A Model for the Prediction of R-tree Performance

Yannis Theodoridis

Computer Science Division
Department of Electrical and Computer Engineering
National Technical University of Athens
Zographou 15773, GREECE
e-mail: theodor@cs.ntua.gr

Timos Sellis

Computer Science Division
Department of Electrical and Computer Engineering
National Technical University of Athens
Zographou 15773, GREECE
e-mail: timos@cs.ntua.gr

Abstract: In this paper we present an analytical model that predicts the performance of R-trees (and its variants) when a *range query* needs to be answered. The cost model uses knowledge of the dataset only, i.e., the proposed formula that estimates the number of disk accesses is a function of data properties, namely, the *amount* of data and their *density* in the work space. In other words, the proposed model is applicable even before the construction of the R-tree index, a fact that makes it a useful tool for *dynamic* spatial databases. Several experiments on synthetic and real datasets show that the proposed analytical model is very accurate, the relative error being usually around 10%-15%, for uniform and non-uniform distributions. We believe that this error is involved with the gap between efficient R-tree variants, like the R*-tree, and an optimum, not implemented yet, method. Our work extends previous research concerning R-tree analysis and constitutes a useful tool for spatial query optimizers that need to evaluate the cost of a complex spatial query and its execution procedure.

1. Introduction

The development of efficient data structures for maintaining large sets of multi-dimensional (*spatial*) data, such as points or regions, is of crucial importance in several applications, including *Spatial, Image* and *Multimedia* Database Systems. In recent years, several data structures have been developed for point [NHS84, Fre87, HSW89] and non-point [Gut84, SRF87, BKSS90, KF94] objects in multi-dimensional space. All these indexing methods use several heuristics to index spatial data efficiently. The large number of spatial data structures proposed indicate that, today, research in this field should turn to the development of powerful analytical models that predict the performance of a data structure instead of the development of one more of them. Powerful analytical models are useful in three ways:

(i) they allow us to better understand the behavior of a data structure under various input datasets and sizes.

(ii) they can play the role of an objective comparison point when various proposals for efficient spatial indexing are compared each other.

(iii) they can be used by a query optimizer in order to evaluate the cost of a complex spatial query and its execution procedure.

The performance of a spatial data structure is usually evaluated by the ability to answer *range queries* by accessing the smallest possible number of pages in the disk. Insertion and deletion operations are also important but not as common in real applications as queries on the spatial database. Other queries, such as *nearest neighbor queries* [RKV95] or topological queries of high resolution (e.g. *meet*, *covers*) [PTSE95] are useful but not representative of the efficiency of a spatial data structure. As a consequence, all the efforts to analytically predict the performance of spatial data structures and, particularly R-trees [Gut84], have concentrated on the range query performance.

Some efforts towards the analytical estimation of the range query performance of R-tree-based data structures have been presented in the past [FSR87, KF93, PSTW93, FK94]. As a further step, in this paper we propose an analytical model that supports datasets of any type (point or region data) and distribution (uniform or non-uniform). In particular, we propose a model that predicts the R-tree performance using only data properties and, especially, the *amount* N and the *density* D of the dataset. The proposed model is shown to be efficient for several distributions of synthetic and real datasets. The relative error, extracted by several experiments, is usually around 10%-15%, for uniform and non-uniform distributions. We believe that this error is involved with the gap between efficient R-tree variants, like the R*-tree, and an optimum, not implemented yet, method.

The paper is organized as follows: In section 2, we provide the background information about R-trees and related work on R-tree analysis. In section 3, we present our analysis for the prediction of the R-tree performance, based on data properties. In section 4, we give experimental results that evaluate our model on different data distributions. We conclude, in section 5, discussing some applications and extensions of the proposed model, and giving hints for future work.

2. Background and Related Work

R-trees were proposed by Guttman [Gut84] as a direct extension of B^+-trees [Knu73, Com79] in n-dimensions. The data structure is a height-balanced tree which consists of intermediate and leaf nodes. A *leaf node* is of the form

$$(oid, R)$$

where *oid* is an object identifier and is used to refer to an object in the database. R is the MBR approximation of the data object, i.e., it is of the form

$$(p_{l\text{-}1}, p_{l\text{-}2}, \ldots, p_{l\text{-}n}, p_{u\text{-}1}, p_{u\text{-}2}, \ldots, p_{u\text{-}n})$$

which represents the $2n$ coordinates of the lower-left (p_l) and the upper-right (p_u) corner of a n-dimensional (hyper-) rectangle p. An *intermediate node* is of the form

$$(ptr, R)$$

where *ptr* is a pointer to a lower level node of the tree and R is a representation of the rectangle that encloses.

Let M be the maximum number of entries in a node and let $m \le M / 2$ be a parameter specifying the minimum number of entries in a node. An R-tree satisfies the following properties:

(i) Every leaf node contains between m and M entries unless it is the root.
(ii) For each entry (*oid*, R) in a leaf node, R is the smallest rectangle that spatially contains the data object represented by *oid*.
(iii) Every intermediate node has between m and M children unless it is the root.
(iv) For each entry (*ptr*, R) in an intermediate node, R is the smallest rectangle that spatially contains the rectangles in the child node.
(v) The root node has at least two children unless it is a leaf.
(vi) All leaves appear in the same level.

Figure 1 shows an example set of data rectangles and the corresponding R-tree built on these rectangles (assuming maximum node capacity $M = 4$).

