Consiglio Nazionale delle Ricerche

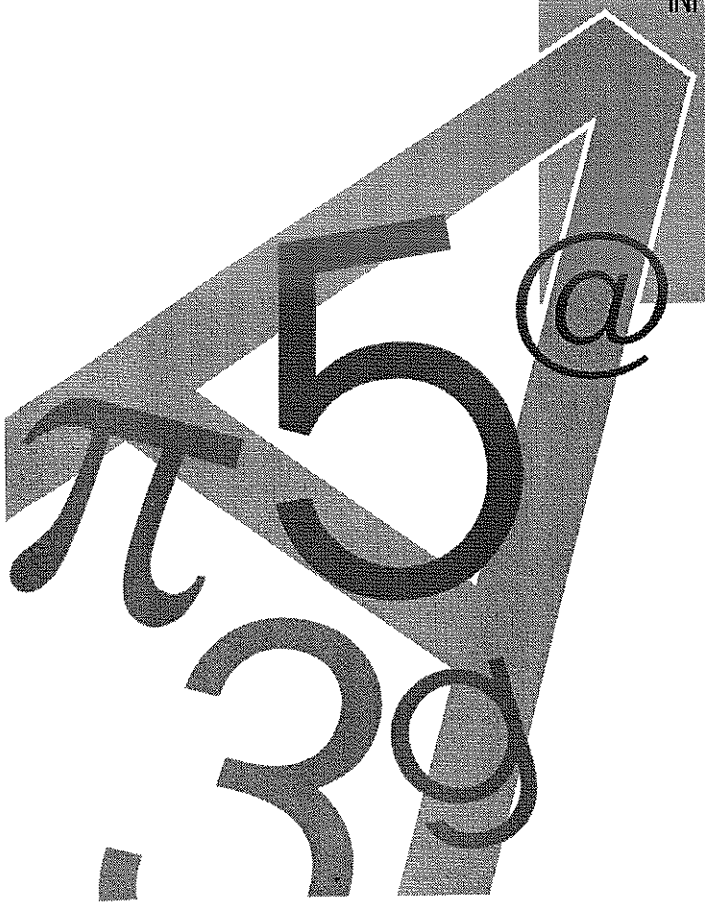
Technical Report

Computing Proximity of Metric Regions

Giuseppe Amato, Fausto Rabitti, Pasquale Savino, Pavel Zezula

I.E.I. B4-16-09-00

I.E.I.
ISTITUTO DI
ELABORAZIONE DELLA
INFORMAZIONE



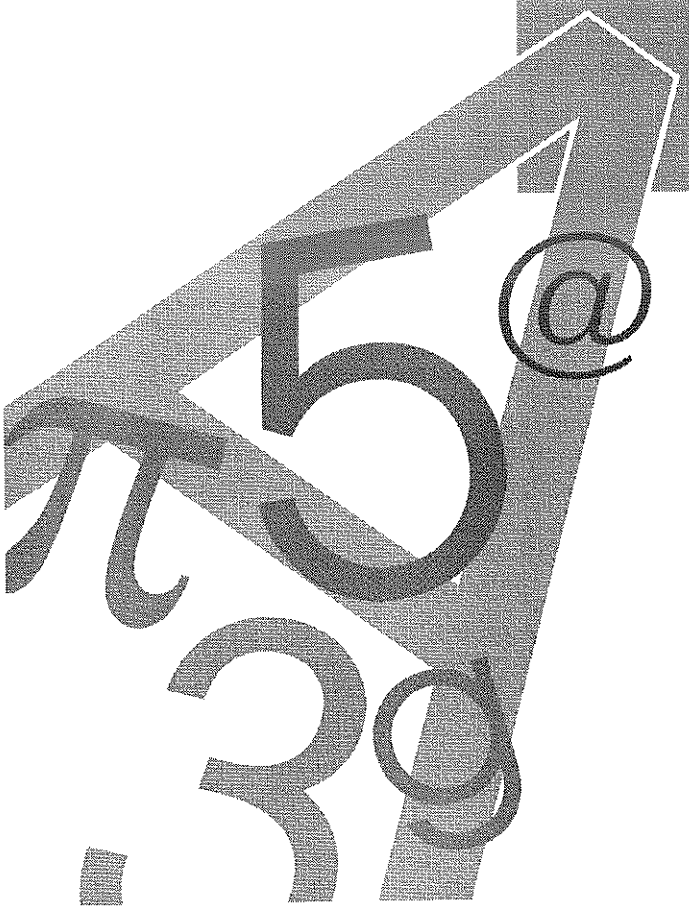
Pisa

Consiglio Nazionale delle Ricerche

Computing Proximity of Metric Regions

Giuseppe Amato, Fausto Rabitti, Pasquale Savino, Pavel Zezula

I.E.I. B4-16-09-00



Computing Proximity of Metric Regions

Giuseppe Amato[1], Fausto Rabitti[2], Pasquale Savino[1], and Pavel Zezula[3]

[1] IEI-CNR, Pisa, Italy,
{G.Amato,P.Savino}@iei.pi.cnr.it
WWW home page: http://www.iei.pi.cnr.it
[2] CNUCE-CNR, Pisa, Italy,
F.Rabitti@cnuce.cnr.it
WWW home page: http://www.cnuce.cnr.it
[3] Masaryk University, Brno, Czech Republic,
zezula@fi.muni.cz
WWW home page: http://www.fi.muni.cz

Abstract. The problem of defining and computing proximity of regions constraining objects from generic metric spaces is investigated. Besides other possibilities, the proximity measure can be applied to improve the performance of metric data indexes through optimized splitting and merging of regions, pruning regions during similarity retrieval, ranking regions for best case matching, and declustering regions to achieve parallelism. Approximate, computationally fast, approach is developed for pairs of metric ball regions, which covers the needs of current systems for processing of distance data. The validity and precision of proposed solution is verified by extensive simulation on three substantially different data files. The results of experiments are very positive.

Categories and Subject Descriptors: E.1 [Data Structures]: *Trees*; **E.5 [Files]:** *Searching*; **H.2.2. [Database Management]:** Physical Design - *Access methods*
Other Keywords: information retrieval, distance-only data, metric regions, algorithms, performance evaluation

1 Introduction

In order to speedup retrieval in large collections of data objects, storage (index) structures are developed and used. Contrary to traditional databases organizing simple sets of formatted items, current multimedia data occupy much more space and have very complex internal structures. The search is not performed at the level of actual (raw) multimedia objects, but on characteristic features that are extracted from these objects. Features are typically high-dimensional vectors or some other objects for which nothing more than pair-wise *distances* in specific *metric space* can be measured. The latter type of data is sometimes designated as the *metric data* or the *distance data*. In such environments, exact match has little meaning and concepts of *similarity*, (*dissimilarity*) are typically used for searching.

Though numerous designs of storage structures exist, the common underlying principle is the *partitioning* of object files into subgroups, called *partitions*, and bounding them in specific *regions*. The motivation is to achieve a structure where, once a query is issued, only some of its regions have to be examined in order to find qualifying objects. However, contrary to partitions, which contain disjoint sets of objects, regions can overlap.

In traditional databases of simple sortable domains, regions do not overlap. But with multi-dimensional keys, non-overlapping regions are difficult to maintain, and structures such as the R-tree [Gu84] or X-tree [BKK96] allow for overlapping regions. Consequently, the number of

searched regions for a query increases, and index search efficiency goes down. In order to devise strategies aiming at creation of non (or little) overlapping regions, the phenomenon of *region proximity* has been defined to provide an objective support for answering questions such as: how close two regions are, how much they overlap? In multi-dimensional vector spaces, the proximity can be expressed in terms of volumes and intersections of hyper-rectangles. As an example, [KF92] shows how proximity can be used to decluster nodes of R-trees to achieve parallelism.

In order to better capture objects' content, thus trying to enlarge the set of data types for which efficient search is possible, more recent approaches to index multimedia, genomic, and many non-traditional databases have considered the case where keys are not restricted to stay in a vector space, and only pair-wise object distances can be computed. This approach, which subsumes the case of multi-dimensional keys, has generalized the notion of *similarity queries* and resulted in the design of so-called *metric trees*. Although several metric storage structures have been proposed, see for example [Ch94,Br95,BO97,CPZ97,BO99], their algorithms for partitioning and organizing objects in regions are based on heuristics that have no clearly defined guiding principles to rely upon. Naturally, the performance of such structures is not optimum, and practical experience confirms that there is still a lot of space for improvement. We believe that the basic reason for this state of affairs is the absence of the notion of *region proximity* in generic metric spaces – due to the absence of coordinates, the techniques used in vector spaces cannot be applied here.

Since our rather theoretical problem is strictly motivated by pragmatic needs, we have sought for practical solutions. As a result, we propose techniques of computing proximity that satisfy the following criteria: (1) the measures are reliable indications of the reality and provide proximity with sufficient precision; (2) the cost of calculating the measures is low; (3) the measures are able to adapt to changed environments, i.e. different metrics and different data files; (4) the necessary storage overhead is moderate.

In the following, Section 2 introduces some application scenarios where region proximity can be useful, Section 3 formalizes the problem and proposes a solution, Section 5 presents experimental results that validate the proposed approach, and finally Section 6 concludes with suggestions for future research.

2 Application Considerations

In order to illustrate the dominant role of metric regions' proximity in the development of search mechanisms, consider the issues of *partitioning*, *allocation*, and *ranking*.

Partitioning Partitions constrained by regions are typically stored in storage buckets (tree nodes, pages, or blocks of data) that require some costs to access. Static regions are not very typical and the evolution process in storage structures is regulated by specific *splitting* and *merging* procedures. When a region $\mathcal{R}$ splits, two new regions, say $\mathcal{R}_1$ and $\mathcal{R}_2$, are created. By analogy, two regions, $\mathcal{R}_1$ and $\mathcal{R}_2$, can merge to form a single region $\mathcal{R}$. Notice that regions $\mathcal{R}_1$ and $\mathcal{R}_2$ can have arbitrary positions, so they are not necessarily disjoint. Considering a specific objective function, one way of splitting a region can be more advantageous than another split - content of a region can typically be split in several ways. In particular, when after split regions overlap a lot, the probability of accessing both of the regions for some queries is high. Similarly, when specific region is to be merged with other region from a candidate set, not all

these possibilities are of equal significance. That means, a quantitative measure of the *quality of partitioning* (splitting or merging) is important.

