CAR–TR–241 DCR–86–05557
CS–TR–1740 December 1986

A POPULATION ANALYSIS OF QUADTREES WITH VARIABLE NODE SIZE

Randal Nelson
Hanan Samet

Department of Computer Science and
Center for Automation Research
University of Maryland
College Park, Maryland 20742 USA

ABSTRACT

A new method termed population analysis is presented for approximating the distribution of node occupancies in hierarchical data structures which store a variable number of geometric data items per node. The basic idea is to describe a dynamic data structure as a set of populations which are permitted to transform into one another according to certain rules. The transformation rules are used to obtain a set of equations describing a population distribution which is stable under insertion of additional information into the structure. These equations can then be solved, either analytically or numerically, to obtain the population distribution. This technique is used to model hierarchical data structures with variable node capacity as a set of interacting populations where each population represents the nodes of a given occupancy. Quadtree data structures for storing point and line data are analyzed in detail, and the results are compared to experimental data. Two phenomena referred to as *aging* and *phasing* are defined and shown to account for the differences between the experimental results and those predicted by the model. The population technique is compared with statistical methods of analyzing similar data structures.

Key words and phrases: analysis of algorithms, bucketing methods, multidimensional attributes, hierarchical data structures, line representations, quadtrees.

The support of the National Science Foundation under Grant DCR–86–05557 is gratefully acknowledged.

I. Introduction

A large group of data structures employ a representational scheme which can be applied at different spatial resolutions to allow the structure to be adapted to the data. Such representations are sometimes referred to as hierarchical data structures because the different levels of resolution can be considered to form a hierarchy. The group includes structures such as quadtree and octree varieties [Same84a], bintrees [Same84c], grid files [Niev84], and also techniques which are less explicitly based on spatial decomposition such as extendible hashing [Fagi79]. However, all these structures are variable resolution representations which have locally similar structure at different resolutions.

Because of their adaptive nature, the structure of these representations depends on the data being represented. This makes it difficult to analyze their performance. Traditional worst-case analysis is often inappropriate because the worst case tends to be both very bad and highly improbable. What is generally desired is some sort of a "typical case" description of properties of interest such as required storage per data item or access time.

Most approaches to the analysis of hierarchical structures have thus been statistical in nature, most notably, Fagin *et al.* in their analysis of extendible hashing [Fagi79] which also applies to certain types of quadtrees. Regnier [Regn85] and Tamminen [Tamm83] have also published statistical analyses of grid files and a structure called EXCELL respectively. These analyses are based on the computation, relative to some model of data distribution, of statistical sums over the space of possible data structure configurations.

A major drawback of such statistical analyses is that they are complicated to perform, even for the case of uniformly distributed point data which can be treated one-dimensionally. The prospect of attempting a similar analysis for more complicated data primitives (e.g., line segments or polygons) interacting in higher dimensional spaces is discouraging. Furthermore, since such analyses depend on some model of data distribution, they will at best yield approximations to the expected values when real data is used.

The original motivation for this study was an effort to analyze the storage behavior of certain quadtree data structures which we used in the implementation of a geographic information system [Same85c]; specifically, the PMR quadtree [Nels86] which organizes line segment data using quadtree nodes of variable capacity. A straightforward statistical analysis of the structure in terms of its configuration space promised to be an exceedingly laborious method of obtaining what would be, in any case, an approximation based on a data model. Since we were interested in the distribution of the nodes as a function of their occupancy, we decided instead to model a quadtree as a collection of populations where each population represented the nodes in the quadtree having a particular occupancy. Thus the set of all empty nodes constitutes one such population, the set of all nodes containing a single line segment another, and so forth. A certain approximation is involved since the set of all nodes of a given occupancy contains nodes at many different levels in the quadtree. Since the nodes at different levels represent physical blocks of different areas, the population is not, strictly speaking, homogeneous. However, since the structure of hierarchical data structures is similar at different resolutions, it was expected that the effect of this approximation would be relatively minor.

As information is added to a quadtree with variable capacity nodes, each population grows in a manner which depends on the other populations. For instance, consider a structure in which nodes can hold up to m data elements and are split when the capacity is exceeded. In this case, the probability of an insertion producing a new node with occupancy i depends both on the fraction of the nodes with occupancy $i-1$ and on the population of full nodes (occupancy m) since nodes of any occupancy can be produced when a full node splits. Our goal is to determine a steady state, where the proportions of the various populations are constant under addition of new information according to some data model. If such a steady state exists, then it can be taken as a representative distribution of populations from which expected values for structure parameters such as average node occupancy can be calculated.

The population analysis described above may be viewed as an alternative method (*vis-a-vis* statistical sums) of defining a "typical" structure. This method has the advantage that the dependencies between various populations can be determined, and the steady state computed, with relative ease in cases where statistical analysis would be difficult. Furthermore, the method is flexible, and is adaptable to any hierarchical data structure where the decomposition is determined adaptively by the local concentration of the data. Finally, the notion of a steady state corresponds to the intuitive notion of a typical structure more closely than does a statistical average, which is defined even if the structure changes radically from insertion to insertion. Such a formalization of intuitive expectations is useful because it is frequently easier to understand the behavior of a complicated system in terms of a simple model with added corrections, than in terms of a more accurate but more complicated monolith.

The remainder of the paper is organized as follows. Section II gives a brief overview of quadtrees. Section III describes the use of the population model in detail, and illustrates its use by analyzing the simple PR quadtree for storing points. Section IV extends the analysis to the generalized PR quadtree, and compares the results with experimental data. Section V accounts for the discrepancy in the model in terms of two phenomena termed *aging* and *phasing* which are characteristic of hierarchical data structures in general. Section VI considers the asymptotic case of the PR quadtree. Section VII applies the method to the analysis of the PMR quadtree, a structure for storing line data, and compares the results with experimental data. Section VIII presents our conclusions and suggestions for further work.

II. Quadtrees

A quadtree [Same84a] is a hierarchical, variable resolution data structure based on the recursive partitioning of the plane into quadrants. It can be viewed as a 4-ary tree where each node represents a region in the plane called a block, and the sons of each node represent a partition of that region into four pieces. This scheme is useful for representing data having geometric distribution at variable resolution. Variations exist for representing planar regions [Klin71], collections of points [Fink74, Same84b], and collections of line segments [Same85b, Nels86] as well as more complicated objects (e.g., rectangles [Kede81]). Generalizations of the principle to three and higher dimensions (e.g., octrees [Hunt78, Jack80, Meag82] and bintrees [Know80, Tamm84, Same85a]) have also been investigated, and have many of the same basic properties.

Quadtrees can be divided into two types: those based on regular decomposition of space using pre-defined boundaries, and those where the partition is determined explicitly by the data as it is entered into the structure. The usual method of regular decomposition is to start with a unit square region, and partition blocks into equal quarters as additional resolution is required locally. In this case, all blocks are square, and have a side length which is an integral power of 1/2. Figure 1 shows a quadtree representation for a set of points based on regular decomposition. The region of interest has been recursively partitioned into quadrants until no quadrant contains more than a single point. This structure is known as the PR quadtree [Oren82, Same84b]. A second decomposition method has been investigated in applications where it is desired to adapt the structure closely to the data (e.g., the classical point quadtree [Fink74]). When such a method is used, the partitions are typically irregular and of odd sizes, and the shape of the final structure depends critically on the order in which the information was inserted into the tree. In exceptional cases this dependency can lead to badly balanced structures (e.g., consider the case of a binary search tree when the items are inserted in sorted order). For most applications, regular decomposition works at least as well as data-based, and is easier to implement and analyze. In this paper, only structures utilizing regular decomposition are considered.

The condition used to determine when a quadtree block should be partitioned is called the *splitting rule*. This rule is usually a function of the data stored in the corresponding node, thus making it easy to evaluate using local information. The form of the rule depends on the type of data being stored. For instance, if a quadtree is being used to store a collection of of points, one possible rule is "split until no block contains more than one distinct point". This is reasonable since any pair of points can eventually be separated by a sufficiently fine dissection. The structure produced by the use of this rule is the PR quadtree described above. On the other hand, if a quadtree is used to store line segments, then some other condition should be used, since intersecting segments cannot be separated no matter how far the decomposition is carried. In this case, a rule of the form "split once if an inserted segment intersects a block already containing one or more segments" is appropriate. This procedure will eventually separate disjoint segments, and isolate the intersection points of segments that intersect. It requires the capability to store an arbitrary quantity of information at each node; however, the average node occupancy remains low. The structure produced by this rule is known as a PMR quadtree [Nels86]. These data structures are described fully in the referenced papers.

