Analysis of n-dimensional Quadtrees Using the Hausdorff Fractal Dimension

*Christos Faloutsos**
Department of Computer Science
and Institute for Systems Research
University of Maryland
College Park, MD 20742-3255
christos@cs.umd.edu

Volker Gaede
Institut für Wirtschaftsinformatik
Humboldt-Universität zu Berlin
Spandauer Str. 1
10178 Berlin, Germany
gaede@wiwi.hu-berlin.de

Abstract

There is mounting evidence [Man77, Sch91] that real datasets are statistically self-similar, and thus, 'fractal'. This is an important insight since it permits a compact statistical description of spatial datasets; subsequently, as we show, it also forms the basis for the theoretical analysis of spatial access methods, without using the typical, but unrealistic, uniformity assumption.

In this paper, we focus on the estimation of the number of quadtree blocks that a real, spatial dataset will require. Using the the well-known Hausdorff fractal dimension, we derive some closed formulas which allow us to predict the number of quadtree blocks, given some few parameters. Using our formulas, it is possible to predict the space overhead and the response time of linear quadtrees/z-ordering [OM88], which are widely used in practice. In order to verify our analytical model, we performed an extensive experimental investigation using several real datasets coming from different domains. In these experiments, we found that our analytical model agrees well with our experiments as well as with older empirical observations on 2-d [Gae95b] and 3-d [ACF+94] data.

*This work was partially supported by the National Science Foundation under Grants No. CDR-8803012, EEC-94-02384, IRI-8958546 and IRI-9205273), with matching funds from Empress Software Inc. and Thinking Machines Inc. Some of the work was performed while he was visiting AT&T Bell Laboratories, Murray Hill, NJ.

**Proceedings of the 22nd VLDB Conference
Mumbai(Bombay), India, 1996**

1 Introduction

Spatial databases have numerous applications, including geographic information systems, medical image databases [ACF+94], multimedia databases (after extracting n features from each object, and mapping it into a point in n-d space [Jag91, FRM94]), as well as traditional databases, where each record with n attributes can be considered as a point in n-dimensional space [Güt94].

In order to guarantee the fast retrieval of the data stored in these databases, spatial access methods are typically used. In practice, the prevailing methods seem to be two: (a) the R-trees [Gut84] and its variants [SRF87, BKSS90], and (b) methods based on a regular subdivision of the data space such as linear quadtrees [Gar82] and z-ordering [OM84]. The terms 'linear quadtrees' and 'z-ordering' essentially denote the same method and therefore, will be used interchangeably.

Linear quadtrees have been very popular for 2-dimensional spaces. One of the major application is in geographic information systems: linear quadtrees have been used both in production systems, like the TIGER system at the U.S. Bureau of Census [Whi81] (`http://tiger.census.gov/tiger/tiger.html`), which stores the map and statistical data of the U.S.A., as well as research prototypes such as QUILT [SSN87], PROBE [OM88], and

GODOT [GR94]. For higher dimensions, oct-trees have been used in 3-d graphics and robotics [BB82]; in databases of 3-d medical images [ACF+94], etc. There are several good reasons for the popularity of these methods, such as their simplicity, their robustness and that the indexing keys can be inserted into ubiquitous *one*-dimensional access methods, e.g., B-trees.

In all the above cases, it is important to know the number of index entries (z-values) that a specified region will be decomposed into, since the performance of the spatial access methods is correlated with the number of index entries [Ore89, Gae95b, Gae95a]. For a query region, the number of z-values is related to the number of disk accesses that will be required; for a data region, the number of z-values is directly related to the storage requirements for this region, as we describe in subsection 2.2. Ideally, one could predict for each given dataset the number of indexing entries by using some parameter.

In a previous paper [FK94], we presented an analysis of R-trees using the Hausdorff fractal dimension. Recently, we also showed that the theory of fractals can be successfully used for estimating the selectivity of spatial queries [BF95]. The results presented are very encouraging, since the estimates based on fractals yielded very good results compared with other assumptions typically made. In [BF95], we already pointed out that fractals may also be a suitable tool for the analysis of spatial access methods based on a regular decomposition of the data space such as z-ordering [OM84]. Here, we will substantiate this claim by providing an analysis of linear quadtrees that uses the theory of the well-known Hausdorff fractal dimension.

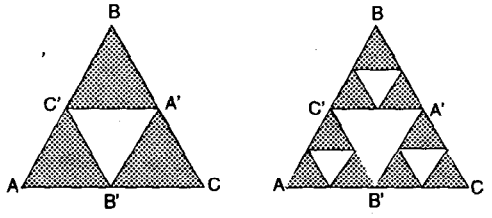
The contributions of this work are the following: First, we highlight and exploit the fact that most real datasets are self-similar (*fractal*). Second, using an existing, successful assumption for 'random quadtrees' [VM96], we show that the number of index entries for an object follows a power law, with exponent the so-called 'Hausdorff' fractal dimension of the object's boundary. Third, we show that our result agree perfectly with previous analytical results and that it explains our older empirical work [Gae95b, ACF+94], where we first pointed out the existence of the power law. Lastly, we present experimental results, to demonstrate the accuracy of our formulas for real datasets.

The remainder of this paper is organized as follows: After giving an introduction to fractals and to quadtrees in Section 2, we present our analysis in Section 3. Section 4 presents some experimental results on real datasets. Section 5 compares our results with older ones, and describes how a practitioner could utilize our formulas. Section 6 lists the conclusions and future work.

