Transactions on Machine Learning
and Data Mining
Vol. 6, No.1 (2013) 19-42
ISSN: 1865-6781,
ISBN: 978-3-942952-24-8

ibai Publishing
www.ibai-publishing.org

Reduction of Distance Computations in Selection of Pivot Elements for Balanced GHT Structure

László Kovács

University of Miskolc, Department of Information Technology,
3515 Miskolc-Egyetemváros, Hungary
Phone: +36-20-3319765
kovacs@iit.uni-miskolc.hu

Abstract. For objects in general metric spaces, the generalized hyperplane indexing is one of the most widely used indexing techniques In the paper, some methods are presented to improve the quality of the partitioning in generalized hyperplane tree structure from the viewpoint of balancing factor. The proposed method to represent the elements in the target domain metric space is the usage of a distance matrix as it can model the distance relationship without any information loss. The efficiency of the partitioning depends on the appropriate selection of the pivot elements. As this method requires the knowledge of a great number of distance values between the objects, the paper proposes an interval-based distance value representation. Based on the test results, the given method dominates the usual techniques if the fullness factor of the distance matrix is between 3% and 35%.

Keywords: pivot based indexing, general metric space, interval based computation

1 Introduction

Objects under investigation are usually represented by real valued feature vectors. The generation of appropriate feature vectors in some complex problem domains is a very hard or impossible task. For such complex objects as x-ray photos, large documents or the feeling of humans, there exists no simple and all-covering feature representation. Our investigation focuses on the cases, where the only information on the objects is the distance values between them. The applied distance function *d()* is called metric if it fulfills the following conditions:

$$\begin{gathered} d(x,y) \geq 0 \\ d(x,y) = 0 \leftrightarrow x = y \\ d(x,y) = d(y,x) \\ d(x,z) + d(z,y) \geq d(x,y). \end{gathered} \tag{1}$$

In this case the container space is a general metric space (GMS). The different application areas may have very different and complex distance functions. Usually the storage and comparison of the objects in GMS are relatively expensive operations. In the case of applications with huge collection of these objects, the objects are clustered and indexed to reduce the query and object retrieval costs. The traditional one dimensional indexing techniques cannot be used in GMS. The most widely used indexing methods in general metric spaces use pivot elements. A pivot element p is a distinguished object from the object-set. The distance from an object x to p is used as the indexing key value of x to locate the bucket containing x. Usually more than one single pivot element are used in the partitioning algorithms.

There are many variants of pivot-based index trees in general metric spaces. The Generalized Hyperplane Tree (GHT) [29] and Bisector Tree [14] tree are widely used alternatives. These structures are binary trees where each node of the tree is assigned to a pair of pivot elements (p_1,p_2). If the distance of the object to p_1 is smaller than the distance to p_2, then the object is assigned to the left subtree, otherwise it is sent to the right subtree. According to experiments [29], the GHT provides a better indexing structure than the usual vantage point trees.

A key element in the efficiency of indexing is the appropriate selection of the pivot elements. Based on the survey of [5], the following methods are usually used for pivot selection. The simplest solution is the random selection of the pivot elements. In this approach, more tests are run and the pivot set with best parameter is selected. The second method is the incremental selection method. In the first step of this algorithm, a p_1 with optimal fitness is selected. In the next step, the pivot set is extended with p_2, generated by a new optimization process where p_1 is fixed already. On this way, the pivot set is extended incrementally to the required size. The third way is the local optimization method. In this case, an initial pivot set is generated on some arbitrary way. In the next step, the pivot element with worst contribution is removed from the set and a new pivot element is selected into the set. This phase is repeated until a given quality level is reached.

The work [5] analyzed the pivot selection methods from the viewpoint of subtree pruning operation. Usually a heuristic approach is used in the applications. The core elements of the heuristics are the following rules: the pivot elements should be far from the other not pivot elements and they should be far from each others too. The paper concluded that the incremental selection method provides the optimal solution of this heuristics. An improved pivot selection method called Sparse Spatial Selection (SSS) is presented in [6]. The SSS method generates the pivot elements dynamically when a new outlier element is inserted into the object pool. The new incoming element is selected as a new pivot if it is far enough from the other pivot elements. A loss minimization method was proposed by [13] for optimization of the output index

structure, where the loss is measured as the real distance between the object and its nearest neighbor in the index tree.

The main goal of this paper is to analyze the pivot selection methods from a different viewpoint, namely from the viewpoint of balancing factor of the generated index tree. The object set to be indexed is represented with a distance matrix as only this method can preserve the distance relationship found in the domain space. The assumption in the investigation is, that initially this distance matrix is empty and the calculation of the distance value is a relatively high cost operation. This condition is usually met in the case of complex objects. The balancing factor is an important parameter of the traditional search trees. In the case of well balanced tree, the cost of search operation is low and stable [26]. Another aspect of the investigation is the cost reduction of distance computations during the selection of pivot set. The proposed method optimizes the process of pivot selection to achieve a balanced GHT tree using a minimum number of distance calculations. The investigation addresses the split operation of a GHT node when the bucket gets full. In this process, two new pivot elements should be selected. As the bucket contains only the objects of a single node, it can be assumed that the size of the object set is limited. Based on these consideration, the presented method is a combination of algorithms on distance matrices and direct pivot selection. The proposed method uses a interval-based distance value representation form in order to reduce the number of required distance computations. The performed tests demonstrate the efficiency of the presented method.

2 Distance matrix for representation of elements in GMS

There are two main different approaches for representation of objects in general metric space. The first method is based on the mapping of the objects into some Euclidean space as this representation form provides a large set of operators on the object set. The generation of aggregated elements or of sample objects, the construction of container objects and ordering of the objects by some of the dimensions are among the most frequently used operators. The applied mapping function is denoted with $\mu: GMS \rightarrow E_N$, where N denotes the dimensionality of the target Euclidean space.

The most widely used mapping algorithm is the Fréchet-embedding [4]. The algorithm is based on the following considerations. First, A_i $(i=1,..,N)$ groups of elements in GMS are selected as groups of pivot elements. The group A_i is assigned to the i^{th} coordinate in the target Euclidean space. The i^{th} coordinate of a target object u is calculated with

$$\mu(u)_i = min_{v \in A_i}\{d(u,v)\}. \quad (2)$$

Another mapping option is the application of a multidimensional scaling (MDS) [17] method. The initial positions in the target Euclidean space are generated randomly. The final positions are calculated in a iterative process which optimizes the stress mapping error function. The stress measure [17] is calculated with

$$s_\mu = \sqrt{\frac{\sum_{u,v}(D\big(\mu(u),\mu(v)\big) - d(u,v))^2}{\sum_{u,v} d(u,v)^2}} \qquad 2(3)$$

where

μ : the mapping function
u,v : objects in GMS
d: distance in GMS
D : distance in L_2.