The PR and the PMR quadtrees described above can be generalized by permitting the nodes to contain more than one data item before splitting. The rule for the generalized PR quadtree then becomes "split until no block contains more than m points". This principle is similar to that used by Tamminen in his EXCELL system [Tamm81] and by Nievergelt *et al.* in the grid file [Niev84]. Analogously, for the generalized PMR quadtree, the rule becomes "split once if an inserted segment intersects a block already containing m or more segments". The generalization of the point representation is of interest because it is possible to decrease the number of nodes by allowing several points to be stored per node. If access to a node is slow in comparison with the time needed to examine the contents, (e.g., if accessing a node requires one or more page faults), then storing several points per node can lead to a considerable reduction in the cost of operations such as region queries which involve local search. The same principle applies in the case of line data. Furthermore, in the line representation, permitting several lines to occupy a node before its block is split reduces the amount of splitting required to isolate intersection points, and hence reduces the amount of storage necessary to store the data. This strategy is particularly useful if there is some bound on the number of segments likely to intersect at a point (e.g., three for a Dirichlet tesselation or four for a collection

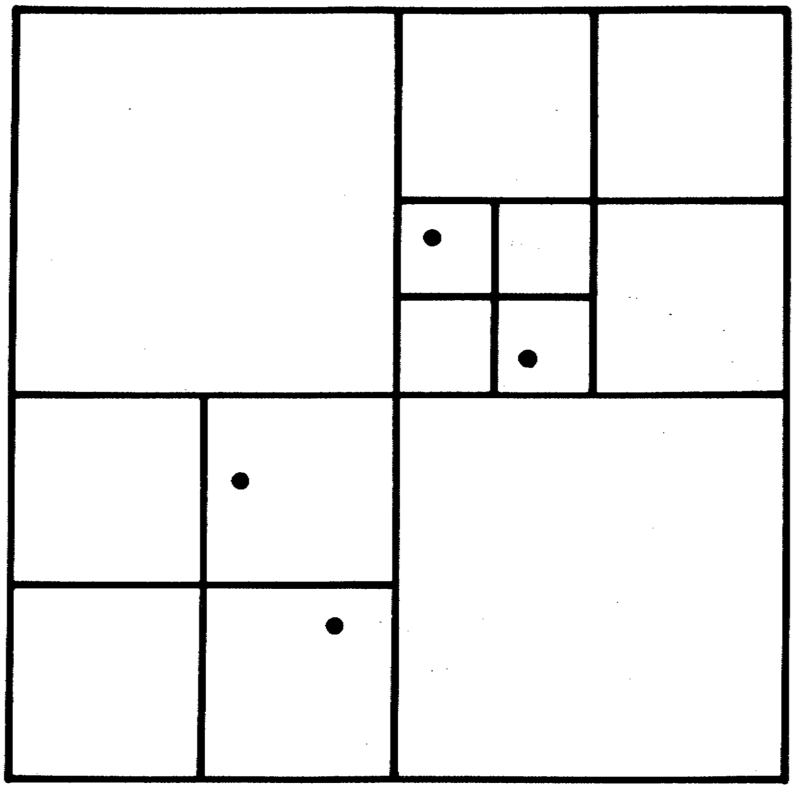

Figure 1. PR quadtree for four points. Blocks are recursively quartered until no block contains more than one point.

of line segments representing a network of roads) since the node capacity can be set to that bound.

For the purposes of predicting the expense of processing information stored in data structures such as those described above, and for determining the optimal node capacity for a particular application, it is desirable to be able to predict the storage characteristics for different node capacities. For example, the obvious method of determining all intersections of a set of line segments stored in a PMR quadtree requires time proportional both to the total number of nodes and to the square of the average node occupancy. Since larger node capacities generally result in a smaller total node count, there is a tradeoff involving the node capacity that must be considered.

In this paper we describe a framework for predicting the distribution of node occupancies based on a steady state model of the populations in a dynamic quadtree, and use it to compute population distributions for some instances of the generalized PM and PMR quadtrees described above. The results are compared with distributions in actual quadtrees, and are found to agree overall, within approximately 10 percent. We show that the discrepancy is attributable to a pair of properties termed *aging* and *phasing*, which are typical of hierarchical data structures in general. Thus the population model provides both a means of obtaining useful quantitative predictions about the expected storage properties, and a framework for understanding some of the more complex behavior of quadtrees and similar hierarchical data structures.

III. Computation of the expected distribution

Consider a quadtree data structure whose leaf nodes each contain between 0 and m data items which are members of some set A (e.g., the PR quadtree for points). The number of data items stored in a node is the occupancy of the node. We can describe the distribution of node occupancies in a particular quadtree Q by a state vector $\vec{d} = (p_0, p_1, \cdots, p_m)$ where p_i is the proportion of the nodes having occupancy i. As a consequence of this definition $p_0+p_1+\cdots+p_m = 1$. We would like to define an average or typical state vector, say $\vec{e}$, for the data structure which could be used to predict storage properties with some degree of accuracy. We will call this vector the expected distribution. The following discussion makes specific reference to quadtrees, but the same principles apply in the case of octrees and higher dimensional data structures.

A typical statistical approach to defining $\vec{e}$ would be the following. Let $\vec{d}_i$ be the average value of the state vector $\vec{d}$ over all quadtrees which represent a subset of A containing i elements. Generally, $\vec{d}_i$ is defined relative to some model of data distribution. Define the expected distribution $\vec{e}$ as the limit of the sequence $\vec{d}_1$, $\vec{d}_2, \cdots$. If $\vec{e}$ exists and can be calculated, then it could serve as a representative distribution for quadtrees of a given type. Fagin *et al.* have used a statistical approach to calculate average node occupancies as a function of the number of data points in the context of extendible hashing [Fagi79]. The application of their results, with slight modifications, to the PR quadtree, indicates that the limit $\vec{e}$ does not exist. Specifically, the vector sequence $\vec{d}_1$, $\vec{d}_2, \cdots$ undergoes oscillatory behavior of increasing period and non-decreasing amplitude. We will show in Section V that this sort of oscillatory behavior, which we refer to as *phasing*, is typical of hierarchical data structures in general when a uniform data distribution is present.

The above statistical calculation represents a considerable mathematical effort. Furthermore, it cannot be easily generalized to hierarchical representations for other data primitives (e.g., line segments). Calculating the vectors $\vec{d}$ directly seems to be difficult in general, especially if the data primitives are non-trivial. For these reasons, we decided to pursue an alternative definition for a typical distribution.

We consider a quadtree as a set of populations of nodes where each population consists of all nodes having a given occupancy. Thus empty nodes form one population, nodes containing one point a second, and so forth. Insertion of a point into a node of occupancy i either transforms it into a node with occupancy $i+1$ or else causes the node to split, increasing several populations. The expected distribution $\vec{e}$ is defined by the condition that the proportion of each population making up the structure remains unchanged if data is added to the tree in accordance with statistical expectations, i.e., $\vec{e}$ is a fixed point under the operation of insertion. This condition can be used to determine $\vec{e}$ if the data distribution and the statistical results of adding a datum to a node can be calculated. The key difference between our method and the statistical approach of Fagin *et al.* is that our method considers only the local probabilities for the distribution of data items in a single node into its quadrants rather than a distribution for the whole population. It is more tractable than the direct statistical approach, and yields results which are in reasonable agreement with experimental data for several quadtree data structures. We first illustrate the use of this technique by means of a specific example, and then demonstrate its application to generalized PR and PMR quadtrees.

Recall the simple PR quadtree described above. Every node contains either zero or one data points. There are thus two distinct populations. We refer to these as types n_0 and n_1 respectively. If a point is added to a node of type n_0, that node is transformed into a single node of type n_1. On the other hand, if a point is added to a node of type n_1, then the block must be split, perhaps several times, until the two points lie in separate blocks. In this case, several nodes, of both types, are generated. For any node type, the average result of adding a point to the node can be described by a transform vector $\vec{t}=(t_0,t_1)$ where t_0 is the average number of nodes of type n_0 produced by the insertion of a point, and t_1 is the average number of n_1 nodes. Let $\vec{t_i}$ be the transform vector describing the results of adding a point to a node of type n_i. The vectors $\vec{t_i}$ form the rows of a matrix **T** called the transform matrix. We already know that $\vec{t_0} = (0,1)$. To determine $\vec{t_1}$, we use the geometry of the situation to write a recurrence relation. If the distribution of data points is uniform, then in 3/4 of the cases, a single split will suffice, dividing the block into four quadrants, two of which are empty, and two of which contain a single point. In 1/4 of the cases, both points will fall in the same quadrant which must be split again under the same conditions as the original split. This allows us to write the recurrence relation

$$\vec{t_1} = \frac{3}{4}\,(2,2) + \frac{1}{4}\,(3,0) + \frac{1}{4}\vec{t_1}.$$

Solving this vector equation gives $\vec{t_1} = (3,2)$.

The transform matrix **T** is thus given by

$$\mathbf{T} \;=\; \begin{matrix} \mathrm{T}_{00} & \mathrm{T}_{01} \\ \mathrm{T}_{10} & \mathrm{T}_{11} \end{matrix} \;=\; \begin{matrix} 0 & 1 \\ \frac{9}{3} & \frac{6}{3} \end{matrix}$$

Note that the rows of the transform matrix describe the results of the transformation of a node of a given type, while the columns describe all transformations which can produce a node of a given type.

If we now assume that the probability of a data point being inserted into a node of a given occupancy is proportional to the numerical fraction of nodes of that type in the tree, then we can use the above results to write equations which $\vec{e}$ must satisfy. Note that this assumption is equivalent to the assumption that the distribution of node occupancies is independent of the geometric size of the corresponding block – i.e., that nodes at depth n in the tree do not have different occupancy distributions than those at level $n+1$. Empirical studies indicate that this is not strictly true – larger nodes tend to have slightly higher average occupancies. We refer to this phenomenon as *aging*, and will examine it in more detail later. However, for PR quadtrees, the approximation is close enough to be useful.