Allocation When a new region appears, it must be placed in storage system. In such situation, metric region measures can be useful for finding a suitable storage bucket in which the partition is to be allocated. Obviously, the strategy is different for single and multiple (independent) disks – multiple disks can support parallel processing. If parallel disks are available, regions with high proximity should not be put on the same disk, i.e. the regions should be *declustered*. On a single disk, regions with high probability to be accessed together should be placed as close as possible, i.e. *clustered*. Naturally, the problems of clustering and declustering of data on secondary storage are quite complex, but main problem again is to quantify the regions' proximity.

Ranking Queries in traditional databases divide objects into two parts. One part contains objects that do not satisfy the query while the other part contains objects satisfying the query and forms the query response set. Objects in both the groups are of equal relative importance. In metric databases, queries are based on similarity and a degree of membership is important. Response to a similarity query is a *ranked set*, ordered on relative distances, which is actually the only possible linear arrangement of objects in such case. However, there are also very good reasons for ranking regions. Examples of such situations include ranking of object clusters [GRG+99], organizing priority queues for searching [HS99], and approximate retrieval [ZSA+98]. In all these cases, the proximity of regions determines efficiency of proper algorithms.

3 The Problem of Metric Region Measures

Suppose a *metric space* $\mathcal{M} = (\mathcal{D}, d)$, defined by a domain of objects, $\mathcal{D}$, (i.e. the *keys* or indexed features) and by a total (distance) function, d, which satisfies for each triple of objects $O_x, O_y, O_z \in \mathcal{D}$ the following properties:

(i) $d(O_x, O_y) = d(O_y, O_x)$ (*symmetry*)
(ii) $0 < d(O_x, O_y) < \infty, O_x \neq O_y$ and $d(O_x, O_x) = 0$ (*non negativity*)
(iii) $d(O_x, O_y) \leq d(O_x, O_z) + d(O_z, O_y)$ (*triangle inequality*)

Provided the objects are vectors, the traditional way to measure distances in vector spaces is to use a Minkowski-form distance. This set of distance measures is often designated as the L_p distance and is defined for vectors v_x and v_y as $L_p(v_x, v_y) = (\sum_{j=1}^{n} \mid v_x[j] - v_y[j] \mid^p)^{1/p}, p \geq 1$, with L_1 known as the *city-block* or *Manhattan* distance and L_2 the *Euclidean* distance. Since all coordinates of the vectors are assumed independent, L_p distances are proportional to closeness of vectors in multi-dimensional space.

However, vector coordinates can be dependent or correlated. Good examples of such data are color histograms with each dimension representing a color. Obviously, orange and pink are certainly more similar than red and blue colors. In order to measure a distance between histograms, this natural (though also subjective) *cross-talk* of dimensions should properly be taken into account [Fa96]. A way to handle this problem is to use the *quadratic-form* distance.

$$d_{qf}^2(v_x, v_y) = (v_x - v_y)^T \mathbf{A} (v_x - v_y), \tag{1}$$

where $\mathbf{A} = [a_{i,j}]$ is a similarity matrix between dimensions of vectors v_x and v_y, and the superscript T idicates matrix transposition. Naturally, there is no linear correspondence between

distances and positions of vectors in the space, though the measure is still a distance metric provided the matrix is symmetric and $a_{i,i} = 1$.

Other example of a distance only measure is the *Levenstein* (also called the *edit*) distance to quantify similarity over strings. It is defined as the minimal number of string symbols that have to be inserted, deleted, or substituted to transform a string O_x into a string O_y, see e.g. [HD80].

Similarity of sets is another measure that is still a metric and applies for non-vector data. Given two sets A and B, the similarity is defined as the ratio of the number of their common elements to the number of all different elements.

$$S_T(A, B) = \frac{n(A \cap B)}{n(A \cup B)} \tag{2}$$

where $n(X)$ is the nuber of elements in set X. Notice that a generalization of this measure is the *Tanimoto* similarity measure [Ko84]

$$S_T(v_x, v_y) = \frac{(v_x, v_y)}{\|v_x\|^2 + \|v_y\|^2 - (v_x, v_y)} \tag{3}$$

which is defined for vectors with (v_x, v_y) being the scalar product of v_x and v_y, and $\|v_x\|$ the Euclidean norm of v_x. As a final example, consider the *Hausdorff distance*, which is used to compare shapes of images [HKR93]. Here the compared objects are sets of relevant, e.g. high curvature, points.

3.1 Partitions and regions

Given a file of *metric data* $\mathcal{F} \subseteq \mathcal{D}$, it is convenient to pre- process (or partition) $\mathcal{F}$ into smaller non-redundant units so that the retrieval process might perform in sub-linear time.

Definition 1. *Partitioning is a separation of a set into subsets such that every element belongs to one subset and no two subsets have an element in common.* □

According to [Uh91], there are two elementary strategies how to partition a set of metric data into two subsets:

ball partitioning choose an object from $O \in \mathcal{F}$ and compute the average distance with respect to O. Then, one partition contains object with distance smaller than the average and the other one contains the rest of the file.

generalized hyperplane choose two objects $O_1, O_2 \in \mathcal{F}$. Then for all objects, O_i, in the first partition $d(O_1, O_i) < d(O_2, O_i)$ is true while for objects in the second partition the predicate is false.

Naturally, the content of a partition is sufficiently defined by explicitly listing all its elements. Notice that the relationships between elements are implicitly given by the distance function. Naturally, such representation is not very practical. In order to characterize generic properties of partitions, space effective abstractions are used in practice. To this aim, partitions are constrained by *regions* satisfying specific properties.

Definition 2. *A region* $\mathcal{R} = \{O \in \mathcal{D} \mid \mathcal{C}_{\mathcal{R}}(O)\}$ *is the set of objects of* $\mathcal{D}$ *which satisfy the constraint* $\mathcal{C}_{\mathcal{R}}(\cdot)$. □

In order the constraints to be efficient (i.e. simple and small), they usually define regions which are bigger than necessary. Consequently, a region of partition contains all objects of this partition, but it might contain other objects that also satisfy the constraint. In any case, a region covers a certain amount of total object space, the fraction of which is designated as the *coverage*.

Contrary to disjoined partitions of data elements, regions of the same metric space can have significantly different relative positions. Regions can be quite far from each other, they can overlap, or one of the regions can even be included in other regions.

3.2 The approach taken

Though the volume of metric regions cannot be decided, it is obvious that regions *intersect* if an object belongs to more regions. It could be correctly argued that the proximity of regions should be proportional to the amount of space shared by the two regions, and the larger their intersection is the higher the proximity of these regions. However, in order to implement such idea, the following three arguments should carefully be considered:

- no space coordinate system for computing a region volume can be used, since only relative distances, constrained by the triangle inequality property, define the objects' geometry. In particular, there is no generic formula for computing the volume of a metric space and no volume of a region can be computed.
- zero proximity is typically assigned to disjoint regions, regardless of how "far" they actually are. Intuitively, this is only appropriate for *exact-match* (point) queries – non-intersecting data regions cannot share a point. However, a third region might contain points, which are also shared by the other two, though nonintersecting, regions. A correct proximity measure must be able to reflect such situation.
- depending on metric, some distances are far more frequent than the others. For example, in high-dimensional space, distances for sets of uniformly distributed elements are practically the same. In real files, data objects are not uniformly distributed, they typically occur in clusters. In any case, distance distributions are skewed and this fact must carefully be taken into account.

Inspired by [KF92], where a proximity measure for vector spaces was proposed, we define proximity of metric regions with respect to another subject. It is again a metric region, but this region is a random variable. For convenience, we call it the *query region*. Then, we define the proximity measure as the relative number of cases in which a query region intersects the compared regions to the total number of possible query regions. Notice that a query region has got the general metric region properties as given by Definition 2.

Definition 3. *The n-proximity* $X^n(\mathcal{R}_1, \mathcal{R}_2 \ldots \mathcal{R}_n)$ *of regions* $\mathcal{R}_1$ *to* $\mathcal{R}_n$ *is the probability that a randomly chosen query region* $\mathcal{Q}$ *over the same metric space* $\mathcal{M}$ *finds qualifying objects in all regions* $\mathcal{R}_1$ *to* $\mathcal{R}_n$*, i.e.* $\exists O_{i_1}, \ldots, O_{i_n} \mid O_{i_1} \in \mathcal{R}_1, \ldots, O_{i_n} \in \mathcal{R}_n$ *and* $O_{i_1}, \ldots, O_{i_n} \in \mathcal{Q}$. □

Since our rather theoretical problem is motivated by purely pragmatic needs, we search for practical solutions to be used in the field of storage structures for metric data. In particular, the required measures should satisfy the following criteria:

Accuracy In order to be useful, the measures must be accurate indications of the reality as formalized by Definition 3.

Fast computation The cost of calculating the measures should be "low". Such requirement is necessary so that the measures could also be used at run- time.

Flexibility Good measures should easily be able to adapt to changed environments. They should work equally well for different metrics. The measures should also be able to reflect peculiarities of specific files, such as distance distribution.

Low storage cost Though some use of pre-calculated (auxiliary) data is fully acceptable, a possible storage overhead, needed to support the computation, should certainly not be excessive.

3.3 Ball Regions

Up to now, we have not considered any specific type of regions. In order to come out with a solution, let us concentrate on the *ball* regions for measures with levels $n \leq 2$. To the best of our knowledge, ball regions are practically the only type of regions which are used in practice.

Definition 4. *A ball* $\mathcal{B}_x = \mathcal{B}_x(O_x, r_x) = \{O_i \in \mathcal{D} \mid \mathcal{C}_{\mathcal{B}_x}(O_i) = d(O_x, O_i) \leq r_x\}$, *is the region, determined by a center* $O_x \in \mathcal{D}$ *and a radius* $r_x \geq 0$, *defined as the set of objects in* $\mathcal{D}$ *for which the distance to* O_x *is less than or equal to* r_x. □

Ball regions are more amenable to effective analysis, because they are the simplest region types that can be defined in a metric space. Since 1-proximity is not only equal but also quite easy to solve, we mostly concentrate on case of $n = 2$. Before we proceed, let's consider some facts that appear on the qualitative level of analysis.

