A Cost Model for Query Processing in High-Dimensional Data Spaces

Christian Böhm

Ludwig Maximilians Universität München

This is a preliminary release of an article accepted by ACM Transactions on Database Systems. The definitive version is currently in production at ACM and, when released, will supersede this version.

Name: Christian Böhm

Affiliation: Ludwig Maximilians Universität München

Address: Institut für Informatik, Oettingenstr. 67, 80538 München, Germany, boehm@informatik.uni-muenchen.de

---

During the last decade, multimedia databases have become increasingly important in many application areas such as medicine, CAD, geography or molecular biology. An important research issue in the field of multimedia databases is similarity search in large data sets. Most current approaches addressing similarity search use the so-called feature approach which transforms important properties of the stored objects into points of a high-dimensional space (feature vectors). Thus, the similarity search is transformed into a neighborhood search in the feature space. For the management of the feature vectors, multidimensional index structures are usually applied. The performance of query processing can be substantially improved by optimization techniques such as the blocksize optimization, data space quantization or dimension reduction. To determine optimal parameters an accurate estimation of the performance of index-based query processing is crucial. In this paper, we develop a cost model for index structures for point databases such as the R^*-tree or the X-tree. It provides accurate estimations of the number of data page accesses for range queries and nearest neighbor queries under Euclidean metric and maximum metric. The problems spe-

cific to high-dimensional data spaces, called boundary effects, are considered. The concept of the fractal dimension is used to take the effects of correlated data into account

Categories and Subject Descriptors: H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing H.2.8 [**Database Management**]: Database Applications
General Terms: Performance, Theory
Additional Key Words and Phrases: Cost model, multidimensional index.

---

1. INTRODUCTION

1.1 Motivation

Indexing high-dimensional data spaces is an emerging research domain. It gains increasing importance by the need to support modern applications with powerful search tools. In the so-called non-standard applications of database systems such as multimedia [Faloutsos *et al.* 1994a, Shawney and Hafner 1994, Seidl and Kriegel 1997], CAD [Berchtold 1997, Berchtold and Kriegel 1997, Berchtold *et al.* 1997c, Jagadish 1991, Gary and Mehrotra 1993, Mehrotra and Gary 1993, Mehrotra and Gary 1995], molecular biology [Altschul *et al.* 1990, Kastenmüller *et al.* 1998, Kriegel *et al.* 1997, Kriegel and Seidl 1998, Seidl 1997], medical imaging [Keim 1997, Korn *et al.* 1996], time series analysis [Agrawal *et al.* 1995, Agrawal *et al.* 1993, Faloutsos *et al.* 1994b], and many others, similarity search in large data sets is required as a basic functionality.

A technique widely applied for similarity search is the so-called feature transformation, where important properties of the objects in the database are mapped into points of a multidimensional vector space, the so-called feature vectors. Thus, similarity queries are naturally translated into neighborhood queries in the feature space.

In order to achieve a high performance in query processing, multidimensional index structures [Gaede and Günther 1998] are applied for the management of the feature vectors. Unfortunately, multidimensional index structures deteriorate in performance when the dimension of the data space increases, because they are primarily designed for low dimensional data spaces (2D and 3D) prevalent in spatial database systems whereas feature vectors are usually high-dimensional. Therefore, a number of specialized index structures for high-dimensional data spaces have been proposed. Most high-dimensional index structures are variants of the R-tree family [Guttman 1984, Beckmann *et al.* 1990, Sellis *et al.* 1987] using either rectangular or spherical page regions. Minimum bounding rectangles are used by the X-tree [Berchtold *et al.* 1996]. The SS-tree [White and Jain 1996] and the TV-tree [Lin *et al.* 1995] use spheres to describe the regions of the space covered by data pages and directory pages. The SR-tree [Katayama and Satoh 1997] uses a combination of a sphere and an MBR, the intersection solid, as page region. Other high-dimensional index structures which are not directly related to R-trees are the LSD^h-tree [Henrich 1998] and the Pyramid-tree [Berchtold *et al.* 1998b]. The Hybrid Tree combines the properties of data partitioning (e.g. R-tree) and space partitioning (e.g. k-d-B trees) index structures [Chakrabarti and Mehrotra 1999]. An index structure applicable in a parallel computer is the parallel X-tree [Berchtold *et al.* 1997a].

In spite of these efforts, there are still high-dimensional indexing problems under which even specialized index structures deteriorate in performance. Therefore, various optimization techniques for index structures have been proposed. The most common optimization parameter in index structures is the logical page size, i.e. the basic unit of transfer between

main memory and secondary storage. [Böhm 1998] and [Böhm and Kriegel 2000] propose a dynamic and independent page size optimization. It turns out that large page sizes are optimal in high-dimensional data spaces while in medium-dimensional cases smaller page sizes are optimal. The optimum depends not only on the dimension but also on the number of objects currently stored in the database and on the underlying data distribution. Weber, Schek and Blott [Weber *et al.* 1998] give evidence that there always exists a dimension which is high enough to make any thinkable index structure inefficient and propose the VA-file, a method which is based on the sequential scan of the data set (combined with a data compression method). [Böhm 1998] considers the sequential scan as a special case of an optimized index having an infinite page size.

A second optimization technique is the data space quantization which is essentially a data compression technique. It is applied in the VA-file [Weber *et al.* 1998] and in the IQ-tree [Berchtold *et al.* 2000]. The basic idea is to reduce the I/O time by representing the attributes not by their full 32 bit values but only with between 4 and 8 bits. As the representation of the vectors is inexact in this concept, a refinement step is required for the results, which causes additional I/O cost. A further price to pay is a greater CPU overhead for the decompression. If query processing is I/O-bound, the cost balance of data space quantization can be positive.

A third optimization technique for indexes is dimension reduction. The TV-tree [Lin *et al.* 1995] uses dimension reduction in the directory. Tree striping [Böhm 1998] is based on the idea of decomposing the feature vectors and storing the sub-vectors in separate indexes with moderate dimensionality. The results of query processing in the separate indexes must be joined. Seidl [Seidl 1997] and Faloutsos et al. [Faloutsos *et al.* 1994a] apply dimension reduction in the filter step of a multi-step query processing architecture.

For both the choice of a suitable index structure or query processing technique as well as for the determination of optimal parameters in optimization techniques it is crucial to provide accurate estimations of the cost of query processing. In this paper we present a methodology for cost modeling of high-dimensional index structures.

1.2 Influence Factors on the Performance of Query Processing

There are various factors which have an influence on the performance of index-based query processing. First of all the data set. The efficiency of index-based query processing depends on the dimension of the data space, the number of points in the database and on the data distribution from which the points are taken. Especially the correlation of dimensions is of high importance for the efficiency. Correlation means that the attributes of some dimensions are statistically not independent from each other. The value of one attribute is more or less determined by the values of one or more other attributes. In most cases, this dependency is not strict, but rather observable by the means of statistics. From a geometric point of view, correlation means that the data points are not spread over the complete data space. Instead, they are located on a lower-dimensional subset of the data space which is not necessarily a single linear subspace of the data space. For most index structures the performance of query processing improves if this intrinsic dimension of the data set is lower than the dimension of the data space. Our cost model takes these properties of the data set into account. The impact of the dimension and the number of data points is considered separately for low-dimensional data spaces (sections 3-4) and for high-dimensional data spaces (section 5). Section 5 develops also a criterion for distinguishing between the low-dimensional and the high-dimensional case. Correlation effects are handled by the concept of the fractal dimension in section 6.

The metric for measuring the distance between two data points (Euclidean metric, Manhattan metric, maximum metric) has an important influence on the query performance, too.

The cost formulas are thus presented for the two most important metrics, the Euclidean and the maximum metric throughout the whole paper. While the Manhattan metric is relatively straightforward, modeling more complex kinds of metrics such as quadratic form distance functions [Seidl 1997, Seidl and Kriegel 1997] is future work.

A second set of influence factors is connected with the index structure. Most important is the shape of the page regions: It can be a rectangle, a sphere or a composed page region. If it is a rectangle, it can be a minimum bounding rectangle or it can be part of a complete decomposition of the data space such as in the k-d-B-tree or in the LSD^h-tree. Most difficult to capture in a model are the various heuristics which are applied during insert processing and index construction. The impact of the heuristic on the volume and extension of page regions is very hard to quantify. A further influence factor is the layout of pages on the secondary storage. If the layout is clustered, i.e. pairs of adjacent pages are likely to be near by each other on disk, the performance can be improved if the query processing algorithm is conscious of this clustering effect. Our cost estimations are presented for rectangular page regions. Considering spherical page regions or cylinders is straightforward, whereas more complex shapes such as composed or intersected solids are difficult. The impact of insertion heuristics is considered neither in this paper nor in any cost model for multidimensional query processing known to the author. In fact, these cost models assume idealized index structures which do not yield deteriorations such as heavy overlap among the page regions. Such parameters could eventually be measured for a specific index and be integrated into the formulas, but this is future work.

A third set of influence factors is due to the choice of the query processing algorithm. The HS algorithm proposed by Hjaltason and Samet [Hjaltason and Samet 1995] yields a better performance in terms of page accesses than the RKV algorithm by Roussopoulos, Kelley and Vincent [Roussopoulos *et al.* 1995]. Disk clustering effects can be exploited by algorithms considering the relative positions of pages on the background storage. We will assume a depth-first search strategy for range queries and the HS algorithm for nearest neighbor queries.

1.3 Paper Outline

After reviewing some related work on cost models, we start with the introduction of modeling range queries assuming an independent and uniform distribution of the data points. Moreover, we assume in the beginning that queries do not touch the boundary of the data space. Range queries are transformed into equivalent point queries by accordingly adapting the page regions. The central concept for this compensating adaptation is called *Minkowski sum* or *Minkowski enlargement*. We determine from the Minkowski sum the access probability of pages and use this access probability to develop an expectation for the number of page accesses. In the next section, nearest neighbor queries evaluated by the HS algorithm are modeled. This is conceptually done by a reduction step which transforms nearest neighbor queries into an equivalent range query. The corresponding range r can be estimated by a probability density function $p(r)$ using r as variable.

The simplifying assumptions of a uniform and independent data distribution and the ignorance of data space boundaries will be dropped step by step in the subsequent sections. First, the so-called boundary effects are introduced and shown to be important in high-dimensional data spaces. Our model is modified to take boundary effects into account. Then, we consider non-uniform data distributions which are independent in all dimensions. Last, we formalize correlations by the means of the *fractal dimension* and integrate this concept into our cost models for range queries and nearest neighbor queries. Experimental evalua-

tions showing the practical applicability of the cost models and their superiority over related approaches are provided in each section separately.

1.4 Terminology

Due to the high variety of cost formulas we will develop in this paper the notation and the identifiers we have to use are complex. There are a few general identifiers for basic cost measures which will be used throughout this paper:

V	for some volume
R	for the expected distance between a query point and its nearest neighbor
P	for some probability distribution function
X	for the access probability of an individual page
A	for the expectation of the number of page accesses.

The boundary conditions for the corresponding cost measure such as the distance metric and basic assumptions such as uniformity or independence in the distribution of data points are denoted by subscripted indices of the general identifiers. For instance, $X_{\text{nn,em,ld,ui}}$ means the access probability for a nearest neighbor query (nn) using the Euclidean metric (em) on a low-dimensional data space (ld) under uniformity and independence assumption (ui). We distinguish:

- the query type: range query (r), nearest neighbor (nn) and *k*-nearest neighbor (knn)
- the applied metric: Euclidean metric (em) and maximum metric (mm)
- the assumed dimensionality: low-dimensional (ld) and high-dimensional (hd)
- the data distribution assumption: uniform/independent (ui), correlated (cor).

Especially the metric identifier (em or mm) is sometimes left out if an equation is valid for all applied metrics. In this case, all terms lacking the metric specification are understood to be substituted by the same metric, of course. The volume V is specified by a few indices indicating the geometric shape of the volume. Basic shapes are the hyper-sphere (S), the hyper-cube (C) and the hyper-rectangle (R). The Minkowski sum (cf. section 3) of two solids o_1 and o_2 is marked by a plus symbol ($o_1 \oplus o_2$). The clipping operation (cf. section 5) is marked by the intersection symbol ($o_1 \cap o_2$). Often, indices denote only the type of some geometrical object: $V_{R \oplus S}$ stands for the Volume of a Minkowski sum of a rectangle and a sphere.

We use the following notational conventions throughout this paper: Our database consists of a set of N points in a d-dimensional data space $DS \subseteq \Re^d$. Usually, we will restrict ourselves to the data spaces normalized to the unit hypercube $[0..1]^d$. It is unclear to what extent the normalization of the data space affects the correctness of the result set. This depends on the mapping between the object space and the feature space in the feature transformation which is out of the scope of this paper. Generally, the multi-step architecture of query processing requires that in the feature space more objects must be retrieved than the user requests as similar objects. The ratio is called the filter selectivity. The mapping into a normalized data space may affect this filter selectivity positively or negatively. The problem of modeling the filter selectivity for various mappings and similarity measures is subject to our future work. We will use the Euclidean metric δ_{em} and the maximum metric δ_{mm} for distance calculations between two points P and Q:

$$\delta_{\text{em}}(P, Q) = \sqrt[2]{\sum_{0 \le i < d} (Q_i - P_i)^2} \quad \text{and} \quad \delta_{\text{mm}}(P, Q) = \max_{0 \le i < d} \{|Q_i - P_i|\} \quad .$$

The average number of points stored in a data page will be denoted as the *effective data page capacity* $C_{\text{eff,data}}$. In our cost estimations, we do not consider directory page accesses for two reasons: First, directory pages are often assumed to be resident in the database cache after their first access [Berchtold *et al.* 1996]. But even if the cache is too small to store the complete directory, the number of accessed data pages is often even larger than the number of *available* directory pages. Even if the ratio of accessed pages decreases when moving from the root level to the leaf level, the data page accesses are the predominant factor in the query processing cost. For concreteness, we restrict ourselves to point access methods with rectilinear page regions minimizing the overlap in the index construction. The model can be extended for structures with spherical regions in a straightforward way.

1.5 Algorithms for Query Processing in High-Dimensional Spaces

In this paper, we will consider two kinds of queries: range queries and nearest neighbor queries. A range query is defined as follows: We are given a database DB of N points in a d-dimensional data space, a *query point* Q, a radius r called *query range* and a distance metric δ. Retrieve all points in the database which have a δ-distance from Q not greater than r:

$$\text{RangeQuery}(\text{DB},Q,r,\delta) = \{P \in \text{DB} | \delta(P, Q) \leq r\}$$

By Q and r, a high-dimensional hypersphere is defined, the *query sphere*. An index based algorithm for range queries traverses the tree in a depth-first search. All pages intersecting the query sphere are accessed.

In a k-nearest neighbor search, the region of the data space where data points are searched is not previously known. Given is also the database DB, the *query point* Q, the distance metric δ and a number k of objects to be retrieved from the database. The task is to select the k points from the database with minimum distance from Q:

$$\text{kNNQuery}(\text{DB}, Q, k, M) = \{P_0 \ldots P_{k-1} \in \text{DB} \mid \neg\exists P' \in \text{DB} \setminus \{P_0 \ldots P_{k-1}\} \wedge \neg\exists i, 0 \leq i < k : \delta(P_i, Q) > \delta(P', Q)\}$$

This can be determined in a depth-first search, as in the RKV algorithm by Roussopoulos, Kelley and Vincent [Roussopoulos *et al.* 1995]. This algorithm keeps a list of the k closest points which is updated whenever a new, closer point is found. The distance of the last point in the list is used for pruning branches of the tree from being processed. Although several further information is used for pruning, the algorithm cannot guarantee a minimum number of page accesses. The HS algorithm proposed by Hjaltason and Samet [Hjaltason and Samet 1995] and in a similar form earlier by Henrich [Henrich 1994] performs neither a depth-first search nor a breadth-first search, but accesses pages in an order by increasing distance from the query point. This can be implemented using a priority queue. The algorithm stops when all pages with a distance less than the current pruning distance are processed. The details of this algorithm can be taken from the original literature [Henrich 1994, Hjaltason and Samet 1995] or one of the secondary publications [Berchtold *et al.* 1997b, Böhm 1998]. In contrast to the RKV algorithm, the optimality of the HS algorithm is provable (cf. [Berchtold *et al.* 1997b]), because it accesses exactly the pages in the *knn-sphere*, i.e. a sphere around the query point which touches the k-th nearest neighbor. In contrast, RKV accesses a number of additional pages which depends on the effectivity of the applied heuristics. Therefore, the performance of the RKV algorithm is difficult to estimate, and we will restrict ourselves to the HS algorithm in our cost model.

