the following is by no means exhaustive. It is our intention to evaluate different techniques which return a range of average numbers of clusters per peer when applied to our scenario.

Rule of thumb (r.o.t.): A coarse rule of thumb is presented in [MKB79, p. 365]. It is suggested to calculate the number of clusters of a data set of size $|O|$ as $k \approx \sqrt{|O|/2}$. Thus, we use $k = \lceil \sqrt{|O|/2} \rceil$.

This rule of thumb directly calculates the number of desired clusters. In contrast, the techniques presented in the following are applied in an iterative process. A single key figure results for a specific value of k. Various values of k are thus to be tested to select the best k minimizing/maximizing the key figure. To reduce runtime performance, even when applying the rule of thumb, an adaptation of the original k-medoid clustering algorithm is used. The FAMES extension to k-medoid clustering uses pivots in order to speed-up k-medoid clustering [PNT11]. FAMES avoids the calculation of all pair-wise distances when computing the medoid of a certain cluster. In addition to improving efficiency, it is shown in [PNT11] that FAMES can also increase the effectiveness of the clustering since the efficiency gain is not due to the consideration of a random sample of database

objects as medoid candidates—which is the approach of some traditional algorithms. For determining the initial candidate set of medoids, we minimize in 10 runs the sum over all clusters of within-cluster object-to-medoid distances.

Besides the rule of thumb, we apply three variants of the well-known GAP statistic. The GAP statistic [TWH01] is frequently used and offers the property that—in contrast to many alternative approaches—it can also detect the presence of only a single cluster.

GAP: The GAP statistic as originally defined in [TWH01] is based on a sampling process which is not directly applicable in all metric spaces. However, as suggested in [TWH01], when only distance information is available, a specific mapping technique such as multidimensional scaling can be used to obtain feature vectors in a low dimensional space, which provide the basis for the sampling process. In our experiments, we directly apply ten sampling steps on the feature vectors without the use of an additional mapping technique in order to obtain a best case comparison baseline against which we can compare our approach RS4MI.

GAP$_w$, introduced in [YY07], modifies the weighting scheme of the GAP statistic.

GAP$_n$ represents another slight modification of the GAP statistic, where all logarithms used in the formulae of the GAP statistic are removed [MES10].

Sil$_1$ **and Sil**$_2$**:** The Silhouette technique [Rou87] is also adapted as a means for calculating the desired number of clusters of a peer. It is only applicable in case of $k > 1$. Thus, two alternatives are used in our experiments. If two is indicated as optimum cluster number, we set $k = 1$ in case of Sil$_1$; $k = 2$ is used in case of Sil$_2$. Peers with only a single database object—of course—only encode a single cluster in the resource description.

To determine an appropriate value for k, the above mentioned approaches based on the GAP statistic and the Silhouette technique are iteratively tested on every peer till $k = \lfloor \min(2\sqrt{nDocs}, nDocs) \rfloor$ with *nDocs* denoting the number of documents/images of a peer.

Analysis of local k-medoid clustering

Tab. 2 (top) shows the average fraction of visited peers, the average number of clusters per peer, as well as average summary sizes in case of local k-medoid clustering when automatically determining the number of clusters of a peer. The rule of thumb (r.o.t.) leads to decent retrieval performance at the cost of comparatively large summaries. Better peer selectivity is achievable by the SIL$_2$ approach with more space efficient resource descriptions.

Using the GAP statistic for determining the number of clusters of a peer results in average summary sizes of approximately 1 kB and 73.7% of peers being contacted for retrieving all relevant documents. GAP$_w$ and GAP$_n$ lead to fewer numbers of clusters per peer and thus more space efficient resource descriptions. However, both perform worse than GAP.