To find the optimum positions, a gradient method is applied. The method is based on the analogy from physics, where each pair-wise connection can be treated as a spring. Another standard method is the FastMap mapping [10], which is based on the standard dimension reduction method in Euclidean space. The dimension value in a lower dimensional space can be calculated from the distance values in the higher dimensional space using the known cosine low. In this approach, two pivot elements are assigned to a target dimension. The new coordinate value is equal to

$$\mu(u)_i = \frac{d(u,p_{i1})^2 - d(u,p_{i2})^2 + d(p_{i1},p_{i2})^2}{2d(p_{i1},p_{i2})} \qquad (4)$$

where

p_{i1}: the first pivot element of i^{th} dimension,
p_{i2}: the second pivot element of i^{th} dimension.

This mapping uses the projection of (p_{i1},u) onto the line (p_{i1}, p_{i2}) as coordinate value $\mu(u)_i$. This projection can be calculated from the cosine law and applying the Pythagoras law. The distance values on the hyperplane perpendicular to the selected line (p_{i1}, $p_{i2)}$ are calculated with

$$d'(u,v) = \sqrt{d(u,v)^2 - (\mu_i(u) - \mu_i(v))^2} \qquad (5)$$

Taking two another pivot elements in this perpendicular space, the dimension values for the next dimension of the target Euclidean space can be generated on a similar way as it was presented in the previous steps. A main drawback of the presented mapping approach is that the distance values in the source metric space may significantly differ from the distance values in the target Euclidean space. For example, the Fréchet-embedding is no relation preserving, i.e. it is not ensured that

$$\forall a,b,c,d\colon d(a,b) \geq d(c,d) \rightarrow D\big(\mu(a),\mu(b)\big) \geq D(\mu(c),\mu(d)) \qquad (6)$$

There exist many object distributions in GMS, which can't mapped to a corresponding point set in Euclidean space. The Fig. 1 shows this fact with a simple example for object set with four objects. The given distance values are valid in GMS, but they are no valid in Euclidean space. The experiences show that for GMS with larger object sets only a small subset of valid GMS distance distribution can be represented without information loss in Euclidean space. Thus, the direct mapping of the objects into a Euclidean space causes a significant loss of information, the resulted object set is a rough approximation of the original object set.

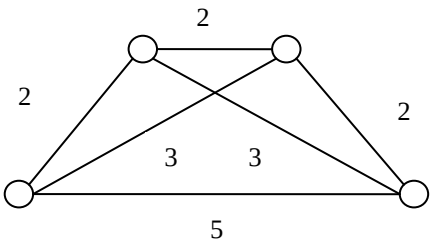

Fig. **1.** , Object set valid in GMS but not valid in Euclidean space

Another representation approach is the distance matrix form, where the distance values between every objects are saved without any modification in a matrix. Let $\mathcal{H} \subset \Re^{N\times N}$ denotes the set of distance matrices meeting the axioms of the distance functions. Let $\hat{d}$ denote the upper triangle part of $\hat{h}$ and $\mathcal{H}^u$ the set of these matrices. With corresponding mapping of indexes the formula (1) can be converted into the following form:

$$\begin{gathered} \forall\, d_{ij}, d_{jk}, d_{ki} \in \hat{d} \;\in\; \mathcal{H}^u : d_{ik} + d_{kj} - d_{ij} \geq 0 \\ \forall\, d_{ij} \in \hat{d} \in \mathcal{H}^u : d_{ij} \geq 0 \\ \mathcal{H}^u \subset \Re^{\binom{N}{2}} \end{gathered} \tag{7}$$

As it can be seen the set of valid distance matrices is equal to the solution set of the linear homogenous inequality system (7). In this formula we allow to have a zero distance value between any objects. This difference enables the investigation of degenerate cases where two objects may be overlapped, i.e. they are the same object. It follows from this fact that if

$$\hat{x}, \hat{y} \in \mathcal{H}^u, \alpha, \beta \in \mathcal{R}^+ \tag{8}$$

then

$$\alpha\hat{x} + \beta\hat{y} \in \mathcal{H}^u \tag{9}$$

is also met. Thus $\mathcal{H}^u$ is a convex cone in $\Re^{\binom{N}{2}}$ containing the zero element of $\Re^{\binom{N}{2}}$ too. A ray of $\mathcal{H}^u$ for direction $\hat{\mathrm{d}} \in \mathrm{H}^{\mathrm{u}}$ is defined as

$$\alpha\hat{d} \in \mathcal{H}^u, \alpha \in \mathcal{R}^+ \tag{10}$$

The direction $\hat{\mathrm{d}}$ is an extreme direction of a convex cone if it cannot be expressed as a conic combination of directions of any rays in the cone distinct from it:

$$\forall \hat{a}, \hat{b} \in \mathcal{H}^u, \alpha, \beta, \gamma \in \mathcal{R}^+, \hat{a} \neq \gamma\hat{d}, \hat{b} \neq \gamma\hat{d}: \alpha\hat{a} + \beta\hat{b} \neq \hat{d} \tag{11}$$

According to the theory of Klee [27], any closed convex set containing no lines can be expressed as the convex hull of its extreme points and extreme rays.

Given a matrix $\hat{A} = [\bar{a}_1, \bar{a}_k, \ldots, \bar{a}_k]$, a cone *C(Â)* and its polar cone *P(Â)* can be defined as follows [5]:

$$\begin{gathered} C(\hat{A}) = \left\{\bar{x} | \bar{x} = \sum\nolimits_{i=1}^{k} \beta_i \bar{a}_{i,} \quad \beta_i \geq 0\right\} \\ P(\hat{A}) = \left\{\bar{x} | \hat{A}^T \bar{x} \geq 0\right\}. \end{gathered} \tag{12}$$

The vector $\overline{w}$ is called as edge vector of the polar cone given in (3) if $\overline{w}$ is not positively covered by any vectors in *P(Â)* \ *C($\overline{w}$)*, where *C($\overline{w}$)* denotes the one dimensional cone generated by $\overline{w}$.

The theorem by Tamura [27] determines the conditions for a given vector $\overline{w}$ in *P(Â)* to be an edge vector of *P(Â)*. Assume that rank(*Â*) = M. Then the vector $\overline{w}$ with unit norm is an edge vector of *P(Â)* if and only if there exists subcollections $\{\bar{\mathrm{a}}_{\mathrm{i1}}, .., \bar{\mathrm{a}}_{\mathrm{iq}}\}$ and $\{\bar{\mathrm{a}}_{\mathrm{iq+1}}, .., \bar{\mathrm{a}}_{\mathrm{ik}}\}$ of $\hat{A}$ such that

$$\begin{gathered} \langle \overline{w}, \bar{a}_{ij} \rangle = 0, \qquad j = 1, .., q \\ \langle \overline{w}, \bar{a}_{ij} \rangle > 0, \qquad j = q+1, .., k \\ rank\left(\left[\bar{a}_{i1}, .., \bar{a}_{iq}\right]\right) = M - 1 \end{gathered} \tag{13}$$

and the polyhedral cone *P(Â)* can be expressed in terms of edge vectors as

$$P(\hat{A}) = \left\{\bar{x} | \bar{x} = \sum\nolimits_{i=1}^{q} \beta_i \overline{w}_i, \quad \beta_i \geq 0\right\} \tag{14}$$

if $P(\hat{A})$ does not contain any subspace. Unfortunately, the extreme rays of the metric cone can't be generated directly for larger N values as the number of extreme rays is a $O(2^{N^2})$[31].