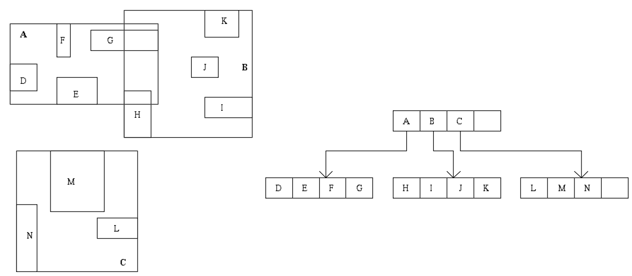


Figure 1: Some rectangles, organized to form an R-tree, and the corresponding R-tree

After Guttman's proposal, several researchers proposed their own improvements on the basic idea. Among others, Roussopoulos and Leifker [RL85] proposed the packed R-tree, for the case that data objects are known in advance (i.e., it is applicable only to static databases), Sellis et al. [SRF87] proposed the R^+-tree, a variant of R-trees that guarantees disjointness of nodes, Beckmann et al. [BKSS90] proposed the R*-tree, an R-tree-based method that uses a rather complex but effective grouping algorithm, Kamel and Faloutsos [KF94] proposed the Hilbert R-tree, an improved R-tree variant using fractals, and so on.

The R-tree family of spatial data structures seem to be the most promising ones and those that most research efforts have concentrated on. Recently, the interest turns to the proposal of efficient analytical models for R-tree cost estimation, rather than the proposal of one more effective variant.

Faloutsos et al. [FSR87] presented a model that estimates the performance of R-trees and R^+-trees [SRF87] assuming uniform distribution of data and *packed* trees (i.e., all the nodes of the tree are full of data). Later, Kamel and Faloutsos [KF93] and Pagel et al. [PSTW93] independently presented the following formula that gives the average number *DA* of disk accesses in a n-dimensional R-tree index accessed by a query window $q = (q_1, \ldots, q_n)$, provided that the sides $(s_{j,1}, \ldots, s_{j,n})$ of each R-tree node s_j are known (the summation extends over all the nodes of the tree).

$$DA(q) = \sum_j \left\{ \prod_{i=1}^{n} (s_{j,i} + q_i) \right\}$$

The above formula allows us to estimate the number of disk accesses for a query window q but assumes that the R-tree has been built and that the MBR of each node s_j of the R-tree can be measured. In other words, the proposed formula is *qualitative*, i.e., it does not predict the average number of disk accesses but, intuitively, presents the effects of the sizes of the nodes and the query to the R-tree performance. Moreover, the influence of the node perimeters is revealed, that helps us to understand the R*-tree efficiency as, only in that method among the R-tree variants, the node perimeter has been taken into account [PSTW93].

Faloutsos and Kamel [FK94] extended the above formula to really predict the number of disk accesses using a property of the dataset, called *fractal dimension*. The fractal dimension d of a dataset (consisting of points) can be mathematically computed and constitutes a simple way to describe non-uniform datasets, using just a single number. The estimation of the number of disk accesses $DA(q)$ at level 1 (i.e., leaf level) according to the model proposed in [FK94] (f is the average capacity - fanout - of the R-tree nodes) is given by:

$$DA(q) = \frac{N}{f} \cdot \prod_{i=1}^{n} (s_{1,i} + q_i)$$

where $$s_{1,i} = \left(\frac{f}{N} \right)^{1/d}, \ \forall\, i = 1, \ldots, n$$

The above model constitutes the first attempt to model R-tree performance for non-uniform distributions of data (including the uniform distribution as a special case: $d = n$) superseding an older analysis [FSR87] that assumed uniformity. However the model is applicable only to point datasets which are not the majority in real spatial applications.

Extending work of [PSTW93], Pagel et al. [PSW95] recently proposed an optimal algorithm that establishes a lower bound result for *static* R-tree performance. They also show by experimental results that the best known static and dynamic R-tree variants, the *packed* R-tree [KF93] and the R*-tree respectively, perform about 10%-20% worse than the lower bound.

Symbols	Definitions
n	number of dimensions
N	amount of a dataset
D	density of a dataset
$q = (q_1, \ldots, q_n)$	query window
m	minimum R-tree node capacity
M	maximum R-tree node capacity
f	average node capacity - fanout - of the R-tree
h	height of the R-tree
N_j	number of R-tree nodes at level j
$s_j = (s_{j,1}, \ldots, s_{j,n})$	average size of an R-tree node at level j
DA	number of disk accesses for a query window q
S	selectivity for a query window q

Table 1: Table of symbols and definitions.

3. R-tree Analysis

Formally, the problem of the R-tree cost analysis is defined as follows: Let n be the dimensionality of the data space and $WS = [0,1)^n$ the n-dimensional unit work space. Let us assume that N (point or non-point) data objects of average size $s = (s_1, \ldots, s_n)$ are stored in an R-tree index and a query asking for all objects that intersect a query window $q = (q_1, \ldots, q_n)$ needs to be answered. A formula that estimates the average number *DA* of disk accesses with only the knowledge of the data properties is the goal to be reached. Although we refer to dynamic indexing we assume that we can use some data properties for our prediction, such as the expected number N and average size s of data, since these properties can be usually computed using a sample of the dataset (efficient sampling algorithms have been proposed, among others, by Vitter in [Vit84, Vit85]).