Under the above assumption, the insertion of new data points into a PR quadtree will transform nodes of types n_0 and n_1 with relative frequency e_0 and e_1 respectively (recall that $\vec{e}$ is the expected distribution). The number of nodes in each population which receive new points and thus are transformed is proportional to the size of that population. Consequently, the population distribution of the remaining, untransformed nodes remains $\vec{e}$. Thus in order for $\vec{e}$ to be a fixed point, the distribution of node types produced from the transformed nodes must also be $\vec{e}$. This allows us to formulate equations which can be solved for $\vec{e}$ as follows. Suppose that the number of data points is increased by Δn. In this case, the expected number of new nodes of each type can be calculated from the distribution of node types in the tree (assumed to be $\vec{e}$) and the transform vectors (represented by the matrix $\mathbf{T}$). Specifically, the expected number of new n_0 nodes is $\mathbf{T}_{00}e_0\Delta n + \mathbf{T}_{10}e_1\Delta n$, and the expected number of new n_1 nodes is $\mathbf{T}_{01}e_0\Delta n + \mathbf{T}_{11}e_1\Delta n$. The expected total number of new nodes is the sum of these two quantities. Requiring the proportion of new n_0 nodes to be e_0 gives

$$\frac{\mathbf{T}_{00}e_0+\mathbf{T}_{10}e_1}{\mathbf{T}_{00}e_0+\mathbf{T}_{10}e_1+\mathbf{T}_{01}e_0+\mathbf{T}_{11}e_1} = e_0.$$

Similarly, requiring the proportion of new n_1 nodes to be e_1 gives

$$\frac{\mathbf{T}_{01}e_0+\mathbf{T}_{11}e_1}{\mathbf{T}_{00}e_0+\mathbf{T}_{10}e_1+\mathbf{T}_{01}e_0+\mathbf{T}_{11}e_1} = e_1.$$

These equations can be concisely expressed in matrix notation by the formula

$$\vec{e}\mathbf{T} = a\vec{e} \tag{1}$$

where a is the scalar $\mathbf{T}_{00}e_0+\mathbf{T}_{10}e_1+\mathbf{T}_{01}e_0+\mathbf{T}_{11}e_1$. Note that since a is a function of $\vec{e}$, (1) does not represent a set of linear equations, but rather a set of quadratic equations involving the components of $\vec{e}$. This particular example can be solved analytically to yield $\vec{e}=(1/2,1/2)$, the only positive solution. Thus a PR quadtree with a maximum of one point per node, should have approximately equal numbers of full and empty nodes. This agrees fairly well with experiments in storing random points in PR quadtrees where we found approximately 53% empty and 47% full nodes. The particular causes of discrepancy are examined later.

IV. Analysis of generalized PR quadtrees

The above technique can be used to analyze the generalized PR quadtree where a node may contain up to m points. The integer m is known as the node capacity. The expected distribution $\vec{e}$ has $m+1$ components, there are $m+1$ node types n_0 through n_m, and there are $m+1$ transform vectors $\vec{t}_0$ through $\vec{t}_m$ which form the rows of the $m+1 \times m+1$ transform matrix $\mathbf{T}^m$. The transform vectors $\vec{t}_0$ through $\vec{t}_{m-1}$ are simple because the new data point is just added to the node without causing a split, so that a node of type n_i becomes a node of type n_{i+1}. The vectors $\vec{t}_0$ through $\vec{t}_{m-1}$ thus have the form

$$\vec{t}_i = (0,\ldots,0,1,0,\ldots,0).$$

where there are $m+1$ components and 1 is in the $i+1^{st}$ position.

The vector $\vec{t}_m$ which represents the splitting of a node into four quarters when its capacity is exceeded is found by determining the expected distribution of the points into the quadrants, and using this information to write a recurrence relation for $\vec{t}_m$ as illustrated above in the case of the PR quadtree for $m=1$.

As another example, consider the case of $m=2$, i.e., nodes are split when their occupancy reaches 3. Splitting a node containing 3 points once will produce one n_0 node and three n_1 nodes with probability 3/8; two n_0 nodes, one n_1 node, and one n_2 node with probability 9/16; and three n_0 nodes and one n_3 node with probability 1/16. The last case supplies the recurrence since it must be split again. Thus when $m=2$, $\vec{t}_2$ satisfies

$$\vec{t}_2 = \frac{3}{8}(1,3,0) + \frac{9}{16}(2,1,1) + \frac{1}{16}(3,0,0) + \frac{1}{16}\vec{t}_2$$

which can be solved to obtain $\vec{t}_2 = (27/15,\ 27/15,\ 9/15)$. Note that for m greater than about 4, the effect of the recurrence term is effectively zero since its coefficient is equal to four times the probability of $m+1$ points falling into the same quadrant, i.e., 4^{-m}.

In general, the expected distribution of $m+1$ points into quadrants is just the binomial distribution for $m+1$ objects placed independently into four buckets. The expected number of buckets containing i items, P_i, is thus given by

$$P_i = \binom{m+1}{i} \frac{3^{m+1-i}}{4^m}.$$

The term P_{m+1} represents the case where all $m+1$ points end up in the same quadrant so that recursive splitting occurs. A recursive relation for $\vec{t}_m$ can thus be written

$$\vec{t}_m = (P_0, P_1, \ldots, P_m) + P_{m+1}\vec{t}_m.$$

The value of P_{m+1} is 4^{-m} which is a common factor in all the factors P_i. By multiplying both sides by 4^{-m}, subtracting $\vec{t}_m$ and dividing by $4^{-m}-1$ we obtain the following expression for the components $\mathbf{T}_{mi}$ of $\vec{t}_m$.

$$\mathbf{T}_{mi} = \binom{m+1}{i} \frac{3^{m+1-i}}{4^m - 1}.$$

Note that for m larger than three or four, the probability of splitting more than once is negligible, and $\mathbf{T}_{mi}$ is closely approximated by P_i.

The transform matrices for the first few value of m are as follows:

$$\mathbf{T}^1 = \begin{matrix} 0 & 1 \\ \frac{9}{3} & \frac{6}{3} \end{matrix} \quad \mathbf{T}^2 = \begin{matrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ \frac{27}{15} & \frac{27}{15} & \frac{9}{15} \end{matrix} \quad \mathbf{T}^3 = \begin{matrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{81}{63} & \frac{108}{63} & \frac{54}{63} & \frac{4}{21} \end{matrix}$$

This should give a general idea of the pattern in the first m rows. The last rows of the transform matrices for $1 \leq m \leq 6$ are given below. In order to show the regularity in the denominators, the fractions have not been reduced.

$$m = 1: \quad \vec{t}_1 = \left\{ \frac{9}{3}, \frac{6}{3} \right\}$$

$$m = 2: \quad \vec{t}_2 = \left\{ \frac{27}{15}, \frac{27}{15}, \frac{9}{15} \right\}$$

$$m = 3: \quad \vec{t}_3 = \left\{ \frac{81}{63}, \frac{108}{63}, \frac{54}{63}, \frac{12}{63} \right\}$$

$$m = 4: \quad \vec{t}_4 = \left\{ \frac{243}{255}, \frac{405}{255}, \frac{270}{255}, \frac{90}{255}, \frac{15}{255} \right\}$$

$$m = 5: \quad \vec{t}_5 = \left\{ \frac{729}{1023}, \frac{1458}{1023}, \frac{1215}{1023}, \frac{540}{1023}, \frac{135}{1023}, \frac{18}{1023} \right\}$$

$$m = 6: \quad \vec{t}_6 = \left\{ \frac{2187}{4095}, \frac{5103}{4095}, \frac{5103}{4095}, \frac{2835}{4095}, \frac{945}{4095}, \frac{189}{4095}, \frac{21}{4095} \right\}$$

As in the $m=1$ case, a set of equations for the components of $\vec{e}$ can be expressed in terms of the transform matrix $\mathbf{T}$.

$$\vec{e}\mathbf{T} = a\vec{e} \tag{1}$$

The scalar a is the normalizing factor in computing the proportions and is given by

$$a = \sum_{i=0}^{m} \sum_{j=0}^{m} \mathbf{T}_{ij}\, e_i$$

that is, the sum of the coefficients of $\vec{e}$ multiplied by the sum of the components of the transform matrix in the corresponding row. The row sums of the transform matrix represent the number of nodes produced by a node of type n_i upon absorbing an additional point, and hence all are equal to unity with the exception of row m whose sum is

$(4^{m+1}-1)/(4^m-1)$ (i.e., slightly greater than four). Thus a can be written

$$a = e_0 + e_1 + \cdots + \frac{4^{m+1}-1}{4^m-1} e_m$$

The matrix equation (1) represents a set of $m+1$ quadratic equations for the components of $\vec{e}$. In general, such a set of equations can have up to 2^{m+1} solution vectors; however, since the components of $\vec{e}$ represent proportions, we are interested only in solutions for which all the components are positive. It can be shown that for sets of equations of the above form, at most one positive solution is possible. To see this, consider the general form of the individual equations. For every $0<i<m$ there is an equation of the form

$$ae_i = \mathbf{T}_{i-1i}\, e_{i-1} + \mathbf{T}_{mi}\, e_m \qquad (2)$$

These equations correspond to columns in the matrix $\mathbf{T}$. Expanding a and grouping similar powers of e_i gives

$$e_i^2 + Be_i + C = 0$$

Where B is the part of a which does not involve e_i, and C is the negative of the right side of (2). This is just a quadratic equation in e_i, and thus e_i must satisfy

$$e_i = \frac{-B \pm\sqrt{B^2-4C}}{2}$$

Now, if all the components of $\vec{e}$ are positive, then B must be positive. In this case, only one of the solutions to the quadratic formula can possibly be positive, hence there is at most one positive value for e_i which satisfies the system. Similar expressions hold for e_0 and e_m, and thus, since there is at most one possible positive value for each of the components, there is at most one solution for $\vec{e}$ in which all the components are positive. We are thus free to solve the equations numerically, with the assurance that any positive solution we find will be appropriate.

The equations were set up and solved for PR quadtrees with node capacities m ranging between one and eight points. For each node capacity $1 \le m \le 8$, the transform matrix $\mathbf{T}$ was used to obtain a system of equations describing the expected distribution $\vec{e}$. The systems were solved numerically using an iterative technique which converged on the positive solution. Experimental data was collected by constructing ten quadtrees of 1000 random points for each case and averaging the results. Corresponding data points from different trees were typically within about 10% of each other. Below, the theoretical and experimental values are compared. In each case, the theoretical result is given first.