SIL$_1$ leads to similar average summary sizes as GAP. The average number of clusters per peer is in both cases approximately 2.3, but GAP offers better peer selectivity. SIL$_1$ always assumes one cluster when there might be two (which GAP might detect). SIL$_2$ shows better peer selectivity than the other competing approaches (even better than the

	r.o.t.	GAP	GAP_w	GAP_n	SIL_1	SIL_2
visited peers	67.4%	73.7%	75.3%	76.4%	76.4%	65.7%
clusters per peer	3.1	2.3	1.9	1.5	2.3	2.8
summary size	1350.3 B	1048.4 B	880.7 B	722.9 B	1029.5 B	1232.2 B

	$k=1$	$k=2$	$k=4$	$k=8$	$k=32$	$k=128$
peers seen	79.9%	68.7%	54.4%	37.7%	13.1%	2.7%
clusters per peer	1.0	1.9	3.4	5.6	12.2	18.2
summary size	525.0 B	867.2 B	1.4 kB	2.3 kB	4.9 kB	7.3 kB

Table 2: Results for local k-medoid clustering with automatic determination of the number of clusters k (top) and all peers using the same global k (bottom).

rule of thumb which identifies on average 3.1 clusters per peer) at the cost of storing on average 2.8 clusters per peer in the summaries. Overall, GAP_n and GAP_w seem promising approaches with average summary sizes clearly below 1 kB.

A main drawback of the k-medoid approaches analyzed in this section so far is that the summary sizes cannot be influenced by any kind of design parameter of the approach. An alternative in this respect is to globally specify k, the maximum allowed number of clusters per peer. In this case, peers with *nDocs* $\leq k$ store all feature objects in their summary. Since for some peers the number of feature objects is smaller than k, the average number of clusters per peer becomes smaller than k as well. This scenario which is thus similar to the special treatment of single node trees in Sect. 4.2 is evaluated in Tab. 2 (bottom).

The explicit definition of an upper bound for the number of clusters allows for a direct and accurate adjustment of summary sizes and selectivity. This gives a clear advantage over the M-tree based approach and also over the approaches which automatically determine a suitable number of clusters per peer. However, if very small summaries are necessary, the flexibility is restricted by the discrete values of k.

It can be observed from Tab. 2 (bottom) that only in cases where the maximum desired number of clusters per peer is set to $k = 1$ or $k = 2$, average summary sizes with less than 1 kB can be achieved. If a maximum of two clusters is allowed, 68.7% of the resources are visited with an average summary size of 867 byte. In order to further reduce this figure, only a single cluster per peer can be allowed. However, almost 80% of peers are contacted in this case with an average summary size of 525 byte.

4.4 RS4MI: Resource Selection for Metric Indexing

RS4MI can make use of all pruning criteria mentioned in Sect. 2. A set of n reference objects C—globally unique for all resources—is applied in order to assign a database object o of a resource to the closest cluster center $c^* = \arg\min_{c_i \in C} d(c_i, o)$. The set of reference objects is transfered to remote peers together with updates of the P2P software,

so that no additional network load is imposed during the operating phase of the P2P IR system. Such an approach is for example proposed in [BEMH07].

Different variants of RS4MI resource descriptions are evaluated in the following to find the best alternative. These variants can make use of different pruning criteria and thus result in different peer selectivity and average summary sizes.

RS4MI$_{\mathbf{1xxxx}}$: Here, only a single bit is stored per cluster in order to indicate if any database objects lie in the very cluster or not. This results in a bit vector of size n and thus resource descriptions with $\mathcal{O}(n)$ space complexity. The double-pivot distance constraint outlined in Sect. 2 is the only pruning constraint which can be used in this case to prune peers from search.

RS4MI$_{\mathbf{x??xx}}$: Resource descriptions offering $\mathcal{O}(n)$ space complexity can also be designed by storing the minimum and/or maximum cluster radii. By doing so, the range-pivot distance constraint can be applied on an intra-cluster level (cf. Sect. 2). In addition to storing both minimum and maximum cluster radii for the n clusters (i.e. RS4MI$_{\text{x11xx}}$), we test parameter settings of RS4MI$_{\text{x1xxx}}$ and RS4MI$_{\text{xx1xx}}$ where only minimum or maximum cluster radii are stored respectively. A single distance value is always represented as a four byte float.

Of course, the double-pivot distance constraint can also be applied in this case. If no minimum/maximum cluster radius is set for a particular cluster, it is indicated by the summary that the corresponding peer does not administer any database objects within the very cluster. So, the double-pivot distance constraint is used by all of the following resource selection schemes whenever applicable.

RS4MI$_{\mathbf{xxx??}}$: If all criteria for cluster pruning described in Sect. 2 should be applied, two matrices *MIN* and *MAX* have to be administered by every database as resource description (RS4MI$_{\text{xxx11}}$). This requires $\mathcal{O}(n^2)$ space per resource. As before, a single matrix cell requires four byte in order to store radius information. Both matrices are sent as resource description and used for the pruning of resources without querying them. We also test parameter settings where only a single matrix *MIN* (RS4MI$_{\text{xxx1x}}$) or *MAX* (RS4MI$_{\text{xxxx1}}$) is used.