2. REVIEW OF RELATED COST MODELS

Due to the high practical relevance of multidimensional indexing, cost models for estimating the number of necessary page accesses have already been proposed several years ago. The first approach is the well-known cost model proposed by Friedman, Bentley and Finkel [Friedman *et al.* 1977] for nearest neighbor query processing using maximum metric. The original model estimates leaf accesses in a kd-tree, but can be easily extended to estimate data page accesses of R-trees and related index structures. This extension was published in 1987 by Faloutsos, Sellis and Roussopoulos [Faloutsos *et al.* 1987] and with slightly different aspects by Aref and Samet in 1991 [Aref and Samet 1991], by Pagel, Six, Toben and Widmayer in 1993 [Pagel *et al.* 1993] and by Theodoridis and Sellis in 1996 [Theodoridis and Sellis 1996]. The expected number of data page accesses in an R-tree is

$$A_{\text{nn,mm,FBF}} = \left(\sqrt[d]{\frac{1}{C_{\text{eff}}}} + 1 \right)^d .$$

This formula is motivated as follows: The query evaluation algorithm is assumed to access an area of the data space which is a hyper cube of the volume $V_1 = 1/N$, where N is the number of objects stored in the database. Analogously, the page region is approximated by a hypercube with the volume $V_2 = C_{\text{eff}}/N$. In each dimension, the chance that the projection of V_1 and V_2 intersect each other, corresponds to $\sqrt[d]{V_1} + \sqrt[d]{V_2}$ if $n \to \infty$. To obtain a probability that V_1 and V_2 intersect in all dimensions, this term must be taken to the power of d. Multiplying this result with the number of data pages N/C_{eff}, yields the expected number of page accesses $A_{\text{nn,mm,FBF}}$. The assumptions of the model, however, are unrealistic for nearest neighbor queries on high-dimensional data for several reasons. First, the number N of objects in the database is assumed to approach to the infinity. Second, effects of high-dimensional data spaces and correlations are not considered by the model. Cleary [Cleary 1979] extends the model of Friedman, Bentley and Finkel [Friedman *et al.* 1977] by allowing non-rectangular page regions, but still does not consider boundary effects and correlations. Eastman [Eastman 1981] uses the existing models for optimizing the bucket size of the kd-tree. Sproull [Sproull 1991] shows that the number of data points must be exponential in the number of dimensions for the models to provide accurate estimations. According to Sproull, boundary effects significantly contribute to the costs unless the following condition holds:

$$N >> C_{eff} \cdot \left(\sqrt[d]{\frac{1}{C_{eff} \cdot V_S\left(\frac{1}{2}\right)}} + 1 \right)^d$$

where $V_S(r)$ is the volume of a hypersphere with radius r which can be computed as

$$V_S(r) = \frac{\sqrt{\pi}^d}{\Gamma(d/2 + 1)} \cdot r^d$$

with the gamma-function $\Gamma(x)$ which is the extension of the factorial operator $x! = \Gamma(x+1)$ into the domain of real numbers: $\Gamma(x+1) = x \cdot \Gamma(x)$, $\Gamma(1) = 1$ and $\Gamma(\frac{1}{2}) = \sqrt{\pi}$.

For example, in a 20-dimensional data space with $C_{\text{eff}} = 20$, Sproull's formula evaluates to $N >> 1.1 \cdot 10^{11}$. We will see later, how bad the cost estimations of the FBF model are if substantially fewer than a hundred billion points are stored in the database. Unfortunately, Sproull still assumes for his analysis uniformity and independence in the distribution of data

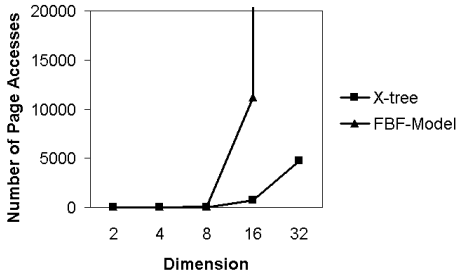

Figure 1: Evaluation of the model of Friedman, Bentley and Finkel.

points and queries, i.e. both data points and the center points of the queries are chosen from a uniform data distribution, whereas the selectivity of the queries ($1/N$) is considered fix. The above formulas are also generalized to k-nearest neighbor queries, where k is also a user-given parameter.

The assumptions made in the existing models do not hold in the high-dimensional case. The main reason for the problems of the existing models is that they do not consider boundary effects. *"Boundary effects"* stands for an exceptional performance behavior, when the query reaches the boundary of the data space. As we show later, boundary effects occur frequently in high-dimensional data spaces and lead to pruning of major amounts of empty search space which is not considered by the existing models. To examine these effects, we performed experiments to compare the necessary page accesses with the model estimations. Figure 1 shows the actual page accesses for uniformly distributed point data versus the estimations of the model of Friedman, Bentley and Finkel. For high-dimensional data, the model completely fails to estimate the number of page accesses.

The basic model of Friedman, Bentley and Finkel has been extended in two different directions. The first is to take correlation effects into account by using the concept of the fractal dimension [Mandelbrot 1977, Schröder 1991]. There are various definitions of the fractal dimension which all capture the relevant aspect (the correlation), but are different in the details, how the correlation is measured. We will not distinguish between these approaches in our subsequent work.

Faloutsos and Kamel [Faloutsos and Kamel 1994] used the *box-counting fractal dimension* (also known as the *Hausdorff fractal dimension*) for modeling the performance of R-trees when processing range queries using maximum metric. In their model they assume to have a correlation in the points stored in the database. For the queries, they still assume a uniform and independent distribution. The analysis does not take into account effects of high-dimensional spaces and the evaluation is limited to data spaces with dimensions less or equal to 3. Belussi and Faloutsos [Belussi and Faloutsos 1995] used in a subsequent paper the fractal dimension with a different definition (the *correlation fractal dimension*) for the selectivity estimation of spatial queries. In this paper, range queries in low-dimensional data spaces using Manhattan metric, Euclidean metric and maximum metric were modeled. Unfortunately, the model only allows the estimation of selectivities. It is not possible to extend the model in a straightforward way to determine expectations of page accesses.

Papadopoulos and Manolopoulos used the results of Faloutsos and Kamel and the results of Belussi and Faloutsos for a new model published in a recent paper [Papadopoulos and

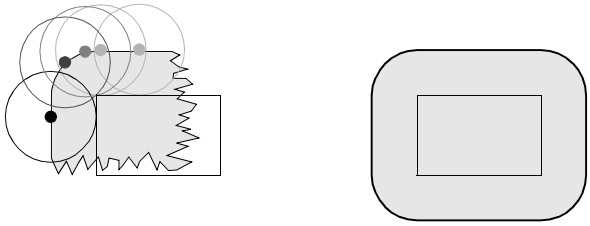

Figure 2: The Minkowski Sum.

Manolopoulos 1997]. Their model is capable of estimating data page accesses of R-trees when processing nearest neighbor queries in a Euclidean space. They estimate the distance of the nearest neighbor by using the selectivity estimation of Belussi and Faloutsos [Belussi and Faloutsos 1995] in the reverse way. We will point out in section 4 that this approach is problematic from a statistical point of view. As it is difficult to determine accesses to pages with rectangular regions for spherical queries, they approximate query spheres by minimum bounding and maximum enclosed cubes and determine upper and lower bounds of the number of page accesses in this way. This approach makes the model inoperative for high-dimensional data spaces, because the approximation error grows exponentially with increasing dimension. Note that in a 20-dimensional data space, the volume of the minimum bounding cube of a sphere is by a factor of $1/V_S(1/2) = 4.1 \cdot 10^7$ larger than the volume of the sphere. The sphere volume, in turn, is by $V_S(\sqrt{d}/2) = 27,000$ times larger than the greatest enclosed cube. An asset of the model of Papadopoulos and Manolopoulos is that queries are no longer assumed to be taken from a uniform and independent distribution. Instead, the authors assume that the query distribution follows the data distribution.

The concept of fractal dimension is also widely used in the domain of spatial databases, where the complexity of stored polygons is modeled [Gaede 1995, Faloutsos and Gaede 1996]. These approaches are of minor importance for point databases.

The second direction, where the basic model of Friedman, Bentley and Finkel needs extension, are the boundary effects occurring when indexing data spaces of higher dimensionality.

Arya, Mount and Narayan [Arya *et al.* 1995, Arya 1995] presented a new cost model for processing nearest neighbor queries in the context of the application domain of vector quantization. Arya, Mount and Narayan restricted their model to the maximum metric and neglected correlation effects. Unfortunately, they still assume that the number of points is exponential with the dimension of the data space. This assumption is justified in their application domain, but it is unrealistic for database applications.

Berchtold, Böhm, Keim and Kriegel [Berchtold *et al.* 1997b] presented in 1997 a cost model for query processing in high-dimensional data spaces, in the following called *BBKK model*. It provides accurate estimations for nearest neighbor queries and range queries using the Euclidean metric and considers boundary effects. To cope with correlation, the authors propose to use the fractal dimension without presenting the details. The main limitation of the model are (1) that no estimation for the maximum metric is presented, (2) that the number of data pages is assumed to be a power of two and (3) that a complete, overlap-free coverage of the data space with data pages is assumed. Weber, Schek and Blott [Weber *et al.* 1998] use the cost model by Berchtold et al. without the extension for correlated data to

show the superiority of the sequential scan in sufficiently high dimensions. They present the VA-file, an improvement of the sequential scan. Ciaccia, Patella and Zezula [Ciaccia *et al.* 1998] adapt the cost model [Berchtold *et al.* 1997b] to estimate the page accesses of the M-tree, an index structure for data spaces which are metric spaces but not vector spaces (i.e. only the distances between the objects are known, but no explicit positions). Papadopoulos and Manolopoulos [Papadopoulos and Manolopoulos 1998] apply the cost model for declustering of data in a disk array. Two papers [Riedel *et al.* 1998, Agrawal *et al.* 1998] present applications in the data mining domain.

This paper is based on the BBKK cost model which is presented in a comprehensive way and extended in many aspects. The extensions not yet covered by the BBKK model include all estimations for the maximum metric, which are developed additionally throughout the whole paper. The restriction of the BBKK model to numbers of data pages which are a power of two is overcome (cf. sections 5.2 and 5.3). A further extension of the model regards *k*-nearest neighbor queries. The corresponding cost formulas are presented in section 4.4. The numerical methods for integral approximation and for the estimation of the boundary effects were to the largest extent out of the scope of [Berchtold *et al.* 1997b]. Improved evaluation methods which are not contained in the BBKK model are described in sections 4.3, 5.2 and 5.3. These evaluation methods based on precomputed volume functions for intersection solids have a high practical importance, because costly numerical methods such as a Montecarlo integration can be completely performed during the compile time. Finally, the concept of the fractal dimension, which was also used in the BBKK model in a simplified way (the data space dimension is simply replaced by the fractal dimension) is in this paper well established by the consequent application of the fractal power laws.

3. RANGE QUERY

In this section, we assume uniformity and independence in the distribution of both, data and query points. Moreover, we ignore the existence of a boundary of the data space or assume at least that page regions and queries are distant enough from the space boundary that the boundary is not touched. We start with a given page region and a given query range *r* and determine the probability with which the page is accessed under the uniformity and independence assumption for the query point. The uniformity and independence assumption means that the query point is a *d*-dimensional stochastic variable which is distributed such that each position of the data space has the same probability.

3.1 The Minkowski Sum

The corresponding page is accessed, whenever the query sphere intersects with the page region. To illustrate this, cf. figure 2. In all figures throughout this paper, we will symbolize queries by spheres and page regions by rectangles. Also our terminology (“*query sphere*”, for example) will often reflect this symbolization. We should note that queries using the maximum metric rather correspond to hypercubes than hyperspheres. We should further note that not all known index structures use hyperrectangles as page regions. Our concepts presented here are also applicable if the shape of the query is cubical or the shape of the page region is spherical.

For estimating the access probability of a page, we make conceptually the following transformation: We enlarge the page region so that if the original page touched any point of the query sphere, then the enlarged page touches the center point of the query. It is obvious from figure 2 that the page region becomes enlarged by a sphere of the same radius *r* whose center point is drawn over the surface of the page region. All positions inside the grey

marked region of figure 2 are the positions of the query point, where the page is accessed, and all positions outside the grey area are the positions of the query point, where the page is not accessed. Therefore, the marked volume divided by the data space volume directly corresponds to the access probability of the page. As we assume for simplicity that the unit hypercube $[0..1]^d$ is the data space, the data space volume corresponds to 1.

The concept of enlarging a given geometrical object by another object in the described way is called Minkowski sum [Kaul *et al.* 1991]. The Minkowski sum of two objects A and B is defined as the vector sum of all points of the objects:

$$A \oplus B = \{a + b \mid a \in A, b \in B\}$$

The main application of the Minkowski sum is robot motion planning. In the context of cost modeling, only the volume $V_{A \oplus B}$ of the Minkowski sum is needed. Therefore, we will not strictly distinguish between the Minkowski sum as a spatial object (solid) and its volume. For the determination of the volume, we have to consider various cases. The simplest case is that both solids are hyperrectangles with side lengths a_i and b_i, for $0 \le i < d$. In this case, the volume $V_{R \oplus R}$ of the Minkowski sum of two rectangles is the rectangle with the side lengths c_i, where each c_i corresponds to the sum of a_i and b_i:

$$V_{R \oplus R}((a_0, \ldots, a_{d-1}), (b_0, \ldots, b_{d-1})) = \prod_{0 \le i < d} (a_i + b_i).$$

If both solids, query sphere and page region are hyperspheres with radius r_q and r_p, the Minkowski enlargement corresponds to a hypersphere with the radius $r_q + r_p$. The corresponding volume of the hypersphere can be evaluated by the following formula:

$$V_{S \oplus S}(r_q, r_p) = \frac{\sqrt{\pi}^d}{\Gamma\left(\frac{d}{2} + 1\right)} \cdot (r_q + r_p)^d.$$

The evaluation of the volume becomes more complex if query and page region are shaped differently. This is illustrated in a 3-dimensional example in figure 3, which depicts some of the solids by which the page region (gray) becomes enlarged: At each corner of the gray solid, we have a part (1/8) of a sphere (drawn is only one part s_1). The composed parts would form one complete sphere of radius r (the query radius). At each edge of the region, we have a part (1/4) of a cylinder. The parts form three complete cylinders where the radius is the query radius. The length of c_1, for instance, is the width of the page region. Last, at each surface of the region, we enlarge the region by a cuboid where two side lengths are defined by the surface of the region and the remaining side length is again r. In the general d-dimensional case, our page region has not only corners, edges and 2-dimensional surfaces, but surface segments with dimensions up to $d - 1$. Each surface with dimensionality k is enlarged by a part of a hypercylinder which is spherical in $d - k$ dimensions. In the remaining k dimensions, the hypercylinder has the shape of the surface segment to which it is connected.