Definition: The *density* D of a set of N boxes with average size s is the average number of boxes that contain a given point of n-dimensional space. Equivalently, density D can be expressed as the ratio of the global data area over the work space area. If we assume that the work space is the unit (hyper-) rectangle $[0,1)^n$ then the density $D(N,s)$ is given by the following formula:

$$D(N,s)=\sum_{N}\prod_{i=1}^{n} s_i = N\cdot\prod_{i=1}^{n} s_i \qquad (1)$$

Assume now an R-tree of height h (the root is assumed to be at level h and leaf-nodes are assumed to be at level 1). If N_j is the number of nodes at level j and s_j their average size then the expected number DA of disk accesses in order to answer a query $q = (q_1, ..., q_n)$ is defined as follows:

$$DA = 1+\sum_{j=1}^{h-1}\text{intsect}(N_j, s_j, q) \qquad (2)$$

where intsect(N_j, s_j, q) is a function that returns the number of nodes at level j intersected by the query window q. In other words, Eq. 2 expresses the fact that the expected number of disk accesses is equal to 1 (the root access) plus the expected number of intersected nodes at each level j (j = 1, ..., h-1). In order to estimate the function *intsect* we provide the following lemma:

Lemma: Given a set of N boxes with average size s and a query window q, the average number of intersected boxes is:

$$\text{intsect}(N,s,q) = N\cdot\prod_{i=1}^{n}(s_i+q_i) \qquad (3)$$

Proof: The average number of intersected boxes is equal to the density D of the boxes inflated by q_i at each direction, i.e.,

$$\text{intsect}(N,s,q)=\text{intsect}(N,s',0)=D(N,s')=N\cdot\prod_{i=1}^{n} s_i'$$

where $s_i' = s_i + q_i$. ■

Combining Eq. 2 and 3 we have:

$$DA = 1+\sum_{j=1}^{h-1}\left\{N_j\cdot\prod_{i=1}^{n}(s_{j,i}+q_i)\right\}. \qquad (4)$$

Our goal is to express Eq. 4 as a function of the data properties N (number of data) and s (average size of data boxes) or, in other words, to express the R-tree properties h, N_j and s_j as functions of N and s.

The height h of an R-tree with average node capacity (fan-out) f that stores N data entries is given by the following formula [FSR87]:

$$h = 1+\left\lceil \log_f \frac{N}{f}\right\rceil \qquad (5)$$

Since a node organizes, on the average, f entries then we can assume that the average number of leaf-nodes is $N_1 = N / f$, the average number of leaf-nodes' parents is $N_2 = N_1 / f$, etc. In general, the average number of nodes at level j is

$$N_j = \frac{N}{f^j} \qquad (6)$$

If we assume D_j to be the density of nodes at level j (j=1, ..., h-1) then (from Eq. 1, 6)

$$D_j = N_j\cdot\prod_{i=1}^{n} s_{j,i} = \frac{N}{f^j}\cdot(s_{j,i})^n \Rightarrow$$

$$s_{j,i} = \left(D_j\cdot\frac{f^j}{N}\right)^{1/n} \qquad (7)$$

In order to reach Eq. 7 we assumed that the sizes of the node sides are equal (i.e., $s_{j,1} = s_{j,2} = ... = s_{j,n}$, $\forall j$). This simplification is a reasonable property for a "good" R-tree [KF93]. What remains is an estimation of D_j using data properties.

Suppose that, at level j, N_j nodes with average size $(s_{j,i})^n$ are organized in N_{j+1} parent nodes with average size $(s_{j+1,i})^n$. Each parent node groups, on the average, f nodes. The average size $s_{j+1,i}$ of a parent node at each direction is given by:

$$s_{j+1,i} = \left(f^{1/n}-1\right)\cdot t_{j,i} + s_{j,i} \qquad (8)$$

where $t_{j,i}$ is the distance between the centers of two consecutive boxes' projections at direction i and its average value is given by:

$$t_{j,i} = \frac{1}{(N_j)^{1/n}}, \quad \forall j \qquad (9)$$

Intuitively, Eq. 8 and 9 express that, among the f nodes that construct one parent node, $f^{1/n}$ nodes are responsible for the size of the parent node at each direction. The centers of the N_j nodes' projections are equally distanced and this distance ($t_{j,i}$) depends on $N_j^{1/n}$ nodes at each direction (see Figure 2 for an illustration when n = 2).

Definition: The density $D_{j,i}$ of a set of N_j nodes' projections (line segments) with average size $s_{j,i}$ is the average number of segments that contain a point of 1-dimensional space. Assuming that the work space at each direction is the unit segment [0,1) we have:

$$D_{j,i} = N_j\cdot s_{j,i} \qquad (10)$$

Lemma: The density $D_{j,i}$ of a set of N_j nodes' projections is expressed as a function of the density D_j of the N_j nodes by the following equation:

$$D_{j,i} = \left(D_j\cdot N_j^{n-1}\right)^{1/n} \qquad (11)$$

Proof:

$$D_{j,i} = N_j \cdot s_{j,i} \Rightarrow \prod_n D_{j,i} = \prod_n \left(N_j \cdot s_{j,i}\right) = N_j^{\,n} \cdot \prod_n s_{j,i} \Rightarrow$$

$$\Rightarrow D_{j,i}^{\,n} = N_j^{\,n} \cdot \frac{D_j}{N_j} = N_j^{\,n-1} \cdot D_j \,. \qquad \blacksquare$$