Two further combinations are included in the analysis. RS4MI$_{\text{x1xx1}}$ stores minimum cluster radii and the matrix *MAX* as resource description. In opposition, RS4MI$_{\text{xx11x}}$ applies maximum cluster radii and the *MIN* matrix to prune peers during search.

We also evaluate a hybrid resource selection scheme where either per-cluster or per-object information is stored in the resource description of a peer.

Analysis of RS4MI

In the following, different ways of how to best design summaries in case of RS4MI are evaluated. First, summaries storing only per-cluster information are analyzed. Later, hybrid summaries are evaluated. We should note here that *hybrid* in case of RS4MI means storing per-cluster or per-object information. RS4MI can of course, similar to the ap-

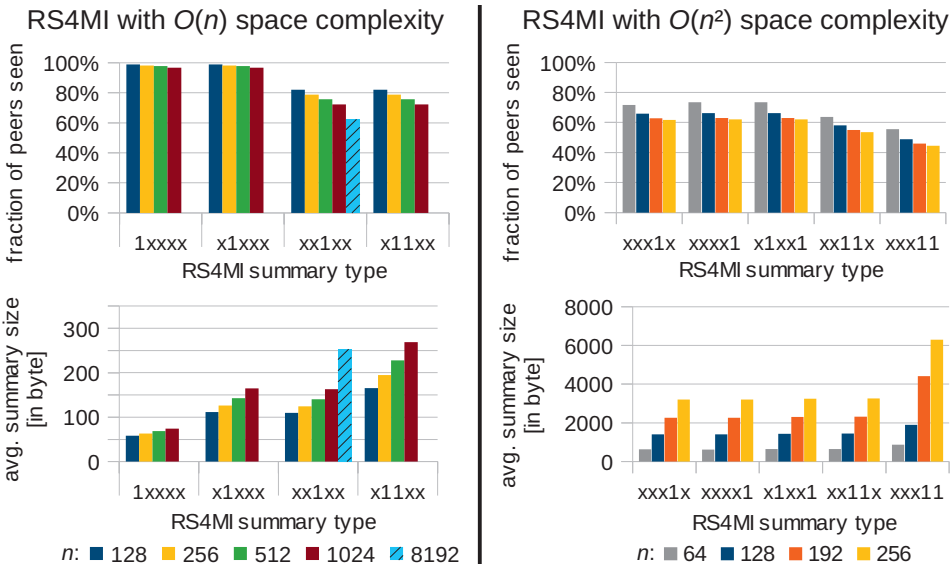

Figure 4: Results of RS4MI for summaries with space complexity $\mathcal{O}(n)$ (left) and $\mathcal{O}(n^2)$ (right).

proaches evaluated in Sect. 4.2 and Sect. 4.3, be extended to store feature objects for peers with few images directly in the summaries. An analysis is part of future work.

RS4MI approaches storing per-cluster information: Fig. 4 (top left) visualizes retrieval performance for resource descriptions with $\mathcal{O}(n)$ space complexity. It can be observed that RS4MI$_{\text{1xxxx}}$ and thus only applying the double-pivot distance constraint does not lead to an acceptable peer selectivity. RS4MI$_{\text{1xxxx}}$ with a bit-vector as underlying data structure however results in very space efficient resource descriptions, even in case of larger values of n (e.g. $n = 1024$ in Fig. 4 (bottom left)).

Comparing RS4MI$_{\text{x1xxx}}$ with RS4MI$_{\text{xx1xx}}$, it can be observed that although both approaches have similar average summary sizes, RS4MI$_{\text{xx1xx}}$ can prune clearly more peers than RS4MI$_{\text{x1xxx}}$. Even RS4MI$_{\text{x11xx}}$ cannot noticeably improve peer selectivity. RS4MI$_{\text{xx1xx}}$ with a very large number of reference objects being used (e.g. $n = 8192$ or even more) seems the best choice amongst the approaches considered in the left part of Fig. 4.