In order to see if two balls, $\mathcal{B}_x, \mathcal{B}_y \subseteq \mathcal{D}$, overlap, i.e. there exists $O_i \in \mathcal{D}$ which belongs to both $\mathcal{B}_x$ and $\mathcal{B}_y$, it is sufficient to check if the sum of their radii is greater than or equal to the distance between the balls' centers, specifically

$$\mathcal{B}_x \cap \mathcal{B}_y \neq \emptyset \quad \Longleftrightarrow \quad r_x + r_y \geq d(O_x, O_y)$$

It is quite intuitive that, for given radii values, the proximity of $\mathcal{B}_x$ and $\mathcal{B}_y$ should increase if $d(O_x, O_y)$ goes down (the two balls' centers get closer). Similarly, $X^2(\mathcal{B}_x, \mathcal{B}_y)$ will decrease if, for a given $d(O_x, O_y)$, the sum $r_x + r_y$ grows.

Query Regions In order to be consistent, we consider the relevant case where query regions are balls too. In this case, each query region $\mathcal{Q}$ is univocally identified by a *query key*, Q, and a query radius, r, thus $\mathcal{Q} = \mathcal{Q}(Q, r)$.

When queries with the radius r are considered, we can explicitly designate the fact that the measures depend on r by using the notation $X^n_r(\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_n)$ and referring to it as the *n-proximity for r-query regions*. Note that when $r = 0$, point queries are used. When the query radius is not constant, the expected value of proximity can still be determined in obvious way.

From the application point of view, point queries in files of objects from complex metric spaces are not very typical – exact match rarely exists thus *similarity range* (or *nearest neighbor*) *queries* are prevalent.

Range queries are characterized by an object and radius, which defines the surroundings of the object, in which, everything found forms the response set. The choice of a proper radius is left on the user. Since the response set should not typically be large, small radii values are more likely than the large ones.

Nearest neighbor queries are again specified by a query object. Instead of limiting the result by a radius, the response set is constrained by the number of best cases, i.e. the most similar objects with respect to the query. However, also in this case, the query object radius plays an important role while evaluating the query. The radius for a given query object is changing dynamically, starting with a very large (usually the maximum) radius and narrowing down its value until the minimum region (containing the required number of neighbors) is reached.

In order to see the effects of query radii on our metric space measures consider the following observation.

Observation 31 *Suppose a query object Q with distances to two specific ball centers O_x and O_y as $d(O_x, Q) = r_x + r$ and $d(O_y, Q) = r_y + r$. Provided $r > 0$, it is obvious that Q is not included in $\mathcal{B}_x$ and $\mathcal{B}_y$. However, in both the balls, there is at least one object, say $O_i \in \mathcal{B}_x$ and $O_j \in \mathcal{B}_y$, which is sure to belong to the query ball, i.e. $O_i, O_j \in \mathcal{Q}(Q, r)$. It follows that such $\mathcal{Q}$ should be considered as a region able to intersect both $\mathcal{B}_x$ and $\mathcal{B}_y$. More precisely, a query ball $\mathcal{Q}(Q, r)$ intersects both $\mathcal{B}_x$ and $\mathcal{B}_y$ if $d(O_x, Q) \leq r_x + r$ and $d(O_y, Q) \leq r_y + r$.*

Accordingly, the following lemma specifies the effects of range queries on the metric ball measures.

Lemma 1. *Proximity for query balls with $r > 0$ can be transformed to point queries by using the following substitution.*

$$X_r^n(\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_n) = X_0^n(\overline{\mathcal{B}_1}, \overline{\mathcal{B}_2}, \ldots, \overline{\mathcal{B}_n})$$

where $\overline{\mathcal{B}_i}$ is an enlarged region $\mathcal{B}_i$, such that $\overline{O_i} = O_i$ and $\overline{r_i} = r_i + r$. □

According to Lemma 1, positive query radii can be transformed to point queries, so it is correct to consider only point queries in the following. For such situation, we simplify notation and use X^n instead of X_0^n.

To conclude this section, we can say, with a slight abuse of terminology, that the proximity strongly depends on the "size" of regions' intersection. However, the problem is to determine which are the actual arguments governing the region proximity.

4 Computing Ball Region Proximity

In this section, we concentrate on developing computational procedures that are able to determine ball region measures for $n = 1$ and 2. From now on, we assume that the maximum distance is $d_m < \infty$, thus consider a *bounded* metric space.

4.1 A note on distance distributions

Before we proceed, we first define necessary terms for precise discussion. Let $f_O(x)$ represent the *distance density* function that x determines the probability of distances from object O. The corresponding *distance distribution*, that is the probability $F_O(x)$ of a distance to O to be at most x, can be determined as

$$F_O(x) = \int_0^x f_O(t)\, dt \tag{4}$$

Remember that x is assumed to be bounded by d_m, i.e. $x \leq d_m$. Notice also that we consider the distribution and density relative to an object, because in general, distributions with respect to different objects can vary.

Given two different objects $O_x, O_y \in \mathcal{D}$, the corresponding distrifutions F_{O_x} and F_{O_y} are generally different functions. We can also say that F_{O_i} represents the O_i's point of view of the domain $\mathcal{D}$. However, it is not possible to know distance distributions with respect to all objects.

An alternative solution suggests to consider the overall distribution of distances over $\mathcal{D}$. This can be defined as

$$F(x) = \mathtt{Pr}\{d(\mathbf{O_1}, \mathbf{O_2}) \leq x\}, \tag{5}$$

where $\mathbf{O_1}$ and $\mathbf{O_2}$ are two independent random objects of $\mathcal{D}$. However, even if we neglect the computational complexity of a procedure that would determine $F(x)$, all objects from $\mathcal{D}$ are simply not known. What only seems feasible to compute is an approximation of $F(x)$, or alternatively $f(x)$, by considering pair-wise distances between a sample of objects of size n.

The problem of distance distribution for metric data has been studied in [CPZ98] to derive a cost model for similarity queries. In particular, to measure the compatibility of two viewpoints in $\mathcal{M}$, the concept of *discrepancy* is defined. Then, in order to quantify the possible variation of different viewpoints, another measure, called the *homogeneity of viewpoints* is also established.

The problem of distance distribution for metric data has been studied in [CPZ98] to derive a cost model for similarity queries. In particular, in order to quantify the homogeneity of behaviour of viewpoints, a measure, called the *homogeneity of viewpoints*, was defined. In order to justify the possibility of using the approximated overall distribution of distances, instead of distribution with respect to specific objects, numerous synthetic and real-life files were tested. For all these data sets, the homogeneity of viewpoints was very high.

Accordingly, we use in our experiments the approximated overall distance distribution instead of F_{O_x} and F_{O_y} for all x and y.

4.2 Definition of the Proximity Measure

Given a ball region $\mathcal{B}_1 = (O_1, r_1)$ and the distance distribution with respect to its center F_{O_1}, the probability that a randomly chosen query region belongs to this region is $F_{O_1}(r_1)$. The proof comes from the definition of distance distribution. Naturally, the probability of any query point in metric space bounded by maximum distance d_m is 1, because $F_{O_1}(d_m) = 1$. Respecting our definition of proximity, the 1- proximity is given by the following equation.

$$X^1(\mathcal{B}_1) = F_{O_1}(r_1)$$

As a consequence of the discussion in Section 4.1, it is approximated by

$$X^1(\mathcal{B}_1) \approx F(r_1) \tag{6}$$

Proximity of a pair of regions is defined as follows:

Definition 5. *The proximity $X(\mathcal{B}_x, \mathcal{B}_y)$ of ball regions $\mathcal{B}_x, \mathcal{B}_y$ is the probability that a randomly chosen object $\mathbf{O}$ over the same metric space $\mathcal{M}$ appears in both regions:*

$$X^2(\mathcal{B}_x, \mathcal{B}_y) = \mathtt{Pr}\{d(\mathbf{O}, O_x) \leq r_x \wedge d(\mathbf{O}, O_y) \leq r_y\}$$

□

4.3 Computational Difficulties

The computation of proximity according to Definition 5 requires the knowledge of distance distributions with respect to regions' centers. Since any object from $\mathcal{M}$ can become a ball center, such knowledge is not realistic to obtain. However, as discussed in [CPZ98], we can assume that the distributions depend on the distance between the centers (d_{xy}), while they are (practically) independent from the centers themselves. Such assumption is realistic when distance distributions with respect to different objects have small *discrepancies*, which was found true in [CPZ98] for many data files. Thus, we can modify our definition as

$$X^2(\mathcal{B}_x, \mathcal{B}_y) \approx X_{d_{xy}}(r_x, r_y) = \Pr\{d(\mathbf{O}, \mathbf{O_x}) \leq r_x \wedge d(\mathbf{O}, \mathbf{O_y}) \leq r_y\}, \tag{7}$$

where $\mathbf{O_x}$, $\mathbf{O_y}$, and $\mathbf{O}$ are random objects such that $d(\mathbf{O_x}, \mathbf{O_y}) = d_{xy}$.

Now, consider the way how $X_{d_{xy}}(r_x, r_y)$ can be computed. Let X, Y and D_{XY} be continuous random variables corresponding, respectively, to the distances $d(\mathbf{O}, \mathbf{O_x})$, $d(\mathbf{O}, \mathbf{O_y})$, and $d(\mathbf{O_x}, \mathbf{O_y})$. The *joint conditional density* $f_{X,Y|D_{XY}}(x, y|d_{xy})$ is the probability[1] that distances $d(\mathbf{O}, \mathbf{O_x})$ and $d(\mathbf{O}, \mathbf{O_y})$ are, respectively, x and y, given that $d(\mathbf{O_x}, \mathbf{O_y}) = d_{xy}$. Then, $X_{d_{xy}}(r_x, r_y)$ can be computed as

$$X_{d_{xy}}(r_x, r_y) = \int_0^{r_x} \int_0^{r_y} f_{X,Y|D_{XY}}(x, y|d_{xy}) dy dx \tag{8}$$

In general, $f_{X,Y|D_{XY}}(x, y|d_{xy}) \neq f_{XY}(x, y)$, because the *joint density* $f_{XY}(x, y)$ gives the probability that the distances $d(\mathbf{O}, \mathbf{O_x})$ and $d(\mathbf{O}, \mathbf{O_y})$ are x and y, no matter what is the distance between $\mathbf{O_x}$ and $\mathbf{O_y}$. The difference between the two densities is immediately obvious when we consider the metric space postulates. Accordingly, $f_{X,Y|D_{XY}}(x, y|d_{xy})$ is 0 if x, y, and d_{xy} do not satisfy the triangular inequality, because such distances cannot simply exist. However, $f_{XY}(x, y)$ is not restricted by such constraint, and any pair of distances $\leq d_m$ is possible. For illustration, Figure 1 shows the joint conditional density $f_{X,Y|D_{XY}}(x, y|d_{xy})$ for a fixed d_{xy} and the joint density $f_{XY}(x, y)$. They are both obtained by sampling from the same data set, but their characteristics are significantly different.