Before we determine the volume of the Minkowski enlargement in the most complex case of a hypersphere and a hyperrectangle, let us solve it for the simpler case that the page region is a hypercube with side length a. In this case, all k-dimensional surface segments have the volume a^k. Still open is the question, how many such surface segments exist and what the volume of the assigned part of the hypercylinder is. The number of surface segments can be determined by a combinatorial consideration. All surface segments (including the corners and the page region itself) can be represented by a d-dimensional vector over the

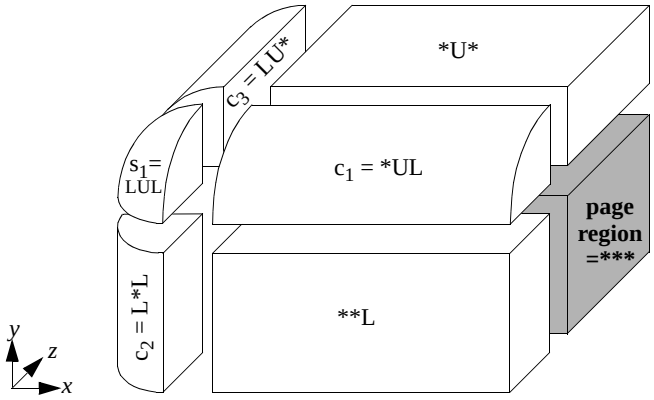

Figure 3: Example of a 3-dimensional Minkowski sum.

symbols ‘L’, ‘U’ and ‘*’. Here, the symbol ‘L’ stands for the *lower* boundary, ‘U’ for the *upper* boundary and the star stands for the complete interval between the lower and upper boundary. Each position of a symbol L, U or * corresponds to one dimension of the original data space. Using this notation, the corners have no star, the edges have one star, the 2-dimensional surfaces have two stars in the vector, and so on. The hyperrectangle itself has *d* stars, no ‘L’ and no ‘U’ in its description vector. In our example in figure 3, s_1 is assigned to the vector (LUL), and c_1 is assigned to the vector (*UL), because it covers the entire range in the *x*-direction and it is fixed at the upper end of the *y*-direction and at the lower end of the *z*-direction. The number of *k*-dimensional surface segments corresponds to the number of different vectors having *k* stars. The positions of the stars can be arbitrarily selected from *d* positions in the vector, yielding a binomial number of possibilities. The remaining $d-k$ positions are filled with the symbols ‘L’ and ‘U’. Therefore, the number of surface segments SSEGM(*k*) of dimension *k* is equal to:

$$\text{SSEGM}(k) = \binom{d}{k} \cdot 2^{d-k}.$$

The fraction of the hypercylinder at each surface segment is $1/2^{d-k}$, because the cylinder is halved for each of the $d-k$ dimensions. Therefore, we get the following formula for the Minkowski sum of a hypersphere with radius *r* and a hypercube with side length *a*:

$$V_{S \oplus C}(r,a) = a^d + \sum_{0 \le k < d} \text{SSEGM}(k) \cdot a^k \cdot \frac{1}{2^{d-k}} \cdot \frac{\sqrt{\pi^{d-k}}}{\Gamma\left(\frac{d-k}{2}+1\right)} \cdot r^{d-k}$$

$$= \sum_{0 \le k \le d} \binom{d}{k} \cdot a^k \cdot \frac{\sqrt{\pi^{d-k}}}{\Gamma\left(\frac{d-k}{2}+1\right)} \cdot r^{d-k}.$$

In the most complex case of non-cubical hyperrectangles, the *k*-dimensional surface segments must be summed up explicitly, which is a costly operation. Instead of the binomial

multiplied with a^k, we have to summarize over all k combinations of the side lengths of the hyperrectangle, i.e. the power set $2^{\{0\ldots d-1\}}$:

$$V_{S \oplus R}(r, \vec{a}) = \sum_{0 \le k \le d} \left(\sum_{\{i_1 \ldots i_k\} \in 2^{\{0\ldots d-1\}}} \left(\prod_{j=1}^{k} a_{i_j} \right) \right) \cdot \frac{\sqrt{\pi}^{d-k}}{\Gamma\left(\frac{d-k}{2}+1\right)} \cdot (r)^{d-k} \quad .$$

We should note that in most cases the determination of the Minkowski sum of a hypersphere and a hyperrectangle is an operation which is too costly, because it involves a number of basic operations (multiplications), which is exponential in the dimension. The explicit determination of the Minkowski sum of a real hyperrectangle and a hypersphere of high dimensionality must be avoided, even if exactness is sacrificed.

A usual work-around is to transform the page region into a hypercube with equivalent volume. However, exactness is sacrificed in this approach, because the Minkowski sum of a non-cubical hyperrectangle is larger than the Minkowski sum of a volume-equivalent hypercube. Our experiments later will show that this approximation does not cause serious errors in the cost estimations for the X-tree, which strives for square-like page regions. For other index structures, this effect could play a more important role. Estimating access probabilities in such cases is subject to future work.

3.2 Estimating Rectangular Page Regions

To make modeling manageable, we assume page regions which have the form of a hypercube. Observe that structures as the X-tree or the R*-tree try to produce such page regions and that our prediction matches the experimental observation. We exploit that the number of points enclosed in an arbitrary hypercube of side length a_{nonbound} is proportional to the volume of this hypercube. By the index nonbound we indicate that this hypercube is not the minimum bounding rectangle of the enclosed points, i.e. a_{nonbound} is the expected side length of the page regions before taking the MBR effect (*minimum bounding rectangle*) into account. Similarly, $V_{\text{R,nonbound}}$ is the volume of the page region without considering the MBR effect. Due to the proportionality, the number C_{eff} of points stored in a page divided by the number N of all points in the database is equal to the volume of the page divided by the volume V_{DS} of the data space:

$$\frac{C_{\text{eff}}}{N} = \frac{V_{\text{R,nonbound}}}{V_{\text{DS}}} = \frac{a_{\text{nonbound}}^d}{1} \quad \Rightarrow \quad a_{\text{nonbound}} = \sqrt[d]{\frac{C_{\text{eff}}}{N}} \quad .$$

In this formula for the side length of a typical page region, we assume a complete coverage of the data space with page regions. This assumption is not meaningful for minimum bounding rectangles. Usually, there is a small gap between two neighboring page regions. An expectation for the width of this gap under uniformity and independence assumption can be determined by projecting all points of a page onto the coordinate axis which is perpendicular to the gap. The average distance between two neighboring projections is $1/C_{\text{eff}}$ times the side length of the region, because the projections of the points are uniformly distributed. This is also the expected value for the width of the gap by which the side length of the page region is decreased compared with a_{nonbound}. Therefore, the side length of a minimum bounding rectangle can be estimated as:

$$a = \left(1 - \frac{1}{C_{\text{eff}}}\right) \cdot a_{\text{nonbound}} = \left(1 - \frac{1}{C_{\text{eff}}}\right) \cdot \sqrt[d]{\frac{C_{\text{eff}}}{N}} \quad .$$

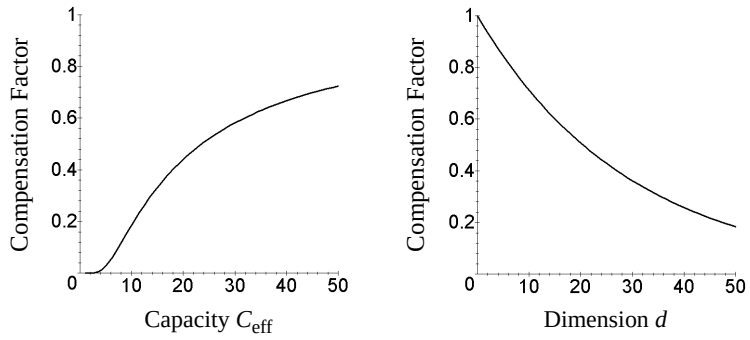

Figure 4: The Compensation Factor for Considering Gaps.

The consideration of gaps between page regions is particularly important if the effective page capacity is low. Figure 4 shows the compensation factor for varying page capacities (left side) and for varying dimensions (right side). It shows the factor which decreases the volume of a page region when gaps are considered. If the compensation factor is 1, no compensation is needed. The smaller the compensation factor is, the higher is the impact of the MBR effect. The left diagram shows the compensation factor for a fixed dimension $d = 16$ with varying capacity C_{eff}. The strongest MBR effect occurs for low capacities. For a typical capacity between 20 and 40 points per data page (which is the most frequently applied capacity according to our experience), the compensation factor ranges between 40% and 70%. The right diagram shows the compensation factor for a fixed effective page capacity (30 points) and varying dimension. Most compensation is necessary for large dimensions.

3.3 Expected Number of Page Accesses

By inserting the expected side length a into the formulas for the Minkowski enlargement, it is possible to determine the access probabilities of typical data pages under uniformity and independence assumption. This is for the maximum metric:

$$X_{\text{r,mm,ui}}(r) \;=\; (2r+a)^d \;=\; \left(2r + \left(1 - \frac{1}{C_{\text{eff}}}\right) \cdot \sqrt[d]{\frac{C_{\text{eff}}}{N}}\right)^d \quad .$$

Note again that the volume corresponds to the access probability as the data space is normalized to the unit hypercube. For Euclidean metric, the access probability for range queries with radius r evaluates to:

$$\begin{aligned} X_{\text{r,em,ui}}(r) &= \sum_{0 \le k \le d} \binom{d}{k} \cdot a^k \cdot \frac{\sqrt{\pi^{d-k}}}{\Gamma\left(\frac{d-k}{2}+1\right)} \cdot r^{d-k} \\ &= \sum_{0 \le k \le d} \binom{d}{k} \cdot \left(\left(1 - \frac{1}{C_{\text{eff}}}\right) \cdot \sqrt[d]{\frac{C_{\text{eff}}}{N}}\right)^k \cdot \frac{\sqrt{\pi^{d-k}}}{\Gamma\left(\frac{d-k}{2}+1\right)} \cdot r^{d-k} \quad . \end{aligned}$$

From these access probabilities, the expected number of page accesses can be determined by multiplying the access probability with the number of data pages N/C_{eff}:

$$A_{\text{r,mm,ui}}(r) = \left(2r \cdot \sqrt[d]{\frac{N}{C_{\text{eff}}}} + 1 - \frac{1}{C_{\text{eff}}}\right)^d .$$

For Euclidean metric, the corresponding result is:

$$A_{\text{r,em,ui}}(r) = \frac{N}{C_{\text{eff}}} \cdot \sum_{0 \le k \le d} \binom{d}{k} \cdot \left(\left(1 - \frac{1}{C_{\text{eff}}}\right) \cdot \sqrt[d]{\frac{C_{\text{eff}}}{N}}\right)^k \cdot \frac{\sqrt{\pi^{d-k}}}{\Gamma\left(\frac{d-k}{2} + 1\right)} \cdot r^{d-k} .$$

4. NEAREST NEIGHBOR QUERY

In [Berchtold *et al.* 1997c] and [Böhm 1998], the optimality of the HS algorithm [Hjaltason and Samet 1995] for nearest neighbor search was proven. The HS algorithm yields exactly the same page accesses as an equivalent range query, i.e. a range query using the distance to the nearest neighbor as query range. This enables us to reduce the problem of modeling nearest neighbor queries to the problem of modeling range queries, which was solved in section 3. Therefore, we have to estimate the nearest neighbor distance and apply this distance as the radius in the range query model.

Like in section 3 we start with the assumptions of an independent, uniform data distribution and we will ignore boundary effects. These effects will be investigated in depth in section 5 and section 6.

4.1 Coarse Estimation of the Nearest Neighbor Distance

A simple way to estimate the nearest neighbor distance is to choose a sphere in the data space such that an expected value of one data point is contained in it according to the current point density and to use the radius of this sphere as an approximation of the actual nearest neighbor distance. In the case of the maximum metric, we get the following formula:

$$\frac{1}{N} = V_q = (2r)^d \qquad r = \frac{1}{2\sqrt[d]{N}} .$$

For Euclidean metric, the corresponding formula is:

$$\frac{1}{N} = V_q = \frac{\sqrt{\pi^d}}{\Gamma(d/2+1)} \cdot r^d \qquad r = \sqrt[d]{\frac{\Gamma(d/2+1)}{N}} \cdot \frac{1}{\sqrt{\pi}} .$$

Unfortunately, this approach is not correct from the point of view of stochastics, because the operation of building an expectation is not invertible, i.e. the expectation of the radius cannot be determined from the expectation of the number of points in the corresponding sphere. The approximation determined by this formula is rather coarse and can be used only if a fast and simple evaluation is of higher importance than the accuracy of the model. The general problem is that even under uniformity and independence assumption the nearest neighbor distance yields a certain variance, when several range queries are executed.

4.2 Exact Estimation of the Nearest Neighbor Distance

A stochastically correct approach is to determine a distribution function for the nearest neighbor distance, and to derive an expectation of the nearest neighbor distance from the

corresponding probability density function. From this probability density function, the expectation of the number of page accesses can also be derived.

According to the rules defining a distribution function $P(r)$ for the nearest neighbor distance we must determine the probability, with which the actual nearest neighbor distance is smaller than the stochastic variable r. The nearest neighbor distance is *larger than r* if and only if no data point is contained in the sphere with radius r. The event 'distance is larger than r' is the opposite of the event needed in our distribution function. Therefore, $P(r)$ is as follows:

$$P(r) = 1 - (1 - V(r))^N.$$

Here, $1 - V(r)$ models the probability with which a point selected from a uniform distribution is not contained in the volume. Therefore, $(1 - V(r))^N$ is the probability with which all points are outside the volume. Subtracting this from 1 is the probability of at least one point being inside. Due to the convergence of the limit

$$\lim_{\nu \to \infty} \left(1 + \frac{k}{\nu}\right)^\nu = e^k$$

the distribution function can be approximated for a large number of objects N by the following formula:

$$P(r) \approx 1 - e^{-N \cdot V(r)}.$$

This approximation yields negligible relative errors for a large N (starting from 100) and will facilitate the evaluation later in this paper.

For maximum metric and Euclidean metric, the probability distribution function $P(r)$ can be evaluated in the following way:

$$P_{mm}(r) = 1 - (1 - (2r)^d)^N,$$

$$P_{em}(r) = 1 - \left(1 - \frac{\sqrt{\pi}^d}{\Gamma(d/2 + 1)} \cdot r^d\right)^N.$$

From the distribution function $P(r)$ a probability density function $p(r)$ can be derived by differentiation:

$$p(r) = \frac{\partial P(r)}{\partial r}.$$

For maximum and Euclidean metric, this evaluates to:

$$p_{mm}(r) = \frac{\partial P_{mm}(r)}{\partial r} = \frac{d \cdot N}{r} \cdot (1 - (2r)^d)^{N-1} \cdot (2r)^d,$$

$$p_{em}(r) = \frac{\partial P_{em}(r)}{\partial r} = \frac{d \cdot N}{r} \cdot \left(1 - \frac{\sqrt{\pi}^d}{\Gamma(d/2 + 1)} r^d\right)^{N-1} \cdot \frac{\sqrt{\pi}^d}{\Gamma(d/2 + 1)} r^d.$$

Figure 5 shows some probability density functions for 100,000 uniformly distributed data points in a two-dimensional and an 8-dimensional data space, respectively.