In addition, resource descriptions with $\mathcal{O}(n^2)$ space complexity are analyzed. For these approaches a binning technique is applied in order to reduce summary sizes. Every four byte distance value is quantized into a single byte. The minimum and maximum distance value from the database objects of the external collection to every cluster center c_i is determined. The range between these two boundaries per reference object c_i is uniformly quantized into 253 intervals. From the remaining three values, two are used to represent distance values below and above the boundaries. The third remaining value is used to indicate an empty cluster with no entry. Here, it is again assumed that the minimum and

	gzip	bzip2	lzma	png	paq8o8	webpll
summary size	867.0 B	1020.1 B	863.4 B	880.4 B	803.1 B	777.7 B

Table 3: Average summary sizes for $\text{RS4MI}_{\text{xxx11}}$ with $n = 64$.

maximum distances from feature objects of the external collection to the cluster centers c_i are known to all peers in advance and transfered to them by updates of the P2P software so that all peers can correctly estimate the true distance from the quantized values. However, this information is also small enough to be transfered to participating peers during the operation phase of the P2P IR system.

Fig. 4 (bottom right) shows that the average summary sizes in case of $\text{RS4MI}_{\text{xxx1x}}$, $\text{RS4MI}_{\text{xxxx1}}$, $\text{RS4MI}_{\text{x1xx1}}$, and $\text{RS4MI}_{\text{xx11x}}$ are very similar. According to retrieval performance (cf. Fig. 4 (top right)) $\text{RS4MI}_{\text{xx11x}}$ applying the *MIN* matrix and an array of length n with maximum cluster radii clearly outperforms the other three approaches. Also $\text{RS4MI}_{\text{xxx11}}$ encoding the quantized *MIN* and *MAX* matrices with a small value of n is promising (e.g. $n = 64$). For the feature set being indexed, $\text{RS4MI}_{\text{xxx11}}$ with n being small or $\text{RS4MI}_{\text{xx1xx}}$ with n being big seem to be the most promising RS4MI approaches.

To further reduce the summary sizes of $\text{RS4MI}_{\text{xx1xx}}$ with $n = 8192$, alternative compression algorithms might be suitable. When changing the compression algorithm to bzip2, summary sizes are reduced on average from 253.2 (gzip) to 222.1 byte and to 234.2 byte in case of lzma. Thus, a reduction of approximately 10% seems easily possible[4].

Summary sizes for $\text{RS4MI}_{\text{xxx11}}$ with $n = 64$ can also be reduced. The bzip2 implementation seems to be inappropriate with average summary size noticeably increasing, and also lzma does not lead to a significant reduction (cf. Tab. 3). Thus, in addition to gzip, bzip2, and lzma, three image compression algorithms are tested, where the concatenation of the quantized *MIN* and *MAX* matrices is interpreted as a 2-dimensional 256 bit gray-scale image of size 64×128 pixels. Tab. 3 shows the results. Standard png compression provides some overhead, but paq8o8[5] and especially webp[6] lossless image compression provide more space efficient resource descriptions; webp in particular by significantly improving the memory requirements of the summaries of the peers with images in few clusters.

Hybrid RS4MI approaches storing per-object information: The RS4MI approaches presented so far solely rely on cluster pruning principles. Object pruning and thus the encoding of per-object information in the resource descriptions is not considered. However, the distribution of peer sizes (cf. Fig. 2) indicates many peers with few documents. Thus, at least for peers with very few documents it might be beneficial to encode per-object summary information and apply pivot filtering (cf. Sect. 2).

[4] Additional compression results are based on the at4j library (*http://at4j.sourceforge.net/*). We acknowledge the contributors of at4j and of contributing libraries such as 7-zip (*http://www.7-zip.org/*) and apache commons compress (*http://commons.apache.org/compress/*).

[5] cf. *http://mattmahoney.net/dc/#paq*

[6] cf. *https://developers.google.com/speed/webp/*

	$n = 1$	$n = 2$	$n = 4$	$n = 8$	$n = 12$	$n = 16$
peers seen	97.3%	95.5%	90.9%	82.5%	75.8%	73.5%
summary size	136.5 B	216.9 B	372.5 B	676.2 B	976.1 B	1274.6 B

Table 4: Results when purely applying pivot filtering.