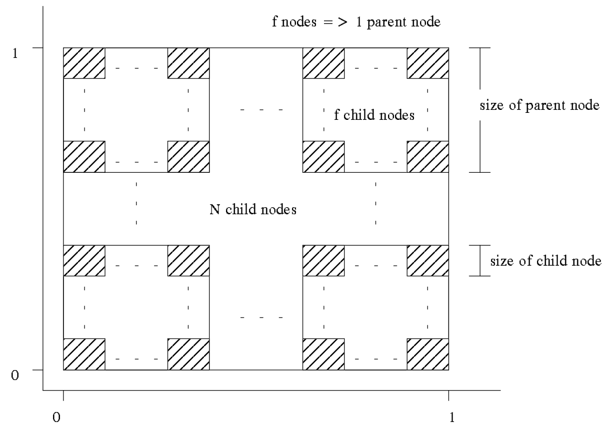


Fig.2: Grouping of *f* nodes into 1 parent node

Using Eq. 8 through 11 we can express the density D_{j+1} of the N_{j+1} parent nodes as a function of the density D_j of the N_j nodes at level j.

$$D_{j+1} = N_{j+1} \cdot s_{j+1}^{\;n} \Rightarrow \ldots \Rightarrow$$

$$D_{j+1} = \left\{ 1 + \frac{D_j^{1/n} - 1}{f^{1/n}} \right\}^n \qquad (12)$$

Eq. 12 expresses the density of the nodes at level j+1 as a function of the density of the nodes at level j and the average node capacity f. Essentially, using Eq. 12 we can compute the density at each level of the R-tree as a function of only the density D of the data objects (which can be named D_0).

At this point, the original goal set has been reached, i.e., to find a formula for the expected number *DA* of disk accesses in order to answer a range query specified by a query window *q* using only the data properties *N*, *D* and the *f* factor. Combining Eq. 4, 5, 6, 7 and 12, the following formula for *DA* can be derived:

$$DA = 1 + \sum_{j=1}^{1+\left\lceil \log_f \frac{N}{f} \right\rceil} \left\{ \frac{N}{f^j} \cdot \prod_{i=1}^{n} \left(\left(D_j \cdot \frac{f^j}{N} \right)^{1/n} + q_i \right) \right\} \qquad (13)$$

Clearly this formula can be computed by only using the dataset properties *N* and *D*, the typical R-tree parameter *f* and the query window *q*.

Finally, we can note that the *selectivity S* of a range query specified by a query window *q*, i.e., the ratio of the expected number of answers in the dataset over the total number *N* of entries, can also be computed using only the data properties *N* and *D*. It is important to be able to predict the selectivity of a query search, as the result of such a search may be input to another operation where a query processor must also be able to predict, and so on, which is a typical procedure in complex queries processing. Obviously, the selectivity *S* for a query window *q* is equal to intsect(*N,s,q*), as defined in Eq. 3, over *N*. In other words, *S* is equal to the ratio of the number of intersected boxes among the *N* boxes of the input dataset over *N*. Combining Eq. 1 and 3 we have:

$$S = \frac{\mathrm{intsect}(N,s,q)}{N} = \frac{N \cdot \prod_{i=1}^{n} (s_i + q_i)}{N} \Rightarrow \ldots \Rightarrow$$

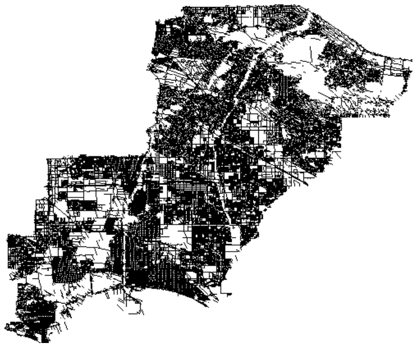
$$S = \prod_{i=1}^{n} \left(\left(\frac{D}{N} \right)^{1/n} + q_i \right) \qquad (14)$$

As will be shown in section 4 the above selectivity estimation is very accurate with the relative error being usually below 5% or 10% for random or non-uniform datasets, respectively.

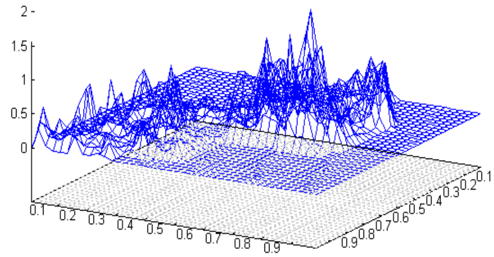
The proposed analytical model assumes uniformity of data in order to express the density of the R-tree nodes at a level j+1 as a function of the density of the child nodes at level j (Eq. 12). In particular, it was assumed that the centers of the N_j nodes' projections are equally distanced, in order to provide Eq. 8 and 9. This assumption (i.e., uniformity of data centers) produces a model that could be efficient for uniform-like data distributions but hardly applicable to non-uniform distributions of data, which are the rule when dealing with real applications [Falo94].

To adapt the model in order to efficiently support any type of datasets (uniform or non-uniform ones) we reduce the uniformity assumption of the analytical model from a *global-effect* factor (i.e., assuming the global work space) to a *local-effect* factor (i.e., assuming a small sub-area of the work space).