To determine the expected value of the nearest neighbor distance, the probability density function multiplied with r must be integrated from 0 to infinity:

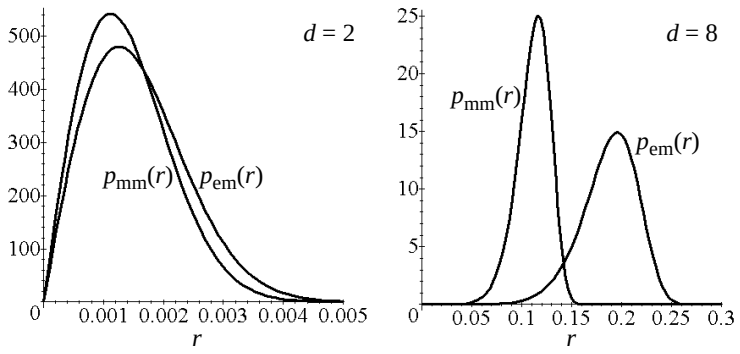

Figure 5: Probability Density Functions.

$$R = \int_0^\infty r \cdot p(r) \partial r,$$

$$R_{mm} = d \cdot N \cdot \int_0^\infty (1-(2r)^d)^{N-1} \cdot (2r)^d \partial r,$$

$$R_{em} = d \cdot N \cdot \int_0^\infty \left(\left(1 - \frac{\sqrt{\pi^d}}{\Gamma(d/2+1)} r^d\right)^{N-1} \cdot \frac{\sqrt{\pi^d}}{\Gamma(d/2+1)} r^d\right) \partial r \ .$$

The integration variable is denoted by '∂' instead of the more usual 'd' to avoid confusion with the identifier 'd' standing for the dimension of the data space. The integral is not easy to evaluate analytically.

4.3 Numerical Evaluation

We present two numerical methods to evaluate the integrals presented at the end of section 4.2 numerically. The first is based on the binomial theorem. Basically, the probability density function $p(r)$ is a polynomial of the degree $d \cdot N$. It can be transformed into the coefficient form $p(r) = a_0 r^{d \cdot N} + a_1 r^{d \cdot N - 1} + \ldots$ using the binomial theorem. We demonstrate this for the maximum metric:

$$(1-(2r)^d)^{N-1} = \sum_{0 \le i \le N-1} \binom{N-1}{i} \cdot (-1)^i \cdot (2r)^{d \cdot i};$$

$$p_{mm}(r) = \frac{d \cdot N}{r} \cdot (2r)^d \cdot \sum_{0 \le i \le N-1} \binom{N-1}{i} \cdot (-1)^i \cdot (2r)^{d \cdot i}.$$

This alternating series can be approximated with low error bounds by the first few summands ($0 \le i \le i_{max}$) if the absolute value of the summands is monotonically decreasing.

This is possible if the power of r decreases in a steeper way with increasing i than the binomial increases. This is guaranteed if the following condition holds:

$$\frac{N}{2} \le \frac{1}{(2r)^d} \Rightarrow r \le \frac{1}{2} \cdot \sqrt[d]{\frac{2}{N}} \Rightarrow \; p_{\mathrm{mm}}(r) \approx \frac{d \cdot N}{r} \cdot (2r)^d \cdot \sum_{0 \le i \le i_{\max}} \binom{N-1}{i} \cdot (-1)^i \cdot (2r)^{d \cdot i} \; .$$

Therefore, we approximate our formula for the expected nearest neighbor distance in the following way:

$$\begin{aligned} R_{\mathrm{mm}} \;&\approx\; d \cdot N \cdot \int_0^{\left(\frac{1}{2} \cdot \sqrt[d]{\frac{2}{N}}\right)} \left((2r)^d \cdot \sum_{0 \le i \le i_{\max}} \binom{N-1}{i} \cdot (-1)^i \cdot (2r)^{d \cdot i} \right) \partial r \\ &\approx\; d \cdot N \cdot \sum_{0 \le i \le i_{\max}} \binom{N-1}{i} \cdot (-1)^i \cdot \int_0^{\left(\frac{1}{2} \cdot \sqrt[d]{\frac{2}{N}}\right)} (2r)^{d \cdot (i+1)} \partial r \\ &\approx\; \sum_{0 \le i \le i_{\max}} \binom{N-1}{i} \cdot \frac{(-1)^i}{i + 1 + \frac{1}{d}} \cdot \left(\frac{2}{N}\right)^{i + \frac{1}{d}} . \end{aligned}$$

The same simplification can be applied for the Euclidean metric. However, an alternative way based on a histogram-approximation of the probability density function yields lower approximation errors and causes even a lower effort in the evaluation.

To facilitate numerical integration methods such as the middlebox approximation, the trapezoid approximation or the combined method according to Simpson's rule [Press *et al.* 1988], we must determine suitable boundaries, where the probability density function has values which are substantially greater than 0. If we consider for example figure 5, we observe that for d=8, $p_{\mathrm{mm}}(r)$ is very close to 0 in the two ranges $0 \le r \le 0.05$ and $0.16 \le r \le \infty$. Only the range between the lower bound $r_{\mathrm{lb}} = 0.05$ and the upper bound $r_{\mathrm{ub}} = 0.16$ contributes significantly. The criterion for a sensible choice of lower and upper bounds is based on the distribution function which corresponds to the area below the density function. We choose the lower bound r_{lb} such that the area in the ignored range $[0..r_{\mathrm{lb}}]$ corresponds to a probability less than 0.1% and do the same for the upper bound r_{ub}. We get the following two conditions, resulting from the approximation of the distribution function:

$$P(r) \approx 1 - \mathrm{e}^{-N \cdot V(r)} \ge 0.001 \qquad P(r) \approx 1 - \mathrm{e}^{-N \cdot V(r)} \le 0.999$$

$$r \ge r_{\mathrm{lb}} = V^{-1}\left(\frac{-\ln 0.999}{N}\right) \qquad r \le r_{\mathrm{ub}} = V^{-1}\left(\frac{-\ln 0.001}{N}\right) .$$

Integration can therefore be limited to the interval from r_{lb} to r_{ub}. The integral can be approximated by a sum of trapezoids or by a sum of rectangles:

$$R = \int_0^{\infty} r \cdot p(r) \partial r \approx \frac{r_{\mathrm{ub}} - r_{\mathrm{lb}}}{i_{\max}} \cdot \sum_{0 \le i < i_{\max}} \left(\frac{r_{\mathrm{ub}} - r_{\mathrm{lb}}}{i_{\max}} \cdot i + r_{\mathrm{lb}} \right) \cdot p\left(\frac{r_{\mathrm{ub}} - r_{\mathrm{lb}}}{i_{\max}} \cdot i + r_{\mathrm{lb}} \right) .$$

As we limit the integration to a small interval, a small number of rectangles or trapezoids is sufficient for a high accuracy. To achieve a relative error less than 1.0%, an approximation by $i_{\max} = 5$ rectangles was sufficient in our experiments.

4.4 K-Nearest Neighbor Query

The cost model developed in the previous sections can also be extended for estimating *k*-nearest neighbor queries. For the coarse model, this is straightforward since the volume must be chosen such that *k* objects are contained rather than one. Therefore, the term $1/N$ must be replaced by k/N. For maximum metric, the estimated distance is:

$$R_{\mathrm{mm}}(k) \approx \frac{1}{2} \cdot \sqrt[d]{\frac{k}{N}}.$$

For Euclidean metric, the result is analogous:

$$R_{\mathrm{em}}(k) \approx \sqrt[d]{\frac{k \cdot \Gamma(d/2+1)}{N}} \cdot \frac{1}{\sqrt{\pi}}.$$

For the exact model, the probability distribution must be modeled as a summation of Bernoulli-chains with lengths ranging from 1 to *k*. The probability that at least *k* points are inside the volume $V(r)$ corresponds to the following formula:

$$P_k(r) \;=\; 1 - \sum_{0 \le i < k} \binom{n}{i} \cdot V(r)^i \cdot (1 - V(r))^{n-i}.$$

For $k = 1$, the formula corresponds to the distribution function $P(r)$ for nearest neighbor queries. The probability density function and the expectation of the nearest neighbor distance are determined in the same way as in section 4.2.

We should note that the peak in the probability density function of a *k*-nearest neighbor query becomes steeper with increasing *k* (decreasing variance). Therefore, the approximation by the coarse model which is bad for high, asymmetric variances, becomes better with increasing *k*. For sufficiently large $k > 10$ the coarse model and the exact model yield comparable accuracy.

4.5 Expectation of the Number of Page Accesses

As initially mentioned, the number of page accesses of a nearest neighbor query is equivalent to the number of page accesses of a range query when the nearest neighbor distance is used for the query range. An obvious approach to modeling is therefore to use the expectation of the nearest neighbor distance and to insert it into the expectation of the number of page accesses using range queries:

$$A_{\mathrm{nn}} \approx A_{\mathrm{r}}(R).$$

However, this approach reveals similar statistical problems and leads to similar inaccuracies as the coarse estimation approach in section 4.1. The problem is that the number of page accesses is not linear in the query range. Once again, the approach can be taken if high accuracy is not required or if the variance of the nearest neighbor distance is low.

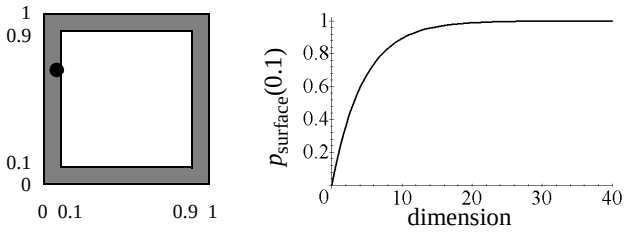

Figure 6: Probability that a Point is Near by the Data Space Boundary.

Instead, we have once again to apply the distribution function $P(r)$ to determine an expectation of the number of page accesses by integration as follows:

$$A_{\text{nn}} = \int_0^\infty A_{\text{range}}(r) \cdot p(r) \partial r \,.$$

For maximum and Euclidean metric, this formula evaluates to:

$$A_{\text{nn,mm}} = \int_0^\infty \left(2r \cdot \sqrt[d]{\frac{N}{C_{\text{eff}}}} + 1 - \frac{1}{C_{\text{eff}}} \right)^d \cdot \frac{d \cdot N}{r} \cdot (1 - (2r)^d)^{N-1} \cdot (2r)^d \partial r \,,$$

$$A_{\text{nn,em}} = \int_0^\infty \frac{N}{C_{\text{eff}}} \cdot \sum_{0 \le k \le d} \binom{d}{k} \cdot \left(\left(1 - \frac{1}{C_{\text{eff}}} \right) \cdot \sqrt[d]{\frac{C_{\text{eff}}}{N}} \right)^k \cdot \frac{\sqrt{\pi^{d-k}}}{\Gamma\left(\frac{d-k}{2} + 1 \right)} \cdot r^{d-k}$$

$$\cdot \frac{d \cdot N}{r} \cdot \left(1 - \frac{\sqrt{\pi^d}}{\Gamma(d/2+1)} r^d \right)^{N-1} \cdot \frac{\sqrt{\pi^d}}{\Gamma(d/2+1)} r^d \partial r \,.$$

This result can be simplified by a similar technique as in section 4.3.

5. EFFECTS IN HIGH-DIMENSIONAL DATA SPACES

In this section, we describe some effects occurring in high-dimensional data space which are not sufficiently considered in our models of the previous sections. We still assume a uniform and independent distribution of data and query points in this section. The models developed in the previous sections will be modified to take the described effects into account.

5.1 Problems specific to High-Dimensional Data Spaces

The first effect occurring especially in high-dimensional data spaces is that all data and query points are likely to be near by the boundary of the data space. The probability that a point randomly taken from a uniform and independent distribution in a *d*-dimensional data

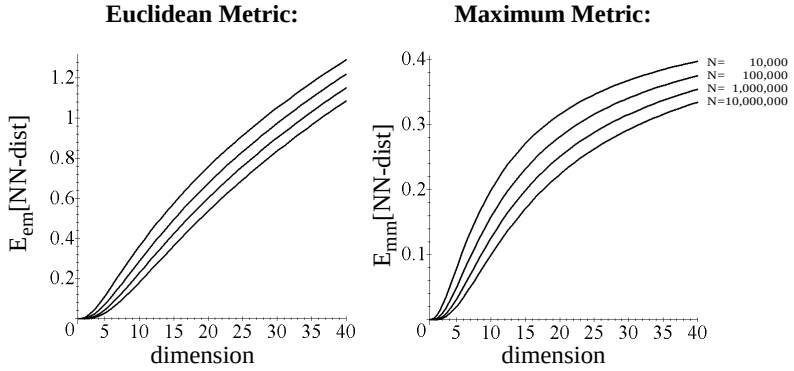

Figure 7: Expected Nearest Neighbor Distance with Varying Dimension.

space has a distance of r or less to the space boundary can be determined by the following formula:

$$P_{\text{surface}}(r) = 1 - (1 - 2 \cdot r)^d.$$

As figure 6 shows, the probability that a point is inside a 10% border of the data space boundary increases rapidly with increasing dimension. It reaches 97% for a 16-dimensional data space.

A second effect which is even more important, is the large extension of query regions. If we use our model for determining an expected value of the nearest neighbor distance, we observe that the expectation fast approaches surprisingly high values. Figure 7 shows the expected values for the nearest neighbor distance with varying dimension for the maximum metric and the Euclidean metric for several databases containing between 10,000 and 10,000,000 points. In particular, when applying the Euclidean metric in a data space of a dimension between 13 and 19, the nearest neighbor distance reaches a value of 0.5, i.e. the nearest neighbor sphere has the same diameter as the complete data space. The size of the database has a minor influence on this effect.

The combination of the two effects described above leads us to the observation that large parts of a typical nearest neighbor sphere must be outside the boundary of the data space (cf. figure 8). The consequences arising from this fact are commonly referred to as *boundary effects*. As we will investigate in depth in the subsequent sections, the most important con-

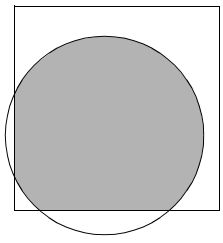

Figure 8: Parts of the Query Sphere are out of the data space boundary.

sequence is that in our models all volume determinations must consider clipping at the boundary of the data space. On the one hand, the expectation of the nearest neighbor distance increases by boundary effects, but on the other hand, access probabilities of data pages decrease because large parts of the Minkowski sum are clipped away.

If the dimension further increases, the typical nearest neighbor distance grows to values much greater than 1/2. In this case, it becomes very likely that the nearest neighbor sphere exceeds most of the data space boundary areas.

A similar effect is observable for the page regions. If we assume, following our initial model, hypercube shaped page regions, the side length of such a region quickly exceeds 0.5. However, it is impossible that the data space is covered only with pages having side lengths between 0.5 and 1. Basically, the pagination arises from a recursive decomposition of the data space into parts of approximately the same volume (by uniformity and independence assumption). Therefore, each page is split several times in each dimension. That means, only the side lengths 1, 1/2, 1/4, 1/8,... (approximately) can occur. In sufficiently high dimensions, the page regions do not have side lengths of 1/2 or smaller in all dimensions, because if every data page is split at least once in each dimension, we need a total number of at least 2^d data pages to cover the complete data space. For example, in a 30-dimensional data space, we would need one billion data pages to reach such a pagination, resulting in a database size of 4,000 GBytes (assuming a page size of 4KBytes).

Therefore, we will modify our cost models such that for database sizes N of less than $N_{\text{Singlesplit}}$ objects with

$$N \leq N_{\text{Singlesplit}} = C_{\text{eff}} \cdot 2^d$$

this effect is considered.

5.2 Range Query

We still assume uniformity and independence in the distribution of data and query points. For sufficiently high dimension d (such that the inequality above holds), we observe that the data space is only split in a number $d' < d$ of dimensions. The maximum split dimension d' can be determined by using the following formula:

$$d' = \left\lceil \log_2\left(\frac{N}{C_{\text{eff}}}\right) \right\rceil .$$

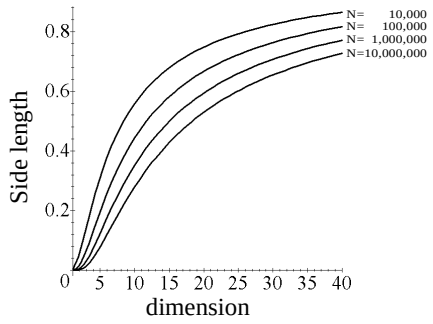

Figure 9: Side Lengths of Page Regions for C_{eff}=30.