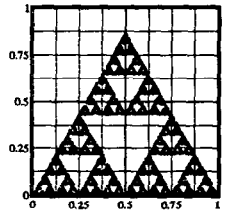
2 Survey - Background

2.1 Introduction to fractals

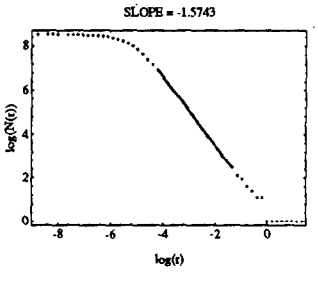
Intuitively, a set of points is a fractal if it exhibits self-similarity over all scales. This is illustrated by an example: Figure 1(a) shows the first few steps in constructing the so-called *Sierpinski triangle*. Figure 1(b) gives a 5,000-point sample, termed 'Sierpinski5K' dataset from now on. Theoretically, the Sierpinski triangle is derived from an equilateral triangle ABC by excluding its middle (triangle A'B'C') and by recursively repeating this procedure for each of the resulting smaller triangles. The resulting set of points exhibits 'holes' in any scale; moreover, each smaller triangle is a *miniature replica* of the whole triangle. In general, the characteristic of fractals is this *self-similarity* property: parts of the fractal are similar (exactly or statistically) to the whole fractal.



(a) Sierpinski triangle



(b) 'Sierpinski5K' dataset



(c) its box-counting plot

Figure 1: *The Sierpinski triangle, a theoretical fractal: (a) the first steps of its recursive construction (b) a sample of 5,000 points and (c) its box-counting plot*

Like all fractals, the Sierpinski triangle is a rich source of paradoxes: it is a point-set with area zero and with infinite-length perimeter. Thus, it is not a 1-dimensional Euclidean object (otherwise it would have finite length perimeter), but it is not a 2-dimensional Euclidean object either, since it has zero area.

The way to resolve the issue is to consider *fractional* values for the dimensionality, which are called *fractal dimensions*. There are more than one fractal dimensions [BF95], but among them the *Hausdorff* or *box-counting* fractal dimension D_H is the one that is suitable for our application.

It is quite easy to compute for a given data set embedded in an E-dimensional address space its fractal dimension D_H by using the box-counting method [Sch91]. This method imposes an E-dimensional grid with (hyper-)cubic grid cells of side r and counts the number $N(r)$ of cells that are penetrated by the set of points (i.e., that contain one or more of its points). By repeating the above, for grids of difference sides, and we can plot the $N(r)$ versus r, in log-log scales. This plot is often called the *box-counting plot.*

If the point-set is self-similar for a range of scales $r \in (r_1, r_2)$, then its box-counting plot will be a straight line for this range. Its negated slope is defined as the Hausdorff fractal dimension D_H of the point-set for the range of scales (r_1, r_2):

Definition 1 (Hausdorff fractal dimension) *For a point-set that has the self-similarity property in the range of scales (r_1, r_2), its Hausdorff fractal dimension D_H for this range is measured as*

$$D_H = -\frac{\partial \log(N(r))}{\partial \log(r)} = \text{constant} \quad r_1 < r < r_2$$

Notice that, for Euclidean objects, their fractal dimension equals their Euclidean dimension. Thus, lines, line segments, circles, and all the standard curves have D_H=1; planes, disks and standard surfaces have D_H=2; Euclidean volumes in E-dimensional space have $D_H = E$.

Figure 1(b) shows the box-counting plot for D_H for the Sierpinski5K dataset. Notice that the slope for $r \in (e^{-4.5}, e^{-1})$ is 1.574, very close to the theoretical value of $\log 3/\log 2 = 1.585$ [Man77].

2.2 Quadtrees and z-ordering

The terminology is easiest described in 2-d address space; the generalizations to E dimensions should be obvious. Following the quadtree literature, the address space is a square, called an *image*, and it is represented as a $2^K \times 2^K$ array of 1×1 squares. Each such square is called a *pixel.*

Consider the four equal squares that the image can be decomposed into. Each such square is called a *level-1 block*; a *level-k block* can be recursively defined as one of the four equal squares that constitute a level-$(k-1)$ block. Thus, the pixels are level-K blocks; the image is the (only) level-0 block. We can represent this process as a 4-way tree: the root is at level 0, and it has four children, the four level-1 blocks. The edges of this tree can be labeled with 2-bit binary strings, where the first bit indicates the horizontal direction ('left/right', for '0/1' respectively) and the second bit indicates the vertical direction ('down/up', for '0/1' respectively). Then, we have:

Definition 2 *The z-value of a level-k block is the concatenation of the labels of the edges, from the root to the node of the quadtree that corresponds to this level-k block.*

An object in the image is represented by turning the appropriate pixels to 'black'; the rest (i.e., background) pixels remain 'white'.

Definition 3 *The level-K quadtree decomposition of an object within an image is the unique, minimal set of blocks of levels 0 through K that cover the object exactly, without covering extra space.*

By 'minimal set of blocks' we mean 'minimum cardinality': That is, the target set of blocks does not contain any quadruplet of level-k blocks that can be consolidated to a single, level-(k-1) block. The efficient way to obtain the quadtree decomposition is by recursively dividing the object into blocks, until they are homogeneous or until we reach the pixel level (level-K blocks). For a 2-dimensional object, the result of such a decomposition is a 4-way tree, which is termed as the *region quadtree* [Kli71]. Blocks that are empty/full/partially-full are represented as white, black and gray nodes in the quadtree, respectively. See Figure 2(b) for an example.