The idea is the following: The "density" factor of the dataset is involved in the formula that estimates the R-tree performance (Eq. 13, combined with Eq. 12) as a single number. However, for non-uniform datasets, density is a varying parameter (graphically, a surface in 2-dimensional space) that shows strong deviations, if measured in different points of the work space, compared to the average value. For example, in Figure 3, a real dataset[1] is illustrated together with its *density surface*.



(a) LBcounty dataset



(b) LBcounty density surface

Fig.3: A real dataset and its density surface

The average density of this dataset is $D_{avg} = 0.13$. However, actual density values vary from $D = 0$ (in zero-populated areas, such as the upper-left and bottom-right corners) up to $D = 2$ (in high-populated areas), with respect to the reference point. It is evident that using the D_{avg} value in the cost formula would sometimes lead to inaccurate estimations. On the other hand, a satisfactory image of the density surface provides more accurate D values, with respect to the specified query window q.

Based on the above, Eq. 13 could efficiently support either uniform or non-uniform distributions of data if the following modifications are made:

(i) the average density D_0 of the dataset is replaced by the actual density D_0' of the dataset within the area of the query window q.

(ii) the amount N of the dataset is replaced by a transformation N' of that, computed as follows[2]: $N' = \left(D_0' / D_0 \right) \cdot N$

In the next section, the proposed analytical model is evaluated by comparing the expected disk accesses (Eq. 13) to actual tests using the R*-tree indexing method, which is generally accepted to be one of the most efficient R-tree variants.

4. Evaluation of the Model

In order to evaluate our analytical model we performed several experiments using synthetic and real datasets in 2-dimensional space. We consider that the conclusions of the evaluation are also applicable to higher-dimensionality data. For the 2-dimensional case, the formula for the expected disk accesses DA, Eq. 13, is simplified to the following one (we assume that the sides of the query windows are equal-sized, i.e., $q_i = q$, $\forall\ i = 1, \ldots, n$):

$$DA = 1 + \sum_{j=1}^{1 + \left\lceil \log_f \frac{N}{f} \right\rceil} \left\{ \left(\sqrt{D_j} + q \cdot \sqrt{\frac{N}{f^j}} \right)^2 \right\} \quad (15)$$

where
$$D_j = \left\{ 1 + \frac{\sqrt{D_{j-1}} - 1}{\sqrt{f}} \right\}^2 \quad (15a)$$

For our tests we used two groups of *synthetic* datasets (typical samples of the synthetic datasets are illustrated in Figure 4):

- *random data* : rectangles with centers following random distribution in the work space.
- *skewed data* : rectangles with centers following skewed distribution (according to Zipf's "law") [Knu81] in the work space.

We also used two *real* datasets from the TIGER database:

- *LBcounty dataset* : 53,143 line segments (stored as rectangles) indicating roads of Long Beach county - California (Figure 3a).
- *MGcounty dataset* : 39,221 line segments (stored as rectangles) indicating roads of Montgomery county - Maryland (Figure 5a).

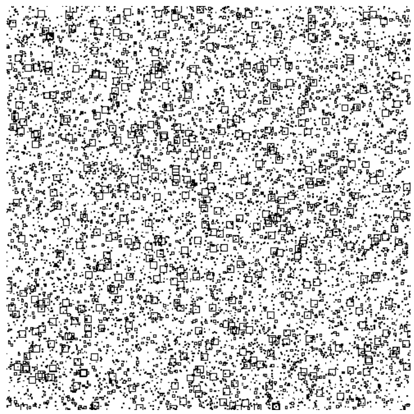
The estimated disk accesses for each test were computed by using Eq. 15 where the average node capacity f was set to 34 entries (i.e., the typical 68% average capacity for nodes that contain at most M = 50 entries). The experimental disk accesses for R*-trees were based on several query-sets consisting of

---

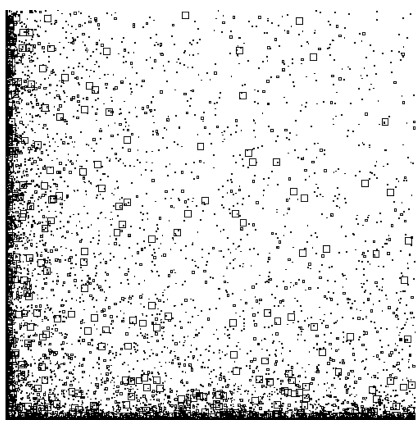
[1]from the TIGER database of the U.S. Bureau of Census.

[2]The average density D_0 of a dataset is considered to be always $D_0 > 0$, even for point datasets. Zero density corresponds to zero-populated work area.

randomly generated query windows q, with sizes q_x, q_y that ranged from 0.0 to 0.4 at each direction (in the [0,1) work space).