In the distance matrix representation form, a useful aggregated property of the object set is the distance distribution function. As the experiences show the efficiency of the algorithms are significantly influenced by the characteristics of the distribution. In the test generations, three main types of distribution were selected: uni-polar, bi-polar and multi-polar distributions with uniform distribution within the clusters. The shape of the functions for the investigated uni-polar and bi-polar cases is shown in Figure 2.

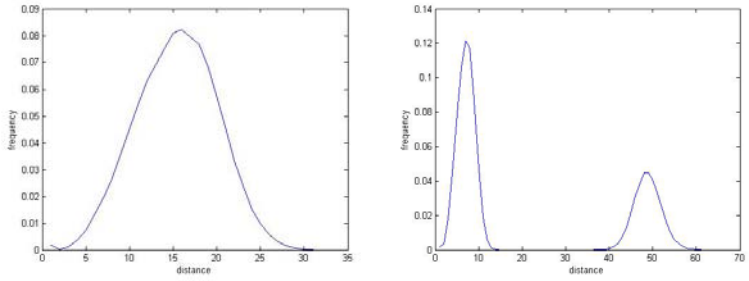

Fig. 2. , Distance distributions of uni-polar and bi-polar cases

3 Indexing methods in general metric space

The most important functionality of an information system is the retrieval of the required data or information items. The retrieval operation can be given as a function

$$f(D, \bar{q}) \tag{15}$$

where D denotes the domain of target objects and the vector $\bar{q}$ is the predicate vector. The result set of the query is defined with

$$R_{f(D,\bar{q})} = \{o | o \in D \wedge \bar{q}(o)\}. \tag{16}$$

In the case of GMS, the predicate refers only to the distance values between the objects:

$$\bar{q}(o) = \bar{q}(\{d(o,x) | \, x \in GMS\}). \tag{17}$$

One important characteristic of the different retrieval methods is the functionality of the matching operation [24]. The exact search uses an exact matching, while the neighbor search allows a fuzzy matching too. The term "fuzzy" means in this context that not an exact matching is required. For example, in [23], the category of fuzzy matching includes among others the substring matching or the string matching with regular expression.

The main goal of the indexing structure is to provide an efficient method for element retrieval. The first indexing methods were developed for elements in one-dimensional Euclidean space. The most widely used index structure of this kind is the B-tree [24] structure. The index structure provides a O(log(N)) cost for exact matching. To support interval matching, the base B-tree structure is extended with an additional pointer chain between the sibling nodes. The main benefit of the B-tree structure is that it provides a well-balanced structure with a stabile cost value. The retrieval from B-tree structure is based on the pair-wise comparisons of the objects. In the literature, there are methods for comparison-less structures too, like the family of the hash functions [18]. The main drawback of this structure is that it is very sensitive to the value distribution of the target objects.

In the case of multi-dimensional Euclidean space, more candidate index structures are used as there is no simple dominating index method. The kd-tree [24] structure uses a similar structure as the B-tree has, but every level corresponds to different dimensions. This tree structure provides a O(log(N)) for exact matching, but it has a significantly higher cost for interval matching. The R-tree [12] structure is based on a different concept. Every node corresponds here to a cube in the high dimensional space. The hierarchy of embedded cubes provides an efficient exact search with O(log(N)) cost, but the interval-based search may increase to linear cost.

In the case of general metric space, neither of the given methods can be used for object retrieval. Four main types of indexing methods can be distinguished for this domain. The first group contains so called ball-based partitioning methods [9]. As the ball can be defined in GMS too, this shape can be used to separate the set of objects into two distinct subsets: objects within a ball and objects outside of the ball:

$$\begin{aligned} R_{in,c} &= \{o | o \in D \wedge d(c,o) \leq r\} \\ R_{out,c} &= \{o | o \in D \wedge d(c,o) > r\}. \end{aligned} \tag{18}$$

The center of the ball is called usually pivot element of the index structure. The main benefit of the structure is that it can be implemented on a simple way. The drawback of the method is that the roles of the inner and outer clusters are asymmetric, i.e. there is an intra-cluster distance threshold (2r) for the objects within the inner cluster and there is no such threshold for the outer cluster. An improvement of the base ball-partitioning method is the M-tree index structure [7] which works similar to the R-tree structure. The M-tree provides a balanced structure, the algorithm for insertion of a new element is based on the concepts implemented in B-tree. An entry in the M-tree stores also the distance values between the parent and child pivot objects. This value can be used to prune some branches of the index tree in the case of neighbor search. The elimination of the branch is based on the following consideration:

$$d(a,b) > r_1, d(a,c) < r_2, r_1 > r_2 \rightarrow d(b,c) > r_1 - r_2. \tag{19}$$

Having a current closest object at a distance r_c, all subtrees with $r_p - r_d > r_c$ are eliminated, where r_p is the distance to the pivot object and r_d is the radius of the corresponding ball.

A conceptually different approach for object indexing is the family of computation methods based only on the distance matrix. In the AESA [21] algorithm, the distances between every pairs of objects are known and thus every objects can be considered as a candidate pivot element The method provides the best query results for small object sets but it can't be applied to larger sets because of the $O(N^2)$ number of distance computations.

Another important approach is the Generalized Hyperplane Tree (GHT) partitioning [29]. In this approach, two pivot elements are applied for separation of the object set. The first cluster contains objects near to the first pivot element, while the second cluster includes the elements closer to the second pivot element. Basically, in this approach, both clusters have symmetric roles, thus the element distribution is more suitable for recursive partitioning than the ball-based architecture. Another benefit of the GHT structure is that the generated regions do not overlap unlike the ball-based partitioning methods. The GHT structure stores also distance information in the tree nodes, the maximum distance to the pivot element within a cluster is saved as a cluster parameter. Having a range query with a radius r_q, the right subtree at the current node should be parsed if

$$d(q, p_r) - r_q \leq d(q, p_l) + r_q \tag{20}$$

is met. In this case, the intersection of the neighborhood with radius r_q and r_q.The left subtree is traversed if

$$d(q, p_l) - r_q \leq d(q, p_r) + r_q \tag{21}$$

is met. The symbols p_l and p_r denote the pivot elements of the left and right hyperplanes; q is query object. In some cases, both branches should be processed, causing an increased query cost. The cost of tree construction is in $O(N \cdot logN)$ [29].

The literature contains some extensions of the base GHT structure. In [3], the base structure was extended to a distributed architecture. The proposed GHT* structure is optimized for range query and it is scalable to manage huge amount of objects without a central directory. The nodes of the tree are distributed on a cluster of servers. The synchronization of data content during the search operation is performed with a message forwarding technique. Later, a more refined dynamic hyperplane index structure was presented in [22]. The paper invented a fully dynamic data structure. The proposed ghost hyperplane technique uses a disk-memory bound structure and provides high scalability. According to the test experiments, the hyperplane-based dynamic partitioning requires less distance computation operations but the disk utilization is less efficient than the ball-based partitioning methods.