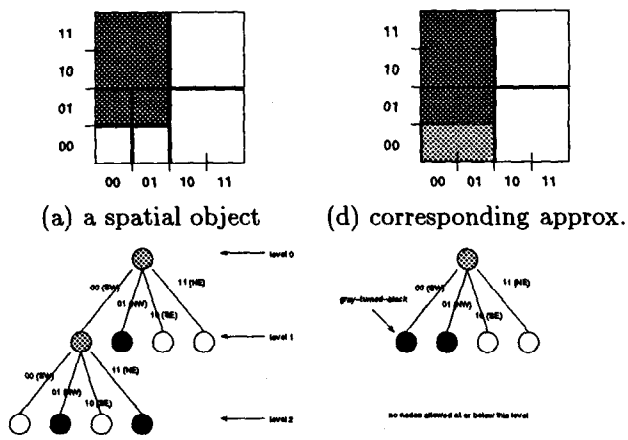
For efficiency reasons (eg., see [Ore89]), we often approximate an object with a 'coarser resolution' object. Formally, we have:

Definition 4 *The level-k quadtree decomposition of an object ($k < K$) is the minimal set of blocks of levels 0 through k that cover the object completely, while they cover the smallest possible additional area.*

Thus, given the level-K quadtree decomposition, represented as a four-way tree, the level-k decomposition is derived by (a) dropping all the nodes at levels j ($j > k$) (b) turning the gray nodes at level-k to black nodes and (c) consolidating black nodes, if necessary. Figure 2 shows an object in a 4×4 image (K=2), its level-2 decomposition, its level-1 decomposition and the corresponding approximation of the object. Notice how the block with z-value '00' turned from 'gray' to 'black', to create the level-1 quadtree decomposition of the original shaded rectangle.

Definition 5 *Let $N_b(k)$ denote the total number of blocks of a level-k quadtree decomposition.*

Since every block has a unique z-value, we can represent a level-k quadtree decomposition of an object by listing the corresponding z-values. Thus, the z-values of the shaded rectangle in figure 2(a) are '0001' (for 'left-down; left-up') '0011' and '01'. Following



(a) a spatial object (d) corresponding approx.

(b) its level-2 quadtree (c) its level-1 quadtree

Figure 2: *Counter-clockwise, from top-left: (a) The shaded rectangle is decomposed into three blocks. (b) the corresponding level-2 quadtree, with z-values* 01, 0001 *and* 0011 *(c) the level-1 quadtree, with z-values* 01, 00 *(d) the corresponding approximate spatial object - the lightly-shaded region is the enlargement, due to the approximation.*

[Gae95b], the maximum permissible length of the z-values that correspond to a level-k quadtree decomposition is called the *granularity* g. Obviously, for a 2-d address space:

$$g = 2 \cdot k$$

In general, for an E-d address space

$$g = E \cdot k \tag{1}$$

As described above, the lengths of the z-values have to be even (in general, multiples of E). There are variations of z-ordering, where the z-values can have arbitrary lengths. For simplicity, we ignore those variations in this paper, although we believe that they are amenable to a similar analysis like the upcoming one.

As described above, quadtrees have been used to store objects in main memory. For disk storage, the prevailing approach is the so-called *linear* quadtree [Gar82], or, equivalently the z-ordering method [OM84]. Using the z-values as just described, each object (and range query) can be uniquely represented by a set of z-values, namely the z-values of the blocks of its level-i quadtree decomposition. Each such z-value can be treated as a key of a record of the form (z-value, object-id, *other attributes* ...), and it can be inserted in a file structure such as a B^+-tree. Table 1 illustrates such a relation, containing the z-values of the shaded rectangle of Figure 2(a).

Additional objects in the same address space can be handled in the same way; their z-values will be inserted into the same B^+-tree. Thus, spatial queries can be served by operations on the B^+-tree: For example, a range query which specifies a region and asks for all the objects in it, will be decomposed into a set of z-values; the B^+-tree can be searched to retrieve matching z-values and the corresponding objects.

z-value	object id	(other attributes)
...	...	...
0001	'ShadedRectangle'	...
...	...	...
0011	'ShadedRectangle'	...
...	...	...
01	'ShadedRectangle'	...
...	...	...

Table 1: *Illustration of the relational table that will store the z-values of the sample shaded rectangle.*

Given the above discussion, the terms 'number of quadtree blocks' and 'number of z-values' are *identical*, and are used interchangeably for the rest of this paper.

Before presenting our analysis, we first want to recall some previous empirical and analytical results, in order to put our work into context.

2.3 Analysis of quadtree decomposition

Previous attempts have been restricted to 2-dimensional polygons [HS79], squares [Dye82, Sha88], 2-d rectangles [Fal92] and E-d hyper-rectangles [FJM94]. In a closely related previous paper [Gae95b], we showed experimentally that the number of blocks $N_b(k)$ of a level-k quadtree decomposition grows exponentially with the level k of the quadtree. Adapting the notation, we showed that

$$N_b(k) \propto (D_z)^g = (D_z)^{2 \cdot k} \tag{2}$$

where the granularity is $g = 2 \cdot k$ for a 2-d address space, and where the value D_z was defined as the *fractal z-ordering dimension* of the specific spatial object. D_z was shown to be the major determinant for the number of index entries resulting from a level-K decomposition. However, the relationship of D_z to D_H was not clear and therefore, we will clarify this point in this paper.

Figure 3(a-b) shows the boundary of Middle Franconia in Germany along with a plot of the natural logarithm of the number of z-values versus the granularity g, i.e., the maximum permissible length for z-values. Similar experimental data on 3-d human brain images [ACF+94] used oct-trees (= 3-d quadtrees), and showed that the number of oct-tree blocks for brain organs grows as

$$N_b(k) \propto 2^{2.63k}$$

Figure 4(a) shows such a brain organ; Figure 4(b) shows the (logarithm-base-2 $\log_2$ of the) number of oct-tree leaf nodes, as a function of the level k, along with the regression line.