(a) Random data

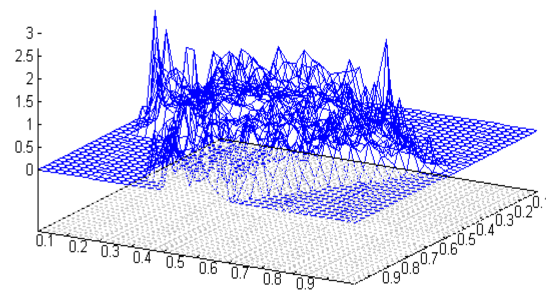


(b) Skewed (Zipf) data

Fig.4: Synthetic datasets

As explained in section 3, instead of computing just the average density D_{avg} of the dataset, the density surface *D(x,y)* was generated by appropriate sampling. The cost formula (Eq. 15) takes into consideration the value of *D(x,y)* that corresponds to the specified query window $q(q_x, q_y)$. For evaluation purposes, we selected 9 representative points of the work space (all combinations of (x,y) where x in {0.25, 0.50, 0.75} and y in {0.25, 0.50, 0.75}). An example of this strategy is illustrated in Figure 5 where the density surface of the *MGcounty* real dataset appears, together with the values of *D(x,y)* on the 9 representative points. Clearly, the density surface *D(x,y)* of a dataset provides a better estimation of the low- and high-populated areas of the work space.

(a) MGcounty dataset

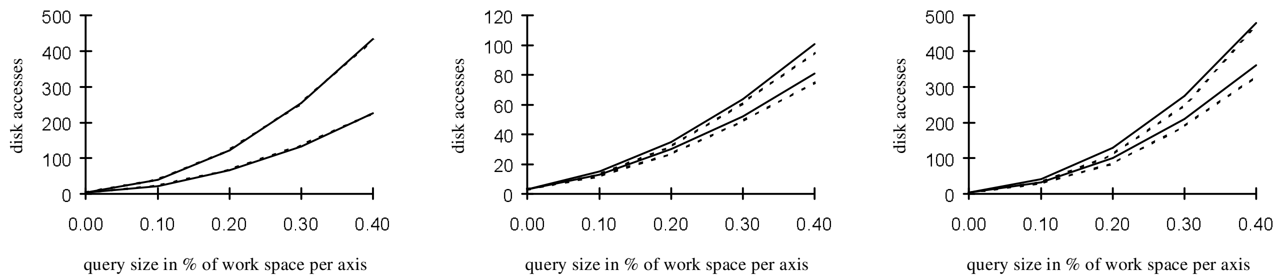


	x = 0.25	x = 0.50	x = 0.75
y = 0.25	0.549	0.709	0
y = 0.50	0.799	0.670	0.349
y = 0.75	0	0.488	0.174

(b) MGcounty density surface (and 9 representative values)

Fig.5: The MGcounty dataset and its density surface with its 9 representative values

We ran several tests in order to compare the experimental results with the analytical ones using query windows around the representative points. Figure 6 illustrates the results for random, skewed and real datasets. The analytical results are plotted with dotted lines and the experimental results for R*-trees with solid lines. The estimation of the number of disk accesses *DA* is very accurate, compared to the actual results using R*-trees, since the relative error is usually around 10%-15%. This was the rule for all datasets that we tested.



(a) Random data: D = 0.1, N = 40K (lower pair) and 80K (upper pair)

(b) Skewed data: D = 0.1, N = 40K (lower pair) and 80K (upper pair)

(c) Real data: LBcounty (upper pair) and MGcounty (lower pair) datasets

Fig.6: Comparison of average experimental and analytical performance: analytical estimates (dotted lines) and actual results using R^*-trees (solid lines).

The flexibility of the proposed analytical model on non-uniform distributions of data, using the "*density surface*", is also extracted from the results of our experiments. Figure 7 illustrates the results for typical point (q = (0,0)) or range queries (q = (0.1,0.1) on random, skewed and real datasets. The analytical results are plotted with dotted lines while the experimental results using R*-trees are plotted with solid lines. Note also that the skewed and real distributions show similar irregularities for both the analytical and the experimental values.

Summarizing the results of our tests, we list in Table 2 the average relative errors of the actual results compared to the predictions of our model.

Datasets	**Relative Error**	
	point queries	range queries
Random data	0%-10%	0%-5%
Skewed data	0%-15%	0%-10%
Real data	0%-15%	0%-20%

Table 2: Average relative error in estimating the number *DA* of disk accesses.

In general, the main conclusion that arises from the tests is that the proposed analytical model is applicable to spatial data that follow a wide range of distributions, either "good" (such as uniform-like random) or "bad" (such as Zipf skewed) ones. As discussed earlier, the uniformity assumption of the analysis is also adaptive to non-uniform distributions of data using the "density surface" property of the dataset. Compared to related work, our model is not limited to point datasets (as in [FK94]) or uniform distributions (as in [FSR87]) and can be easily used as a guideline by a spatial query optimizer to estimate the retrieval cost of a range query.

A second estimation which is supported by the proposed analysis concerns the *selectivity* parameter. Using Eq. 14 we estimate the qualifying hits that will be retrieved by a range query specified by a query window q. This estimation, called *selectivity* estimation, is also important for a query optimizer in order to decide the execution steps of a complex spatial query. We evaluated the analytical estimate of Eq. 14 using the same datasets as before. As listed in Table 3, the relative errors are very low, i.e., the analytical estimate is almost identical to the actual results. In other words, Eq. 14 can be used successfully to the selectivity estimation of a range query.