Another GHT alternative is the multi-pivot variant tree where *m* pivots divide the space into *m* Voronoi-like partitions. The parameterized GHT structure was proposed in [19]. This GHT structure enables a flexible region topology, where the borders may differ from the plane shape. The separation condition is given with

$$\begin{aligned} R_A &= \{o | o \in D, d(p_A, o) < d(p_B, o) + c\} \\ R_B &= \{o | o \in D, d(p_A, o) \geq d(p_B, o) + c\}. \end{aligned} \tag{22}$$

It can be mentioned here, that the hyperplane partitioning technique is used also in the case of Euclidean domain. In this case, the hyperplane is defined on a way different from the GMS approach. For example in [8], the hyperplane is the optimal separator plane closest to the selected pivot elements.

An important characteristic of every partitioning structure is the balancing factor. The cost of a query operation depends on the actual balancing factor of the tree, the optimal cost is yielded in the case of perfect balancing. The balancing factor can be quantified with some measures. The usual approach is to calculate the difference of the subtree heights. If the leaf nodes can store variant number of elements, the difference of the weighted heights is used as balancing measure. Another approach is presented in [8], where a δ-safe property is defined. The partitioning is δ-safe, if both open half-spaces contain at least $\delta \cdot (N - N_d)$ objects, where N_d denotes the number of objects for constructing the separator hyperplane.

4 Selection of pivot elements

It is known that the efficiency of indexing methods depends significantly on the position of the pivot elements [21], thus the appropriate selection of the pivot elements is a crucial optimization component. The usual measure to calculate the fitness of a pivot-set is the average of maximum distance differences [5]:

$$\mu_{p_1,..,p_M} = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{j=1}^{N} max_{k \in 1..M} \{ |d(x_i, p_k) - d(x_j, p_k)| \} \tag{23}$$

where M denotes the number of pivot objects. It can be shown that $\mu_{p_1,..,p_M}$ is always between 0 and the average object distance. This statement follows from the facts, that a)

$$|d(x_i, p_k) - d(x_j, p_k)| \geq 0 \rightarrow max\{|d(x_i, p_k) - d(x_j, p_k)|\} \geq 0 \rightarrow \mu_{p_1,..,p_M} \geq 0; \tag{24}$$

b) and using the triangle inequality, we get that

$$d(x_i, x_j) \geq \left| d(x_i, p_k) - d(x_j, p_k) \right| \rightarrow \mu_{p_1,..,p_M} \leq \frac{\sum_i \sum_j d(x_i, x_j)}{N^2} = \bar{d}. \quad (25)$$

The $\mu_{p_1,..,p_M}$ measure can be used to approximate the search cost of the nearest neighbor search. As it was shown, both child hyperplanes of a node should be tested if

$$\begin{aligned} d(q, p_r) &\leq d(q, p_l) + 2r_q \\ d(q, p_l) - 2r_q &\leq d(q, p_r). \end{aligned} \quad (26)$$

This means that in this case

$$|d(q, p_r) - d(q, p_l)| \leq 2r_q. \quad (27)$$

Considering ξ as stochastic variable of the neighborhood radius, we get that

$$P\left(|d(x, p_r) - d(x, p_l)| \leq 2r_q\right) = F_\xi(2r_q). \quad (28)$$

The multiplication of the variable with $1/\bar{d}$ preserves the monotony of F(), thus the smaller is the search cost the smaller is $2r_q/\bar{d}$. This means that for large $\bar{d}$, i.e. $\mu_{p_1,..,p_M}$ value, the search cost is low.

On the other hand, the $\mu_{p_1,..,p_M}$ measure does not provide an appropriate measure for our GHT-balancing problem, as there is no strong correlation between the $\mu_{p_1,..,p_M}$ measure and the balancing factor of the tree. As the main goal of the investigation is to provide a well-balanced distribution of the objects, a new balancing criterion measure is introduced with

$$\mu = 2 \cdot \frac{min\{|B_L|, |B_R|\}}{|B_L| + |B_R|} \quad (29)$$

where B_L and B_R denote the left and right side subtrees. This measure corresponds to the global objective function. In the optimal case, the value of μ is equal to 1. If $\mu = 0$ then all objects are assigned to one of the child branches.

For implementation of a μ-optimal partitioning algorithm, a novel extension of the multi-phase pivot selection method [30] is applied. The proposed method is the combination of the distance-maximization heuristic and the hill-climbing method. Despite the simplicity, the hill-climbing method is considered as a very efficient and low cost local optimization algorithm. In the first phase the usual heuristic step is applied: the object pair with largest intra-distance will be selected. In order to minimize the computation cost, only an approximation is performed in the followings steps to find the object pair with maximum distance:

- random selection of an object p_0
- selection of p_1 with $d(p_1,p_0) \rightarrow$ maximum
- selection of p_2 with $d(p_1,p_2) \rightarrow$ maximum.

The object pair (p_1,p_2) is selected as initial pivots. Based on the experiences, the fitness of the random selection largely depends on the object distribution. In the case of uniform distribution it provides a relatively good result but if the distribution is bipolar a poor results is yielded. To improve the efficiency a local optimization process is performed in the second phase. The main steps of this phase are the followings, where p1,p2 denote the current best pivot candidate pair:

```
1: mumax = mu (p1,p2);
2:   selection of p3 where mu(p1,p3) is maximum;
3:   selection of p4 where mu(p4,p2) is maximum;
4:   mu = max (mu(p1,p3),mu(p4,p2));
5:   if mu > mumax then
6:     mumax = mu;
7:     replace the old pivot pair with the new one;
8:     go back to step 1
9:   else
10:     terminate the procedure;
11:   end;
```

In the first line of the algorithm, the current $\mu(p_1,p_2)$ value is stored into variable mumax. In line 2 and 3 two new candidate elements are selected with

$$p_3 = \operatorname{argmax}_p\{\mu(p,p_1)\}$$
$$p_4 = \operatorname{argmax}_p\{\mu(p,p_2)\} \tag{30}$$

If one of the new objects yields a better balancing factor, it will selected into the candidate pivot pair.

In the tests, three algorithms were compared. The first algorithm is the brute force search where for every object pair (p_1,p_2) the $\mu(p_1,p_2)$ value is evaluated and the best pair is selected. This method provides a global optimum but it requires the largest execution cost. The second method is the random pivot selection method with maximum intra-distance criteria. The third method is the proposed local optimization algorithm. Regarding the cost factors of the tested methods, the brute-force method has the largest cost with

$O(N^3)$

as the number of measure calculation is equal to $O(N^2)$ and the cost to determine the μ measure is in O(N). The proposed combined method belongs to the

$O(N^2)$

complexity class, as in a optimization cycle, only O(N) μ measure calculations are executed. The random search method requires only

O(N)

cost.

In the tests, two parameters were measured: the efficiency factor μ and the execution costs *t*. The test results are summarized in Table1. The first table (Table 1a) is for the uni-polar case, the second table (Table 1b) shows the bi-polar case with parameter value: $|B_L|/|B_R|=4$. For the multi-polar cases, the results always lay between these values.