Unfortunately, an analytic form of $f_{X,Y|D_{XY}}(x, y|d_{xy})$ is unknown. In addition, computing and maintaining it as a discrete function would result in very high number of values. Indeed, the function depends on three arguments so that the storage space required is $O(n^3)$, where n is the number of samples for each argument. This makes such approach of obtaining and maintaining the probabilities totally unacceptable.

On the other hand, the joint density is simpler to obtain. Since X and Y are independent random variables, $f_{XY}(x, y) = f_X(x) \cdot f_Y(y)$. Given the definition of the random variables X and Y, it is easy to show that $f_X(d) = f_Y(d)$, so we can omit the name of the random variable and designate the joint density as $f(d)$. Notice that $f(d)$ can be easily obtained by sampling from the data set.

In this article, we develop an approach able to compute the proximity measure by expressing $f_{X,Y|D_{XY}}(x, y|d_{xy})$ in terms of $f_{XY}(x, y)$, which is available by means of $f(d)$. From the storage point of view, such approach is feasible, but the problem is to find the necessary transform. In the Appendix A we show how the joint conditional density can be obtained for a two dimensional

[1] We are using continuous random variables so, to be rigorous, their probability is by definition always 0. However, in order to simplify the explanation, we slightly abuse the terminology and use the term probability to give an intuitive idea of the behavior of the density function being defined.

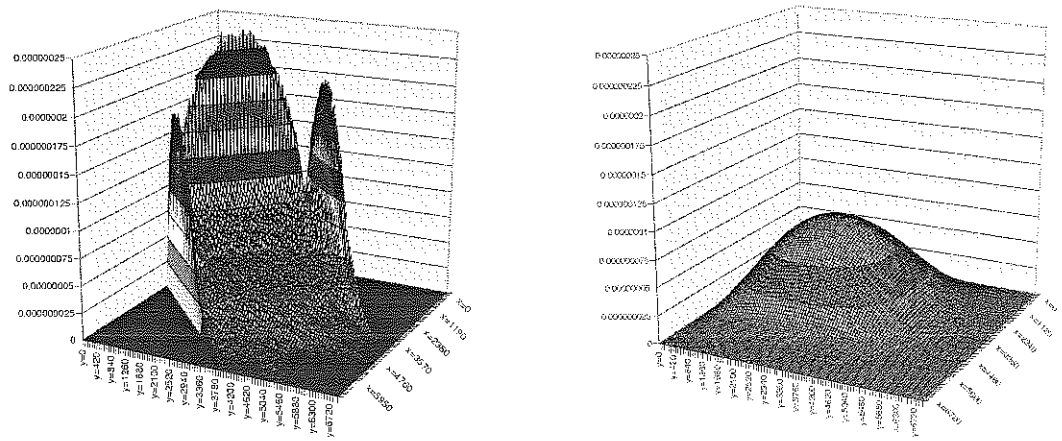


Fig. 1. Comparation of $f_{X,Y|D_{XY}}(x, y|d_{xy})$ and $f_{XY}(x, y)$

Euclidean space. However, even in this special case, a correct evaluation of $f_{X,Y|D_{XY}}(x, y|d_{xy})$ is computationally untractable and, as a consequence, not suitable for a correct evaluation of proximity. That is why we have decided to investigate some approximations that would satisfy efficiency requirements, and at the same time guarantee good quality of results.

Before we proceed, we define as reference an approximation of the proximity that is generally used in current applications.

$$X_{d_{xy}}^{trivial}(r_x, r_y) = \begin{cases} 0 & \text{if } r_x + r_y < d_{xy} \\ \frac{2 \cdot \min\{r_x, r_y\}}{2 \cdot d_m - d_{xy}} & \text{if } \max(r_x, r_y) > \min(r_x, r_y) + d_{xy} \\ \frac{r_x + r_y - d_{xy}}{2 \cdot d_m - d_{xy}} & \text{otherwise} \end{cases} \tag{9}$$

For convenience, we call this approximation *trivial*, because it completely ignores distributions of distances though, as Figure 1 confirms, distances in specific files can have very peculiar distributions, so their omission in proximity measures must result in high errors.

4.4 Approximate Proximity

Given two objects O_x and O_y such that $d(O_x, O_y) = d_{xy}$, the space of possible distances $x = d(O, O_x)$ and $y = d(O, O_y)$, measured from the object O, is constrained by the triangular inequality, i.e. $x + y \geq d_{xy}$, $x + d_{xy} \geq y$, and $y + d_{xy} \geq x$. Figure 2 helps to visually identify these constraints: in the gray area, called the *bounded area*, the triangular inequality is satisfied; in the white area, called the *external area*, the triangular inequality is not satisfied. Notice that the graph of the joint conditional density in Figure 1 has values greater than zero only in the bounded area, and that quite high values are located near the edges.

Such observations form the basis for our heuristics to approximate the joint conditional density by means of the joint density. The intuitive idea can be outlined as follows:

> *Collect values of $f_{XY}(x, y)$ for x, y, and d_{xy} from the external area and add them inside the bounded area.*

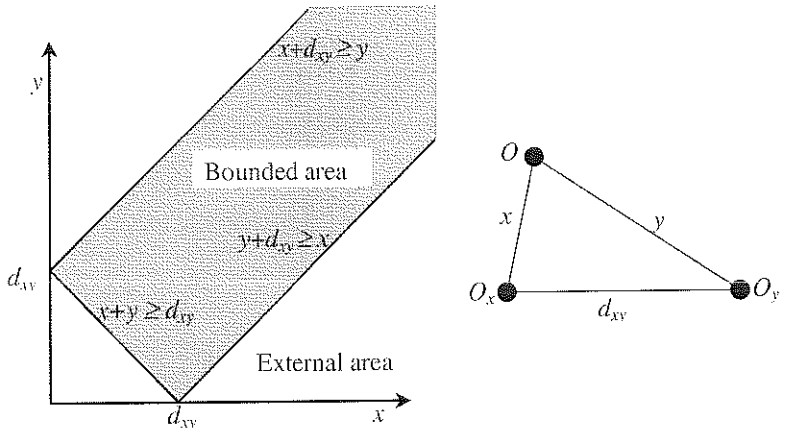


Fig. 2. Area bounded by the triangular inequality for a given d_{xy}

When x, y, and d_{xy} satisfy the triangular inequality, the value of $f^{appr}_{XY|D_{XY}}(x, y|d_{xy})$ depends on specific strategy used to implement the previous idea, but $f^{appr}_{XY|D_{XY}}(x, y|d_{xy}) = 0$ when x, y, and d_{xy} do not satisfy the triangular inequality. Notice that our approximations preserve the properties of density functions, and the integral over the bounded area is 1. This is the basic assumption of any probabilistic model that would be violated provided the joint densities were simply trimmed out by the triangle inequality constraints.

In order to come out with specific solutions, we have tried four different implementations of this heuristic, varying the strategy applied to move density values. Figure 3 provides a visual representation of the methods. The circles represent the joint density function, while the arrows show how points are moved from the external area to the bounded area.

Orthogonal approximation collects points outside the bounded area and moves them on top of the corresponding constraint following a direction that is orthogonal to the constraint.

Parallel approximation collects points outside the bounded area and moves them on top of the corresponding constraint following a direction that is parallel to the axis.

Diagonal approximation collects points outside the bounded area and moves them on top of the corresponding constraint following a direction that always passes through d_m.

Normalized approximation eliminates densities outside the constrained space and normalizes the ones found inside so that the integral over the whole constrained space is equal to one.

In this way, an approximation of the proximity can be computed according to Equation 8, but using $f^{appr}_{X,Y|D_{XY}}(x, y|d_{xy})$ instead of $f_{X,Y|D_{XY}}(x, y|d_{xy})$. Moreover, the orthogonal, parallel and diagonal approximations can be computed directly through the joint density $f_{X,Y}(x, y)$, provided the integration limits are modified as follows:

$$X^{appr}_{d_{xy}}(r_x, r_y) = \int_0^{b_x(d_{xy}, r_x, r_y)} \int_{b^1_y(x, d_{xy}, r_x, r_y)}^{b^2_y(x, d_{xy}, r_x, r_y)} f_{X,Y}(x, y) dy dx \tag{10}$$

In the following, we simplify the terminology by omitting the d_{xy}, r_x, r_y parameters and use for the integration bounds only the symbols b_x, $b^1_y(x)$ and $b^2_y(x)$.

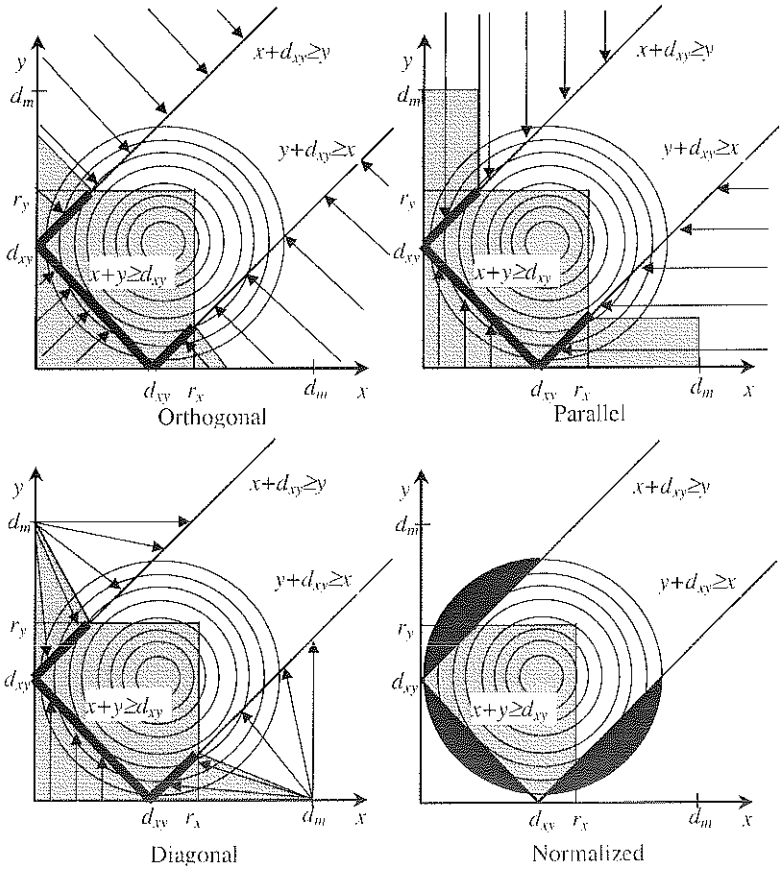


Fig. 3. moving from joint density to joint conditional density

Equation 10, integrates the function $f_{XY}(x, y)$ in an area that is larger than the original one ($[0..r_x]$ and $[0..r_y]$), so that all points that should be moved to produce $f^{appr}_{X,Y|D_{XY}}(x, y|d_{xy})$ are in fact correctly considered, obtaining the same result as of integrating $f^{appr}_{X,Y|D_{XY}}(x, y|d_{xy})$ itself. In fact, b_x, $b^1_y(x)$, and $b^2_y(x)$ are functions, specific for each approximation method, that bound the integral as illustrated by the gray marked areas highlighted in Figure 3. In general, b_x gives the integration range along the x axis, $b^1_y(x)$ is the lower bound of the gray area, and $b^2_y(x)$ is the upper bound of the gray area.