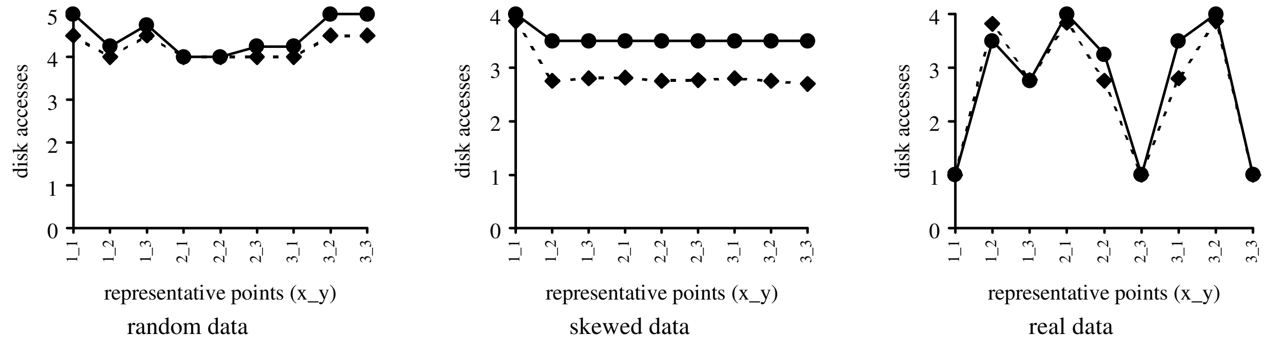
Datasets	Relative Error
Random data	0%-5%
Skewed data	5%-10%
Real data	5%-10%

Table 3: Average relative error in estimating selectivity S

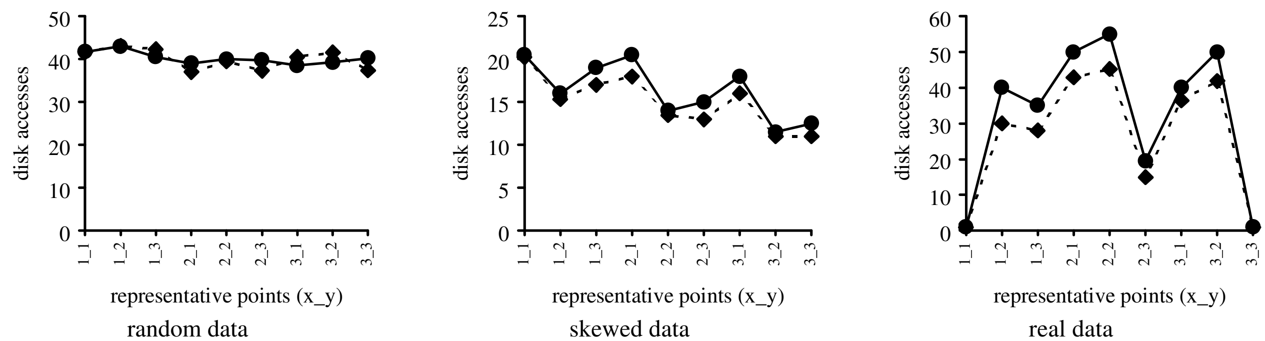
Observing Eq. 14 we extract two useful results:

- if the dataset includes only points (*point dataset*) then $D = 0$ which implies that $S = q^n$.
- if the query is a point with zero size (*point query*) then $q = 0$ which implies that $S = D/N$.

Both formulas are very simple but powerful enough for many real applications which are characterized by one of the two properties (point datasets and/or point queries).



(a) point queries



(b) range queries

Fig.7: Comparison of experimental and analytical performance around representative points: analytical estimates (dotted lines) and actual results using R^*-trees (solid lines).

5. Conclusion

We presented a model that predicts the performance of R-trees (and its variants) when a point or range query needs to be answered. The formula that estimates the number of disk accesses is a function of data properties only, namely, the *amount* N of data and their *density* D in the work space, and can be used without any knowledge of the R-tree index properties. It is applicable to any size of data or query and, although it makes use of the uniformity assumption, it is also adaptive to non-uniform (skewed) distributions. The main point of our analysis is the *density* of a dataset. We showed how this parameter can be used to compute the average sizes of the R-tree nodes at each level, and, as a consequence, the average number of disk accesses at each level. Although we refer to dynamic indexing we consider that we can estimate the "*density surface*" of a dataset, using appropriate sampling methods.

Our experimental results showed that the proposed analytical model is very accurate, the relative error being usually around 10%-15% when the analytical estimate is compared to the experimental results of the R*-tree, one of the most efficient R-tree variants. We believe that this relative error is involved with the gap between efficient existing R-tree variants and an optimum, not implemented yet, method. Relevant conclusions, concerning *static* spatial data structures, can be found in [PSW95] where an efficient static R-tree variant, packed R-trees [KF93], performed about 10%-20% worse than the theoretical lower bound.

A second estimation that is supported by our model concerns the *selectivity* S of a query q, i.e., the ratio of the expected number of answers in the dataset over the total number N of entries. Relevant experimental results showed very low relative error (below 5% or 10%) for all types of datasets. The prediction of the selectivity of a query window is important in spatial query processing, especially when complex queries are involved.

In this work we focused on range queries (and point queries as a special case: $q = 0$). Future work can extend the analytical model in order to make cost and selectivity estimation for other types of queries in a multi-dimensional space, involving topological (*meet*, *covers*, etc.), directional (*north*, *northeast*, etc.) or distance (*near*, *nearest*, etc.) information. We have already adapted the model to estimate the cost of (a) several representative direction and distance relations on GIS applications with random (uniform-like) datasets [TP95] and (b) spatio-temporal relations between objects of large Multimedia applications [TVS96], by handling such relations as range queries with an appropriate transformation of the query window q.