Expected distribution in PR quadtrees – theoretical (*thy*) and experimental (*exp*)

$m = 1$ *thy*: $\vec{e} = (.500\ ,\ .500)$

exp: $\vec{e} = (.536\ ,\ .464)$

$m = 2$ *thy*: $\vec{e} = (.278\ ,\ .418\ ,\ .304)$

$exp: \vec{e} = (.326 \, , .427 \, , .247)$

$m = 3 \; thy: \vec{e} = (.165 \, , .320 \, , .305 \, , .210)$

$exp: \vec{e} = (.213 \, , .364 \, , .273 \, , .149)$

$m = 4 \; thy: \vec{e} = (.102 \, , .239 \, , .276 \, , .225 \, , .158)$

$exp: \vec{e} = (.139 \, , .293 \, , .264 \, , .184 \, , .120)$

$m = 5 \; thy: \vec{e} = (.065 \, , .179 \, , .238 \, , .220 \, , .172 \, , .126)$

$exp: \vec{e} = (.084 \, , .217 \, , .241 \, , .204 \, , .151 \, , .104)$

$m = 6 \; thy: \vec{e} = (.043 \, , .132 \, , .200 \, , .207 \, , .176 \, , .137 \, , .105)$

$exp: \vec{e} = (.050 \, , .150 \, , .201 \, , .215 \, , .176 \, , .127 \, , .081)$

$m = 7 \; thy: \vec{e} = (.028 \, , .098 \, , .165 \, , .189 \, , .173 \, , .143 \, , .114 \, , .090)$

$exp: \vec{e} = (.034 \, , .110 \, , .177 \, , .214 \, , .187 \, , .143 \, , .091 \, , .044)$

$m = 8 \; thy: \vec{e} = (.019 \, , .073 \, , .135 \, , .168 \, , .166 \, , .145 \, , .119 \, , .097 \, , .078)$

$exp: \vec{e} = (.024 \, , .086 \, , .151 \, , .206 \, , .194 \, , .156 \, , .100 \, , .049 \, , .034)$

Experiment and theory agree fairly well as to the general form of the expected distribution $\vec{e}$. Both show, for all node capacities m, a distribution which has a small value for low occupancies, rises to a peak, and decreases again for high occupancies. The values for individual components of $\vec{e}$ are in rough agreement, but in isolated cases, differ by as much as 20% of the peak value.

A quantity which conveniently summarizes the information contained in $\vec{e}$ for many practical applications is the average node occupancy. This value is calculated from $\vec{e}$ by adding e_1 to twice e_2 and so forth, i.e., the dot product of $\vec{e}$ with the vector $(0,1,2,\ldots,m)$. A good general idea of the accuracy of the theoretical model is obtained by comparing, for each m, the average node occupancy predicted by the model with that observed in the experiments. These values are tabulated in Table 1. The agreement is not exact, but it is close enough to be useful, and to establish the utility of the underlying model.

Two characteristics evident in Table 1 are noteworthy. First, the theoretical occupancy predictions are slightly but uniformly higher than the experimental values. Second, the size of the discrepancy seems to have a cyclical structure. This behavior is primarily due to two phenomena, exhibited by hierarchical data structures under certain

Table 1
Average Node Occupancy

node capacity	experimental occupancy	theoretical occupancy	percent difference
1	0.46	0.50	7.2
2	0.92	1.03	10.8
3	1.36	1.56	12.9
4	1.85	2.10	11.6
5	2.44	2.63	7.4
6	3.03	3.17	4.4
7	3.44	3.72	7.5
8	3.79	4.25	10.8

circumstances, which are not taken into account in the rather simple model derived above. The explanation of these phenomena is the subject of the next section.

V. Sources of discrepancy: aging and phasing

We examine first a phenomenon referred to as *aging*. It accounts for the consistent over-estimation of the average node occupancy by the theoretical model. Recall that in the derivation of the model, an assumption was made that the probability of a point, randomly selected from a uniform distribution, falling into a node of type n_i (occupancy i) was proportional to the fraction of total nodes which were of type n_i. Since the probability of a point falling into a node of type n_i is actually proportional to the fraction of the total area occupied by nodes of type n_i, this was equivalent to the assumption that the distribution of node types within the population of nodes of area $(1/2^{2i})$ was independent of i. In fact, nodes having greater area tend to have a slightly higher average occupancy.

The higher occupancy of large nodes can be understood in two ways. If the quadtree is viewed as a static structure where a set of points is given and splitting of quadrants takes place until no block contains more than m points, then for a random, uniform distribution, the bigger nodes will tend to be better filled simply because their area is larger and the point density is (more or less) uniform. This occurs in spite of the fact that the splitting is adaptive.

Another way of viewing the same phenomenon is to consider the quadtree as a dynamic structure which has reached its present state through a history of point insertions. A particular leaf node can be considered to have a lifetime extending from the time it is created by the splitting of its parent, until it absorbs enough points to fill it to capacity and is itself split. This "filling up" process will be referred to as *aging*. Clearly, as a population of nodes representing blocks of a given area ages, the average occupancy increases. Consider a time scale defined by the rate of insertion of points, where each point insertion represents a tick of the clock. On such a scale, if the points are drawn from a uniform distribution, large nodes will, on the average, age faster than small ones since their area is greater and more points will be inserted into them in a given interval of time. Now, consider a quadtree which has populations of nodes of two or more sizes. Since nodes at any level are created by the splitting of full nodes in the previous generation, the average occupancy of a hypothetical age-zero population of nodes is the same for every generation. Since large nodes are formed before small ones, the average number of clock ticks which have passed since the creation of the node is greater for large nodes than for small ones. Combined with the fact that large nodes fill up faster than small ones, this implies that the effects of aging (i.e., increased occupancy) are always more pronounced in the large node population. Thus the average occupancy of large nodes would be expected to be higher.

Table 2 demonstrates that the relative, average occupancy of nodes does indeed decrease with decreasing block size. The data represent averages over 10 PR quadtrees of 1000 points with $m=1$. Block area is proportional to 4^{-depth}, hence the large nodes appear first in the table. The general tendency is for node occupancy to decrease with block size towards the expected value for a population created by splitting a set of full nodes. This value is given by the dot product $\vec{t}_m \cdot (0,1,...,m-1,m)$ which is .40 for $m=1$. The experimental data shows the predicted decrease towards this value which is reached at depths 7 and 8. The anomalously high value for the average occupancy at the deepest level (depth = 9) is an artifact of the implementation which truncates the tree at that depth. However, since there are only a few nodes at the maximum depth, the net effect of this perturbation on the experimental data is negligible.

If larger nodes have a higher average occupancy, then conversely, nodes with higher occupancies tend to have larger average sizes. Corrections to our theoretical model to account for the effects of aging can thus be qualitatively described as follows. Since nodes of higher occupancy are larger, they are individually more likely to be encountered when a point is inserted. Thus to maintain a steady state, the fraction of high-occupancy nodes must be less than predicted by the model, which assumed the average sizes of the nodes to be independent of the occupancy. Conversely, the fraction of low occupancy nodes must be higher. Examination of the occupancy data indicates that this correction is consistent with the observed discrepancy between the theoretical and experimental values. The proportion of high occupancy nodes observed in the experiments is uniformly lower than predicted by the model, while the proportion of low occupancy nodes (e.g., n_0) is uniformly higher. The effect of the correction on the modeled average occupancy would be to decrease it, which is also consistent with the observed

Table 2
Occupancy by node size

Depth	n_0 nodes	n_1 nodes	occupancy
4	6.6	20.1	.75
5	300.2	354.3	.54
6	533.7	411.6	.44
7	225.4	144.9	.39
8	71.5	49.6	.41
9	16.1	19.5	.55

discrepancy.

A second phenomenon referred to as *phasing* is responsible for the periodic behavior of the discrepancy between the observed and theoretical average occupancies considered as a function of node capacity m. This effect is due to the fact that for a uniform distribution of points, the nodes in the quadtree tend to stay about the same size. Moreover, all nodes of the same size tend to fill up and split at about the same time. As a quadtree is built, there will be cycles of relative activity as the uniform point density approaches a value at which nodes of a certain size split, followed by periods of inactivity as the new, smaller nodes fill up. Thus, when the points in the quadtree are drawn from a uniform distribution, the nodes will tend to split and fill in phase with a logarithmic period which repeats every time the number of points increases by a factor of four. The average occupancy will follow a similar cycle, attaining its highest value just before a group of uniformly sized nodes begins to split up, and its lowest value just after most of the nodes have been split. This effect becomes more pronounced as the node capacity increases since the probability of having a local density fluctuation which places all m points in one quadrant, thus causing splitting at more than one level, decreases exponentially with increasing m. Because the distribution of node sizes depends only on the statistical fluctuations in point density which is scale invariant for a uniform distribution, the oscillations will not damp out. Thus the limit of the sequence $\vec{d}_1, \vec{d}_2, \cdots$ mentioned in section III does not exist.

Table 3 illustrates the cyclical variation of the average occupancy with the number of points for a node capacity of 8. The data were generated by averaging the results from 10 quadtrees built from the specified number of points drawn from a uniform distribution. The sample sizes were chosen along a logarithmic scale so that the number of points in the samples quadruples over four steps. Note that relative maxima and minima are separated by factors of four (four steps) as hypothesized. The data is shown plotted on a semi-log scale in Figure 2 which illustrates the cyclical behavior more clearly.