In fact, a similar technique can also be adopted for the normalized method, if we consider the approximation as the ratio between the integral on the gray area and the integral on the whole bounded area. Let's define $I_{d_{xy}}(r_x, r_y)$, using Equation 10, as the integral on the gray area highlighted for the normalized method, and $T_{d_{xy}} = I_{d_{xy}}(d_m, d_m)$, as the integral on the whole bounded area. Then the normalized approximate proximity is $X^{appr}_{d_{xy}}(r_x, r_y) = \frac{I_{d_{xy}}(r_x, r_y)}{T_{d_{xy}}}$. In this way, all our techniques are based on the joint density $f_{XY}(x, y)$.

The definition of the bounding functions b_x, $b_y^1(x)$ and $b_y^2(x)$, which depend on the approximation method used, is discussed in Section 4.5.

The proposed way of approximation can also significantly reduce the computational complexity by using the fact that $f_{X,Y}(x,y) = f(x) \cdot f(y)$ as follows

$$\int_0^{b_x} \int_{b_y^1(x)}^{b_y^2(x)} f_{XY}(x,y)dydx = \int_0^{b_x} f(x) \cdot (F(b_y^2(x)) - F(b_y^1(x)))dx \tag{11}$$

The computational complexity of Equation 11 is $O(n)$, where n is the number of samples (granularity) used to compute the integral as a discrete function. Naturally, the distance density function $f(d)$ and the distance distribution function $F(d)$ of high granularity can easily be kept in the main memory. Concerning the Normalized method, we can see that $T_{d_{xy}}$ only depends on d_{xy} thus can also be maintained in main memory. Consequently, it can also be computed in $O(n)$.

4.5 Bounding functions

In this section we will formally define the bounding functions b_x, $b_y^1(x)$ and $b_y^2(x)$ of the four approximation methods described above. Even though the graphical representation of the integration areas seems to be easy and clear, its formalization is not straightforward, because several special cases should be taken into account to obtain the correct behaviour. Notice that in our simplified formalisation of the problem, the function $f_{XY}(x,y)$ can assume arguments outside the range $[0, d_m]$. In those cases we suppose that the returned value is 0.

We decompose the problem in subcases that can be considered separately – see Figure 4 as a convenient graphical reference. First, we identify two different situations: (i) $r_y < d_{xy}$ and (ii) $r_y \geq d_{xy}$. Distinguishing these two situations, we identify some intervals along the x axis:

1. $I'_1 = [0, d_{xy} - r_y)$,
2. $I'_2 = [d_{xy} - r_y, min(d_{xy} + r_y, r_x))$, and
3. $I'_3 = [min(d_{xy} + r_y, r_x), d_m]$.

In case (ii), we identify another three intervals:

1. $I''_1 = [0, min(r_y - d_{xy}, r_x))$,
2. $I''_2 = [min(r_y - d_{xy}, r_x), min(d_{xy} + r_y, r_x))$, and
3. $I''_3 = [min(d_{xy} + r_y, r_x), d_m]$.

Using the intervals defined above, we define the upper bound $b_y^2(x)$. First suppose that $r_y \leq d_{xy}$. When $x \in I'_1$, and the regions do not intersect (that is $d_{xy} - r_x > r_y$), then proximity is 0 (see Figure 4-a_3) so the upper bound is 0 too. Otherwise, the upper bound is the stright line, which is specific for a method used, and passing by point A shown in Figure 4-(a_1 and a_2). We call $a(x)$ that stright line. When $x \in I'_2$, the upper bound is always equal to r_y. When $x \in I'_3$, the upper bound is the minimum between the two, method specific, stright lines passing respectively, by point B_1 and B_2. We call $b(x)$ the stright line passing by B_1 and $c(x)$ the one passing by B_2. Figures 4-(a_1 and a_2) show two different situations. In the first case, the minimum is $b(x)$, in the other case, the minimum is $c(x)$.

Now suppose that $r_y > d_{xy}$, that is there is always an intersection between the two regions. When $x \in I''_1$, the upper bound is the minimum between the two, method dependent, stright

lines passing, respectively, by point A_1 and A_2. We call $d(x)$ the line passing by A_1 and $e(x)$ the one passing by A_2. Figures 4-(b_2 and b_3) show two different situations. In the first case the minimum is $e(x)$, while in the other case the minimum is $d(x)$. When $x \in I''_2$, the upper bound is always equal to r_y. Since I''_3 is defined exactly as I'_3, the same arguments apply as can be seen in Figures 4-(b_1 and b_2).

More formally, $b^2_y(x)$ can be defined as follows:

$$b^2_y(x) = \begin{cases} \begin{cases} \begin{cases} a(x) \text{ if } d_{xy} - r_x \leq r_y & \text{if } x \in I'_1 \\ 0 \quad \text{elsewhere} \end{cases} & \\ r_y & \text{if } x \in I'_2 \\ min(b(x), c(x)) & \text{if } x \in I'_3 \end{cases} & \text{if } r_y < d_{x,y} \\ \begin{cases} min(d(x), e(x)) & \text{if } x \in I''_1 \\ r_y & \text{if } x \in I''_2 \\ min(b(x), c(x)) & \text{if } x \in I''_3 \end{cases} & \text{elsewhere} \end{cases} \tag{12}$$

where $a(x)$, $b(x)$, $c(x)$, $d(x)$, and $e(x)$ are defined for the individual approximation methods as follows.

First, we start with the Orthogonal method. To define $b^2_y(x)$, that is the upper bound of the integration area, the required stright lines are the following:

$$\begin{aligned} a(x) &= 2 \cdot r_y - d_{xy} + x \\ b(x) &= 2 \cdot r_y + d_{xy} - x \\ c(x) &= 2 \cdot r_x - d_{xy} - x \\ d(x) &= 2 \cdot r_x + d_{xy} - x \\ e(x) &= 2 \cdot r_y - d_{xy} - x \end{aligned}$$

On the other hand, $b^1_y(x)$, that is the lower bound of the integration area, can be defined as

$$b^1_y(x) = \begin{cases} \begin{cases} d_{xy} - 2 \cdot r_x + x \text{ if } d_{x,y} - r_x \leq r_y & \\ 0 \qquad\qquad \text{elsewhere} & \end{cases} & \text{if } r_x < d_{x,y} \\ 0 & \text{elsewhere} \end{cases}$$

That is, when r_x is smaller than d_{xy}, and r_x is such that $d_{xy} - r_x \leq r_y$, the lower bound is the stright line $y = x + d_{xy} - 2 \cdot r_x$ passing by point B see Figure 4-b_3. In all other cases the lower bound is 0.

Last, we need to define b_x, that is the range of the integration. If r_x is smaller than d_{xy} the integration is made in the interval $[0, r_x]$, otherwise in the interval $[0, min(r_1, r_2)]$, where r_1 and r_2 are such that, respectively, $b(r_1) = 0$ and $c(r_2) = 0$. We can make it explicit as follows:

$$b_x = \begin{cases} r_x & \text{if } r_x < d_{x,y} \\ min(2 \cdot r_y + d_{xy}, 2 \cdot r_x - d_{xy}) & \text{if } r_x \geq d_{x,y} \end{cases}$$

To define $b^2_y(x)$ for the Parallel method, we use again the outline of equation 12 and define $a(x)$, $b(x)$, $c(x)$, $d(x)$ and $e(x)$ as follows:

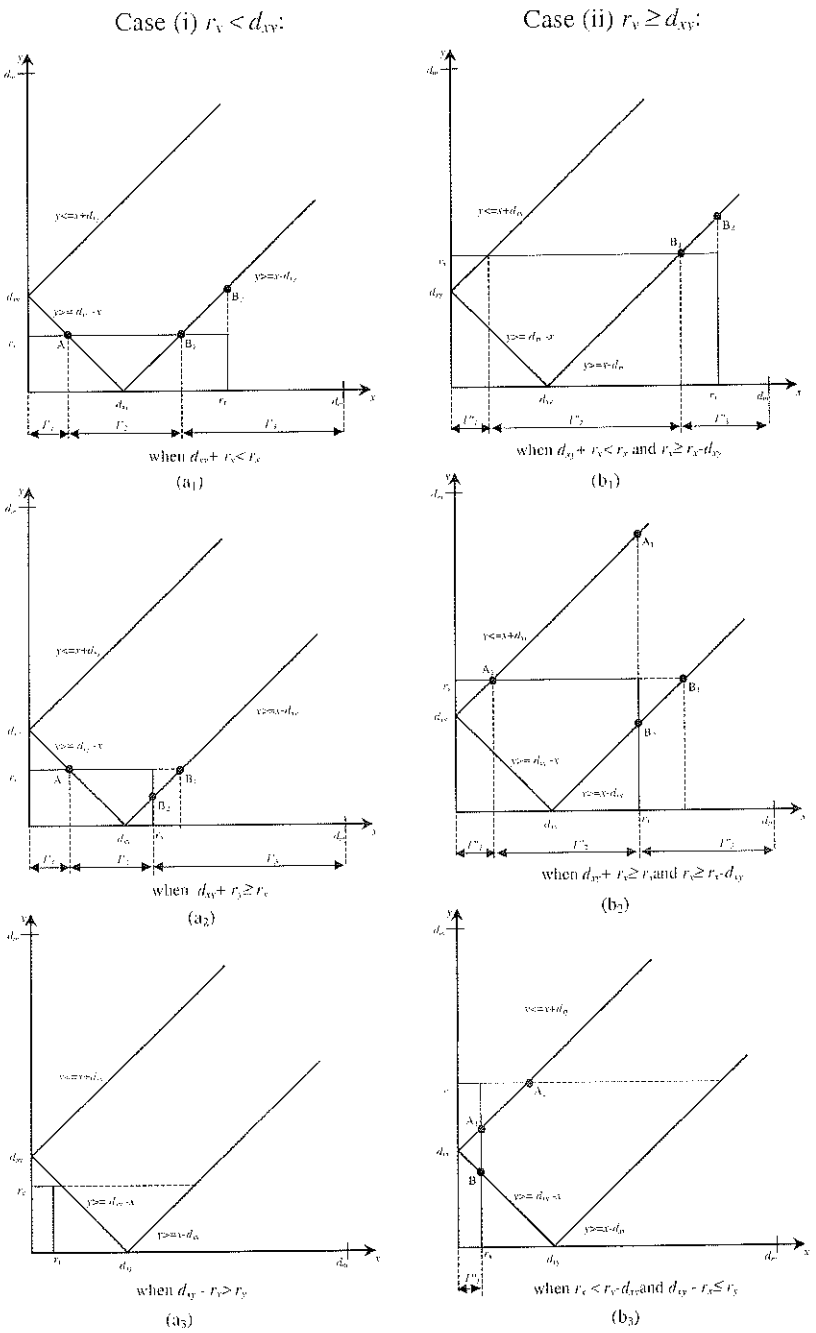


Fig. 4. Situations to be taken into account when defining bounding functions

$$\begin{aligned} a(x) = b(x) &= r_y \\ c(x) &= r_x - d_{xy} \\ d(x) = e(x) &= d_m \end{aligned}$$

$b_y^1(x)$ is always 0 so:

$$b_y^1(x) = 0$$

Last, b_x is defined as follows:

$$b_x = \begin{cases} r_x & \text{if } r_x < d_{x,y} \\ d_m & \text{if } r_x \geq d_{x,y} \end{cases}$$

That is, if r_x is smaller than d_{xy}, then the integral is made in the interval $[0, r_x]$, otherwise in the interval $[0, d_m]$.

To define $b_y^2(x)$ for the Diagonal method, we use again the layout of equation 12 and define $a(x)$, $b(x)$, $c(x)$, $d(x)$ and $e(x)$ as follows:

$$\begin{aligned} a(x) &= r_y \\ b(x) &= -\frac{r_y}{d_m - r_y - d_{xy}} \cdot x + \frac{r_y}{d_m - r_y - d_{xy}} \cdot d_m \\ c(x) &= -\frac{r_x - d_{xy}}{d_m - r_x} \cdot x + \frac{r_x - d_{xy}}{d_m - r_x} \cdot d_m \\ d(x) &= -\frac{d_m - d_{xy} - r_x}{r_x} \cdot x + d_m \\ e(x) &= -\frac{d_m - r_y}{r_y - d_{xy}} \cdot x + d_m \end{aligned}$$

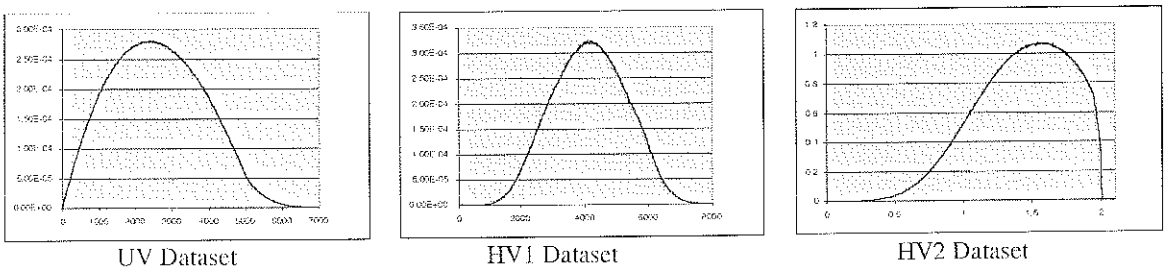
Bounding functions $b_y^1(x)$ and b_x for the Diagonal method are defined exactly the same as for the Parallel method so we will make no further discussion on them.

Last, we have to consider the Normalized method. Because of the different nature of this method, also the definitions of the bounding functions have a different layout. In particular, there is not need of extending the integration area, so just the intersection between the original constraints ($x \leq r_x$ and $y \leq r_y$) of the integration area and the triangular inequality constraint are needed. The result is the following:

$$b_x = r_x$$

$$b_y^1(x) = |x - d_{xy}|$$

$$b_y^2(x) = x + d_{xy}$$



UV Dataset (a) HV1 Dataset (b) HV2 Dataset (c)

Fig. 5. Overall distance density functions of the used data sets

5 Verification

In this section, we investigate the effectiveness of the proposed approaches. Before presenting our simulation results, we first characterize the data sets, describe the evaluation process, and define comparison metrics.

5.1 Data sets

To be more confident on obtained results, we have used three data sets: one synthetic and two real-life data sets representing color features of images. Each of the data sets contained 10.000 objects.

The synthetic data set, called UV, is a set of vectors uniformly distributed in 2-dimensional space where vectors are compared through the Euclidean (L_2) distance. The second data set, designated as HV1, contains color features of images. Color features are represented as 9-dimensional vectors containing the *average*, *standard deviation*, and *skewness of pixel values* for each of the red, green, and blue channels, see [SO95]. An image is divided into five overlapping regions, each one represented by a 9-dimensional color feature vector. That results in a 45-dimensional vector as a descriptor of one image. The distance function used to compare two feature vectors is again the Euclidean (L_2) distance. The third data set, called HV2, contains color histograms represented in 32-dimensions. This data set was obtained from the UCI Knowledge Discovery in Databases Archive ([Bay99]). The color histograms were extracted from the Corel image collection as follows: the HSV space is divided into 32 subspaces (32 colors: 8 ranges of hue and 4 ranges of saturation). The value in each dimension of the vector is the density of each color in the entire image. The distance function used to compare two feature vectors is the histogram intersection implemented as L_1.

The range of distances and corresponding distance density functions can be seen in Figure 5. Notice the differences in densities for individual files: the UV data set presents the most frequent distances on the left of the distances range, the HV1 on the center, and the HV2 on the right. In this way we have tried to cover a large spectrum of possible data.

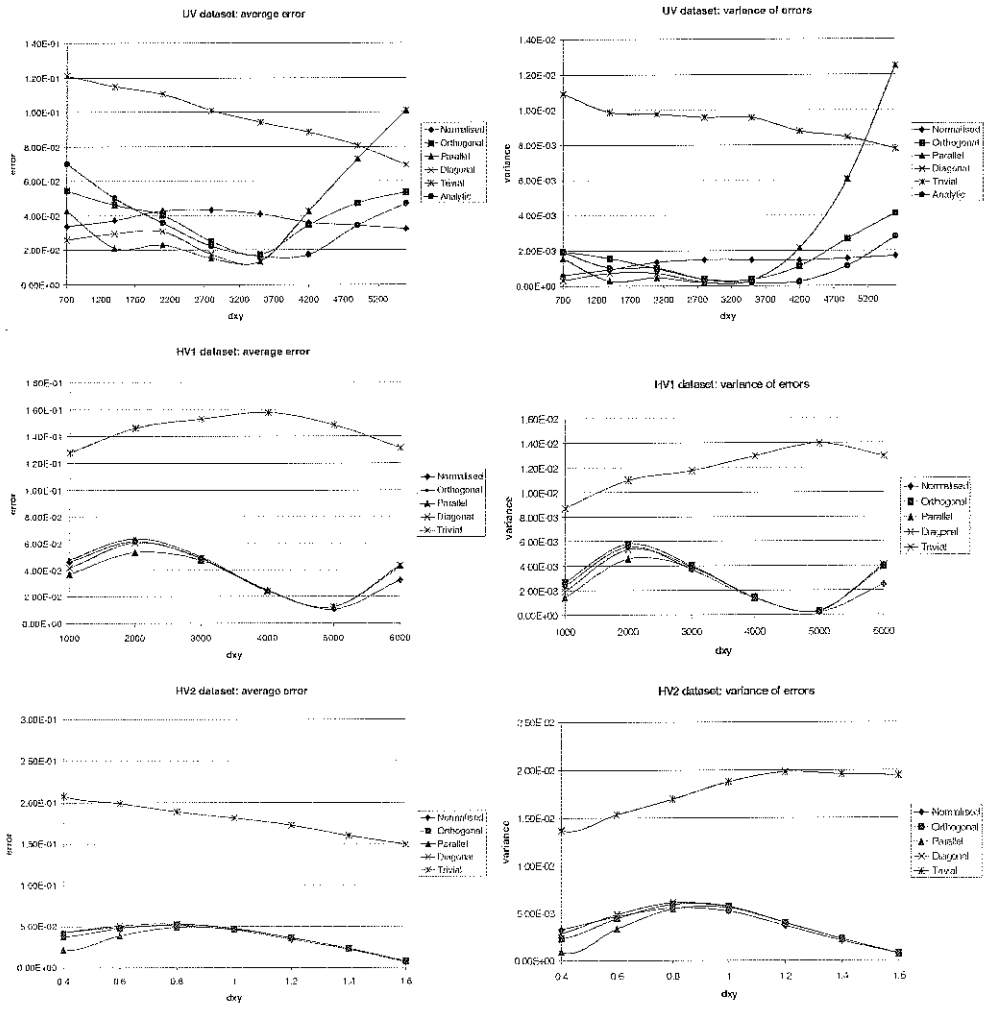


Fig. 6. Average and variance of errors

5.2 Experiments and comparison metrics

In order to form the basis for comparison, we have experimentally computed the actual proximity $X_{d_{xy}}^{actual}(r_x, r_y)$ for all data sets. We have chosen several values of d_{xy} in the range of possible distances and 100×100 values of (r_x, r_y). The value of $X_{d_{xy}}^{actual}(r_x, r_y)$ was computed for all their possible combinations. To accomplish this task, we have found, for each d_{xy}, 400 pairs of objects (O_x, O_y), i.e. the balls' centers, such that $|d(O_x, O_y) - d_{xy}| \leq \rho$, where ρ was the smallest real number that allowed to obtain at least 400 pairs. For each pair of objects we have generated 100×100 balls by varying correspondingly r_x and r_y in the range of possible radii. For each pair of balls we have counted the number of objects in their intersection. $X_{d_{xy}}^{actual}(r_x, r_y)$ was finally obtained by computing the average number of objects in the intersection for each generated configuration of d_{xy}, r_x, and r_y and normalizing such values to obtain the probability.