	$s_d = 100$ (8439)	$s_d = 200$ (9612)	$s_d = 400$ (10202)	$s_d = 600$ (10385)	$s_d = 800$ (10459)	$s_d = 1000$ (10503)
$n = 512$	81.1%	76.0%	69.9%	66.0%	63.4%	61.4%
	174.9 B	251.9 B	426.9 B	585.1 B	759.6 B	919.2 B
$n = 4096$	80.3%	75.7%	69.8%	65.9%	63.4%	61.4%
	214.8 B	277.8 B	443.0 B	596.8 B	768.9 B	926.8 B

Table 5: Results for hybrid summaries. Table cells show the fraction of contacted peers (top) and the avg. summary size (bottom). The number of peers applying pivot filtering is given in brackets.

First, we analyze settings where only object-pivot distances (and thus no per-cluster information) are used in the resource descriptions. Tab. 4 shows the results for different numbers of reference objects. Such an undifferentiated approach is inappropriate and results in very big summary sizes for peers with many documents. When using 16 reference objects and thus encoding 16 object-to-pivot distance values per database object, 73.5% of peers are contacted with resource descriptions of 1.3 kB on average.

We also analyze a hybrid resource description scheme with peers choosing between either per-object or per-cluster summarization, depending on *nDocs*, the number of images a peer administers. In order to very roughly estimate the number of possible reference objects per database object for which object-pivot-distances are stored in the summary, the formula *nRefsPerObject* = $\lfloor \frac{s_d}{4 \cdot nDocs} \rfloor$ is applied. The parameter s_d hereby denotes the desired average summary size in byte and a factor of four in the denominator is used since a single distance value is represented as a four byte float. From Tab. 5, it can be seen that this estimate of the average summary size roughly holds. If *nRefsPerObject* > 0, pivot filtering is applied on the basis of per-object resource descriptions. Otherwise, per-cluster summaries $\text{RS4MI}_{\text{xx1xx}}$ are applied as before.

Table 5 visualizes results of the hybrid resource selection scheme when varying s_d and n. The number of peers applying pivot filtering is denoted in brackets. If these results are compared with the ones applying only per-cluster information, several approaches can be outperformed; for example a parameter settings with $n = 512$ and $s_d = 600$ seems promising. However, peer selectivity of $\text{RS4MI}_{\text{xx1xx}}(n = 8192)$ can only be achieved with much bigger average summary sizes, since compression techniques in case of $\text{RS4MI}_{\text{xx1xx}}(n = 8192)$ can dramatically reduce the summary size of peers with documents in only few clusters.

4.5 Brief comparison of approaches

In Sects. 4.2 and 4.3, we saw that techniques yielding an exact representation of small peers, either applying a special treatment for single node M-trees or defining a desired value for k, are promising in situations with a long-tail distribution of the objects over the peers. Of course, such techniques can also be applied for RS4MI—a consideration of this approach is planned for the near future. To assess the potential in large scale networks, let us concentrate on the summary sizes and the selectivity of the basic techniques here.

Table 6 gives a brief overview of different approaches discussed in Sect. 4.2, Sect. 4.3, and Sect. 4.4. All of them result in average summary sizes below 1 kB. Conceptually, the source selection techniques based on M-tree and k-medoid clustering are similar to each other both applying local clustering and transferring medoids and cluster radii. The parametrization of the techniques is crucial for both approaches. In this regard, the k-medoid based local clustering approach with its easy to interpret design parameter k is more handy than M-tree based clustering and also retrieval performance (as briefly summarized in Tab. 6 and in more detail outlined in Sect. 4.2 and Sect. 4.3) does not give a clear evidence for using the approach based on the M-tree. $\text{RS4MI}_{\text{xx1xx}}(n = 8192)$ leads to better retrieval results with significantly smaller average resource description sizes. The number of contacted peers is further reduced by $\text{RS4MI}_{\text{xxx11}}(n = 64)$ at the cost of larger summaries, comparable with those of 2-medoid. Of course, it is also possible to use different RS4MI summary types within a single P2P IR system. We will analyze this in future work.

	peers seen	avg. summary size
M-tree(576;16384)	75.8%	813.2 byte
2-medoid	68.7%	867.7 byte
$\text{RS4MI}_{\text{xx1xx}}(n = 8192)$	62.2%	253.0 byte
$\text{RS4MI}_{\text{xxx11}}(n = 64)$	57.2%	880.7 byte

Table 6: Comparison of the different approaches with results averaged over ten runs.