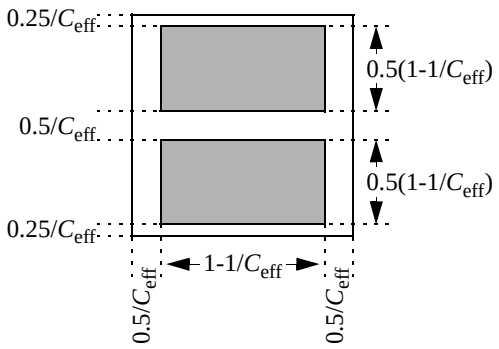

Figure 10: Side Lengths and Positions of Page Regions in the Modified Model.

The data-pages have an average extension a_{split} with

$$a_{\text{split}} = 0.5 \cdot \left(1 - \frac{1}{C_{\text{eff}}}\right)$$

in d' dimensions and an average extension a_{unsplit} with

$$a_{\text{unsplit}} = 1 - \frac{1}{C_{\text{eff}}}$$

in the remaining $d' - d$ dimensions. Figure 10 clarifies the position of two typical page regions in the data space for split (y-axis) and unsplit (x-axis) dimensions. The projection on an axis of a split dimension shows 2 page regions. Between these two regions, there is a gap of the average width $0.5/C_{\text{eff}}$ which is caused by the property of the page region to be a *minimum* bounding rectangle (called the *MBR property,* cf. section 3.2). The distance of $0.25/C_{\text{eff}}$ from the data space boundary is also due to the MBR property. In contrast, the projection on an axis of an unsplit dimension shows only one page region with a distance of $0.5/C_{\text{eff}}$ from the lower and from the upper space boundary, respectively.

Now, we mark the Minkowski sum of the lower page region (cf. figure 11a). We observe that large parts of the Minkowski sum are located outside the data space. The Minkowski sum is the volume, from which a query point has to be taken such that the corresponding page is accessed. However, we assume that only query points inside the data space boundary are used as query points. Therefore, the Minkowski sum has to be clipped at the data space boundary in order to determine the probability that a randomly selected query point accesses the page (cf. figure 11b). We can express the clipping on the boundary with the following integral formula which summarizes all points v in the data space (i.e. all possible positions of query points) with a distance less or equal to the query range r from the page region R:

$$X_{\text{r,hd,ui}}(r) = V_{(R \oplus S) \cap \text{DS}}(r) = \int_0^1 \cdots \int_0^1 \left(\begin{cases} 1 & \text{if } \delta_M(v, R) \le r \\ 0 & \text{otherwise} \end{cases} \right) \partial v_0 \ldots \partial v_{d-1} .$$

Here, the distance function $\delta_M(v, R)$ denotes the smallest distance between the point v and the page region R according to the metric M (Euclidean metric or maximum metric). Unfor-

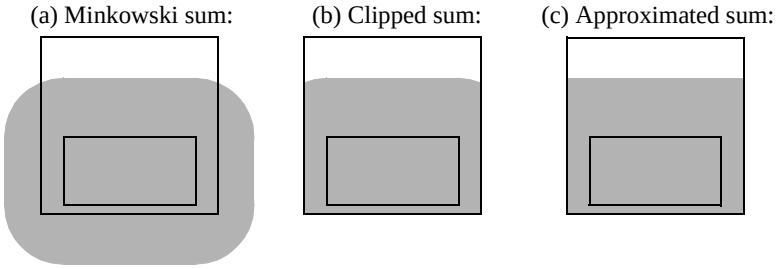

Figure 11: Minkowski Sum Outside the Boundary of the Data Space.

tunately, this integral is difficult to evaluate. Therefore, it has to be simplified. The first observation usable for this purpose is that the distance between the data space boundary and the page region ($0.5/C_{\text{eff}}$ for unsplit dimensions, $0.25/C_{\text{eff}}$ for split dimensions) is small compared to a typical radius r (at least if there should be a realistic chance that at points are retrieved). Therefore, the corresponding gap is filled up completely by the Minkowski enlargement (the projection of the clipped Minkowski sum to an unsplit dimension is 1). Unsplit dimensions can be ignored for the determination of the access probability (cf. figure 11c).

For maximum metric, the clipped Minkowski sum can be determined in the following way (cf. figure 12a): We take the side length of the page region in all halved dimensions. The gap between the region and the data space boundary must not be considered, because it is filled by the Minkowski sum. We add the query radius only one time instead of two times (as we did in our initial model), because only the Minkowski enlargement in the middle of the data spaces applies (the rest is clipped). Here, we have also to consider the MBR effect. The result is taken to the power of the number of dimensions which are split:

$$X_{\text{r,mm,hd,ui}}(r) = \left(\min\left\{0.5 - \frac{0.25}{C_{\text{eff}}} + r,\ 1\right\}\right)^{d'} = \left(\min\left\{0.5 - \frac{0.25}{C_{\text{eff}}} + r,\ 1\right\}\right)^{\left\lceil \log_2\left(\frac{N}{C_{\text{eff}}}\right)\right\rceil}.$$

For a radius greater than $0.5 + 0.25 / C_{\text{eff}}$, the Minkowski enlargement also reaches the data space boundary on the opposite side. This is taken into account by the application of the minimum function in the equation above. In this case, the page has an access probability of 100%.

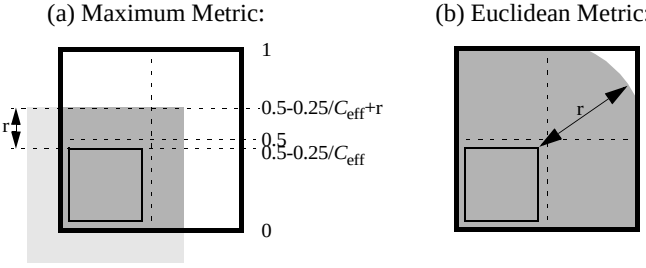

Figure 12: The Modified Minkowski Sum for the Maximum and Euclidean Metric.

If we apply the Euclidean metric, an additional complication arises as depicted on figure 12b. Since the radius r is typically much greater than 0.5 (cf. section 5.1 and figure 7), the sphere itself must be clipped. The volume of a clipped hypersphere cannot be determined analytically. However, we will show, how this volume can be simplified such that a precomputed volume function can be applied to improve the efficiency of the evaluation. The volume function can be precomputed numerically.

The volume of the clipped sphere depends on the dimension, the radius and the size of the clipping window. This means that we have to store the precomputed values in a 3-dimensional array. The dependency of the size of the clipping window can be eliminated by suitable scaling operations. We scale our clipping region (and also r) such that it is mapped to the unit hypercube. Then, we determine the corresponding volume by looking up in the table of precomputed volumes. After that, we apply the inverse scaling to the volume in the table. Since our clipping window is now the unit hypercube, we need only a 2-dimensional array for storing the precomputed values.

By $V_{csi}(d,r)$ we define the volume of the intersection of a sphere with radius r and the unit hypercube in d-dimensional space. Figure 13 depicts the intersection volume $V_{csi}(2, r)$ in 2-dimensional data space. We let the origin be the center of the sphere. Obviously, $V_{csi}(d,0) = 0$ and $V_{csi}(d, \sqrt[2]{d}) = 1$. Between these points, V_{csi} is monotonically increasing.

Definition 1: Cube-Sphere Intersection

$V_{csi}(d,r)$ denotes the intersection volume between the unit hypercube $[0..1]^d$ and a d-dimensional sphere with radius r around the origin:

$$V_{csi}(d, r) = \int_0^1 \dots \int_0^1 \left(\begin{cases} 1 & \text{if } |(v_0 \dots v_{d-1})| \le r \\ 0 & \text{otherwise} \end{cases} \right) \partial v_0 \dots \partial v_{d-1}.$$

$V_{csi}(d,r)$ can be materialized into an array of all relevant dimensions d (e.g. ranging from 1 to 100) and for a discretization of the relevant r between 0 and $\sqrt{d}$ in a sufficiently high number of steps (e.g. 10,000). For the determination of the discretization of $V_{csi}(d,r)$, the Montecarlo integration [Kalos and Whitlock 1986] using the integral formula can be used. Sufficient accuracy is achievable with 100,000 points.
Figure 14 depicts $V_{csi}(d,r)$ for various dimensions d.

$V_{csi}(d,r)$ is used to determine the access probability of a page when a range query using the Euclidean metric is performed. As we pointed out in the previous discussion, the range query behaves like a range query in d'-dimensional space, because all dimensions, where the

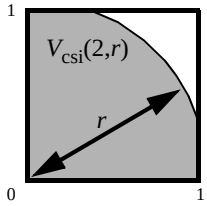

Figure 13: The Volume of the Intersection between Sphere and the Unit Hypercube.

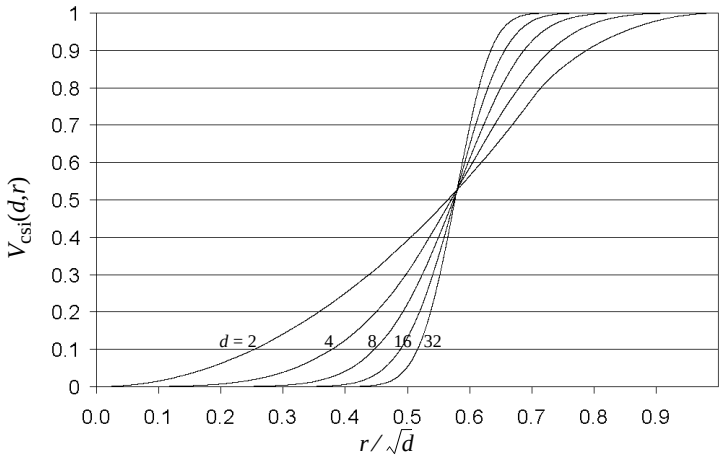

Figure 14: The Volume of the Intersection between a Sphere and the Unit Hypercube.

page region has full extension, can be projected out without affecting the volume of the Minkowski enlargement.

The query sphere, however, is not clipped by the unit hypercube, but rather by the hypercube representing the empty space between the page region and the opposite boundary of the data space. The side length of this cube a_{empty} is (cf. figures 11-12):

$$a_{\text{empty}} = \frac{1}{2} + \frac{1}{4C_{\text{eff}}}.$$

To determine clipping on such a hypercube, we have to scale the radius accordingly before applying $V_{\text{csi}}(d,r)$:

$$V_{\text{csi}}(d, r, a) = a^d \cdot V_{\text{csi}}(d, \frac{r}{a}).$$

The resulting formula for the access probability using Euclidean range queries is:

$$X_{\text{r,em,hd,ui}}(d', r) = \sum_{0 \le k \le d'} \binom{d'}{k} \cdot \left(\frac{1}{2} - \frac{1}{4C_{\text{eff}}}\right)^{d'-k} \cdot \left(\frac{1}{2} + \frac{1}{4C_{\text{eff}}}\right)^{k} \cdot V_{\text{csi}}(k, \frac{r}{\frac{1}{2} + \frac{1}{4C_{\text{eff}}}}) \quad .$$

Here, we again sum up over all hypercylinders with k round dimensions and $d' - k$ rectangularly shaped dimensions. The rectangular part has a side length of $(1/2 - 1/(4C_{\text{eff}}))^{d'-k}$. The volume of the round part is determined by the volume function $V_{\text{csi}}(d,r)$. The radius must be scaled before applying V_{csi} to take into account the normalized clipping window. Last, the obtained volume must be brought back to the original scale by multiplying it with $(1/2 + 1/(4C_{\text{eff}}))^k$.

To show the impact of the effects in high-dimensional spaces on the estimation of the access probability, figure 15 compares our new function $X_{\text{r,em,hd,ui}}(r)$ with the low-dimensional estimation $X_{\text{r,em,ld,ui}}(r)$ and with an intermediate form, where the volume function V_{csi} is replaced by the unclipped sphere volume V_{s}. In the intermediate form only hyper-

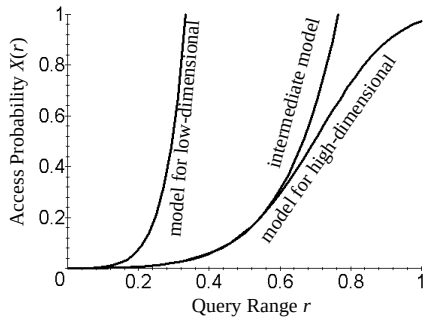

Figure 15: Various Models in High-Dimensional Data Spaces.

sphere segments completely lying outside the data space are clipped. The database in this experiment contains 280,000 points in a 16-dimensional data space. Whereas the cost model for low-dimensional query processing quickly yields unrealistic access probabilities that are larger than 100%, the intermediate model is accurate for ranges less than $r = 0.6$. The intermediate model avoids the precomputed discretization of the volume function V_{csi}.

In [Berchtold *et al.* 1997b], we have given the cost formula for the special case that the number of data pages N / C_{eff} is a power of two. In this case, the number of split dimensions is equal for all data pages (although the dimensions on which the data pages are actually split, may vary). If the number of data pages is not a power of 2, a number of data pages must be split once more than the rest. As the number of all data pages is N/C_{eff}, the number of data pages split once more than the others $n_{d'}$ is equal to:

$$n_{d'} = 2 \cdot \left(\frac{N}{C_{\text{eff}}} - 2^{\left\lfloor \log_2(\frac{N}{C_{\text{eff}}}) \right\rfloor} \right).$$

Likewise, the number of data pages split one time fewer $n_{d'\text{-}1}$ is equal to:

$$n_{d'-1} = N/C_{\text{eff}} - n_{d'}.$$

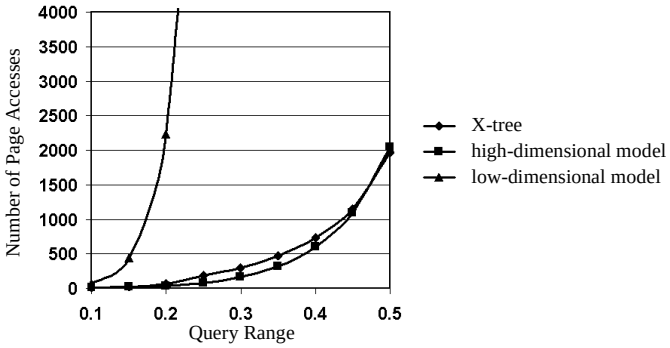

Figure 16: Accuracy of the Models in a 16-dimensional Data Space.

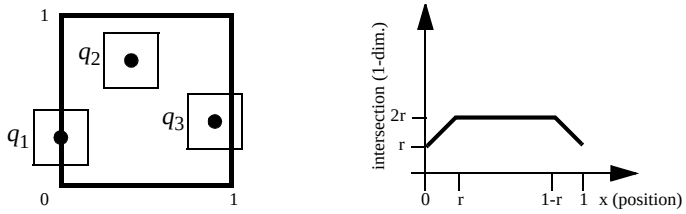

Figure 17: The Intersection Volume for Maximum Metric and Arbitrary Center Point.

Then, the expected number of page accesses is equal to:

$$A_{\mathrm{hd}}(r) \;=\; n_{d'} \cdot X_{\mathrm{hd}}\left(\left\lceil \log_2\left(\frac{N}{C_{\mathrm{eff}}}\right)\right\rceil, r\right) + n_{d'-1} \cdot X_{\mathrm{hd}}\left(\left\lfloor \log_2\left(\frac{N}{C_{\mathrm{eff}}}\right)\right\rfloor, r\right).$$

This equation holds for maximum metric as well as Euclidean metric and for range queries as well as nearest neighbor queries.

The accuracy of the low-dimensional and the high-dimensional cost models for range query processing was compared by using a database of 100,000 points taken from a uniform, independent data distribution in the 16-dimensional data space. The query range was varied from 0.1 to 0.5 using the maximum metric, yielding selectivities between $6.5 \cdot 10^{-12}$ and 11.8%. The results are depicted in figure 16. As expected, the high-dimensional model yields a reasonable accuracy, whereas the low-dimensional model completely fails in this case.