The Fig. 3 shows the comparison of the μ values for the random and local search methods for bi-polar distribution. It can be seen from the results that the random search method work weak in the case of bi-polar distribution and can work well in uni-polar case. The local optimum search method provides always a good result and it requires significant less time than the brute force algorithm.

Table 1.a

sample size	brute force		random		local optimization	
	μ	t	μ	t	μ	t
300	1	0.910	0.89	0.001	1	0.018
400	1	2.312	0.93	0.001	0.99	0.026
500	1	4.321	0.93	0.001	1	0.054
600	1	7.916	0.95	0.001	1	0.081
700	1	12.532	0.94	0.001	1	0.120
800	1	19.166	0.91	0.001	0.98	0.167
900	1	27.331	0.93	0.001	0.99	0.188
1000	1	38.182	0.94	0.001	1	0.221
1200	1	63.529	0.93	0.001	1	0.340
1400	1	105.117	0.94	0.001	0.98	0.442

Table 1.b

sample size	brute force		random		local optimization	
	μ	t	μ	t	μ	T
300	1	0.932	0.40	0.001	1	0.017
400	1	2.212	0.36	0.001	1	0.024
500	1	4.413	0.39	0.001	0.99	0.054
600	1	7.806	0.39	0.001	1	0.077
700	1	12.731	0.38	0.001	1	0.112
800	1	18.463	0.44	0.001	1	0.163
900	1	27.625	0.44	0.001	0.99	0.181
1000	1	37.458	0.42	0.001	1	0.218
1200	1	62.715	0.41	0.001	0.99	0.351
1400	1	103.563	0.41	0.001	1	0.438

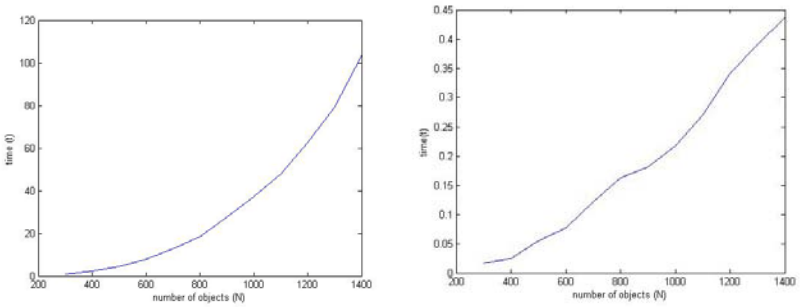

Fig. 3. Time cost of the brute force and local optimum search methods

5 Interval model of distance calculations

The cost models presented in the previous section, are valid only for the case when all the *d(x, y)* distance values are already known. On the other hand, in real applications the generation of the distance matrix is also a high cost operation. For an object set with *N* objects, the distance matrix contains $\binom{N}{2}$ distance values, thus the generation of the matrix is an $O(N^2)$ cost operation. As the calculation of the distance value for complex object is a high cost operation, the reduction of the redundant distance values is an important optimizations step.

The first question of this research phase was to investigate how the single distance values restrict each other within the object set. As the distance value between two objects is constrained by the triangle inequalities of the metric function, the values already in the matrix will constrain the values not already filled in. Every new value entered into the matrix will reduce the uncertainty on the still empty positions.

There are many approaches in the literature to manage value uncertainty. One dominant option is the usage of stochastic variables instead of strict scalar values. In these approaches [15], a distribution function is defined for every stochastic variable. Another widely used approach is the fuzzy modeling of the values. In fuzzy logic [28], a membership function is used instead of the single value. As it can be seen, in both cases, a complex description is required. The main drawbacks of these approaches are that it requires higher computational costs and a more precise knowledge on the domain. If a low cost solution is required then a simplified model should be used. A low cost solution is the application of interval arithmetic [11]. The interval value representation requires only two strict values instead of detailed function description. The model was tested in many application areas. For example in [25], the domain of linear interval number programming was studied, where the coefficients are all interval numbers.

In GMS, every object triplet can be represented by a triangle with sides (*a*, *b*, *c*), where the sides denote the distances between the objects. According to the triangle inequality, the range condition for a given side *c* can be given with

$$\begin{gathered} c \leq a + b = c_{max} \\ c \geq \max(a,b) - \min(a,b) = \max(a-b, b-a) = c_{min}. \end{gathered} \tag{31}$$

Thus, for every object pair, the distance between the objects can be given with a range (c_{min}, c_{max}). Using the interval value for all sides of the given triangle, the range inequality for side *c* has the following form:

$$\begin{gathered} c_{max} = a_{max} + b_{max} \\ c_{min} = \max\left(a_{min} - b_{max}, b_{min} - a_{max}, 0\right). \end{gathered} \tag{32}$$

In order to avoid the upper range explosion, an upper threshold D_M is set for every valid distance value, as it is usual in interval arithmetic [11]. Taking this threshold into account, the range constraint can be transformed into

$$\begin{gathered} c_{max} = \min\left(a_{max} + b_{max}, D_M\right) \\ c_{min} = \max\left(a_{min} - b_{max}, b_{min} - a_{max}, 0\right). \end{gathered} \tag{33}$$

If a distance relation *c* takes part in *M* triangles, then every triangle yields a range constraint for *c*. For a relation (*i,j*) in GMS, the single constraints can be aggregated into the following form:

$$\begin{gathered} d_{i,j,max} = min_k(\min\left(d_{i,k,max} + d_{j,k,max}, D_M\right)), \\ d_{i,j,min} = max_k(\max\left(d_{i,k,min} - d_{j,k,max}, d_{j,k,min} - d_{i,k,max}, 0\right)). \end{gathered} \tag{34}$$

It can be seen, that the interval is different from the base range, if exists a containing triangle where both other sides have non-base range. These constraints imply the rule

$$b_{min} = 0, b_{max} = D_M \;\rightarrow\; c_{min} = 0, c = D_M. \tag{35}$$

This rule is derived from the following consideration:

$$\begin{gathered} c_{min} = \max(a_{min} - b_{max}, b_{min} - a_{max}, 0) = \max(a_{min} - D_M, -a_{max}, 0) = 0 \\ c_{max} = \min(a_{max} + b_{max}, D_M) = \min(a_{max} + D_M, D_M) = D_M. \end{gathered} \tag{36}$$

The interval distance model can be used to measure the uncertainty at a given saturation state of the distance matrix. The first approach is to measure directly the lengths of the intervals. Initially, when no distance value is known yet, every value is given with $[0,D_M]$ where D_M denotes the largest possible value. In this case, the uncertainty is the largest. If a distance value is set to given value *v*, the corresponding interval contains only one element: $[v,v]$. For this relation, the uncertainty is zero. To indicate the level of aggregated uncertainty of the whole distance matrix, an ϕ measure is introduced on the following way:

$$\phi_{avg} = \frac{\sum_{i,j} \phi_{i,j}}{N^2}$$
$$\phi_{i,j} = d_{i,j,max} - d_{i,j,min} \quad (37)$$

The ϕ measure is the largest at the initial state and it decreases when an previously unknown distance is set to shorter range value. For a given relation d_{ij}, not every (d_{ij}, d_{jk}, d_{ik}) triangle has the same importance. The triangle with extreme value is the dominant one to determine the actual range values. As it can be seen, the range values at a given relation change monotonously. Thus the total uncertainty value function must be also monotonous.