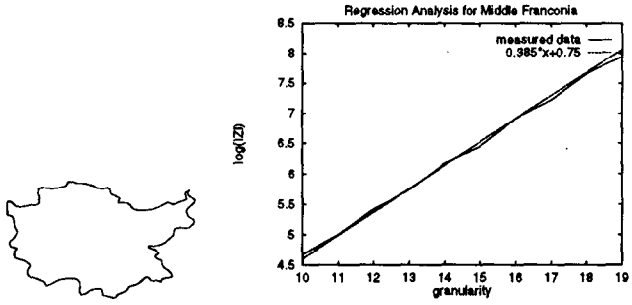


Figure 3: *(a) 'Franconia' dataset: Boundary of Middle Franconia. (b) natural logarithm of the number of its z-values vs. granularity $g = 2 \cdot k$).*

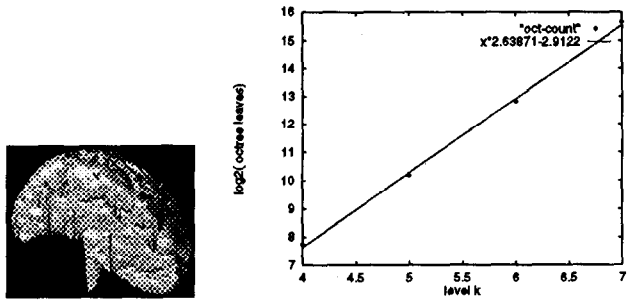


Figure 4: *3-d medical data from [ACF+94]: (a) One brain hemisphere from an atlas (b) ($\log_2$) of number of its oct-tree blocks $N_b(k)$ vs. level k*

Assumptions for Random Quadtrees:

In the analysis of quadtrees and their related algorithms, it is important to have a statistical model of typical, real quadtrees. Such a successful assumption [VM96, SS85] postulates that the probabilities p_w, p_b, p_g of white, black and gray nodes (respectively) is *independent* of the position and of the level. Experiments in [SS85] for nearest neighbor queries showed that the 'level-independence' assumption leads to accurate predictions, for main-memory 2-d quadtrees.

This is the fundamental assumption for the upcoming analysis, which, as we show, agrees well with the experiments.

3 Analysis

The goal of this section is to determine the number $N_b(k)$ of index entries that a spatial object will require in its level-k quadtree decomposition. We would like this formula to be a function of a few, easy-to-estimate parameters of the object, such as the nature of its boundary (as measured by its Hausdorff fractal dimension D_H), the area or more generally, the hyper-volume of the object A_b, etc.

Symbols	Definitions.
E	dimension of the embedding address space
$g(i)$	expected number of gray nodes at level i
$b(i)$	expected number of black nodes at level i
p_w	probability to have a white node
p_b	probability to have a black node
p_g	probability to have a gray node
D_H	Hausdorff fractal dimension of boundary
A_b	total area (hyper-volume) of black pixels at the first k levels
$N_b(k)$	number of z-values (=quadtree blocks) for the level-k decomposition
g	granularity: length of longest permissible z-value ($=E \times k$)

Table 2: *Summary of Symbols and Definitions.*

The strategy we use is to exploit the self-similarity that most real datasets exhibit, and to express both the input (A_b, D_H), as well as the output parameters ($N_b(k)$) in terms of the level-independent probabilities p_g, p_b for gray/black nodes. We make the following conventions:

- the root of the quadtree is at level 0, and it is always gray
- the address space has been normalized to the unit hyper-cube.

The fundamental assumption [VM96] is that the black/gray/white probabilities are level-independent. The only exceptions are the root, which is gray, and the last level K, which has only black pixels. For the intermediate levels, we have:

Assumption 1 (Level-independence) *A (gray) parent node has black, gray and white children with probabilities p_b, p_g, p_w respectively, independent of the level of the parent node.*

We note that the corresponding three probabilities sum to unity: $p_b + p_g + p_w = 1$. Using the above assumption, we can estimate the expected number of black and gray nodes $b(i)$, $g(i)$ at level i:

Lemma 1 *The expected number of gray nodes $g(i)$ at level i is given by:*

$$g(i) = (2^E p_g)^i \tag{3}$$

Proof: By solving the recursion

$$g(i) = g(i-1) \cdot 2^E p_g \tag{4}$$

The idea is that each gray node at level $i-1$ has 2^E children, out of which a fraction p_g are gray. The initial condition is

$$g(0) = 1 \tag{5}$$

since the root at level 0 is always gray. **QED**

Lemma 2 *The expected number of black nodes at a given level i is given by*

$$b(i) = 2^E p_b \ (2^E p_g)^{(i-1)} \tag{6}$$

Proof: There are $g(i-1)$ parents at the $(i-1)$-th level, each with 2^E children; out of them, a fraction p_b are black. **QED**

Next, we need a 'macroscopic' parameter to help us estimate p_b. This parameter is the total black area (hyper-volume) A_b.

Lemma 3 *The hyper-volume ('black area') A_b is given by*

$$A_b = \frac{p_b}{1-p_g} \tag{7}$$

Proof: Combining the hyper-volumes of the individual black nodes at all levels $i = 1, \ldots \infty$ gives:

$$A_b = \sum_{i=1}^{\infty} b(i) 2^{-iE}$$

Substituting $b(i)$ from Lemma 2 and adding the terms of the geometric series, we complete the proof. **QED**

Now we need a second 'macroscopic' parameter, to help us estimate p_g. This parameter is the Hausdorff fractal dimension D_H of the boundary.