For different node capacities, the relative maxima and minima of the average occupancy occur at different numbers of data points. Thus when the size of the data sample is fixed and the node capacity is allowed to vary, the average occupancy will be observed at different points along the cyclical curve. Since the prediction of the population model is independent of the number of data points, the discrepancy between the observed and predicted values would be expected to vary cyclicly with node capacity if the data is gathered for a fixed number of points. The smooth oscillation in the percent difference between theoretical and experimental results in Table 1 represents approximately one cycle of such a variation.

The oscillatory behavior results from the semi-synchronous splitting of nodes of a given generation when the density of a uniform distribution of points reaches a certain value. If a non-uniform distribution were used, the effect would be expected to disappear. Table 4 and Figure 3 show the results of the same experiment using a Gaussian distribution of points two standard deviations wide centered in the square region covered by the quadtree. Oscillatory behavior is observed while the number of points is relatively small, but in this case it damps out as node populations in regions of different densities get out of phase. For the case of the Gaussian, a small residual oscillation might be expected because of the large central area of near constant density.

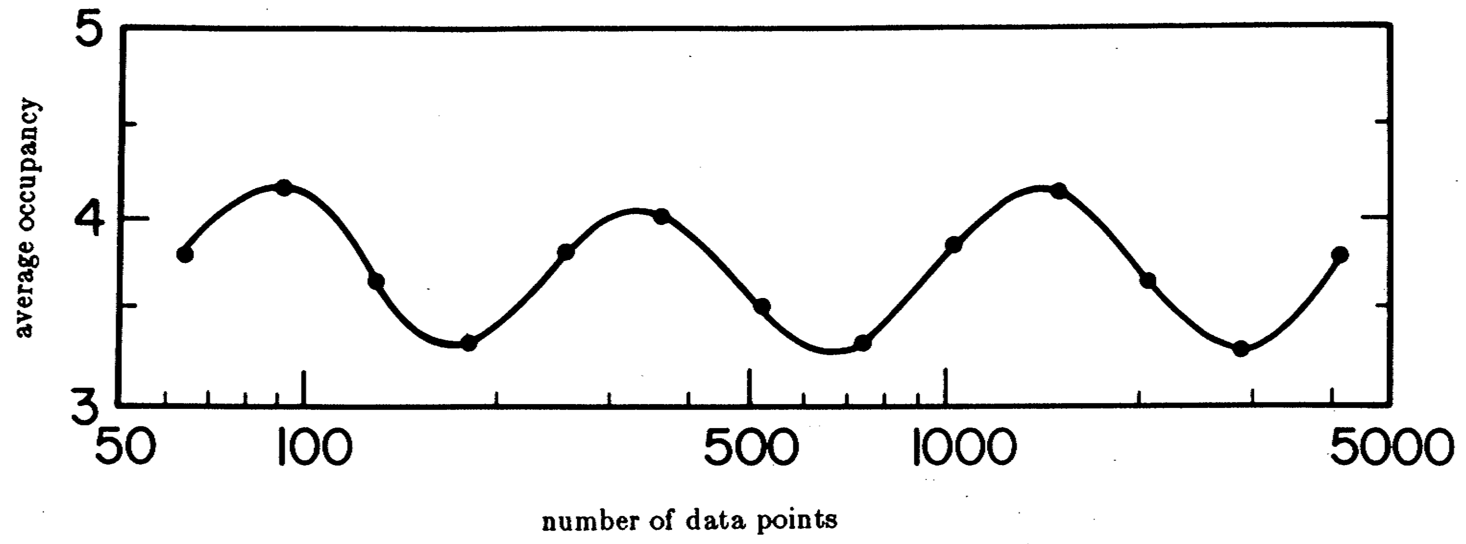

Figure 2. Experimental results and interpolated curve showing variation of average node occupancy with number of data points for uniform distribution of data.

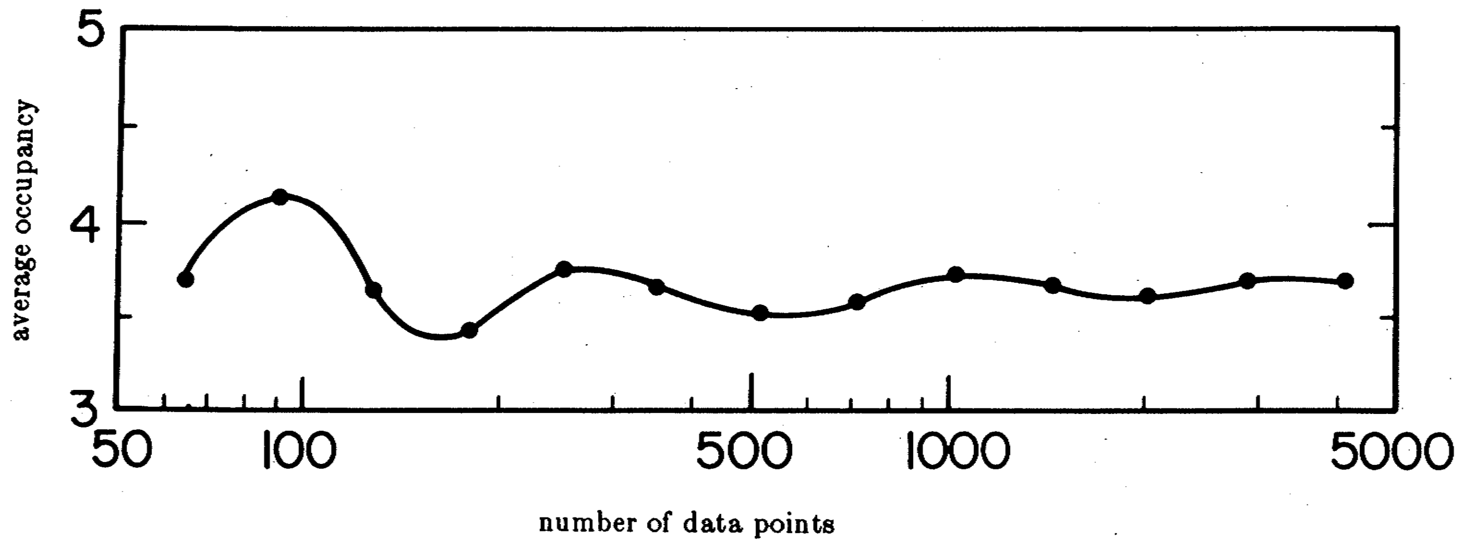

Figure 3. Experimental results and interpolated curve showing variation of average node occupancy with number of data points for Gaussian distribution of data.

Table 3
Variation of occupancy with tree size
averages for 10 trees

points	nodes	occupancy
64	16.9	3.79
90	21.7	4.15
128	35.2	3.64
181	54.4	3.33
256	67.3	3.80
362	90.7	3.99
512	145.0	3.53
724	216.4	3.35
1024	266.5	3.84
1448	350.8	4.13
2048	560.5	3.65
2896	876.6	3.30
4096	1075.6	3.81

Table 4
Variation of occupancy with tree size
Gaussian distribution

points	nodes	occupancy
64	17.2	3.72
90	21.7	4.15
128	35.2	3.63
181	52.3	3.46
256	68.2	3.75
362	99.1	3.65
512	144.1	3.55
724	203.5	3.56
1024	275.5	3.72
1448	393.4	3.68
2048	565.3	3.62
2896	784.9	3.69
4096	1104.7	3.71

The cyclicity observed in our results is the same effect predicted by Fagin *et al.* [Fagi79] in their analysis of extendible hashing, where it appears as higher terms in a Fourier series expansion. Our discussion indicates that such cyclical behavior will tend to appear in data structures based on regular spatial decomposition whenever a data model assuming uniform distribution is incorporated.

Obtaining quantitative expressions for the effects of aging and phasing is a topic for future research. However, rough bounds can be easily obtained to show that the effects are limited. For the case of PR quadtrees with a uniform data distribution, the average node occupancy must be less than m because a full node is always split into nodes whose average occupancy is less than m. On the other hand, since not every insertion causes a node to split, the average occupancy must be greater than the expected occupancy of nodes produced by splitting a full node. For $m=1$ this expected occupancy is 2/5, and for higher m it approaches $(m+1)/4$. These bounds are crude indeed for the case of the PR quadtree, for which they do little more than demonstrate that the effects of aging and phasing do not get completely out of hand. However, the same principle can be used for other structures to show that the average node occupancy is bounded, including some involving more complex data primitives for which complete statistical analysis may be impractical and the behavior non-obvious.

VI. The asymptotic case

In this section we consider the behavior of the expected distribution of occupancies in the PR quadtree as the node capacity m approaches infinity. In this limit, the possible occupancies are approximated by a continuum. Let the continuous variable x on the interval $0 \leq x \leq 1$ represent the fullness of a node. Thus $x=0$ refers to empty nodes, and $x=1$ to completely full nodes. The expected distribution can then be represented by a function f where $f(x)$ is proportional to the probability of an arbitrary node having an occupancy between x and $x + \Delta x$. For convenience, we consider $f(x)$ to be normalized so that it takes on a maximum value of 1.

Now, consider the expected occupancy of the offspring nodes when a node of occupancy m is split. The most probable occupancy is $m/4$, i.e., $x=1/4$. Moreover, as m approaches infinity, the width of the distribution about $x=1/4$ relative to the entire interval from $x=0$ to $x=1$ decreases. Thus, in the limit, full nodes at $x=1$ can be considered to split into four equal-sized nodes with occupancy $x=1/4$. One consequence of this is that $f(x)$ is essentially 0 for $x<1/4$ since it is improbable that nodes of smaller occupancy will be created.