For investigation of the ϕ_{avg} measure function, it is assumed that from (*N*-2) container triangles of a given relation d_{ij}, *K* triangles are near-dominant ones. The shape of the measure function depends on the position when one of these dominant triangles is selected first. The possibility that in the *k-th* step is a dominant is first selected:

$$\frac{\binom{N-K}{k} k! \binom{K}{1} (N-k-1)!}{N!} \quad (38)$$

Based on the formula, it can be seen that if *K* is small, the ϕ_{avg} measure function will decrease smoothly with a quasi linear shape. Otherwise, if the distance distribution is uniform with a small distribution, the function will decrease steeply at the beginning of the matrix feeding process.

Considering the efficiency of the proposed interval-based distance matrix, the structure provides a maximum information content. This representation form stores more information on the distance values than the strict value representation. The strict value mode is a special case of the interval mode. Thus, the benefit of the proposed structure is the information gain compared with the traditional representation method. The uncertainty factor could be reduced to 40% in our test experiences (see Fig 6.-7.). This reduction gain depends on the current saturation level of the distance matrix. The weakness of the proposed method is the increased operation costs. The proposed model performs a value interval adjustment process. The setting or modification of a given distance element in the matrix triggers the modification of other matrix elements. As only the neighboring relations are affected by a value modification, the cost of the triggered adjustment process is in *O(N)*. Although this linear cost can be reduced using optimization methods presented in the next section, the interval based representation method is not superior in all situations. The model is suitable for those cases where the cost of distance calculation is high and the saturation factor of the matrix is relative low.

6 Optimization of the pivot selection method

As it was shown, the execution cost of the base local optimization method has a $O(N^2)$ characteristic. Thus some additional modules were included into the algorithm to reduce the cost value. The implemented reduction methods relate to the calculation of the μ value as this module has the largest cost portion within the pivot search algorithm. The first optimization step relates to the reduction of the candidate set in selection of p_3 and p_4. In the initial version, all objects of the domain belong to the candidate set. On the other hand, it follows from the triangle inequality that if

$$min_k\{|d(x_i, x_k) - d(x_j, x_k)|\} > d(x_i, x_j) \tag{39}$$

is satisfied then the partitioning sets of x_i and x_j are the same, i.e. the left (right) subtrees are the same for both elements. Thus if x_i is already tested and x_j meets the condition (39) then the testing of x_j can be omitted.

In the next table, the cost reduction factor of this elimination step is shown for different object distributions. As it can be seen this step is effective only if the distribution is bi-polar. The reason of this experience is the fact: the smaller is the relative distance d_{ij} the higher is the chance that inequality (39) can be used for test elimination. In the case of bi-polar distribution the chance to have large distance differences is greater than in the case of uni-polar distribution.

Table 2. Reduction values for different distributions

distribution	reduction factor (in percent)	
	average	deviation
uni-polar	0.2%	0.04%
bi-polar	36.4%	5.4%

The second method for candidate set reduction is the application of sampling technique instead of full scan of the objects. In this method the μ is calculated with

$$\mu = 2 \cdot \frac{min\{|B'_L|, |B'_R|\}}{|B'_L| + |B'_R|} \tag{40}$$

where B' denotes subset generated by random sampling. The next table (Table 3) summarizes the achieved accuracy at different sample sizes (reduction levels). The table contains the accuracy error values in percentage.

Test data show that sampling of the object distribution has some similarity with the standard theories of determining the optimal sample size for normal distributions. For example, the Cochran's formula [1] gives the sample size as

$$\gamma = \frac{t^2 \cdot s}{d^2} \tag{41}$$

where

t : value for selected alpha level for each tail
s: estimation for variance
d: acceptable margin of error

The formula of Krejcie [16] provides a different approach:

$$n = \frac{\chi^2 \cdot N}{d^2 \cdot 4 \cdot (N-1) + \chi^2} \tag{42}$$

where χ^2 denotes the Chi-square of the given confidence level. The optimal sample size depends on many factors and its value changes only very slow for increase of *N*. For example the optimal sample size for *N* = 1000 lies between 210 and 270. In our experiment, the optimal sample size is about 140 for *N* = 1000.

Table 3. The efficiency of sampling for different sample sizes

sample size	error for set A (N=200)		error for set B (N=1000)	
	average	deviation	average	deviation
sqrt(N)	42%	27%	19%	16%
2 sqrt(N)	30%	21%	17%	13%
4 sqrt(N)	15%	7%	7%	6%
8 sqrt(N)	13%	8%	7%	5%
16 sqrt(N)	-	-	7%	5%
24 sqrt(N)	-	-	6%	5%

In the computation of the μ fitness value, the distances from a given object *x* to both pivot objects p_1, p_2 are considered to check which pivot is closer to *x*. On the other hand, the distance value calculation can be omitted in some situations. Let p_1, p_2 denote the current pivot candidate objects. Let q denote the current object to be tested. The test returns 1 if q is close to p_1, otherwise it returns 2. It is assumed that exists a *r* object for which the distances *d(q,r)* and $d(p_2,r)$ are already known. The distance $d(p_1,q)$ is known also. It follows from the triangle inequalities that

$$|d(p_2,r) - d(q,r)| \leq d(p_2,q) \leq |d(p_2,r) + d(q,r)|. \tag{43}$$

Thus if

$$d(p_1,q) < |d(p_2,r) - d(q,r)| \tag{44}$$

then

$$d(p_1,q) < d(p_2,q). \tag{45}$$

If

$$d(p_1, q) > |d(p_2, r) + d(q, r)| \tag{46}$$

then

$$d(p_1, q) > d(p_2, q). \tag{47}$$

Thus in this cases, the object q can be assigned to the corresponding subset without calculating the actual $d(p_2,q)$ distance value.

The next optimization module relates to the interval value distance model. In the base model, the modification of a given relation (i,j), will trigger the recalculation of the range borders at the neighboring relations (i,k) and (j,k) for every k value. This step requires a $O(N)$ cost as number of possible k values is equal to (N-2) and at a given k value, the threshold test takes $O(1)$ elementary operations. The only way to reduce the total cost is to eliminate the tests in some container triangles. In the proposed system, a treap [20] structure was implemented for test reduction. The treap structure is a combination of a search tree and a heap structure. Every node has two base attributes: p and k. The symbol p denotes a priority value and k stores the key values. If x is the left child node of x then

$$x_p \leq y_p \, , x_k \; \leq \; y_k \tag{48}$$

is met; if x on the left side of the parent, then

$$x_p \leq y_p \, , x_k \; \geq \; y_k \tag{49}$$

holds. In the implementation, the priority attribute contains the required minimum change of the base relation distance (d_{ij}) in order to became a dominant relation for a given target relation (d_{ik} or d_{jk}). The reduction is based on the following concept: if the change at the base relation is less than the p value then the subtree under the target element can be eliminated from the processing. The k value corresponds to the identification of the relations.