Lemma 4 *The Hausdorff fractal dimension D_H of the boundary is given by:*

$$D_H = E + \log_2(p_g) \tag{8}$$

Proof: ¿From Lemma 1 we have that the number $g(i)$ of gray nodes at level i grows as

$$\begin{aligned} g(i) &= (2^E p_g)^i \\ &= (2^i)^{E+\log_2(p_g)} \end{aligned} \tag{9}$$

The crucial observation is that $g(i)$ is *almost exactly* the number of cells of side $r = 2^{-i}$ that the boundary penetrates. The only exceptions occur when a stretch of the boundary coincides with a (horizontal or vertical) dividing line of the quadtree decomposition. If we neglect these rare cases, the exponent is by definition the Hausdorff fractal dimension of the boundary. **QED**

We are ready for the main theorem:

Theorem 1 *For an E-dimensional spatial object, whose level-K quadtree decomposition obeys the the 'level-independence' assumption, the number of blocks $N_b(k)$ for any level-k decomposition ($k \leq K$) is given by*

$$N_b(k) = 2^{kD_H} \cdot C_1 - C_2 \tag{10}$$

where

$$C_1 = 1 + \frac{A_b(2^E - 2^{D_H})}{2^{D_H} - 1} \tag{11}$$

and

$$C_2 = \frac{A_b(2^E - 2^{D_H})}{2^{D_H} - 1} \tag{12}$$

Proof: Recall that the gray nodes at level k are turned to black, and, possibly, consolidated, to form larger blocks. Assuming that consolidation is rare (at least for shapes without extremely convoluted boundary), the desired number of blocks $N_b(k)$ is the number of black nodes at levels $1 \ldots k$, plus the number of gray nodes at the last level k, which are 'treated as black' ones:

$$N_b(k) = g(k) + \sum_{i=1}^{k} b(i) \tag{13}$$

Substituting the values of $g(k)$ and $b(i)$ from Lemmas (1) and (2), we obtain a geometric progression. Adding its terms, we obtain:

$$\begin{aligned} N_b(k) &= (2^E p_g)^k + (2^E p_b) \frac{(2^E p_g)^k - 1}{2^E p_g - 1} \\ &= (2^E p_g)^k (1 + \frac{2^E p_b}{2^E p_g - 1}) - \frac{2^E p_b}{2^E p_g - 1} \end{aligned} \tag{14}$$

Using Lemma (3) and Lemma (4) we obtain Eq. (11) and (12). **QED**

Thus, we have achieved our goal: we have expressed the number of blocks $N_b(k)$ only in terms of 'macroscopic' parameters, and specifically, the Hausdorff fractal dimension D_H of its boundary and the total area (hyper-volume) A_b of the object. For large k, the constant C_2 contributes little since it is smaller than C_1, which is multiplied by an exponentially growing term. Ignoring C_2 leads to a power law:

$$\boxed{N_b(k) \approx 2^{kD_H} \cdot C_1} \tag{15}$$

with C_1 given by Eq. (11).

4 Experiments on Real Datasets

In this section, we provide the results of some of the experiments we undertook in order to verify the accuracy of our analytical formulas. We used real, as well as synthetic data sets (with known fractal dimensions, as 'sanity checks'). For the real data sets, we tried to

use as diverse data as we had available: 2-d and 3-d address spaces, with points, lines and volumes. Specifically, we used:

- 2-d points:
  - **'IUE'**: longitude-latitude co-ordinates of stars in the sky, from NASAs Infrared-Ultraviolet Explorer (Figure 4.1(a), 15,135 points, $D_H = 1.88$).
  - **'MGcounty'**: a point data set with cross-roads of the Montgomery County of Maryland, USA (Figure 4.1(b), 27,282 points, $D_H = 1.71$),
  - **'LBcounty'**: cross-roads of the Long-Beach County of California, USA (Figure 4.1(c), 36,548 points, $D_H = 1.70$)
- 2-d segments:
  - **'Franconia'**: the boundary of Middle Franconia of Germany (Figure 3(a), 147 edges, $D_H = 1.14$)
- 3-d volume: the brain-atlas data: the dataset and the results are shown in Figure 4.

The synthetic data set was

- **Sierpinski5K**: the Sierpinski triangle (Figure 1(a), 5,000 points, $D_H = 1.58$)

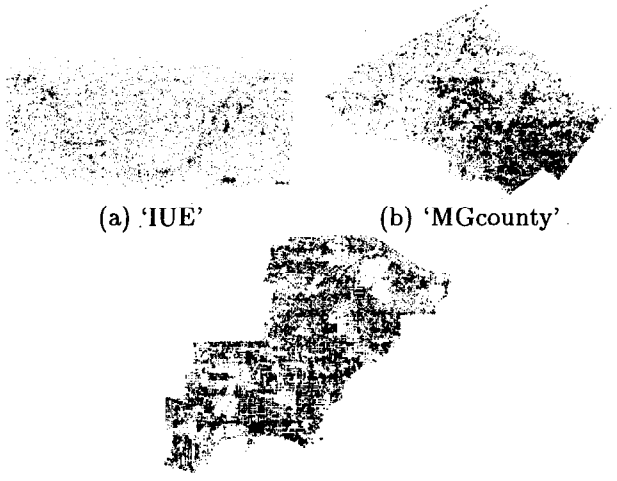
4.1 Check for self-similarity

In a first experiment, we checked each dataset to see whether it is self similar and then we computed the Hausdorff fractal dimension of its boundary using the box-counting plot. Figure 4.1(a-c) show some of the corresponding box-counting plots, e.g., Figure 3(b) and Figure 1(c) show the plots for the 'Franconia' and 'Sierpinski5K' datasets respectively. Notice that they all exhibit fractal (i.e., self-similar) behavior for several scales.