Notice that we did not consider distances d_{xy} of very low densities. In such cases, 400 pairs were only possible to obtain for large values of ρ, thus the actual proximity was not possible

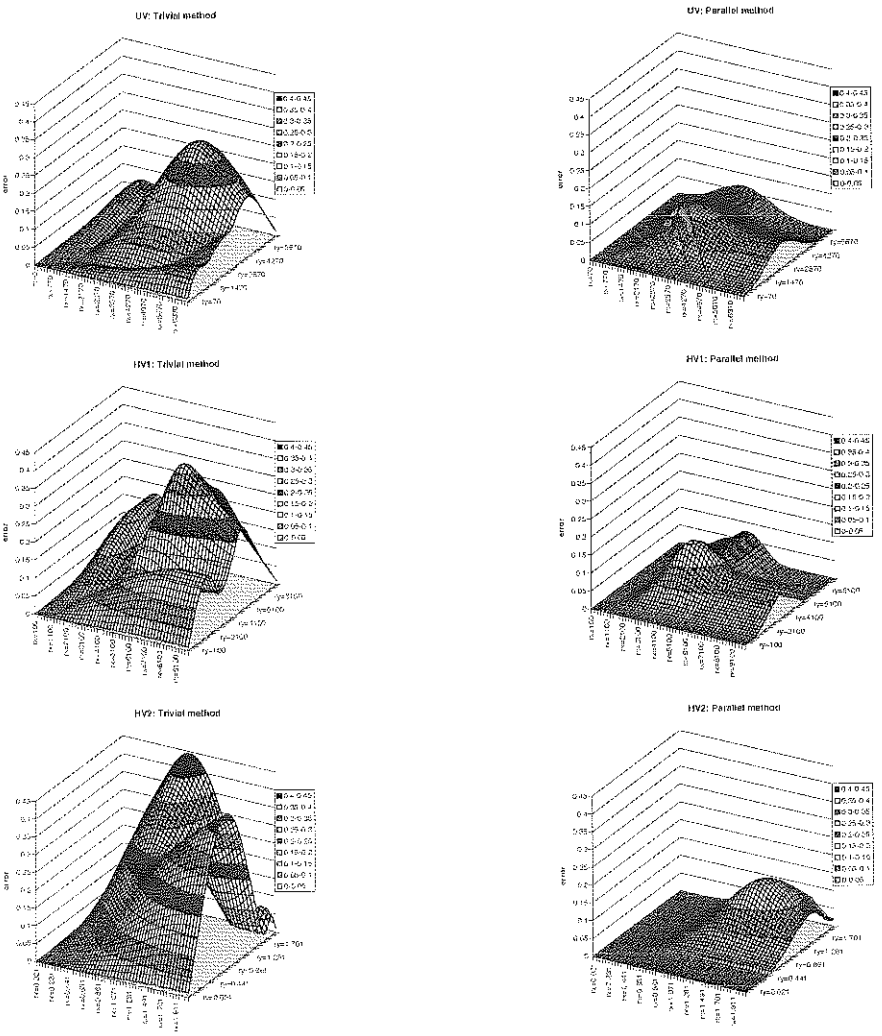


Fig. 7. Comparison between the errors of the trivial method and the parallel method

to establish with sufficient precision. However, such situations, i.e. relative positions of regions' centers, are not likely to occur in reality.

Having obtained the actual proximity, we have computed the approximate proximities proposed in this article for the same values of variables d_{xy}, r_x, and r_y. The comparison between the actual and the approximate proximity was quantified for each possible configuration as the error $\epsilon(r_x, r_y, d_{xy}) = |X^{actual}_{d_{xy}}(r_x, r_y) - X^{appr}_{d_{xy}}(r_x, r_y)|$. Given the high amount of resulting data, we have summarized them by computing the average over the radii r_x and r_y at a given distance between the centers, $\epsilon'_\mu(d_{xy})$, and the average over the distance between the balls' centers d_{xy} at a given pair of radii, $\epsilon''_\mu(r_x, r_y)$, specifically:

$$\epsilon'_\mu(d_{xy}) = Avg_{\mathbf{r_x r_y}}(\epsilon(\mathbf{r_x}, \mathbf{r_y}, d_{xy}))$$

and

$$\epsilon''_\mu(r_x, r_y) = Avg_{\mathbf{d_{xy}}}(\epsilon(r_x, r_y, \mathbf{d_{xy}})).$$

In similar way, we have computed the variance of the error for a given distance d_{xy}:

$$\epsilon'_{\sigma}(d_{xy}) = Var_{\mathbf{r_x}\mathbf{r_y}}(\epsilon(\mathbf{r_x}, \mathbf{r_y}, d_{xy}))$$

The value of ϵ'_{μ} makes possible to evaluate the average error of approximations for specific distance between the balls' centers. However, ϵ'_{μ} alone is not sufficient to correctly judge upon the quality of approximation. In fact, it is obtained as the average error for all possible values of r_x and r_y so that some peculiar behavior can remain hidden. In this respect, the stability of the error must also be considered. Such measure can be obtained by using the variance ϵ'_{σ}. Notice that high average errors and small variances may still provide good approximations. To illustrate, suppose that we want to use the proximity to order (rank) a set of regions with respect to a reference region. It can happen that the ranking results obtained through the actual and approximate proximity are identical even though ϵ'_{μ} is quite high. In fact, when the variance of error is very small, it means that the error is almost constant, and the approximation somehow follows the behavior of the actual proximity. In this case, it is highly probable that the approximated proximity increases (or decreases) according to the behavior of the actual one, guaranteeing the correct ordering.

On the other hand, the value of ϵ''_{μ} represents the average error from a different point of view. It is determined for a given pair of radii (r_x, r_y) varying d_{xy}, thus it can indicate the quality of approximation as a function of the sizes (radii) of balls. This measure offers a finer grained view on the error behavior, since the average is only computed varying the distance d_{xy}.

In fact, the measures ϵ'_{μ} and ϵ''_{μ} are complementary. The first allows one to judge the quality of approximation as a function of the distance between the centers of the balls, while the second helps to judge the quality of approximation as a function of balls' sizes.

5.3 Simulation results

For all data sets, the actual proximity was compared with the proposed approximations, the trivial approximation, and, only for the UV data set, with a proximity measure obtained using the analytic technique described in the Appendix A that as we have mentioned earlier has untrectable computational costs. Figure 6 presents the average error ϵ'_{μ} and its variance ϵ'_{σ}. It is immediate that all approximation methods outperform the trivial one, since the error of the trivial method is even one order of magnitude higher compared to all other methods. The same consideration also holds for the variance of the errors: for all proposed techniques, ϵ'_{σ} is one order of magnitude smaller than the value obtained with the trivial technique. This implies that in specific situations the trivial approximation may provide significantly different results with respect to the actual proximity. On the other hand, the proposed approximations show a very good and stable behavior. They have a small variance as well as small errors, so that they can be reliably used in practice.

If we compare the proposed methods for the UV data set, we can see that the parallel method is the best for values of d_{xy} up to the middle of the range of distances. However, the quality of this method decreases, both in terms of ϵ'_{μ} and ϵ'_{σ}, for high values of d_{xy}. In this range of distances, the best method is the analytic method (with the disadvantage of very high computational costs), followed by the normalized method. Notice that the normalized method presents a very stable behavior.

In the HV1 and HV2 data sets, the differences between the various methods are less evident. However, we can see again that the parallel method demonstrates the best performance. In

HV2, the quality reduction is limited and is noticeable just for very high values of d_{xy}. Here, again, the best method is the normalized approximation. On the other hand, the decrease in performance of the parallel method disappears for HV2, and a complete overlap of the graphs can be observed.

Consider now the average error for a given pair of radii ϵ''_μ. Since the parallel method has always been the best, we only consider ϵ''_μ for this method and the trivial one. The results are sketched in Figure 7. As an additional confirmation of the observation that we have made for ϵ'_μ and ϵ'_σ, the error ϵ''_μ for the parallel method is again smaller than the one measured for the trivial method in all three data sets. In particular, the error of the trivial method is never close to 0, while for a substantial range of r_x and r_y values the error of the parallel method is almost 0. This, in particular, is evident for the real-life data sets HV1 and HV2.

In the UV and HV1 data sets, the highest errors measured for the parallel method occur for values of r_x and r_y around the middle of the considered distance range. In the HV2 data set, the error is high for very large balls. The *step effect*, which can be observed in the graph corresponding to the UV data set, is due to the sampling granularity of d_{xy} and the fact that high values of the error ϵ are accumulated near to the constraint $x + y \geq d_{xy}$ of the triangular inequality. This effect is not noticeable in the other data sets, where the error ϵ observerd near that constraint is much smaller.

6 Conclusions

In order to support development of metric data indexes, approximation methods to quantify the proximity of metric ball regions have been proposed and evaluated. In accordance with our objectives, the proposed methods are *flexible* and do not depend on distance measure, provided it is a metric. *Accuracy* of the methods is high and only depends on global distance distribution, which is easy to obtain and store. The computation of proposed proximity measures is *fast*. Its computational complexity is linear, thus it is also applicable at run-time. The storage overhead of distance distribution histograms is *low*.

We are currently working on application of the method to improve the performance of metric trees. The specific problems concern the tree node *split* and *merge* functions, *ranking* of metric regions in priority queue for the *best case matching*, *declustering* of regions (partitions) to achieve parallelism, and *pruning* for approximate similarity retrieval.

Future research should concentrate on proximity measures of regions other than balls and on proximity of more than 2 regions. More effort should also be spend on developing other applications and possibly on developing new, more efficient, metric indexes.

References

[Bay99] Bay, S. D. The UCI KDD Archive [http://kdd.ics.uci.edu]. Irvine, CA: University of California, Department of Information and Computer Science.

[BKK96] S. Berchtold, D.A. Keim, and H.P. Kriegel. The X-tree: An Index Structure for High-Dimensional Data. Proceedings of the *VLDB96*, Bombay, India, 1996.

[BO97] T. Bozkaya and M. Ozsoyoglu. Distance-based indexing for high-dimensional metric spaces. *ACM SIGMOD*, pp.357-368, Tucson, AZ, May1997.