The cost analysis showed that the direct triggering of the recalculations provides a near optimum solution as the calculation cost are relatively low compared with the costs of the administration of the additional status description data.

7 Implementation and test results

In the tests, the average ϕ and the minimum ϕ values were investigated during the generation of the distance matrix. It is clear, the more values are set the less is the uncertainty. The Figures 3 - 5 show the average ϕ value for increasing number of set values (x-axis) for both the uni-polar and the bi-polar object distributions. As the figure demonstrates in the case of bi-polar distribution the average uncertainty is less than in the case of uni-polar distribution.

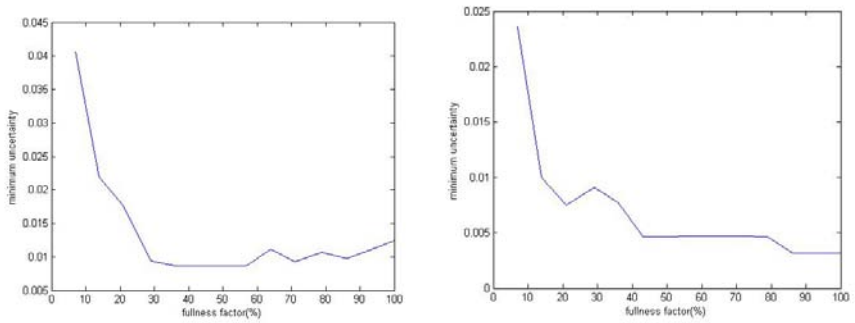

Fig. 3. Minimum uncertainty values for uni-polar and bi-polar distributions

The Fig 3 shows the minimum, not zero ϕ values of the matrix. Based on these results, it can be seen that the smallest value interval is equal to some percents of the average distance value. Thus, if a given level of uncertainty is allowed, some of the distance calculations can be eliminated.

For the case when the distance values are stored with interval values a new definition of the μ fitness measure is introduced. In this approach, an object may belong to both sides with given certainty. Let p_1, p_2 denote the pivot objects and q denotes the current object to be tested. Let *E1* denote the event that q is closer to p_1 than to p_2 and *E2* is the event that q is closer to p_2 than to p_1. If the $d(q,p_1)$ distance has the value $[va_1, vb_1]$ and $d(q,p_2)$ is equal to $[va_2,vb_2]$, then the probability of *E1* and of *E2* can be calculated with the method of geometric probability. The area of valid value pairs is a rectangle with sides corresponding to the intervals. The set of value pairs belonging to *E2* is a half-plane upper the line $y = x$.

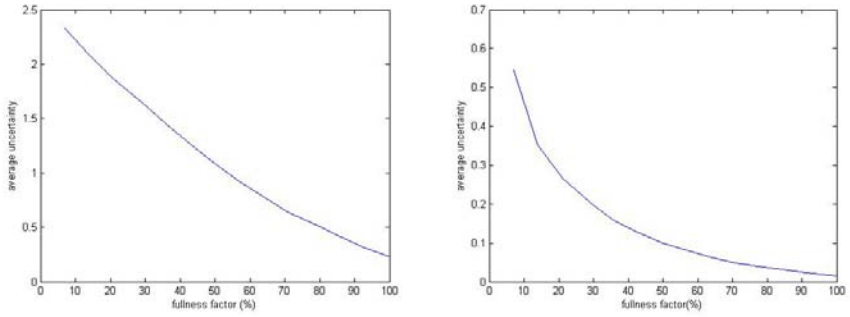

Fig. 4. Average uncertainty values for uni-polar distribution without interval adjustment and with adjustment

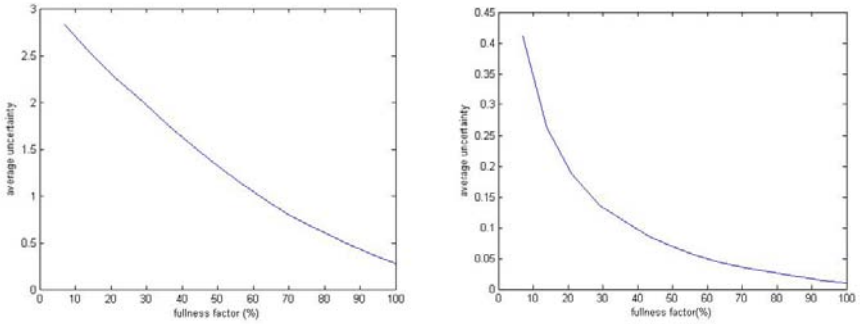

Fig. 5. Average uncertainty values for bi-polar distribution without interval adjustment and with adjustment

Let $p_i(E1)$ denote the probability that the i-th object belongs to the area of p_1. The $p_i(E2)$ is defined on similar way. It can be easily verified that

$$p(E1) + p(E2) = 1 \tag{50}$$

for every object. Based on the previous definitions, the μ value is calculated with

$$\begin{aligned} \mu &= 2 \cdot \frac{min\{P_1, P_2\}}{P_1 + P_2} \\ P_1 &= \sum_{i=1}^{N} p_i(E_1) \\ P_2 &= \sum_{i=1}^{N} p_i(E_2) \end{aligned} \tag{51}$$

The redefined fitness function is a generalization of the base fitness function as it yields the same value for the strict cases when $p_i(E1)$ or $p_i(E2)$ is equal to 1. Using the redefined fitness function, the presented pivot selection algorithm can be executed on the interval-based distance matrix too.

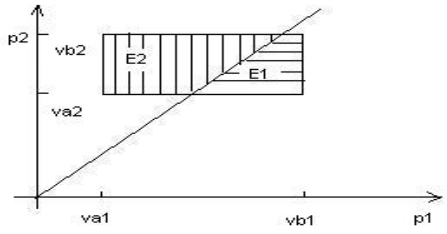

Fig. 6. Geometric probability of subtree assignment

The Table 4 summarizes the test results for the interval-based matrix and it shows the comparison between the strict-valued and the interval-valued matrix approaches.

Table 4. Efficiency of the interval-based approach

proportion of known distances	fitness of random selection	fitness of IV selection
0.01%	.65	0.65
3%	.66	0.81
16%	.62	0.96

The performed tests show that the proposed pivot selection with interval-based distance representation dominates the other methods in accuracy and time if the fullness value of the distance matrix is between 3% and 35%. If the fullness factor is lower then only few information is available and the random selection can provide the same result with less computational cost. Otherwise, if the fullness factor is higher than 35%, then the set of already exactly known distances is enough to get a good approximation for the optimal pivot selection. In the given range of the fullness factor, the interval-based model can provide so much additional information that improves the pivot selection significantly.