4.2 Accuracy of prediction

In the second set of experiments, we compare the actual number of blocks $N_b(k)$ with our analytical predictions. For each dataset, we determined apart from the Hausdorff fractal dimension as shown in the previous section, the area A_b. Next, we computed the number of quadtree leaf nodes $N_b(k)$ using formula and and plotted its base-2 logarithm as a function of the level k.

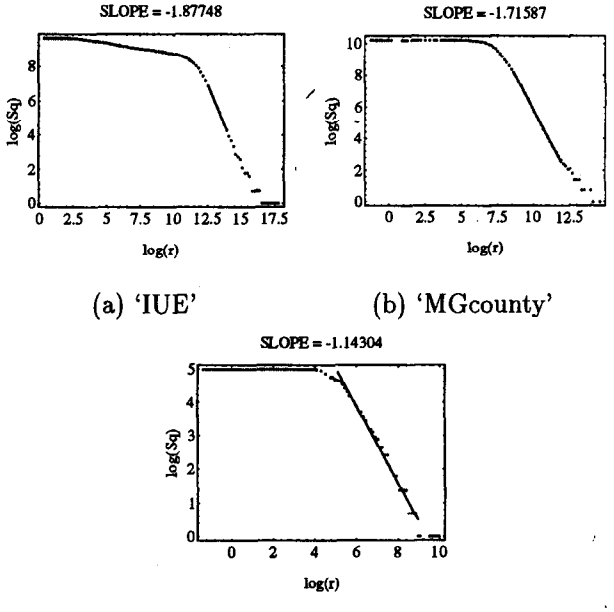
Figures 7(a-e) compare the actual values for the number of leaf nodes depicted as 'bullets' with the analytically predicted values from Theorem 1 (solid



(a) 'IUE' (b) 'MGcounty'

(c) 'LBcounty'

Figure 5: *Three real datasets: (a) star coordinates from NASA (b) crossroads from Montgomery county, Maryland (c) crossroads from Long Beach county, California.*



(a) 'IUE' (b) 'MGcounty'

(c) 'Franconia'

Figure 6: *Box-count plots for the computation of the Hausdorff fractal dimension D_H for selected datasets*

line). The horizontal axis is the level k of the quadtree decomposition; the vertical axis is logarithmic ($\log_2$ specifically) for reasons of presentation.

Notice that the accuracy of the formula is very good for all the datasets. However, this observation is not restricted to the results presented, but also holds for other datasets. According to our experiments, it should further be noticed that our observations also

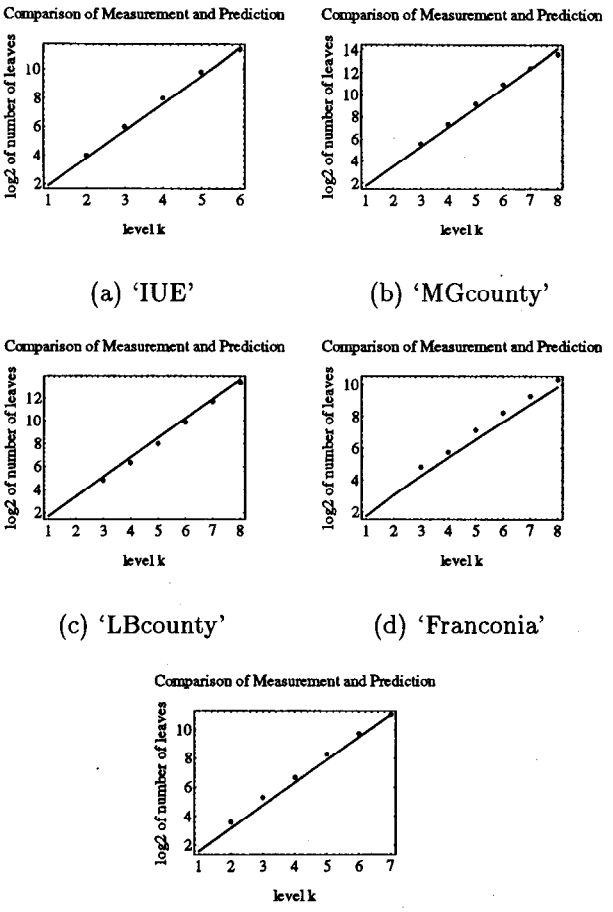


Figure 7: *Accuracy of prediction for the (logarithm of) number of blocks $N_b(k)$, as a function of the level k. Actual values ('bullets'); analytical predictions (solid line).*

apply to more complex datasets composed of objects having strongly varying complexities. The fractal dimension D_H captures the complexity of the datasets.

5 Discussion

Here we discuss the following issues: (a) how do older quadtree analyses compare with our result? (b) how often should we expect to encounter fractal datasets? (c) how would a practitioner benefit from our result?

5.1 Older analyses

Our formula agrees with previous analyses, or even includes them as special cases: Steiglitz and Hunter [HS79] proved that the number of quadtree leaf nodes for a 2-d polygon is proportional to its perimeter. Thus, they showed that it is the boundary that plays crucial role - our formula goes even further, generalizing the result for arbitrary (self-similar) E-d spatial objects, and showing that the major parameter is the 'ruggedness' of the boundary, as measured by the fractal dimension D_H. In [FJM94] we showed that the number of quadtree nodes for an E-d hyper-rectangle is proportional to its hyper-surface: Thus, the number of quadtree blocks will follow a power law with exponent $E-1$, because the hyper-surface is a manifold of dimensionality $E-1$, both traditional, as well as fractal. Again, this agrees with our formula, which predicts that for a given level-i decomposition, the number of z-values will grow exponentially, with exponent $D_H = E-1$.