The condition that the distribution must be stable under insertion of new data into the tree can now be used to determine f. Suppose that the number of points in the tree is increased by Δn. This produces a corresponding number of new nodes of every occupancy. In the continuous approximation the number of new nodes can be represented by a function $\Delta f(x,\Delta n) = c \cdot \Delta n \cdot [G(f_0)](x)$ where c is a constant and the functional G describes how the number of new nodes at any occupancy depends on the existing distribution f_0 of occupancies. Since full nodes split into four new nodes with occupancy 1/4, $[G(f_0)](1/4)$ must be proportional to $4f_0(1)$. To determine the behavior of G for other values of x, we must consider the discrete structure. Recall that a node of occupancy i is formed by the insertion of a point into a node of occupancy $i-1$, or by the splitting of a full node. As m becomes large, the contribution of the splitting factor becomes small for values of i greater than $m/4$. Thus for $i>m/4$ the number of nodes of occupancy i formed by the insertion of new data is proportional to the number of nodes of occupancy $i-1$ prior to insertion. Under the continuous approximation, this is equivalent to making $[G(f_0)](x)$ proportional to $f_0(x-\epsilon)$ for all $1/4<x<+1)$, where $\epsilon=1/m$ and is thus a small but non-zero constant.

Let us assume that the occupancy is stable under insertion of data. Then f_0 is f. If the occupancy is stable, then the ratio of Δf for any two values of x must be the same as the corresponding ratio for f. Hence

$$\frac{f(x_1)}{f(x_2)} = \frac{\Delta f(x_1)}{\Delta f(x_2)} = \frac{[G(f)](x_1)}{[G(f)](x_2)} = \frac{f(x_1-\epsilon)}{f(x_2-\epsilon)}$$

for any values of x_1 and x_2 in the range $1/4 < x \leq 1$. Thus f must be an exponential function of the form $f(x)=ae^{bx}$ in the range $1/4 \leq x \leq 1$.

A node which reaches occupancy $x=1$ splits into four nodes of occupancy $1/4$ when additional data is added. Therefore, there must be four times as many nodes of occupancy $x=1/4$ as there are nodes of occupancy $x=1$. This yields the boundary condition $f(1/4) = 4f(1)$. If we arbitrarily normalize f so that the maximum value, i.e., $f(1/4)$ is 1, then we can solve for the constants a and b, obtaining

$$a = \frac{3}{4}\left[\left(\frac{1}{4}\right)^{\frac{1}{3}} - \left(\frac{1}{4}\right)^{\frac{4}{3}}\right]^{-1} = 1.5874$$

$$b = \left(\frac{4}{3}\right)\ln\left(\frac{1}{4}\right) = -1.8484$$

The average occupancy O_m can be found by computing the normalized first moment of the distribution f:

$$O_m = m\int_{1/4}^{1} xf(x)dx \; / \int_{1/4}^{1} f(x)dx = .541m$$

Thus as the node capacity becomes large, the usage of the nodes under this model approaches $.541m$. This can be compared to the results obtained by Fagin *et al.*, which if adapted to the case of splitting a node into four parts, predict that the occupancy should oscillate about a value of $m/3log_4e = .463m$. This value is the result of a precise statistical computation for uniform distributions of points. The difference is due to the aging phenomenon discussed earlier. For the case of uniformly distributed points, the statistical prediction agrees with the experimental data better than our model. However, even for this simple case, the derivation of the formula is quite involved. Applying such analysis to structures which store more complicated primitives such as line segments or rectangles is difficult. Our model however, is easily adaptable to the analysis of other data structures since only the local statistics of splitting a full node need to be determined to set up the equations. In the next section we show how this is done for the case of the PMR quadtree, a data structure for storing line data.

VII. Analysis of PMR quadtrees

The PMR quadtree is a hierarchical data structure for storing sets of line segments [Nels86]. It is a variant of the PM quadtree introduced by Samet and Webber [Same85b]. The basic idea is to recursively subdivide the plane into blocks in a manner which focusses resolution where the density of line segments is highest, and then store with each block the line segments passing through it. This subdivision is performed

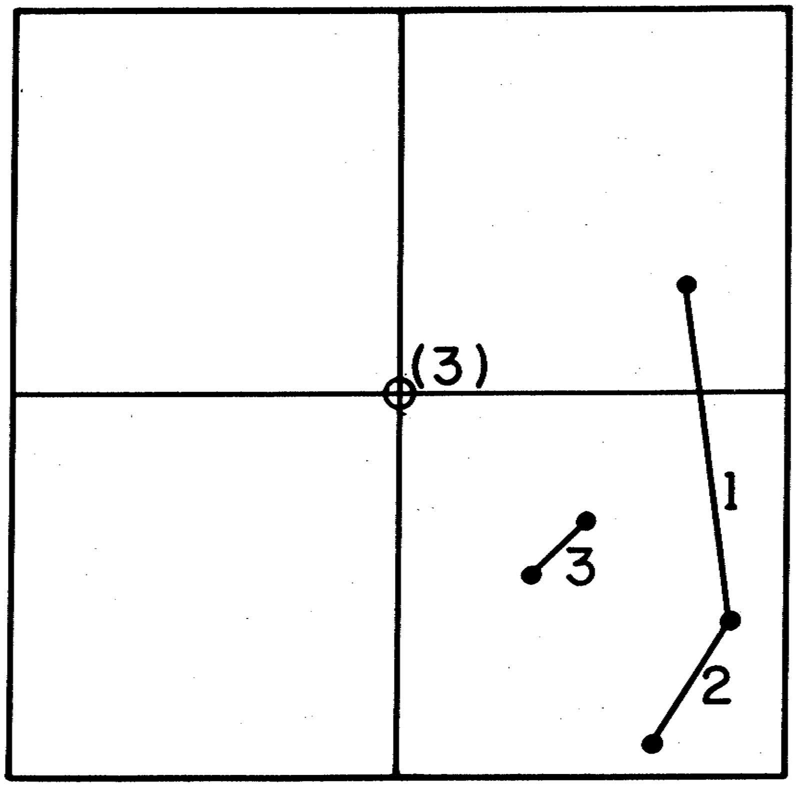

Figure 4. Building a PMR quadtree with node capacity equal to two. Three segments have been inserted causing the plane to be quartered once as indicated by the small circle. Segments were inserted in the order indicated, and the number of the small circle identifies the segment which caused the split.

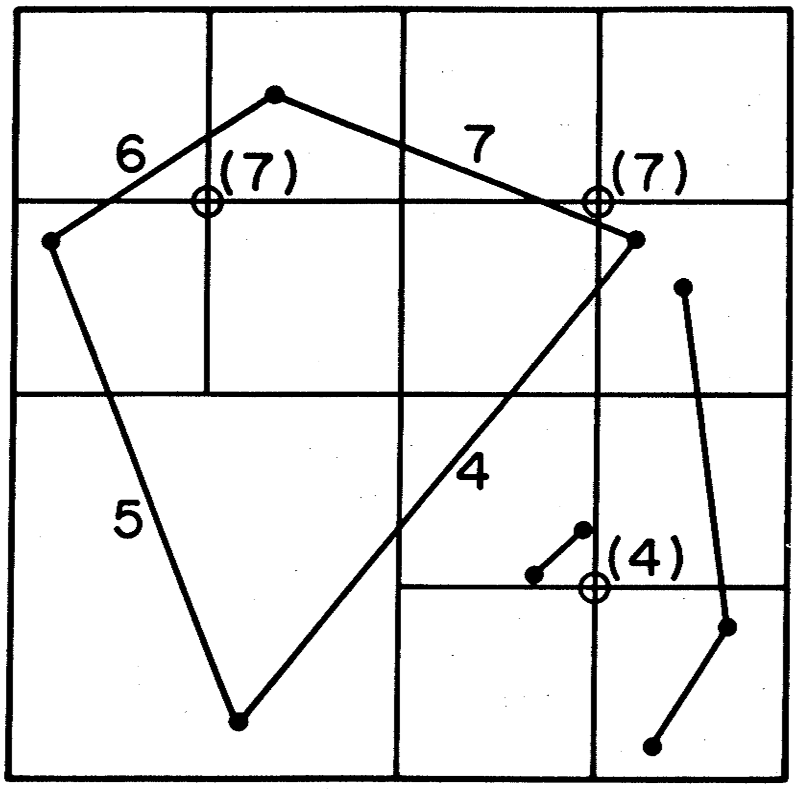

Figure 5. Building a PMR quadtree (continued). Segments 4-7 have been inserted causing three more blocks to be quartered as indicated.

dynamically as the quadtree is constructed. A line segment is stored in a PMR quadtree by inserting it into the nodes corresponding to all the blocks that it intersects. The occupancy of each such node is checked when the insertion occurs, and if it exceeds a preset threshold N, then the node is split *once* into four equal quadrants. An example of the construction of a PMR quadtree is given in Figures 4 and 5. Note that the splitting rule does not guarantee that all nodes will have occupancy less than N as was the case with the PR quadtree for points. In fact, such a subdivision is impossible to achieve for line data since an arbitrary number of line segments can intersect at a single point. However, empirical tests indicate that the average occupancy of nodes produced by this scheme is good, i.e., less than N. We also note that the exact structure of the PMR quadtree produced for a given set of data is dependent on the order of insertion of line segment data. However, since the subdivision is regular, and focussed on regions of high data density, the effect of this on the average occupancy is small. In the following, we show that quantitative predictions for the average occupancy and expected distribution can be made on the basis of our population model.