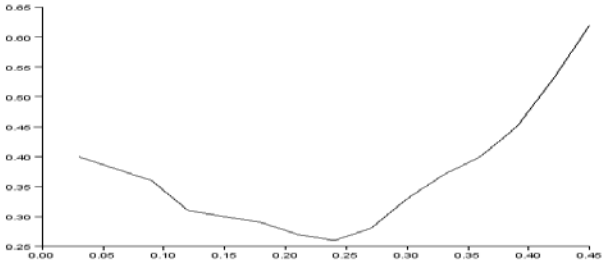

Fig. 7. Dependency of the efficiency from the fullness factor for the interval-based approach

As the result table Table 4, shows, the efficiency of the interval-valued (IV) method depends on the proportion of the known distances that means on the uncertainty of the distance matrix. It yields in good fitness value if the uncertainty of the matrix is low. The exact and formal analysis of this relationship is the goal of further investigations. In the tests, a 16% covering rate resulted in a well balanced tree with a 0.96 balancing factor.

8 Conclusions

The paper presented a detailed analysis of optimal pivot selection in general metric space from the viewpoint of index tree balancing. The analysis focused on the GHT index tree assuming that an index tree node contains a moderate number of objects. In the investigation, two main object distributions were tested: the uni-polar and the bi-polar distributions. The paper proposes a combined heuristic and local search optimization method for selection of pivot objects. For reduction of the search algorithm, some novel optimization methods were introduced. One of the cost reduction methods refers to eliminating of object tests within the calculation of balancing factor. Another important goal is to reduce the number of distance calculations in the object set. The dependencies between the distance values are analyzed in order to eliminate the redundant distance values. Another important reduction method is the application of interval values instead of strict values in order to manage the uncertainty of the distance values. The performed analysis and tests show that the proposed modification improves the efficiency of the standard methods significantly.

Acknowledgements

This research was supported by the Hungarian National Scientific Research Fund Grant OTKA K77809.

References

1. Bartlett, J., Kotrlik, J., Higgins, C.: Organizational Research: Determining Appropriate Sample Size in Survey Research, Information Technology Learning and Performance Vol. 19, 43-51 (2001)
2. Batko, M., Gennaro, C., Zezula, P.: Similarity grid for searching in metric spaces. DELOS Workshop: Digital Library Architectures, LNCS vol 3664, 25–44. (2005)
3. Batko, M., Zezula, P., Gennaro. C.: Scalable Similarity Search in Metric Spaces, Proc. of DELOS Workshop, 213-224 (2004)
4. Bourgain, J.: On Lipschitz embedding of finite metric spaces in Hilbert space, Israel J. Math, 52. (1-2), 46-52 (1985)
5. Bustos, B., Navarro, G., Chavez, E.: Pivot selection Techniques for Proximity Searching in Metric Spaces, Journal Pattern Recognition Letters, 2357-2366 (2003)
6. Bustos, B., Pedreira, O., Brisaboa, N.: A Dynamic Pivot Selection Technique for Similarity Search, Proceedings of ICDE Workshop, 394-400 (2008)
7. Ciaccia, P., Patella, M., Zezula, P.: M-tree An efficient access method for similarity search in metric spaces, Proc. of 23rd Conference on Very Large Data Bases, 426-435 (1997)
8. Devroye, L., King, J., McDiarmid, C.: Random Hyperplane Search Trees, SIAM Journal on Computing, 38(6), 2411-2425 (2009)
9. Dohnal, W.: Indexing Structures for Searching in Metric Spaces, PhD Thesis, Masaryk University, Brno (2004)
10. Faoutsos, C., Liu, K.: FastMap, A Fast Algorithm for Indexing, Data Mining and Visualization of Traditional and Multimedia Datasets, Proc of SIMGMOD '95, 163-174 (1995)

11. Ganessan, K: On arithmetic operations of interval numbers, International Journal of Uncertainty, Fuzziness and Knowledge-based Systems, 13(6), 619-631 (2005)
12. Guttman, A: R-trees A dynamic index structure for spatial searching, Newsletter ACM SIGMOD Records, 14(2), 47-57 (1984)
13. Henning, C., Latecki, J.: The choice of vantage objects for image retrieval, Proc. of Pattern Recognition 2003, 2187-2196 (2003)
14. Kalantari, J., Iraj, McDonald:. A data structure and an algorithm for the nearest point problem. IEEE Transactions on Software Engineering, 9(5), 631–634 (1983)
15. Kall, P.: Stochastic programming. European Journal of Operational Research 10, 125–130 (1982)
16. Krejcie, R., Morgan, D.: Determining sample size for research activities, Educational and Physhological Measurement 30, 607-610 (1970)
17. Kruskal, J., Wish, M.: Multidimensional Scaling, SAGE Publications, Beverly Hills (1978)
18. Litwin, W.: Virtual hashing: a dynamic changing hashing, Proc. of 4th Conference on Very Large Data Bases, 517-523 (1978)
19. Lokoc, J., Skopal, T.: On Applications of Parametrized Hyperplane Partitioning, Proceedings of SISAP Istambul (2010)
20. McCreight, E.M.: Efficient algorithms for enumerating intersecting intervals and rectangles, Technical report, PARC CSL-80-9, Xerox (1980)
21. Mico, L., Oncina, J., Vidal, E.: A new version of the nearest neighbor approximating and eliminating search with linear preprocessing time and memory requirements, Pattern Recognition Letters, 15, 9-17 (1994)
22. Navarro, G., Uribe-Paredes, R.: Fully Dynamic Metric Access Methods based on Hyperplane Partitioning, Journal Information System, 36(4), 734-747 (2011)
23. Oracle Text, Technical white paper, Oracle Comp, http://www.oracle.com/technetwork/database/enterprise-edition/11goracletexttwp-133192.pdf (2007)
24. Ottmann, T., Widmayer, P.: Algorithmen und Datenstrukturen, BI Verlag (1993)
25. Sengupta, A., Pal, T.K., Chakraborty, D., Interpretation of inequality constraints involving interval coefficients and a solution to interval linear programming. Fuzzy Sets and Systems 119, 129–138 (2001)
26. Sprugnoli, R.: Randomly balanced binary trees, Calcolo 17, 99-117 (1981)
27. Tamura, K.: A Method for Constructing the Polar Coneof a Polyhedral Cone, with Applications to Linear Multicriteria Decision Problems, Journal of Optimization Theory and Applications, 19, 547-564 (1976)
28. Tanaka, H., Ukuda, T.,Asal ,K. , On fuzzy mathematical programming. Journal of Cybernetics 3 , 37–46. (1984)
29. Uhlmann, K.: Satisfying general proximity similarity queries with metric trees, Information Processing Letters, 40, 175-179 (1991)
30. Veltkamp, R., van Leuken, R., Typke, R.: Selecting vantage objects for similarity indexing, ACM Transactions on Multimedia Computing, Communications and Applications, Vol 7 (3), Article 16 (2011)
31. Versik, A.: Distance matrices, random metrics and Urysohn space, MPI-2002-8 (2002)
32. Zezula, P., Amato G, Dohnal, V., Batko, M: The metric space approach, Advances in Database Systems, Springer-Verlag, Heidelberg (2006)