As mentioned earlier, in [Gae95b], we showed experimentally that the number of z-values follows a power law, for several 2-d datasets ($E=2$). That is,

$$N_b(k) \propto (D_z)^{2k} = 2^{k(2\log_2 D_z)} \tag{16}$$

Compared with our formula

$$N_b(k) \propto 2^{kD_H}$$

we would have a perfect agreement, if only

$$D_H = 2 \cdot \log_2(D_z)$$

and in general if

$$D_H = E \cdot \log_2(D_z) \tag{17}$$

Table 3 exactly illustrates that this agreement holds, for our experimental datasets. For each dataset, we computed the slope of the regression line, which corresponds to right-hand-side of Eq. 17, i.e., $E \cdot \log_2(D_z)$. Table 3 lists the slope of the regression line (column 2), and the Hausdorff fractal dimension (column 3) of the corresponding dataset. For each dataset that the 'level-independence' assumption holds, the two numbers should be very close. Notice that this is indeed the case; the difference is typically in the third significant digit.

This result is particularly interesting, since it readily allows us to use the analytical model presented in [Gae95b] for determining a good choice of the granularity. In this paper it has been shown experimentally as well as analytically that the performance of the spatial access method z-ordering is sensitive to the chosen granularity.

Finally, in [ACF+94] we observed experimentally that the number of oct-tree blocks for human MRI brain scans was $N_b(k) \propto 2^{2.63k}$. Notice that 2.63 is close to 2.73-2.79, which is the range of the typical fractal dimension of the surface of mammalian brains [Man77].

Data set	actual slope of $\log_2(N_b(k))$ vs k	box-counting Hausdorff frac. dim. D_H
Synthetic dataset		
Sierpinski5K	1.586	1.58
Real datasets		
IUE	1.90	1.88
MGcounty	1.76	1.71
LBcounty	1.72	1.70
Franconia	1.11	1.14

Table 3: *Accuracy of the exponent in our power law: The second column is the slope of the regression line of the graph* $\log_2(N_b(k))$ *versus the level* k. *The third column is the Hausdorff fractal dimension* D_H *of the boundary, using the box counting method.*

5.2 Popularity of fractal datasets

Real datasets seem to be self-similar more often than not: The literature on fractals [Man77, Sch91] provides a long list of real, self-similar structures, including coastlines and country borders (with D_H typically 1.1-1.3); periphery of clouds and rainfall patches ($D_H \approx 1.35$); surface of mammalian brain ($\approx$2.73-2.79); human pulmonary system ($D_H \approx 2.9$); stock-price plots over time ($D_H \approx 1.5$).

5.3 Practical Considerations

The question is 'how would a practitioner use the above results?' Given a data-set (set of points, or a region), the first step is to estimate the fractal dimension D_H. This can be done with an $O(N \log N)$ algorithm [BF95], or even by consulting the literature on fractals, for the typical D_H of the dataset of interest. For example, if our application focuses on 3-d human brain scans, a crude estimate of D_H would be 2.7. The second step is the estimation of the proportionality constant C_1, for which we need an estimate of the total hyper-volume (= black area) A_b. Such an estimate should be easy to obtain, or to approximate: For example, if the minimum bounding rectangle (MBR) of the object of interest is known, we can use the volume of the MBR as a estimate of A_b or the volume of an enclosing polytop.

Thus, a practitioner could have accurate estimates for the number of quadtree blocks that a spatial object will require. This is useful in at least two settings: (a) if the object is a data object, we need to estimate the number of quadtree blocks for a given (exact or approximate) decomposition; this is needed to estimate the space overhead of the resulting B^+-tree index, where each z-value yields a different record (see Table 2.2) (b) for query optimization: given a range query, our formula can predict the number of quadtree blocks it will decompose into, and therefore the number of 'probes' ($\approx$ leaf accesses $\approx$ random disk accesses) that we will have to do in the B^+-tree index. This assumption is justified, since in practice the number of probes is correlated with the number of dik accesses.

As for the most common quadtree decomposition strategies, we notice that our analysis covers for the error-bound strategy.

6 Conclusions

We have derived a closed formula that estimates the number $N_b(k)$ of quadtree blocks for an E- dimensional object that has the 'level independence' property. Our approach unified three observations, which have been made independently:

- the 'level independence' assumption, which is the basis of statistical models for quadtrees [VM96, SS85],
- the theory of fractals, which claims that spatial objects of the real world are often self-similar, and have a non-integer 'Hausdorff' fractal dimension [Man77, Sch91] and
- empirical observations that a power law holds for the number of quadtree blocks for 2-d and 3-d data [Gae95b, ACF+94]

We showed that the 'level independence' assumption *implies* a power law for the number of quadtree nodes, for any dimensionality E of the address space. Moreover, we showed that the exponent of the power law is the Hausdorff fractal dimension of the boundary of the object. Additional, smaller contributions are

- the estimation of the constant of proportionality C_1, which is typically neglected in the literature of fractals;
- the verification of the accuracy of the formula, on several, real data sets. Thus, our formula can help with the estimation of the space requirements for a linear quadtree representation of a dataset, as well as with the selectivity estimation and query optimization for geographic and, in general, spatial databases.

Future work includes the use of fractal concepts for the analysis of other quadtree/z-ordering algorithms, like 'nearest-neighbor' queries and 'spatial joins'.

Acknowledgements

The authors would like to thank Alberto Belussi for his package that computes the fractal dimension.

References

[ACF+94] Manish Arya, William Cody, Christos Faloutsos, Joel Richardson, and Arthur Toga Qbism: Extending a dbms to support 3d medical images. *Tenth Int. Conf. on Data Engineering (ICDE)*, pages 314–325, February 1994.