In order to use the technique that we applied for PR quadtrees, we need to determine the distribution of occupancies that results when we split a node containing m line segments. For points, determining the distribution was relatively simple since the situation was equivalent to the distribution of m points into four buckets. For line data however, the geometry begins to complicate the situation, since a line segment can intersect several quadrants of a block while a point can intersect only one. We begin by considering the case of a block that contains a single line segment. To simplify matters, we assume that the block does not contain either endpoint. This assumption is equivalent to requiring that the average block width be small in comparison with the average length of a line segment in the data set. This will be the case if the data consists of line segments which frequently cross one another as might happen in a map of airline routes or in a photograph of a bubble chamber. If the segments form a planar graph, as in a road map, then the occurrence of endpoints may have to be taken into account.

Consider the set L of all parallel lines passing through a quartered block B at some angle θ (see Figure 6). Every line in L passes through one, two, or three quadrants of B. Let a be a line perpendicular to the set L, and for any point p on a, let $q(p)$ be the number of quadrants intersected by the line perpendicular to a at p. Then a can be partitioned into sections depending on the value of q. Figure 6 shows this partition, with the sections corresponding to $q(p)$ values of 1, 2, or 3 labeled x_1, x_2, and x_3 respectively. Because of the symmetry of the square block, the total length of the sections with $q(p) = 1$, is equal to the total length of the sections with $q(p) = 3$ independent of the value of θ. The value of q over the remainder of the intersection of a with L is 2. Thus the average value of q over the intersection of a with L is 2 for any angle θ. If the lines are drawn from a sample distributed uniformly in space and orientation, this value corresponds to the average number of quadrants intersected by a line passing through the block. Therefore, on the average, a line stored in a node will have to be stored in two of the children if the node is split.

If the line segments stored in the node are uncorrelated, then each quadrant has, independently, a $2/4 = .5$ probability of being intersected by any line segment stored in the parent node. Thus the distribution of occupancies, resulting when a node is split into quadrants, is given by a binomial normalized so that the expected total number of quadrants is 4. In particular, the expected number of quadrants of occupancy i produced by splitting a node of occupancy m is $\binom{m}{i} 2^{(2-i)}$.

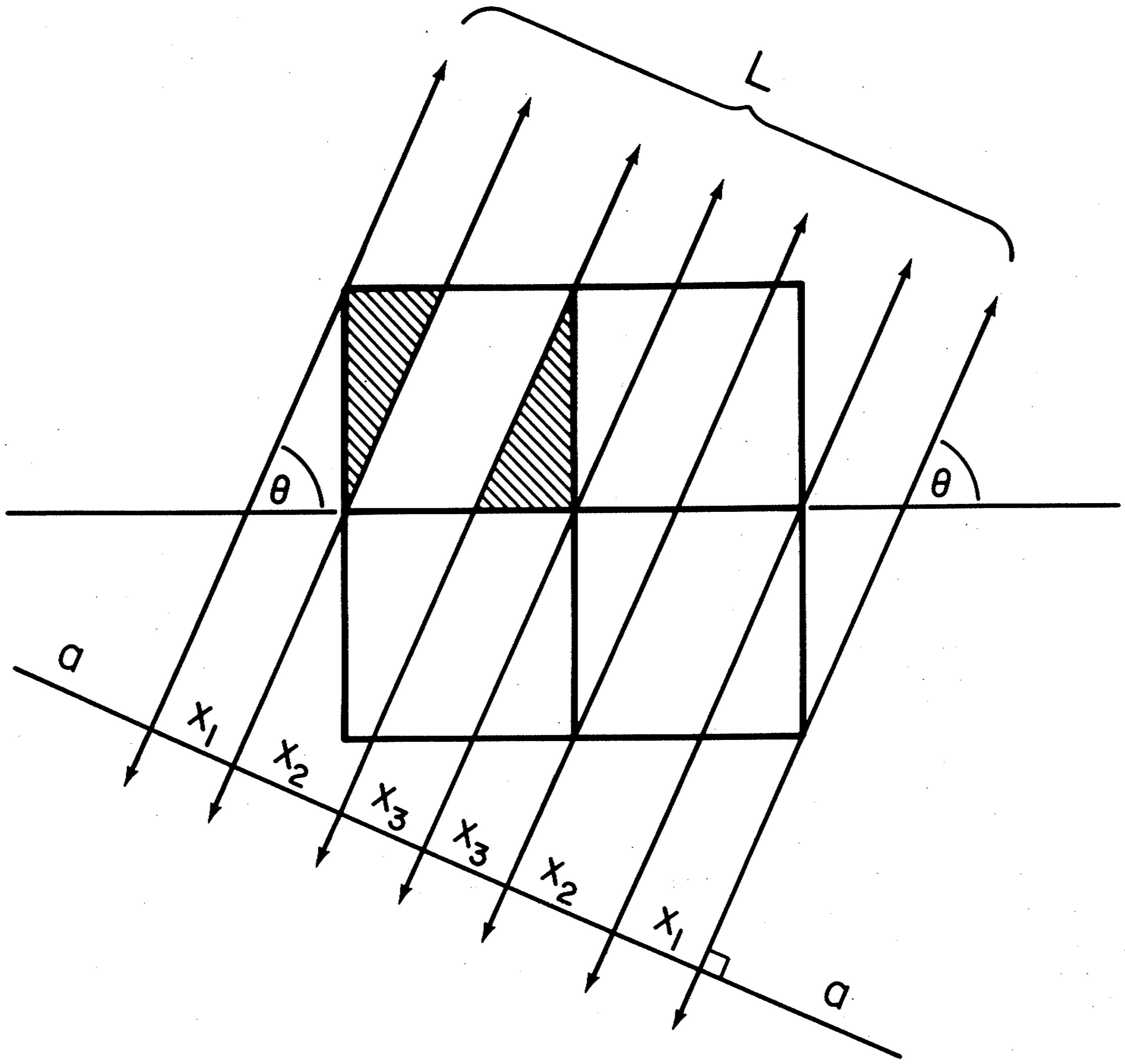

Figure 6. Congruent triangles show that $|x_1|=|x_3|$.

The above results can be used, as in the case of the PR quadtree, to produce a transform matrix T the rows of which describe the results of adding a line segment to nodes of different occupancies. The only difference is that, in the case of the PMR quadtree, there is no absolute upper bound on node occupancy for a given threshold N. Since we do not wish to deal with matrices of infinite size, we choose a cutoff point, and assume that the proportion of nodes with occupancy higher than that cutoff is negligible. We choose to neglect nodes of occupancy higher than $N+2$. This is justified by the observation that for the case of $N=1$, considering nodes of occupancy higher than 3 changes the results of the analysis by less than 1%. For higher thresholds, the effect would be even less since the probability of having all the the segments in a node intersect the same quadrant, producing a node of occupancy higher than N when the node is split, decreases exponentially with increasing N.

Below we give the transform matrices for the first few values of N.

$$\mathbf{T}^1 = \begin{array}{cccc} 0 & 1 & 0 & 0 \\ 1 & 2 & 1 & 0 \\ \frac{1}{2} & \frac{3}{2} & \frac{3}{2} & \frac{1}{2} \\ \frac{1}{4} & \frac{4}{4} & \frac{6}{4} & \frac{4}{4} \end{array} \qquad \mathbf{T}^2 = \begin{array}{ccccc} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \frac{1}{2} & \frac{3}{2} & \frac{3}{2} & \frac{1}{2} & 0 \\ \frac{1}{4} & \frac{4}{4} & \frac{6}{4} & \frac{4}{4} & \frac{1}{4} \\ \frac{1}{8} & \frac{5}{8} & \frac{10}{8} & \frac{10}{8} & \frac{5}{8} \end{array} \qquad \mathbf{T}^3 = \begin{array}{cccccc} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ \frac{1}{4} & \frac{4}{4} & \frac{6}{4} & \frac{4}{4} & \frac{1}{4} & 0 \\ \frac{1}{8} & \frac{5}{8} & \frac{10}{8} & \frac{10}{8} & \frac{5}{8} & \frac{1}{8} \\ \frac{1}{16} & \frac{6}{16} & \frac{15}{16} & \frac{20}{16} & \frac{15}{16} & \frac{6}{16} \end{array}$$

The first N rows indicate the growth of a node as data is added to it below threshold. The last three rows describe the splitting of a node with occupancy equal to, or higher t han the threshold.

As in the case of the PR quadtree, the transform matrices were used to obtain a set of quadratic equations for the components of $\vec{e}$, the expected distribution of occupancies in a steady state. These were solved using the same iterative technique. Experimental data was generated by connecting pairs of points randomly selected from a uniform distribution over a square region, to produce a set of line segments randomly crisscrossing the plane. Ten samples each containing 100 line segments were generated for each value of N and the results averaged. The experimental and theoretical values for the expected distribution $\vec{e}$ are compared below for N ranging between one and eight.