5.3 Nearest Neighbor Query

Typically, query spheres exceed the data space boundary in high-dimensional query processing. For range queries, the consequence is a smaller result set compared with the expectation when neglecting this boundary effect, because only the part of the sphere inside the data space contributes to the result set. In contrast, nearest neighbor queries have a fixed result set size (1 point for a 1-nearest neighbor query). The consequence here is that a greater radius is needed to achieve the same result set size in the presence of boundary effects.

First, we develop an expectation for the volume $V_{\mathrm{csi,a}}(d,r)$ of the intersection volume of the unit hypercube and a sphere with radius r, whose center is arbitrarily chosen in the unit hypercube. We note that this task is similar to the intersection volume $V_{\mathrm{csi}}(d,r)$ in the previous subsection. However, the center of the sphere is now arbitrarily chosen and not fixed in the origin. $V_{\mathrm{csi,a}}(d,r)$ corresponds to the probability that two points arbitrarily chosen from the unit hypercube have a distance less or equal to r from each other.

When the maximum metric is used for the query, the expectation for the intersection volume, which is an intersection of two hypercubes, can be determined analytically. Figure 17 depicts three different positions of queries in the data space. First, we consider only the projection on the x-axis. The center point of q_1 lies exactly on the lower space boundary. Therefore, only half of the side length (r) is inside the data space. The center point of q_2 has a distance greater than r from the data space boundary. Therefore, the complete side length of the cube ($2r$) is inside the data space. Query q_3 intersects the right space boundary, but more than half of the side length is inside the data space. The right diagram of figure 17 depicts the progression of the part of the side length which is inside the data space with varying position of the query point. The side length is r at the points 0 and 1, $2r$ between the

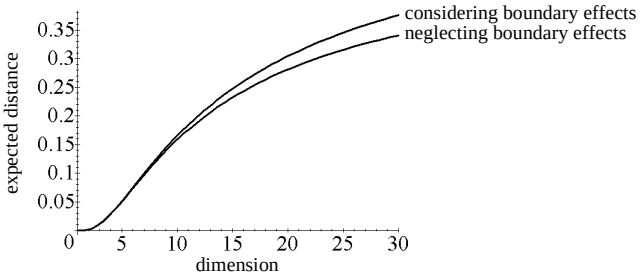

Figure 18: The Impact of Boundary Effects on the Nearest Neighbor Distance.

positions r and $1 - r$. Between 0 and r, the intersection increases linearly. The average of the intersection over all positions is:

$$V_{\text{cci,a}}(1, r) = 2r - r^2.$$

We can extend this result to the d-dimensional case simply by taking the power of d:

$$V_{\text{cci,a}}(d, r) = V_{\text{cci,a}}(1, r)^d = (2r - r^2)^d.$$

This result can be used to determine the expectation of the nearest neighbor distance. A completely analytical solution is possible if we apply our coarse estimation by equalizing $V_{\text{cci,a}}(d,r)$ with $1/N$:

$$1/N = V_{\text{cci,a}}(d, r) = (2r - r^2)^d \qquad r = 1 - \sqrt{1 - \sqrt[d]{1/N}}.$$

The impact of boundary effects on the nearest neighbor distance is shown in figure 18. As expected, boundary effects do not play any role in low dimensions up to 10. With increasing dimension, the effect becomes slightly more important. Neglecting boundary effects, we underestimate the nearest neighbor distance by 10% in 30-dimensional space. This error, however, is also almost negligible. We will see later in this section that it is not the estimation of the nearest neighbor distance that causes problems but the estimation of the Minkowski sum.

The new volume determination $V_{\text{cci,a}}(d,r)$ can also be applied in our exact model for nearest neighbor estimation. The corresponding probability distribution is in this case:

$$P_{\text{mm,hd}}(r) = 1 - (1 - V_{\text{cci,a}}(d, r))^N = 1 - (1 - (2 \cdot r - r^2)^d)^N.$$

The probability density $p_{\text{mm,hd}}(r)$, the expectation for the nearest neighbor distance $R_{\text{mm,hd}}$, and the expectation of the number of page accesses $A_{\text{nn,mm,hd}}$ can be derived from the probability distribution as described in section 4.

When the Euclidean metric is applied, the same problem arises as in section 5.2. It is difficult to determine the intersection volume between the unit hypercube and a sphere with arbitrary center in an analytical way. To cope with this problem, a similar precomputation of the volume may be used. Again, we define the *Cube-Sphere Intersection with Arbitrary Center*, $V_{\text{csi,a}}(d,r)$ by a multidimensional integral which can be evaluated by using the Montecarlo integration [Kalos and Whitlock 1986]. The result can be stored in an array for use by the model.

Definition 2: Cube-Sphere Intersection with Arbitrary Center

$V_{csi,a}(d,r)$ denotes the intersection volume between the unit hypercube $[0..1]^d$ and a d-dimensional sphere with radius r around a point arbitrarily chosen from the unit hypercube:

$$V_{csi,a}(d, r) = \int_{(0,\ldots,0)}^{(1,\ldots,1)} \left(\int_{(0,\ldots,0)}^{(1,\ldots,1)} \left(\begin{cases} 1 & \text{if } |v - w| \le r \\ 0 & \text{otherwise} \end{cases} \right) \partial v \right) \partial w \ .$$

Figure 19 shows $V_{csi,a}(d,r)$ for the dimensions 2, 4, 8, 16 and 32. The intersection volume was determined for all dimensions between 1 and 100 for each radius between 0 and $\sqrt{d}$ in 10,000 intervals using 100,000 steps of the Montecarlo integration [Kalos and Whitlock 1986]. The 10,000 intervals can be used for an efficient numerical evaluation of the expectation. Taking boundary effects into consideration, the probability distribution of the nearest neighbor distance r for the Euclidean metric is:

$$P_{em,hd}(r) = 1 - (1 - V_{csi,a}(d, r))^N .$$

The corresponding density function is:

$$p_{em,hd}(r) = \frac{\partial P_{em,hd}(r)}{\partial r} = N \cdot (1 - V_{csi,a}(d, r))^{N-1} \cdot \frac{\partial V_{csi,a}(d, r)}{\partial r} .$$

Assuming that $V_{csi,a}[d,i]$ is an array with the range $[1..d_{max},0..i_{max}]$ which contains the precomputed values of $V_{csi,a}(d,r)$ for r ranging from 0 to $\sqrt{d}$ with

$$i = \left\lfloor \frac{r \cdot i_{max}}{\sqrt{d}} \right\rfloor ,$$

we are able to replace integral formulas such as the expected value of the nearest neighbor distance by a finite summation:

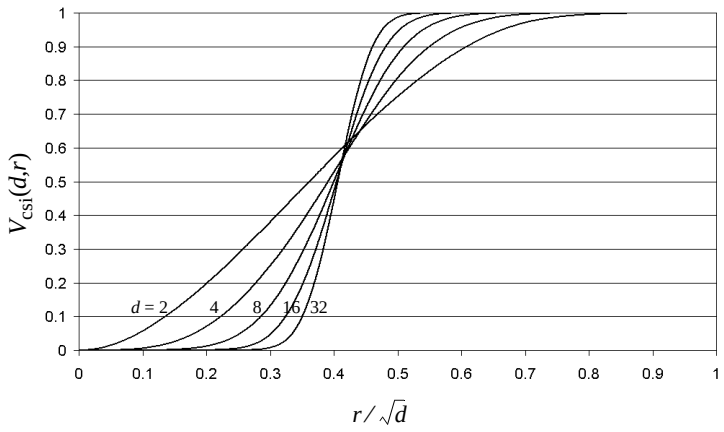

Figure 19: The Intersection Volume for Euclidean Metric and Arbitrary Center Point.

$$R_{\text{em,hd}} = \int_0^{\infty} r \cdot p_{\text{em,hd}}(r)\partial r = N \cdot \int_0^{\sqrt{d}} r \cdot (1 - V_{\text{csi,a}}(d, r))^{N-1} \cdot \frac{\partial V_{\text{csi,a}}(d, r)}{\partial r}\partial r$$

$$\approx \frac{N \cdot \sqrt{d}}{i_{\max}} \cdot \sum_{i=1}^{i_{\max}} i \cdot (1 - V_{\text{csi,a}}[d, i])^{N-1} \cdot (V_{\text{csi,a}}[d, i] - V_{\text{csi,a}}[d, i-1]) .$$

The infinite upper bound of the integral can be replaced by $\sqrt{d}$, because the derivative of $V_{\text{csi,a}}(d,r)$ is constantly 0 for r larger than $\sqrt{d}$, while all other terms have finite values. The derivative is replaced by the local difference. The expected value of the number of page accesses can be determined in the same way:

$$A_{\text{nn,em,hd}} = \int_0^{\infty} A_{\text{r,em,hd}}(r) \cdot p_{\text{em,hd}}(r)\partial r$$

$$= N \cdot \sum_{i=1}^{i_{\max}} A_{\text{r,em,hd}}\left(\frac{i\sqrt{d}}{i_{\max}}\right) \cdot (1 - V_{\text{csi,a}}[d, i])^{N-1} (V_{\text{csi,a}}[d, i] - V_{\text{csi,a}}[d, i-1]) .$$

The evaluation of these formulas is not costly, because the required volume functions $V_{\text{csi,a}}(d,r)$ and $V_{\text{csi}}(d,r)$ are independent of any database specific setting such as the number of points in the database, the point density or the effective page capacity C_{eff}. The predetermined discretization of these functions requires a few megabytes of storage and can be statically linked with the programs for evaluating cost models. Costly Montecarlo integration processes are run only at compile time, not at run-time. Further improvement is achievable if we consider that the probability density only contributes in the interval between r_{lb} and r_{ub} (cf. section 4.3). Integration and summation can be bounded to this area:

$$R_{\text{em,hd}} \approx \frac{N \cdot \sqrt{d}}{i_{\max}} \cdot \sum_{i=\left\lfloor \frac{r_{\text{lb}} \cdot i_{\max}}{\sqrt{d}} \right\rfloor}^{\left\lceil \frac{r_{\text{ub}} \cdot i_{\max}}{\sqrt{d}} \right\rceil} i \cdot (1 - V_{\text{csi,a}}[d, i])^{N-1} \cdot (V_{\text{csi,a}}[d, i] - V_{\text{csi,a}}[d, i-1]) .$$

To evaluate the cost formula for query processing using nearest neighbor queries, we constructed indexes with varying data space dimensionality. All databases contained 100,000 points taken from a uniform and independent distribution. The effective capacity of the data pages was 48.8 in all experiments (the block-size was chosen correspondingly). The dimension varied from 4 to 20. We performed nearest neighbor queries using maximum metric and Euclidean metric on all these indexes and compared the observed page accesses with the predictions of the low-dimensional and the high-dimensional model developed in this paper. The results are depicted in figure 20. The diagram on the left side shows the results for maximum metric, the right diagram shows the results for Euclidean Metric. Whereas the cost model for high-dimensional query processing provides accurate estimations over all dimensions, the low-dimensional model is only accurate in the low-dimensional area up to $d = 6$. Beyond this area, the low-dimensional model completely fails to predict the number of page accesses. Not even the order of magnitude is correctly revealed

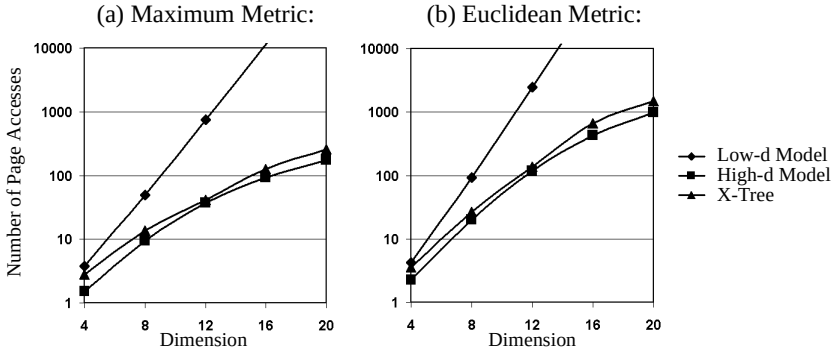

Figure 20: Accuracy of the Cost Models for Nearest Neighbor Queries.

by the low-dimensional model. We should note that the low-dimensional model is mainly related to the original model of Friedman, Bentley and Finkel [Friedman *et al.* 1977] and the extension of Cleary [Cleary 1979].

6. DATA SETS FROM REAL-WORLD-APPLICATIONS

It has been pointed out that data sets from real applications consistently violate the assumptions of uniformity and independence [Faloutsos and Kamel 1994, Belussi and Faloutsos 1995]. In this section, we describe the effects and adapt our models to take non-uniformity and correlation into account.

6.1 Independent Non-Uniformity

It was already proven in the well-known cost model by Friedman, Bentley and Finkel [Friedman *et al.* 1977] that non-uniformity has no influence on the cost of nearest neighbor query processing if no correlation occurs and if the data distribution is smooth. Smoothness means in this context that the point density does not vary severely inside the Minkowski enlargement of a page region. The intuitive reason is the following: Query points are assumed to be taken from the same distribution as data points. For the access probability of a page, we have to determine the fraction of query points which are inside the Minkowski enlargement of the page. If the point density is above the average in some region (say by a factor c) due to non-uniformity, then both, the average volume of the page regions and the average volume of the query regions are scaled by the factor $1/c$. This means that the Minkowski sum is scaled by $1/c$. However, the number of points inside a given volume is by a factor of c higher than in the uniform case. Therefore, the number of points in the Minkowski enlargement is the same as in the uniform case. For smooth distributions for which the independence assumption holds, the performance of the evaluation of nearest neighbor queries is the same as for uniform distributions. In contrast, range queries are difficult to model in the case of non-uniformity, because in the same way as the point density changes with varying location, the size of the result set and the number of page accesses will change. Therefore, range queries will be subject to future work. Likewise, cost modeling in the case that the data distribution is not smooth is an open question.

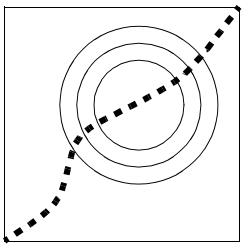

Figure 21: Correlations and their Problems.

6.2 Correlation

For real data the assumption of independent non-uniform data distribution is as unrealistic as the assumption of independent uniform distribution. One of the most important properties of real data is correlation.

If two dimensions are correlated, the value of the one attribute functionally depends on the value of the other attribute. Either it can be directly determined from the other attribute, or there are only a small number of possible values (or a small interval) that the dependent attribute can take on. In contrast to the stochastical definition of correlation taking only linear dependencies into account, we also consider non-linear correlations. The geometrical meaning of a correlation is the following: The *d*-dimensional space is not completely covered with data points. Instead, all points are clustered on a lower-dimensional area which is embedded in the data space. An example is shown in figure 21, where all data points are located on a 1-dimensional line which is embedded in the 2-dimensional data space. As depicted, the line is not necessarily a straight line. It is also possible that there are several lines which are not connected, or that the data points are located in a cloud around the line.

A concept which is often used to handle correlation is the singular value decomposition (*SVD*) [Duda and Hart 1973, Fukunaga 1990, Golup and van Loan 1989] or the principal component analysis (*PCA*) [Faloutsos and Lin 1995, Press *et al.* 1988, Strang 1980]. These techniques transform the data points directly into a lower-dimensional data space by rotation operations and eliminate the correlation in this way. The point set is indexed in lower-dimensional space, and query processing can be modeled by using our techniques presented in sections 3 - 6.

However, SVD and PCA can only detect and eliminate linear correlations. A linear correlation means a single straight line in our example. If there are, for example, two warped lines on which data points are located, SVD and PCA will completely fail. We will show later that the performance of query processing improves even without applying explicit transformations to lower-dimensional data spaces if the intrinsic dimension of the point set is lower than the dimension of the data space.