Finally, besides query optimization, the proposed analytical model could also be useful to the construction of an efficient R-tree. Since the cost formula estimates the number of disk accesses, which is the primary factor for the efficiency of the tree structures, a transformation of that formula could possibly lead to a dynamic and auto-tuned function that will be responsible for the tree construction [TS94] and comparable to the heuristic methods of the existing R-tree variants.

References

[BKSS90] N. Beckmann, H.-P. Kriegel, R. Schneider, B. Seeger, "The R*-tree: an efficient and robust access method for points and rectangles", In *Proceedings of ACM SIGMOD Conference on Management of Data*, May 1990.

[Com79] D. Comer, "The Ubiquitous B-Tree", *ACM Computing Surveys*, vol. 11(2), pp. 121-137, June 1979.

[FK94] C. Faloutsos, I. Kamel, "Beyond Uniformity and Independence: Analysis of R-trees Using the Concept of Fractal Dimension", In *Proceedings of the 13th ACM Symposium on Principles of Database Systems (PODS)*, May 1994.

[Fre87] M. Freeston, "The BANG file: a new kind of grid file", In *Proceedings of ACM SIGMOD Conference on Management of Data*, May 1987.

[FSR87] C. Faloutsos, T. Sellis, N. Roussopoulos, "Analysis of Object Oriented Spatial Access Methods", In *Proceedings of ACM SIGMOD Conference on Management of Data*, May 1987.

[Gut84] A. Guttman, "R-trees: a dynamic index structure for spatial searching", In *Proceedings of ACM SIGMOD Conference on Management of Data*, June 1984.

[Gut94] R.H. Guting, "An Introduction to Spatial Database Systems", *VLDB Journal*, vol.3(4), pp. 357-399, October 1994.

[HSW89] A. Henrich, H.-W. Six, P. Widmayer, "The LSD tree: spatial access to multidimensional point and non point objects", In *Proceedings of the 15th International Conference on Very Large Data Bases (VLDB)*, August 1989.

[KF93] I. Kamel, C. Faloutsos, "On Packing R-trees", In *Proceedings of the 2nd International Conference on Information and Knowledge Management (CIKM)*, November 1993.

[KF94] I. Kamel, C. Faloutsos, "Hilbert R-tree: An Improved R-tree Using Fractals", In *Proceedings of the 20th International Conference on Very Large Data Bases (VLDB)*, September 1994.

[Knu73] D. Knuth, *The Art of Computer Programming, vol. 3: Sorting and Searching*, Addison-Wesley, Reading, MA, 1973.

[Knu81] D. Knuth, *The Art of Computer Programming*, *vol.2*, Addison Wesley, 1981.

[NHS84] J. Nievergelt, H. Hinterberger, K.C. Sevcik, "The Grid File: An Adaptable, Symmetric Multikey file Structure", *ACM Transactions on Database Systems*, vol.9(1), pp. 38-71, March 1984.

[PSTW93] B.-U. Pagel, H.-W. Six, H. Toben, P. Widmayer, "Towards an Analysis of Range Query Performance", In *Proceedings of the 12th ACM Symposium on Principles of Database Systems (PODS)*, May 1993.

[PSW95] B.-U. Pagel, H.-W. Six, M. Winter, "Window Query-Optimal Clustering of Spatial Objects", In *Proceedings of the 14th ACM Symposium on Principles of Database Systems (PODS)*, May 1995.

[PTSE95] D. Papadias, Y. Theodoridis, T. Sellis, M.J. Egenhofer, "Topological Relations in the World of Minimum Bounding Rectangles: a Study with R-trees", In *Proceedings of ACM SIGMOD Conference on Management of Data*, May 1995.

[RKV95] N. Roussopoulos, S. Kelley, F. Vincent, "Nearest Neighbor Queries", In *Proceedings of ACM SIGMOD Conference on Management of Data*, May 1995.

[RL85] N. Roussopoulos, D. Leifker, "Direct Spatial Search on Pictorial Databases Using Packed R-trees", In *Proceedings of ACM SIGMOD*

Conference on Management of Data, May 1985.

[SRF87] T. Sellis, N. Roussopoulos, C. Faloutsos, "The R^+-tree: a dynamic index for multidimensional objects", In *Proceedings of the 13th International Conference on Very Large Data Bases (VLDB)*, September 1987.

[TS94] Y. Theodoridis, T. Sellis, "Optimization Issues in R-tree Construction", In *Proceedings of International Workshop on Geographic Information Systems (IGIS)*, March 1994.

[TP95] Y. Theodoridis, D. Papadias, "Range Queries Involving Spatial Relations: A Performance Analysis", In *Proceedings of the 2nd International Conference on Spatial Information Theory (COSIT)*, September 1995.

[TVS96] Y. Theodoridis, M. Vazirgiannis, T. Sellis, "Spatio-Temporal Indexing for Large Multimedia Applications", In *Proceedings of the 3rd IEEE Conference on Multimedia Computing and Systems (ICMCS)*, June 1996.

[Vit84] J.S. Vitter, "Faster Methods for Random Sampling", *Communications of the ACM*, vol. 27(7), pp. 703-718, July 1984.

[Vit85] J.S. Vitter, "Random Sampling with Reservoir", *ACM Transactions on Mathematical Software*, vol. 11, pp. 37-57, March 1985.