[BO99] T. Bozkaya and Ozsoyoglu. Indexing Large Metric Spaces for Similarity Search Queries. *ACM TODS*, 24(3):361-404, 1999.

[Br95] S. Brin. Near neighbor search in large metric spaces. In *Proceedings of the 21st VLDB International Conference*, pp. 574 -584, Zurich, Switzerland, September 1995.

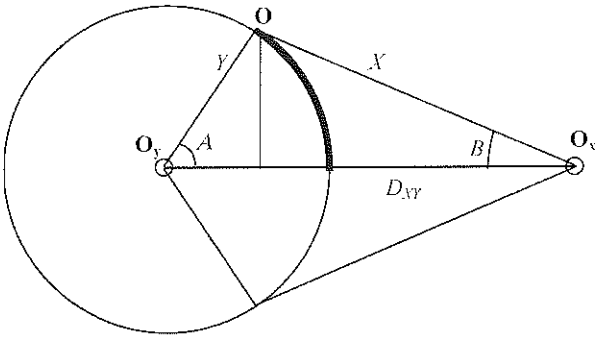


Fig. 8. Analytical approach

[Ch94] T. Chiueh. Content-based image indexing. In *Proceedings of the 20th VLDB International Conference*, pages 582–593, Santiago, Chile, September 1994.

[CPZ97] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. Proceedings of the *23rd VLDB Conference*, Athens, Greece, 1997, pp. 426-435.

[CPZ98] P. Ciaccia, M. Patella, and P. Zezula. A Cost Model for Similarity Queries in Metric Spaces. In *Proceedings of 7th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. PODS 1998*, Seattle, Washington, 1998, pp. 59- 68.

[Fa96] C. Faloutsos. *Searching Multimedia Databases by Content.* Kluwer Academic Publishers, 1996.

[Gu84] A. Guttman. R-trees: A dynamic index structure for spatial searching. In *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, pages 47–57, Boston, MA, June 1984.

[GRG+99] V. Ganti, R. Ramakrishnan, J. Gehrke, A. Powell, and J. French. Clustering Large Data Sets in Arbitrary Metric Spaces. In *Proceedings of the 15th International Conference on Data Engineering, ISDE99*, Sydney, Australia, IEEE, pp. 502-511, 1999.

[HD80] P.A.V Hall and G.R. Dowling. Approximate String Matching. *ACM Computing Surveys*, 12(4):381-402, December 1980.

[HKR93] D.P. Huttenlocker, G.A. Klanderman, and W.J. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850-863, September 1993.

[HS99] G.R. Hjaltason and H. Samet. Distance Browsing in Spatial Databases. *ACM TODS*, 24(2): 265-318 1999.

[KF92] I. Kamel and C. Faloutsos. Parallel R-trees. *Proc. of the ACM SIGMOD Conf.*, June 1992, pp. 195-204.

[Ko84] T. Kohonen. *Self-Organization and Associative Memory* Springer-Verlag, 1984.

[SO95] M. Stricker and M. Orengo. Similarity of Color Images. In: *Storage and Retrieval for Image and Video Databases III*, SPIE Proceedings 2420, 1995, pp. 381-392.

[Uh91] J.K. Uhlmann. Satisfying general proximity/similarity queries with metric trees. *Information Processing Letters*, 40(4):175–179, November 1991.

[ZSA+98] P. Zezula, P. Savino, G. Amato, and F. Rabitti. Approximate Similarity Retrieval with M-trees. *VLDB Journal*, 7(4):275-293, 1998.

A The analytic approach for the two dimension case

Here we discuss how $f_{X,Y|D_{XY}}(x, y|d_{xy})$ can be obtained analytically for the two dimensional case.

Let us suppose to have the continuous random variables X, Y, and D_{XY} as defined in Section 4. In addition let us define the two random variables A and B. A corresponds to the angle between the straight line passing through $\mathbf{O}$ and $\mathbf{O_y}$ and the straight line passing trough $\mathbf{O_x}$ and $\mathbf{O_y}$. B corresponds to the angle between the straight line passing through $\mathbf{O}$ and $\mathbf{O_x}$ and the straight line passing trough $\mathbf{O_x}$ and $\mathbf{O_y}$ Figure 8 sketches of these variables.

If we suppose that A has uniform distribution (this is an approximation, since it has been proven that it is not always true), we can define

$$f_{Y,A}(y,\alpha) = \frac{f(y)}{2\pi y} \tag{13}$$

and by symmetry we have that $f_{X,B}(x,\beta) = f_{Y,A}(y,\alpha)$.

Of course Equation 13 is true only in two dimensional vector spaces and in particular it cannot be applied to pure metric spaces.

Let's suppose that we have fixed y and d_{xy}. Let's suppose that we have an object O such that the angle A is θ. In correspondence of these values we can compute the values of X and B. We denote them $x(\theta)$ and $\beta(\theta)$ respectively. Let's now consider $F_{A|Y,D_{XY}}(\alpha|y,d_{xy})$ that is the probability that a random object $\mathbf{O}$ has and angle smaller than α, given that the distance between O_x and O_y is d_{xy} and the distance between O_y and Q is y. It can be computed as the ration between good cases and all possible cases. Good cases correspond to the probability that $\mathbf{O}$ is on the arc of circumference of radius y, centered in $\mathcal{O}_y$, corresponding to an angle of α and can be computed as the curvilinear integral of $f_{X,B}(x(\theta),\beta(\theta))$ over the circumference when $0 \leq \theta \leq \alpha$. All possible cases correspond to the probability that $\mathbf{O}$ is on the whole circumference of radius y, centered in $\mathcal{O}_y$, and can be computed as the curvilinear integral of $f_{X,B}(x(\theta),\beta(\theta))$ over the circumference when $0 \leq \theta \leq \pi$.

This results in the following:

$$F_{A|Y,D_{XY}}(\alpha|y,d_{xy}) = \frac{\int_0^\alpha f_{X,B}(x(\theta),\beta(\theta)) \cdot y \cdot d\theta}{\int_0^\pi f_{X,B}(x(\theta),\beta(\theta)) \cdot y \cdot d\theta} = \frac{\int_0^\alpha \frac{f(x(\theta))}{2\pi x(\theta)} \cdot y \cdot d\theta}{\int_0^\pi \frac{f(x(\theta))}{2\pi x(\theta)} \cdot y \cdot d\theta}$$

An equation for $x(\theta)$ can easily be obtained as follows:

$$x(\theta) = \sqrt{(y \cdot sen(\theta))^2 + (d_{xy} - y \cdot cos(\theta))^2} = \sqrt{y^2 + d_{xy}^2 - 2yd_{xy} \cdot cos(\theta)}$$

so

$$F_{A|Y,D_{XY}}(\alpha|y,d_{xy}) = \frac{\int_0^\alpha \frac{f(\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)})}{2\pi\cdot\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)}} \cdot d\theta}{\int_0^\pi \frac{f(\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)})}{2\pi\cdot\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)}} \cdot d\theta}$$

and by differentiating we obtain the density

$$f_{A|Y,D_{XY}}(\alpha|y,d_{xy}) = \frac{\frac{f(\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)})}{2\pi\cdot\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)}} \cdot d\theta}{\int_0^\pi \frac{f(\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)})}{2\pi\cdot\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)}} \cdot d\theta}$$

Now using $f_{A|Y,D_{XY}}(\alpha|y,d_{xy})$ we obtain $f_{X|Y,D_{XY}}(x|y,d_{xy})$. In fact we have that

$$F_{X|Y,D_{XY}}(x|y,d_{xy}) = P(X \leq x|y,d_{xy}) = P(\sqrt{y^2 + d_{xy}^2 - 2yd_{xy} \cdot cos(\theta)}) \leq x|y,d_{xy}) =$$

$$= P(-cos(\theta) \leq \frac{x^2 - y^2 - d_{xy}^2}{2yd_{xy}}|y,d_{xy}) = P(cos(\theta) \geq \frac{y^2 + d_{xy}^2 - x^2}{2yd_{xy}}|y,d_{xy}) =$$

$$= P(\theta \leq arccos(\frac{y^2 + d_{xy}^2 - x^2}{2yd_{xy}})|y, d_{xy}) = F_{A|Y,D_{XY}}(arccos(\frac{y^2 + d_{xy}^2 - x^2}{2yd_{xy}})|y, d_{xy})$$

$f_{X|Y,D_{XY}}(x|y, d_{xy})$ can be obtained by differentiating $F_{X|Y,D_{XY}}(x|y, d_{xy})$:

$$f_{X|Y,D_{XY}}(x|y, d_{xy}) = f_{A|Y,D_{XY}}(arccos(\frac{y^2 + d_{xy}^2 - x^2}{2yd_{xy}})|y, d_{xy}) \cdot \frac{2x}{2yd_{xy}\sqrt{1 - (\frac{y^2+d_{xy}^2-x^2}{2yd_{xy}})}} =$$

$$= \frac{\frac{f(x)}{x}}{\int_0^\pi \frac{f(\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)})}{\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)}} \cdot d\theta} \cdot \frac{2x}{\sqrt{4y^2d_{xy}^2 - (y^2 + d_{xy}^2 - x^2)^2}} =$$

$$= \frac{f(x)}{\int_0^\pi \frac{f(\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)})}{\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)}} \cdot d\theta} \cdot \frac{2}{\sqrt{4y^2d_{xy}^2 - (y^2 + d_{xy}^2 - x^2)^2}}$$

Now we can obtain $f_{X,Y|D_{XY}}(x, y|d_{xy})$ that was our initial goal:

$$f_{X,Y|D_{XY}}(x, y|d_{xy}) = f_{X|Y,D_{XY}}(x|y, d_{xy}) f_{Y|D_{XY}}(y|d_{xy}) = f_{X|Y,D_{XY}}(x|y, d_{xy}) f(y)$$

so

$$f_{X,Y|D_{XY}}(x, y|d_{xy}) = \frac{2f(x)f(y)}{\sqrt{4y^2d_{xy}^2 - (y^2 + d_{xy}^2 - x^2)^2} \cdot \int_0^\pi \frac{f(\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)})}{\sqrt{y^2+d_{xy}^2-2yd_{xy}\cdot cos(\theta)}} \cdot d\theta} \tag{14}$$

Notice that the cost of evaluating Equation 14 is $O(n)$, where n is the number of samples used to compute the integral. Computing proximity using Equation 8, when the joint conditional density is obtained by Equation 14, has complexity $O(n^3)$ that is of course not suitable for the applications where it should be used.