The general problem of correlations can also be observed in figure 21. If we consider circles of varying size around some data point, we observe that the number of enclosed points is not proportional to the area of the circle as we would expect. Because the intrinsic dimension of the data set is 1, the number of points enclosed in a circle with radius r is proportional to the radius r. The same observation is valid if we consider cubes or some other d-dimensional objects which are uniformly scaled.

This provides us with a means to define the intrinsic dimension of the data set. Under uniformity and independence assumptions the number of points enclosed in a hypercube with side length s is proportional to the volume of the hypercube:

$$n_{\text{encl}} = \rho \cdot s^d = \rho \cdot V,$$

where $\rho = N / V_{DS}$ is called the *point density*. Real data sets obey a similar power law using the fractal dimension D_F of the data set:

$$n_{\text{encl}} = \rho_F \cdot s^{D_F} = \rho_F \cdot V^{D_F/d}$$

where ρ_F is the fractal analogue to the point density ρ. The power law was used by Faloutsos and Kamel [Faloutsos and Kamel 1994] for the 'box counting' fractal dimension. We use this formula directly for the definition of the fractal dimension:

Definition 3: Fractal Dimension

The fractal dimension D_F of a point set is the exponent for which the following power law holds, provided that the center of the volume is located in the populated part of the data space:

$$n_{\text{encl}} = \rho_F \cdot V^{D_F/d}.$$

The fractal dimension is often not constant over all scales. It is possible that the fractal dimension changes, depending on the size of the volume V. In practice, the fractal dimension is often constant over the wide range of the relevant scales. It is also possible that the fractal dimension is not location-invariant, i.e. a subset of the data set forms a different fractal dimension than the rest of the data set. Intuitively, a reason for this behavior can be that our database contains different kinds of objects (e.g. oil paintings and photos in an image database). The restriction to the populated part of the data space in definition 3 forces us to the assumption that the distribution of the query points must follow the distribution of the data points. In many applications, this assumption seems intuitively reasonable, for instance if in an image database, images of the same type are the query objects. If this assumption does not hold, the performance of query processing is affected in a way that our current model is not able to predict. This is subject to future work.

6.3 Model Dependence on the Fractal Dimension

The first consequence of the existence of the fractal dimension D_F is that the choice of the model to use (high-dimensional or low-dimensional) is dependent on D_F rather than on the embedding dimension d. If D_F is small, then most data points and most queries are far away from the data space boundary. Therefore, we need not to consider clipping in our model. For this case, we must adapt the cost model for low-dimensional data spaces (cf. sections 3-4). In contrast, if D_F is large, effects of high-dimensional data spaces occur. Therefore, the model for high-dimensional data spaces must be adapted for this case (cf. section 5). For moderate D_F both basic models can be applied. As a practical guide, we assume boundary effects if the fractal dimension D_F is greater than or equal to the maximum split dimension d':

$$D_F \geq d' = \left\lceil \log_2\left(\frac{N}{C_{\text{eff}}}\right) \right\rceil.$$

6.4 Range Query

First, we want to determine, how the access probability of a page changes in the presence of a correlation described by the fractal dimension D_F. Let us assume that the fractal point density ρ_F is constant throughout the populated part of the page region and its Minkowski enlargement. For range queries, this assumption is necessary as we have already pointed out in the beginning of section 6. For nearest neighbor queries, a relaxed restriction to smoothly distributed points in the subspace will be sufficient. In the case of low D_F, we can estimate the side length a of a page region according to the power law:

$$C_{\text{eff}} = \rho_F \cdot \left(\frac{a_{\text{ld}}}{1 - 1/C_{eff}}\right)^{D_F} \Rightarrow a_{\text{ld}} = \sqrt[D_F]{\frac{C_{\text{eff}}}{\rho_F}} \cdot \left(1 - \frac{1}{C_{eff}}\right).$$

In the high-dimensional case, d' splits are explicitly applied to achieve data pages with a suitable number of points (C_{eff}). However, we must take into account that a split in some dimension automatically leads to a reduced extension in some correlated dimension. We assume the extension

$$a_{\text{split}} = 0.5 \cdot \left(1 - \frac{1}{C_{\text{eff}}}\right)$$

(cf. section 5.2) in a number d'' of dimensions with

$$d'' = \frac{d' \cdot d}{D_F}$$

and full extension (up to MBR effects, cf. section 5.2)

$$a_{\text{unsplit}} = 1 - 1/C_{\text{eff}}$$

in the remaining $d - d''$ dimensions.

Now we must make an assumption for the distribution of the query points. First, we assume that they are taken from a uniform and independent data distribution. Later, we will assume that data points and query points are selected from the same distribution. For uniformly distributed query points, the Minkowski sum of the page region and a query range r corresponds to the access probability of the page. Following our discussion in section 3.3 and 5.2, we get the following access probabilities for Euclidean metric and maximum metric and for the high-dimensional and the low-dimensional case, respectively:

$$X_{\text{r,mm,ld,c/ui}}(r) = \left(\min\left\{\sqrt[D_F]{\frac{C_{\text{eff}}}{\rho_F}} \cdot \left(1 - \frac{1}{C_{\text{eff}}}\right) + 2r, 1\right\}\right)^d,$$

$$X_{\text{r,mm,hd,c/ui}}(r) = \left(\min\left\{\frac{1}{2} - \frac{1/4}{C_{\text{eff}}} + r, 1\right\}\right)^{\left\lceil \log_2\left(\frac{N}{C_{\text{eff}}}\right)\right\rceil \cdot \frac{d}{D_F}},$$

$$X_{\text{r,em,ld,c/ui}}(r) = \sum_{0 \le k \le d} \binom{d}{k} \cdot \left(\left(1 - \frac{1}{C_{\text{eff}}}\right) \cdot \sqrt[D_F]{\frac{C_{\text{eff}}}{\rho_F}}\right)^k \cdot \frac{\sqrt{\pi^{d-k}}}{\Gamma\left(\frac{d-k}{2} + 1\right)} \cdot r^{d-k},$$

$$X_{\mathrm{r,em,hd,c/ui}}(r) = \sum_{0 \le k \le d''} \binom{d''}{k} \cdot \left(\frac{1}{2} - \frac{1/4}{C_{\mathrm{eff}}}\right)^k \cdot \left(\frac{1}{2} + \frac{1/4}{C_{\mathrm{eff}}}\right)^{d''-k} \cdot V_{\mathrm{csi}}(d'', \frac{r}{\frac{1}{2} + \frac{1/4}{C_{\mathrm{eff}}}}) \ .$$

The expected value of the number of page accesses can be easily determined by multiplication with the number of data pages N/C_{eff}.

For real applications, uniform distribution of the query points is not a realistic assumption. A better alternative is to assume that data points and query points are taken from the same distribution and yield the same fractal dimension D_F. Instead of taking the volume of the Minkowski enlargement for the access probability, we should rather determine the percentage of the query points lying in the Minkowski enlargement. The power law can be used for this purpose, yielding:

$$\begin{aligned} X_{\mathrm{r,mm,ld,c}}(r) &= X_{\mathrm{r,mm,ld,c/ui}}(r)^{D_F/d} \\ &= \left(\min \left\{ \sqrt[D_F]{\frac{C_{\mathrm{eff}}}{\rho_F}} \cdot \left(1 - \frac{1}{C_{eff}}\right) + 2r, 1 \right\} \right)^{D_F} . \end{aligned}$$

The other equations can be modified in the same way.

6.5 Nearest Neighbor Query

Following our coarse model for the estimation of the nearest neighbor distance (cf. section 4.1), we can easily determine a volume having an expectation of 1 point enclosed. As in the preceding section, we assume that the distribution of the query points follows the distribution of the data points. The volume can then be estimated by using the power law:

$$1 = \rho_F \cdot V^{D_F/d} \Rightarrow R_{\mathrm{mm,ld,cor}} \approx \frac{1}{2 \cdot \sqrt[D_F]{\rho_F}}$$

for the maximum metric and

$$\frac{1}{\rho_F} = V^{D_F/d} = \frac{\sqrt{\pi}^{D_F}}{\Gamma(d/2+1)^{D_F/d}} \cdot R_{\mathrm{em,ld,cor}}{}^{D_F} \Rightarrow R_{\mathrm{em,ld,cor}} \approx \frac{\sqrt[d]{\Gamma(d/2+1)}}{\sqrt{\pi}} \cdot \sqrt[D_F]{\frac{1}{\rho_F}}$$

for the Euclidean metric. If D_F is sufficiently large (according to section 6.4), boundary effects must be considered. For the maximum metric, we get the following formula:

$$1 = \rho_F \cdot V^{D_F/d} = \rho_F \cdot (2R_{\mathrm{mm,hd,cor}} - R_{\mathrm{mm,hd,cor}}{}^2)^{D_F} \Rightarrow R_{\mathrm{mm,hd,cor}} \approx 1 - \sqrt{1 - \sqrt[D_F]{1/\rho_F}} .$$

For the Euclidean metric, we need the inverse function of the *cube-sphere intersection with arbitrary center*, $V^{-1}_{\mathrm{csi,a}}(d, V)$ (cf. section 5.3). The corresponding discretization of $V^{-1}_{\mathrm{csi,a}}(d, V)$ can be obtained in a single pass of the discretization of $V_{\mathrm{csi,a}}(d, r)$. The estimation of the nearest neighbor distance is:

$$\frac{1}{\rho_F} = V_{\mathrm{csi,a}}(r)^{D_F/d} \Rightarrow R_{\mathrm{em,hd,cor}} \approx V^{-1}_{\mathrm{csi,a}}\left(\left(\frac{1}{\rho_F}\right)^{d/D_F}\right) .$$

For our exact model, we have to adapt our distribution function in a suitable way. Again, we have to apply the power law:

$$P(r) = 1 - \left(1 - \frac{\rho_F}{N} \cdot V(r)^{\frac{D_F}{d}}\right)^N ,$$

where $V(r)$ is the volume of the d-dimensional hypersphere with radius r in the case of the Euclidean metric and the volume of the d-dimensional hypercube with side length $2r$ in the case of the maximum metric. We have to make a suitable distinction between the low-dimensional and the high-dimensional case when choosing $V(r)$. The rest is straightforward and can be handled as in sections 4-5: An expectation for the nearest neighbor distance can again be gained by integrating r multiplied with the derivative of $P(r)$. The new distribution function must be multiplied with the Minkowski sum as in sections 4-5. For the maximum metric, we get the following formulas for the low-dimensional and the high-dimensional case, respectively:

$$A_{\text{nn,mm,ld,cor}} = \frac{N}{C_{\text{eff}}} \int_0^\infty \left(\left(\min \left\{ \sqrt[D_F]{\frac{C_{\text{eff}}}{\rho_F}} \cdot \left(1 - \frac{1}{C_{eff}}\right) + 2r, 1 \right\} \right)^{D_F} \cdot \frac{\partial}{\partial r} \left(1 - \left(1 - \frac{\rho_F}{N} \cdot (2r)^{D_F}\right)^N\right) \right) \partial r ,$$

$$A_{\text{nn,mm,hd,cor}} = \frac{N}{C_{\text{eff}}} \int_0^\infty \left(\left(\min \left\{ \left(\frac{1}{2} - \frac{1/4}{C_{eff}}\right) + r, 1 \right\} \right)^{\left\lceil \log_2\left(\frac{N}{C_{\text{eff}}}\right) \right\rceil} \cdot \frac{\partial}{\partial r} \left(1 - \left(1 - \frac{\rho_F}{N} \cdot (2r - r^2)^{D_F}\right)^N\right) \right) \partial r .$$

For the Euclidean metric, the corresponding result is

$$A_{\text{nn,em,ld,cor}} = \frac{N}{C_{\text{eff}}} \int_0^\infty \left(\left(\sum_{0 \le k \le d} \binom{d}{k} \cdot \left(\left(1 - \frac{1}{C_{\text{eff}}}\right) \cdot \sqrt[D_F]{\frac{C_{\text{eff}}}{\rho_F}} \right)^k \cdot \frac{\sqrt{\pi}^{d-k}}{\Gamma\left(\frac{d-k}{2} + 1\right)} \cdot r^{d-k} \right)^{\frac{D_F}{d}} \right.$$

$$\left. \cdot \frac{\partial}{\partial r} \left(1 - \left(1 - \frac{\rho_F}{N} \cdot \frac{\sqrt{\pi}^{D_F}}{\Gamma(d/2 + 1)^{D_F/d}} \cdot r^{D_F}\right)^N\right) \right) \partial r ,$$

$$A_{\text{nn,em,hd,cor}} = \frac{N}{C_{\text{eff}}} \int_0^\infty \left(\left(\sum_{0 \le k \le d''} \binom{d''}{k} \cdot \left(\frac{1}{2} - \frac{1/4}{C_{\text{eff}}}\right)^k \cdot \left(\frac{1}{2} + \frac{1/4}{C_{\text{eff}}}\right)^{d''-k} \cdot V_{\text{csi}}\left(d'', \frac{r}{\frac{1}{2} + \frac{1/4}{C_{\text{eff}}}}\right) \right)^{\frac{D_F}{d}} \right.$$

$$\left. \cdot \frac{\partial}{\partial r} \left(1 - \left(1 - \frac{\rho_F}{N} \cdot V_{\text{csi,a}}(d, r)^{\frac{D_F}{d}}\right)^N\right) \right) \partial r .$$

Techniques facilitating the evaluation of these formulas were presented in sections 4-5.

To evaluate our model extension, indexes on several data sets from real-world applications were constructed. Our first application is a similarity search system for CAD drawings provided by a subcontractor of the automobile industry [Berchtold and Kriegel 1997]. The drawings were transformed into 16-dimensional fourier vectors. Our second application is a

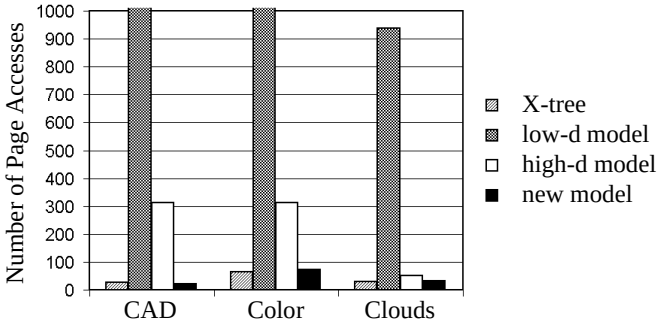

Figure 22: Accuracy for Data Sets from Real-World Applications.

content based image retrieval using color histograms with 16 histogram bins [Seidl 1997]. Both databases contained 50,000 objects. Our third application contained 9-dimensional vectors from weather observations. The fractal dimensions of our data sets are 7.0 (CAD), 8.5 (Color Histograms) and 7.3 (Clouds). We performed nearest neighbor queries using the Euclidean and the maximum metric and compared the results obtained with the predictions of the following 3 cost models:

- The original models by Friedman, Bentley and Finkel [Friedman *et al.* 1977] and Cleary [Cleary 1979], cf. section 4
- our extension to high-dimensional query processing (cf. section 5)
- our extension to non-uniformity and correlation

The results are depicted in figure 22. In contrast to the low-dimensional and the high-dimensional model, the new model considering correlation yields sufficient accuracy in all performed experiments.

7. CONCLUSIONS

7.1 Summary

In this paper, we have proposed a comprehensive cost model for query processing in high-dimensional data spaces. It is based on the concepts of our previous cost model [Berchtold *et al.* 1997b] which were extended in many aspects. The BBKK model has introduced two techniques, Minkowski sum and data space clipping to estimate the number of page accesses when performing range queries and nearest neighbor queries in a high-dimensional Euclidean space. These techniques are explained in detail and were extended to take also the maximum metric into account. Our new model drops the restriction of the BBKK model which could only handle numbers of data pages which are a power of 2. A further extension of this paper are *k*-nearest neighbor queries. We have described in detail and further developed the numerical methods for evaluating the cost model. Computationally expensive steps in the cost estimation are moved to the compile time by applying precomputed volume functions. Correlated data were finally taken into account using the fractal dimension. Experimental evaluations showed the practical applicability of the cost estimations and their superiority over competing techniques in terms of accuracy.