[BB82] D. Ballard and C. Brown. *Computer Vision*. Prentice Hall, 1982.

[BF95] Alberto Belussi and Christos Faloutsos. Estimating the selectivity of spatial queries using the 'correlation' fractal dimension. *Proc. of VLDB*, pages 299–310, September 1995.

[BKSS90] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An efficient and robust access method for points and recta ngles. In *Proc. ACM SIGMOD Conference on Management of Data*, pages 322–331, 1990.

[Dye82] C.R. Dyer. The space efficiency of quadtrees. *Computer Graphics and Image Processing*, 19(4):335–348, August 1982.

[Fal92] C. Faloutsos. Analytical results on the quadtree decomposition of arbitrary rectangles. *Pattern Recognition Letters*, 13(1):31–40, January 1992.

[FJM94] Christos Faloutsos, H.V. Jagadish, and Yannis Manolopoulos. Analysis of the n-dimensional quadtree decomposition for arbitrary hyper-rectangles. CS-TR-3381, UMIACS-TR-94-130, Dept. of Computer Science, Univ. of Maryland, College Park, MD, December 1994. to appear in IEEE TKDE.

[FK94] Christos Faloutsos and Ibrahim Kamel. Beyond Uniformity and Irndependence: Analysis of R-trees Using the Concept of Fractal Dimension. *Proc. ACM SIGACT-SIGMOD-SIGART PODS*, pages 4–13, May 1994. Also available as CS-TR-3198, UMIACS-TR-93-130.

[FRM94] Christos Faloutsos, M. Ranganathan, and Yannis Manolopoulos. Fast subsequence matching in time-series databases. *Proc. ACM SIGMOD*, pages 419–429, May 1994. 'Best Paper' award; also available as CS-TR-3190, UMIACS-TR-93-131, ISR TR-93-86.

[Gae95a] V. Gaede. Geometric information makes spatial query processing more efficient. In *Proc. 3rd ACM International Workshop on Advances in Geographic Information Systems (ACM-GIS'95)*, pages 45–52, Baltimore, Maryland, USA, 1995.

[Gae95b] V. Gaede. Optimal redundancy in spatial database systems. In *Proc. 4th Int. Symp. on Spatial Databases (SSD'95)*, pages 96–116, 1995.

[Gar82] I. Gargantini. An effective way to represent quadtrees. *Comm. of ACM (CACM)*, 25(12):905–910, December 1982.

[GR94] V. Gaede and W.-F. Riekert. Spatial access methods and query processing in the object-oriented GIS GODOT. In *Proc. of the AGDM'94 Workshop*, pages 40–52, Delft, The Netherlands, 1994. Netherlands Geodetic Commission.

[Gut84] A. Guttman. R-trees: a dynamic index structure for spatial searching. *Proc. ACM SIGMOD*, pages 47–57, June 1984.

[Güt94] R. H. Güting. An introduction to spatial database systems. *VLDB Journal*, 3(4):357–399, 1994.

[HS79] G.M. Hunter and K. Steiglitz. Operations on images using quad trees. *IEEE Trans. on PAMI*, PAMI-1(2):145–153, April 1979.

[Jag91] H.V. Jagadish. A retrieval technique for similar shapes. *Proc. ACM SIGMOD Conf.*, pages 208–217, May 1991.

[Kli71] A. Klinger. Pattern and search statistics. In S. Rustagi, editor, *Optimizing Methods in Statistics*, pages 303–337. 1971.

[Man77] B. Mandelbrot. *Fractal Geometry of Nature*. W.H. Freeman, New York, 1977.

[OM84] J.A. Orenstein and T.H. Merrett. A class of data structures for associative searching. *Proc. of SIGACT-SIGMOD*, pages 181–190, April 1984.

[OM88] J.A. Orenstein and F.A. Manola. Probe spatial data modeling and query processing in an image database application. *IEEE Trans. on Software Engineering*, 14(5):611–629, May 1988.

[Ore89] J.A. Orenstein. Redundancy in spatial databases. *Proc. of ACM SIGMOD Conf.*, May 1989.

[Sch91] Manfred Schroeder. *Fractals, Chaos, Power Laws: Minutes From an Infinite Paradise.* W.H. Freeman and Company, New York, 1991.

[Sha88] C.A. Shaffer. A formula for computing the number of quadtree node fragments created by a shift. *Pattern Recognition Letters*, 7(1):45–49, January 1988.

[SRF87] T. Sellis, N. Roussopoulos, and C. Faloutsos. The r+ tree: a dynamic index for multi-dimensional objects. In *Proc. 13th International Conference on VLDB*, pages 507–518, England,, September 1987. also available as SRC-TR-87-32, UMIACS-TR-87-3, CS-TR-1795.

[SS85] Hanan Samet and Clifford A. Shaffer. A model for the analysis of neighbor finding in pointer-based quadtrees. *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)*, 7(6):717–720, 1985.

[SSN87] C.A. Shaffer, H. Samet, and R.C. Nelson. Quilt: a geographic information system based on quadtrees. Technical Report CS-TR-1885.1, Univ. of Maryland, Dept. of Computer Science, July 1987. to appear in the International Journal of Geographic Information Systems.

[VM96] Michael Vassilakopoulos and Yannis Manolopoulos. A random model for analyzing region quadtrees. *Pattern Recognition Letters*, 1996. to appear.

[Whi81] M. White. *N-Trees: Large Ordered Indexes for Multi-Dimensional Space.* Application Mathematics Research Staff, Statistical Research Division, U.S. Bureau of the Census, December 1981.