5 Conclusion and Outlook

We presented RS4MI—an exact resource selection scheme for general metric spaces and showed how the processing of range queries can be performed. RS4MI can outperform an M-tree based exact resource selection scheme and a selection scheme based on k-medoid clustering w.r.t. the number of peers which are contacted and w.r.t. memory requirements. In case of range queries, RS4MI is especially beneficial in scenarios when the memory requirements of the database objects are huge since RS4MI does not store them in the summaries. Furthermore, with large numbers of reference objects fine-grained adaptations are possible w.r.t. the avg. summary size. Peer(s) monitoring the network can adaptively adjust summary sizes since every peer per se knows the summary size of all other peers.

In future work, we will also analyze the processing of k-NN queries and strategies for determining the reference objects. In addition, we will derive approximate extensions providing a good compromise between runtime performance and adequate retrieval quality.

References

[BDP04] S. Berretti, A. Del Bimbo, and P. Pala. Merging Results for Distributed Content Based Image Retrieval. *Multimedia Tools Appl.*, 24(3):215–232, 2004.

[BEMH07] D. Blank, S. El Allali, W. Müller, and A. Henrich. Sample-based Creation of Peer Summaries for Efficient Similarity Search in Scalable Peer-to-Peer Networks. In *Intl. SIGMM Workshop on Multimedia Information Retrieval*, pages 143–152, Augsburg, Germany, 2007. ACM.

[BH10] D. Blank and A. Henrich. Binary Histograms for Resource Selection in Peer-to-Peer Media Retrieval. In *Proc. of LWA Workshop – Lernen, Wissen, Adaptivität*, pages 183–190, Kassel, Germany *www.kde.cs.uni-kassel.de/conf/lwa10/papers/ir4.pdf* (last visit: 27.9.2012), 2010.

[BKSS07] B. Bustos, D. Keim, D. Saupe, and T. Schreck. Content-Based 3D Object Retrieval. *IEEE Comput. Graph. Appl.*, 27(4):22–27, July 2007.

[CNBYM01] E. Chávez, G. Navarro, R. A. Baeza-Yates, and J. L. Marroquín. Searching in Metric Spaces. *ACM Comput. Surv.*, 33(3):273–321, 2001.

[CPZ97] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. In *Proc. of the 23rd Intl. Conf. on Very Large Data Bases*, pages 426–435, Athens, Greece, 1997. Morgan Kaufmann.

[CZBP10] S. A. Chatzichristofis, K. Zagoris, Y. S. Boutalis, and N. Papamarkos. Accurate Image Retrieval Based on Compact Composite Descriptors and Relevance Feedback Information. *Intl. J. of Pattern Recognition and Artificial Intelligence*, 24(2):207–244, 2010.

[DD09] M. M. Deza and E. Deza. *Encyclopedia of Distances*. Springer Berlin Heidelberg, 1st edition, 2009.

[DVNV10] C. Doulkeridis, A. Vlachou, K. Nørvåg, and M. Vazirgiannis. Part 4: Distributed Semantic Overlay Networks. In X. Shen, H. Yu, J. Buford, and M. Akon, editors, *Handbook of Peer-to-Peer Networking*, pages 463–494. Springer Science+Business Media, 1st edition, 2010.

[EBMH08] S. El Allali, D. Blank, W. Müller, and A. Henrich. Image Data Source Selection Using Gaussian Mixture Models. In *Adaptive Multimedia Retrieval: Retrieval, User, and Semantics: 5th International Workshop*, pages 170–181, Berlin, Heidelberg, 2008. Springer LNCS 4918.

[EMH+06] M. Eisenhardt, W. Müller, A. Henrich, D. Blank, and S. El Allali. Clustering-Based Source Selection for Efficient Image Retrieval in Peer-to-Peer Networks. In *Proc. of the 8th Intl. Symp. on Multimedia*, pages 823–830, San Diego, CA, USA, 2006. IEEE.

[ERO+09] B. M. Elahi, K. Römer, B. Ostermaier, M. Fahrmair, and W. Kellerer. Sensor ranking: A primitive for efficient content-based sensor search. In *Proc. of the 8th Intl. Conf. on Information Processing in Sensor Networks*, pages 217–228, San Francisco, CA, USA, 2009. ACM.