$N = 1$ *thy*: $\vec{e} = (.183\ ,\ .470\ ,\ .288\ ,\ .059)$

exp: $\vec{e} = (.192\ ,\ .462\ ,\ .284\ ,\ .057)$

$N = 2$ *thy*: $\vec{e} = (.092\ ,\ .323\ ,\ .434\ ,\ .135,\ .016)$

exp: $\vec{e} = (.103\ ,\ .332\ ,\ .427\ ,\ .132,\ .015)$

$N = 3$ *thy*: $\vec{e} = (.042\ ,\ .192\ ,\ .355\ ,\ .350,\ .057,\ .004)$

exp: $\vec{e} = (.053\ ,\ .209\ ,\ .351\ ,\ .325,\ .058,\ .004)$

$N = 4$ thy: $\vec{e} = (.019\ ,\ .105\ ,\ .249\ ,\ .328,\ .275,\ .023,\ .001)$

exp: $\vec{e} = (.027\ ,\ .126\ ,\ .262\ ,\ .320,\ .240,\ .024,\ .001)$

$N = 5$ thy: $\vec{e} = (.008\ ,\ .055\ ,\ .159\ ,\ .263,\ .285,\ .221,\ .009,\ .000)$

exp: $\vec{e} = (.014\ ,\ .074\ ,\ .181\ ,\ .275,\ .259,\ .189,\ .008,\ .000)$

$N = 6$ thy: $\vec{e} = (.004\ ,\ .028\ ,\ .096\ ,\ .191,\ .253,\ .242,\ .182,\ .004,\ .000)$

exp: $\vec{e} = (.009\ ,\ .046\ ,\ .119\ ,\ .209,\ .251,\ .215,\ .148,\ .003,\ .000)$

$N = 7$ thy: $\vec{e} = (.002\ ,\ .014\ ,\ .056\ ,\ .131,\ .205,\ .232,\ .205,\ .153,\ .002,\ .000)$

exp: $\vec{e} = (.005\ ,\ .031\ ,\ .082\ ,\ .154,\ .212,\ .218,\ .176,\ .121,\ .001,\ .000)$

$N = 8$ thy: $\vec{e} = (.001\ ,\ .007\ ,\ .032\ ,\ .085,\ .154,\ .204,\ .208,\ .176,\ .132,\ .001,\ .000)$

exp: $\vec{e} = (.002\ ,\ .021\ ,\ .056\ ,\ .118,\ .169,\ .201,\ .186,\ .146,\ .100,\ .001,\ .000)$

Again, there is generally good agreement between the experimental and theoretical results. The agreement is excellent for low node capacities. As the node capacity increases, the effect of aging starts to skew the predictions slightly. As in the case of the PR quadtree, this is reflected in the tendency of the model to underestimate the number of low occupancy nodes, and overestimate the number of high occupancy ones.

A better indication of the overall accuracy of the model is obtained by comparing the predicted average occupancy with that measured in the experiments. This is done in Table 5. In this case, the theoretical results are actually closer to the experimental results than for the PR quadtree. Several points should be noted. First, as in the PR quadtree, the theoretical predictions are slightly higher than the measured values. Again, this is due to the implicit assumption in the model that the proportion of nodes of a given occupancy is not correlated with the size of the node. As in the PR quadtree, the phenomenon of aging ensures that this assumption does not strictly hold. Nevertheless, the model yields results which are close enough to be useful. Second, the phasing effect, which was apparent for the point data in the PR quadtree, does not appear to be as significant for line data represented by the PMR quadtree. The apparent reason for this seems to be that the local fluctuations in data density are greater for randomly intersecting line segments than for random points. This has the effect of smoothing out the transition from one generation of nodes to the next as the tree grows in size.

As in the case of the PR quadtree, rough bounds can be placed on the average node occupancy to show that the effects of aging and phasing cannot completely dominate the

Table 5
Average Node Occupancy for PMR Quadtrees

node capacity	experimental occupancy	theoretical occupancy	percent difference
1	1.22	1.22	0.0
2	1.62	1.66	2.4
3	2.14	2.20	2.9
4	2.70	2.81	4.0
5	3.30	3.46	4.7
6	3.90	4.14	5.6
7	4.51	4.81	6.5
8	5.12	5.51	7.0

behavior of the structure. To do this, we show that the average occupancy of the nodes produced when a node is split is bounded, and smaller than the size of the parent. Since each quadrant has a .5 probability of intersecting each line segment in the parent node, the average occupancy of the nodes produced will be 1/2 the occupancy of the parent node. A slight complication arises from the fact that nodes of occupancy higher than N exist in the tree. The average size of parent nodes can be bounded, however, by noting that a node of occupancy $i>N$ must be produced by a split in which all i segments intersect one quadrant. The expected incidence of this situation for $i=N+1$ is $4 \cdot 2^{-(N+1)}$ per split; the expected incidence for $i=N+2$ is $4 \cdot 2^{-(N+1)} 4 \cdot 2^{-(N+2)}$; and similarly for higher occupancies. In other words, the expected number of nodes with occupancy higher than N decreases exponentially with $i-N$. Thus the average size of parent nodes is bounded by a geometric series which converges for all $N>0$, and therefore the average node size in the tree as a whole is bounded.

VIII. Conclusions

We have presented a method for analyzing hierarchical data structures based on a model of such structures as populations of nodes of different occupancies. The method is flexible, and is adaptable to the analysis of any hierarchical data structure in which spatial decomposition is determined by the local distribution of data elements. The model allows the analysis of such data structures without laborious statistical analysis. Only the probabilities of the local interaction of the data primitive with the quadrants of a node need be evaluated. This is generally a much easier task than calculating global statistics. The model represents a formalization of the intuitive notion of a "typical case". By investigating the sources of discrepancy with experimental data, two phenomena which are characteristic of hierarchical data structures were identified: aging and phasing. Aging is responsible for larger nodes having higher than average occupancy even when splitting is adaptive. Phasing causes a cyclical variation in the average occupancy of nodes which is periodic in the logarithm of total number of data items stored in the structure, particularly when the distribution of data is uniform. Generalized PR and PMR quadtrees were analyzed using the population model, and the results compared to experimental data with generally good agreement.

Areas for future research include obtaining expressions quantifying the effects of aging and phasing. The phenomenon of aging, for instance, could be approached by considering the distribution of nodes with respect to their sizes as well as with respect to their occupancies. Another promising area is the application of our method to data structures involving more complex primitives and higher dimensions. An example is the use of octrees to store linear or planar data in three dimensions. A third area of interest is the investigation of alternative models of data distribution. For example, we have empirical data which indicates that when a PR quadtree is used to represent geographic points such as houses or cities, its storage requirements are considerably different than when the points are uniformly distributed. This is apparently due to clustering of the data points. Statistical analysis of hierarchical data structures under non-uniform data distributions would be difficult or at least laborious. Our population method, on the other hand, allows some data models representing clustering (e.g., fractals) to be investigated in a relatively simple and straightforward manner.

References

[Fagi79] - R. Fagin, J. Nievergelt, N. Pippenger, H.R. Strong, Extendible hashing – a fast access method for dynamic files, *ACM Transactions on Database Systems 4,* 3(September 1979), 315-344.

[Fink74] - R.A. Finkel and J.L. Bentley, Quad trees: a data structure for retrieval on composite keys, *Acta Informatica 4,* 1(1974), 1-9.

[Hunt78] - G.M. Hunter, Efficient computation and data structures for graphics, Ph.D. dissertation, Department of Electrical Engineering and Computer Science, Princeton University, Princeton, NJ, 1978.

[Jack80] - C.L. Jackins and S.L. Tanimoto, Oct-trees and their use in representing three-dimensional objects, *Computer Graphics and Image Processing 14,* 3(November 1980), 249-270.

[Kede81] - G. Kedem, The Quad-CIF tree: a data structure for hierarchical on-line algorithms, *Proceedings of the Nineteenth Design Automation Conference,* Las Vegas, June 1982, 352-357.

[Klin71] - A. Klinger, Patterns and search statistics, in *Optimizing Methods in Statistics,* J.S. Rustagi, Ed., Academic Press, New York, 1971, 303-337.

[Know80] - K. Knowlton, Progressive transmission of grey-scale and binary pictures by simple, efficient, and lossless encoding schemes, *Proceedings of the IEEE 68,* 7(July 1980), 885-896.

[Meag82] - D. Meagher, Geometric modeling using octree encoding, *Computer Graphics and Image Processing 19,* 2(June 1982), 129-147.

[Nels86] - R.C. Nelson and H. Samet, A consistent hierarchical representation for vector data, *ACM SIGGRAPH '86*, Dallas, August, 197-206.

[Niev84] - J. Nievergelt, H. Hinterberger, and K.C. Sevcik, The grid file: an adaptable symmetric multi-key file structure, *ACM Transactions on Database Systems 9*, 1(March 1984), 38-71.

[Oren82] - J.A. Orenstein, Multidimensional tries used for associative searching, *Information Processing Letters 14*, 4(June 1982), 150-157.

[Regn85] - M. Regnier, Analysis of Grid File algorithms, *BIT 25*, 2(1985), 335-357.

[Same84a] - H. Samet, The quadtree and related hierarchical data structures, *ACM Computing Surveys 16*, 2(June 1984), 187-260.

[Same84b] - H. Samet, A. Rosenfeld, C.A. Shaffer, R.C. Nelson, and Y-G. Huang, Application of hierarchical data structures to geographic information systems: phase III, Computer Science TR-1457, University of Maryland, College Park, MD, November 1984.

[Same84c] - H. Samet and M. Tamminen, Efficient image component labeling, Computer Science TR-1420, University of Maryland, College Park, MD, July 1984.

[Same85a] - H. Samet and M. Tamminen, Efficient component labeling of images of arbitrary dimension, Computer Science TR-1480, University of Maryland, College Park, MD, February 1985.

[Same85b] - H. Samet and R.E. Webber, Storing a collection of polygons using quadtrees, *ACM Transactions on Graphics 4*, 3(July 1985), 182-222.

[Same85c] - H. Samet, A. Rosenfeld, C.A. Shaffer, R.C. Nelson, Y-G. Huang, and K. Fujimura, Application of hierarchical data structures to geographic information systems: phase IV, Computer Science TR-1578, University of Maryland, College Park, MD, December 1985.

[Tamm81] - M. Tamminen, The EXCELL method for efficient geometric access to data, *Acta Polytechnica Scandinavica*, Mathematics and Computer Science Series No. 34, Helsinki, 1981.

[Tamm83] - M. Tamminen, Performance analysis of cell based geometric file organizations, *Computer Vision, Graphics, and Image Processing 24*, 2(November 1983), 168-181.

[Tamm84] - M. Tamminen, Comment on quad- and octtrees, *Communications of the ACM 27*, 3(March 1984), 248-249.