7.2 Future Work

In the course of this paper, we have stated several limitations of the current approach which are subject to future work. These open issues are summarized here.

We noted in the introduction that query processing in high-dimensional data spaces is often a result of a feature transformation which maps objects of an arbitrary application to points in a high-dimensional vector space. In order to guarantee correctness, the selectivity in the feature space is worse than the selectivity in the object space, such that more feature vectors than objects must be retrieved. How much more objects must be searched, depends on the feature transformation, which subsumes tasks such as dimension reduction and scaling (normalizing) of the data space. The estimation of these effects is beyond the scope of this paper. However, the selectivity of the feature transformation is important for our index cost estimation. Therefore, modeling the feature transformation including effects such as dimension reduction and scaling are subject to our future work.

An additional limitation of our model is the smoothness assumption of the data distribution and in general non-uniformity for range queries. As described in section 6.1, smoothness means that there are no too sharp changes in the point density, for instance between regions which are covered and regions which are not covered by points. Violations of this smoothness assumptions which are not due to correlation are not considered by our model. In the one-dimensional case, such distributions can be handled by models such as the Zipf distribution, by histogram techniques or kernel estimators. We will investigate equivalent multidimensional techniques in future. Similar approaches are needed for modeling range queries in the presence of non-uniformity. This will also be an issue of our further research.

Finally, our model does not take into account the impact of various heuristics in the algorithms for index construction and for query processing. The insert and split strategies are assumed to produce page regions which are cube-like in the dimensions which are affected by the splits. An important question is how the performance changes depending on how different (unbalanced) the side lengths of the page regions are. The next obvious question would be how to determine this asymmetry given the construction heuristics. For the algorithms performing range queries and nearest neighbor queries, we have also assumed optimality, which is met in the depth-first search for range queries and in the HS algorithm for nearest neighbor queries. Another well-known algorithm for the nearest neighbor search is the RKV algorithm (cf. section 1.5). As we have pointed out, the RKV algorithm is difficult to predict. As RKV is known to yield worse performance than HS, there is no reason to apply RKV. More interesting is the development of fast index scans [Berchtold *et al.* 2000] that consider not only the position of a page region in the data space but also the position of the page on disk while planning the disk accesses. Modeling such algorithms is also subject to our future work.

8. ACKNOWLEDGMENT

I'd like to express particular thanks to the reviewers who read the paper extraordinarily carefully and gave valuable hints for substantial improvements.

9. REFERENCES

AGRAWAL R., FALOUTSOS C., SWAMI A. Efficient similarity search in sequence databases. *Proc. 4th Int. Conf. on Foundations of Data Organization and Algorithms*, 1993, 69-84.

AGRAWAL R., GEHRKE J., GUNOPULOS D., RAGHAVAN P. Automatic subspace clustering of high dimensional data for data mining applications. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1998, 94-105.

AGRAWAL R., LIN K., SHAWNEY H., SHIM K. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. *Proc. 21st Int. Conf. on Very Large Databases*, 1995, 490-501.

ALTSCHUL S. F., GISH W., MILLER W., MYERS E. W., LIPMAN D. J. A basic local alignment search tool. *Journal of Molecular Biology*, Vol. 215, No. 3, 1990, 403-410.

AREF W. G., SAMET H. Optimization strategies for spatial query processing. *Proc. 17th Int. Conf. on Very Large Databases*, Barcelona, Catalonia, 1991, 81-90.

ARYA S., MOUNT D.M., NARAYAN O. Accounting for boundary effects in nearest neighbor searching. *Proc. 11th Symp. on Computational Geometry*, Vancouver, Canada, 1995, 336-344.

ARYA S. *Nearest Neighbor Searching and Applications*. Ph.D. thesis, University of Maryland, College Park, MD, 1995.

BELUSSI A., FALOUTSOS C. Estimating the selectivity of spatial queries using the correlation fractal dimension. *Proc. 21st Int. Conf. on Very Large Data Bases*, Zurich, Switzerland, 1995, 299-310.

BERCHTOLD S., BÖHM C., BRAUNMÜLLER B., KEIM D. A., KRIEGEL H.-P. Fast parallel similarity search in multimedia databases. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, Tucson, Arizona, 1997a, 1-12, SIGMOD BEST PAPER AWARD.

BERCHTOLD S., BÖHM C., JAGADISH H. V., KRIEGEL H.-P., SANDER J. Independent quantization: An index compression technique for high-dimensional data spaces. *Proc. 16th Int. Conf. on Data Engineering*, San Diego, CA, 2000.

BERCHTOLD S., BÖHM C., KEIM D., KRIEGEL H.-P. A cost model for nearest neighbor search in high-dimensional data space. *ACM Symposium on Principles of Database Systems*, Tucson, Arizona, 1997b.

BERCHTOLD S., BÖHM C., KEIM D., KRIEGEL H.-P., XU X. Optimal multidimensional query processing using tree striping. Submitted for publication, 1998c.

BERCHTOLD S., BÖHM C., KRIEGEL H.-P. Improving the query performance of high-dimensional index structures using bulk-load operations. *Proc. 6th Int. Conf. on Extending Database Technology*, Valencia, Spain, 1998a.

BERCHTOLD S., BÖHM C., KRIEGEL H.-P. The pyramid-technique: Towards indexing beyond the curse of dimensionality. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, Seattle, WA, 1998b, 142-153.

BERCHTOLD S. *Geometry based search of similar parts*. Ph.D. thesis (in german), University of Munich, 1997.

BERCHTOLD S., KEIM D., KRIEGEL H.-P. The X-tree: An index structure for high-dimensional data. *Proc. 22nd Int. Conf. on Very Large Databases*, Bombay, India, 1996, 28-39.

BERCHTOLD S., KEIM D., KRIEGEL H.-P. Using extended feature objects for partial similarity retrieval. *VLDB Journal* Vol. 6, No. 4, 1997c, 333-348.

BERCHTOLD S., KRIEGEL H.-P. *S3:* Similarity search in CAD database systems. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, Tucson, Arizona, 1997, 564-567.

BECKMANN N., KRIEGEL H.-P., SCHNEIDER R., SEEGER B. The R*-tree: An efficient and robust access method for points and rectangles. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, Atlantic City, NJ, 1990, 322-331.

BEYER K., GOLDSTEIN J., RAMAKRISHNAN R., SHAFT U. When is “nearest neighbor” meaningful? *Proc. Int. Conf. on Database Theory*, Jerusalem, Israel, 1999, 217-235.

BÖHM C. *Efficiently Indexing High-Dimensional Data Spaces.* Ph.D. Thesis, University of Munich, Utz-Verlag München, 1998.

BÖHM C., KRIEGEL H.-P. Dynamically optimizing high-dimensional index structures. *Proc. 7th Int. Conf. on Extending Database Technology*, Konstanz, Germany, 2000.

CHIUEH T. Content-based image indexing. *Proc. 20th Int. Conf. on Very Large Data Bases*, 1994, 582-593.

CIACCIA P., PATELLA M., ZEZULA P. A cost model for similarity queries in metric spaces. *Proc. of the 7th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, 1998, 59-68.

CLEARY J.G. Analysis of an algorithm for finding nearest neighbors in euclidean space. *ACM Trans. on Mathematical Software*, Vol. 5, No.2, 1979, 183-192.

DUDA R. O., HART P. E. *Pattern Classification and Scene Analysis.* Wiley, New York, 1973.

EASTMAN C.M. Optimal bucket size for nearest neighbor searching in k-d trees. *Information Processing Letters* Vol. 12, No. 4, 1981.

FALOUTSOS C., BARBER R., FLICKNER M., HAFNER J., NIBLACK W., PETKOVIC D., EQUITZ W. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, Vol. 3, 1994a, 231-262.

FALOUTSOS C., BHAGWAT P. Declustering using fractals. *Journal of Parallel and Distributed Information Systems*, 1993, 18-25.

FALOUTSOS C., GAEDE V. Analysis of n-dimensional quadtrees using the Hausdorff fractal dimension. *Proc. 22nd Int. Conf. on Very Large Data Bases*, Mumbai (Bombay), India, 1996, 40-50.

FALOUTSOS C., KAMEL I. Beyond uniformity and independence: Analysis of R-trees using the concept of fractal dimension. *Proc. of the Thirteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Minneapolis, Minnesota, 1994, 4-13.

FALOUTSOS C., LIN K.-I. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia data. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, San Jose, CA, 1995, 163-174.

FALOUTSOS C., RANGANATHAN M., MANOLOPOULOS Y. Fast subsequence matching in time-series databases. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1994b, 419-429.

FALOUTSOS C., SELLIS T., ROUSSOPOULOS N. Analysis of object-oriented spatial access methods. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1987.

FRIEDMAN J. H., BENTLEY J. L., FINKEL R. A. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, Vol. 3, No. 3, September 1977, 209-226.

FUKUNAGA K. *Introduction to Statistical Pattern Recognition.* 2nd edition, Academic Press, 1990.

GAEDE V., GÜNTHER O. Multidimensional access methods. *ACM Computing Surveys* 30(2), 1998, 170-231.

GAEDE V. Optimal redundancy in spatial database systems. *Proc. 4th International Symposium on Advances in Spatial Databases*, Portland, Maine, 1995, 96-116.

GARY J. E., MEHROTRA R. Similar shape retrieval using a structural feature index. *Information Systems*, Vol. 18, No. 7, 1993, 525-537.

GOLUB G.H., VAN LOAN C.F. *Matrix Computations.* 2nd edition, John Hopkins Univerity Press, Baltimore, 1989.

GUTTMAN A. *R-trees:* A dynamic index structure for spatial searching. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, Boston, MA, 1984, 47-57.

HENRICH, A. A distance-scan algorithm for spatial access structures. *2nd ACM Workshop on Advances in Geographic Information Systems*, 1994, 136-143.

HENRICH, A. The LSD^h-tree: An access structure for feature vectors. *Proc. 14th Int. Conf. on Data Engineering*, Orlando, 1998.

HJALTASON G. R., SAMET H. Ranking in spatial databases. *Proc. 4th Int. Symp. on Large Spatial Databases*, Portland, ME, 1995, 83-95.

JAGADISH H. V. A retrieval technique for similar shapes. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1991, 208-217.

JAIN R, WHITE D.A. Similarity indexing: Algorithms and performance. *Proc. Storage and Retrieval for Image and Video Databases*, Vol. 2670, San Jose, CA, 1996, 62-75.

KALOS M. H., WHITLOCK P. A. *Monte Carlo Methods.* Wiley, New York, 1986.

KASTENMÜLLER G., KRIEGEL H.-P., SEIDL T. Similarity search in 3d protein databases. *Proc. German Conf. on Bioinformatics*, Köln (Cologne), 1998.

KATAYAMA N., SATOH S. The SR-tree: An index structure for high-dimensional nearest neighbor queries. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1997, 369-380.

KAUL A., O'CONNOR M. A., SRINIVASAN V. Computing minkowski sums of regular polygons. *Proc. 3rd Canad. Conf. on Computing Geometry*, 1991, 74-77.

KEIM D. A. *Efficient similarity search in spatial database systems.* Habilitation thesis, Institute for Computer Science, University of Munich, 1997.

KORN F., SIDIROPOULOS N., FALOUTSOS C., SIEGEL E., PROTOPAPAS Z. Fast nearest neighbor search in medical image databases. *Proc. 22nd Int. Conf. on Very Large Data Bases*, Mumbai (Bombay), India, 1996, 215-226.

KRIEGEL H.-P., SCHMIDT T., SEIDL T. 3d similarity search by shape approximation. *Proc. Fifth Int. Symposium on Large Spatial Databases*, Berlin, Germany, 1997, 11-28.

KRIEGEL H.-P., SEIDL T. Approximation-based similarity search for 3d surface segments. *GeoInformatica Journal*, Kluwer Academic Publishers, 1998.

LIN K., JAGADISH H. V., FALOUTSOS C. The TV-tree: An index structure for high-dimensional data. *VLDB Journal*, Vol. 3, 1995, 517-542.

MANDELBROT B. *Fractal geometry of nature*. W. H. Freeman and Company, New York, 1977.

MEHROTRA R., GARY J. Feature-based retrieval of similar shapes. *Proc. 9th Int. Conf. on Data Engeneering*, 1993.

MEHROTRA R., GARY J. Feature-index-based similar shape retrieval. *Proc. 3rd Working Conf. on Visual Database Systems*, 1995.

PAGEL B.-U., SIX H.-W., TOBEN H., WIDMAYER P. Towards an analysis of range query performance in spatial data structures. *Proc. of the Twelfth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Washington, D.C., 1993, 214-221.

PAPADOPOULOS A., MANOLOPOULOS Y. Performance of nearest neighbor queries in R-trees. *Proc. 6th Int. Conf. on Database Theory*, Delphi, Greece, in: Lecture Notes in Computer Science, Vol.†1186, Springer, 1997, 394-408.

PAPADOPOULOS A., MANOLOPOULOS Y. Similarity query processing using disk arrays. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1998, 225-236.

PRESS W., FLANNERY B. P., TEUKOLSKY S.A., VETTERLING W. T. *Numerical Recipes in C.* Cambridge University Press, 1988.

RIEDEL E., GIBSON G. A., FALOUTSOS C. Active storage for large-scale data mining and multimedia. *Proc. 24th Int. Conf. on Very Large Databases*, 1998, 62-73.

ROUSSOPOULOS N., KELLEY S., VINCENT F. Nearest neighbor queries. *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1995, 71-79.

SCHRÖDER M. *Fractals, Chaos, Power Laws: Minutes from an Infinite Paradise*. W.H. Freeman and Company, New York, 1991.

SEEGER B. *Multidimensional Access Methods and their Applications*. Tutorial, 1991.

SEIDL T. *Adaptable Similarity Search in 3-D Spatial Database Systems*. Ph.D. Thesis, University of Munich, 1997.

SEIDL T., KRIEGEL H.-P. Efficient user-adaptable similarity search in large multimedia databases. *Proc. 23rd Int. Conf. on Very Large Databases*, Athens, Greece, 1997, 506-515.

SELLIS T., ROUSSOPOULOS N., FALOUTSOS C. The R^+-tree: A dynamic index for multi-dimensional objects. *Proc. 13th Int. Conf. on Very Large Databases*, Brighton, England, 1987, 507-518.

SHAWNEY H., HAFNER J. Efficient color histogram indexing. *Proc. Int. Conf. on Image Processing*, 1994, 66-70.

SPROULL R.F. Refinements to nearest neighbor searching in k-dimensional trees. *Algorithmica*, 1991, 579-589.

STRANG G. *Linear Algebra and its Applications*. 2nd edition, Academic Press, 1980.

THEODORIDIS Y., SELLIS T. K. A model for the prediction of R-tree performance. *Proc. of the Fifteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Montreal, Canada, 1996, 161-171.

VAN DEN BERCKEN J., SEEGER B., WIDMAYER P. A general approach to bulk loading multidimensional index structures. *Proc. 23rd Int. Conf. on Very Large Databases*, Athens, Greece, 1997.

WEBER R., SCHEK H.-J., BLOTT S. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces. *Proc. 24th Int. Conf. on Very Large Databases*, New York, 1998.

WHITE D.A., JAIN R. Similarity indexing with the SS-tree. *Proc. 12th Int. Conf on Data Engineering*, New Orleans, LA, 1996.

YAO A. C., YAO F. F. A general approach to d-dimensional geometric queries. *Proc. ACM Symp. on Theory of Computing*, 1985.