[HCS09] X. Hu, T. Chiueh, and K. G. Shin. Large-scale malware indexing using function-call graphs. In *Proc. of the 16th ACM Conf. on Computer and Communications Security*, pages 611–620, New York, NY, USA, 2009. ACM.

[KLC02] D.-H. Kim, S.-L. Lee, and C.-W. Chung. Heterogeneous image database selection on the Web. *The Journal of Systems and Software*, 64:131–149, 2002.

[KW11] M. Kunze and M. Weske. Metric Trees for Efficient Similarity Search in Large Process Model Repositories. *Business Process Management Workshops*, 66:535–546, 2011. Springer Lecture Notes in Business Information Processing.

[LLOS07] M. Lupu, J. Li, B. C. Ooi, and S. Shi. Clustering wavelets to speed-up data dissemination in structured P2P MANETs. In *Proc. of the 23th Intl. Conf. on Data Engineering*, pages 386–395, Istanbul, Turkey, 2007. IEEE.

[LSR+08] H. Liu, D. Song, S. M. Rüger, R. Hu, and V. S. Uren. Comparing Dissimilarity Measures for Content-Based Image Retrieval. In *Proc. of the 4th Asia Infomation Retrieval Symposium*, pages 44–50. Springer LNCS 4993, 2008.

[MEH05a] W. Müller, M. Eisenhardt, and A. Henrich. Fast retrieval of high-dimensional feature vectors in P2P networks using compact peer data summaries. *Multimedia Systems*, 10(6):464–474, 2005.

[MEH05b] W. Müller, M. Eisenhardt, and A. Henrich. Scalable summary based retrieval in P2P networks. In *Proc. of the 14th Intl. Conf. on Information and Knowledge Management*, pages 586–593, Bremen, Germany, 2005. ACM.

[MES10] M. Mohajer, K.-H. Englmeier, and V. J. Schmid. A comparison of Gap statistic definitions with and without logarithm function. *Online:* http://arxiv.org/abs/1103.4767*, (last visit: 27.9.2012)*, 2010.

[MKB79] K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, London, 1979.

[NBZ11] D. Novak, M. Batko, and P. Zezula. Metric Index: An efficient and scalable solution for precise and approximate similarity search. *Inf. Syst.*, 36:721–733, June 2011.

[NF03] H. Nottelmann and N. Fuhr. Decision-theoretic resource selection for different data types in MIND. In *Distributed Multimedia Information Retrieval: Proc. of the Intl. Workshop on Distributed Information Retrieval*, pages 43–57. Springer LNCS 2924, 2003.

[PNT11] A. A. Paterlini, M. A. Nascimento, and C. Traina Jr. Using Pivots to Speed-Up k-Medoids Clustering. *J. of Information and Data Management*, 2(2):221–236, 2011.

[Rou87] P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20:53–65, 1987.

[Sam06] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2006.

[SB11] T. Skopal and B. Bustos. On Nonmetric Similarity Search Problems in Complex Domains. *ACM Computing Surveys*, 43(4):34:1–34:50, October 2011.

[SKG02] S. Saroiu, P. Krishna Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *ACM/SPIE Multimedia Computing and Networking*, pages 156–170, San Jose, CA, USA, 2002.

[Sko06] T. Skopal. *Similarity Search In Multimedia Databases*. PhD thesis, Charles University, Prague, Czech Republic, 2006.

[SS11] M. Shokouhi and L. Si. Federated Search. *Foundations and Trends in Information Retrieval*, 5(1):1–102, 2011.

[TWH01] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a dataset via the Gap statistic. *J. R. Stat. Soc. Ser. B*, 63(2):411–423, 2001.

[Woj02] A. Wojna. Center-Based Indexing in Vector and Metric Spaces. *Fundam. Inf.*, 56:285–310, 2002.

[YSMQ01] C. Yu, P. Sharma, W. Meng, and Y. Qin. Database selection for processing k nearest neighbors queries in distributed environments. In *Proc. of the 1st ACM/IEEE Joint Conf. on Digital Libraries*, pages 215–222, New York, NY, USA, 2001.

[YY07] M. Yan and K. Ye. Determining the Number of Clusters Using the Weighted Gap Statistic. *Biometrics*, 63:1031–1037, 2007.

[ZADB05] P. Zezula, G. Amato, V. Dohnal, and M. Batko. *Similarity Search: The Metric Space Approach*. Springer New York, Inc., Secaucus, NJ, USA, 2005.