A Quadtree-Based Geographical Information System*

Hanan Samet
Azriel Rosenfeld
Clifford Shaffer
Robert E. Webber

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

ABSTRACT

An investigation of the application of hierarchical data structures such as the region quadtree in geographical information systems is currently underway. In this paper we present some results of this effort.

1. INTRODUCTION

An investigation of the application of hierarchical data structures to geographical information systems is currently underway. Its purpose is twofold: (1) to construct a geographic information system based on the quadtree hierarchical data structure, and (2) to gather statistics to allow the evaluation of the usefulness of this approach to geographic information system organization. To accomplish these objectives, a database was built that contained three maps of different features of a small region in Northern California. These maps describe a flood plain, elevation contours, and landuse classes for the region. Programs were then written to perform various analysis and manipulation tasks on the quadtree-encoded regions. This paper presents the results of the preliminary investigation. Further details about the study can be found in [1].

The quadtree representation of regions, first proposed by Klinger [2], has been the subject of numerous papers over the past few years. It is based on the successive subdivision of an image array into quadrants. If the array does not consist entirely of 1's or 0's, then we subdivide it into quadrants, subquadrants, ... until we obtain blocks (possibly single pixels) that consist of 1's or 0's; i.e., they are entirely contained in the region or en-

---

*The support of the U.S. Army Engineer Topographic Laboratory under contract DAAK 70-81-C-0059 is gratefully acknowledged.

tirely disjoint from it. This process is represented by a tree of out degree 4 (i.e., each non-leaf node has four sons) in which the root node represents the entire array. The four sons of the root node represent the quadrants (labeled in order NW, NE, SW, SE), and the leaf nodes correspond to those blocks of the array for which no further subdivision is necessary. Leaf nodes are said to be BLACK or WHITE depending on whether their corresponding blocks are entirely within or outside of the region respectively. All non-leaf nodes are said to be GRAY. For example, see Figure 1, which contains a region and its corresponding quadtree.

Numerous algorithms have been developed for constructing compact quadtree representations, converting between them and other region representations, computing region properties from them, and computing the quadtree representations of Boolean combinations of regions from those of the given regions. For recent overviews of this literature see [3,4].

2. DATABASE

The database used in the study was supplied by the U.S. Army Engineer Topographic Laboratory, Ft. Belvoir, VA. It consisted of three map overlays representing land use classes, terrain elevation contours, and floodplain boundaries. In the case of the elevation contours, only those at multiples of 100 feet were used; and in all cases, only the portions of the overlays bounded by the fiducial marks were used. The data were hand-digitized to 400x450 pixels, and labels were associated with the pixels in each of the resulting regions, specifying the particular land use class or elevation range. The regions were then quadtree-encoded using a 512x512 grid.

In order to be able to handle cartographic data we had to slightly modify the above definition of a quadtree. In particular, we decided to use a different label for each landuse class and 100 foot elevation contour. This means that our quadtree is no longer binary but instead is multicolored. Thus leaf nodes now have a range of values. This did not prove to be a problem. Nevertheless, certain operations, such as union and intersection are not defined in terms of multicolor quadtrees, and thus they had to be modified. In particular, they are defined in our system in terms of binary quadtrees by considering one of the colors as BLACK (usually the color of the object of interest) and the other colors to be WHITE. Note that boundaries between landuse classes and contour elevations are treated as lies of zero width and do not appear explicitly in our quadtrees.

The landuse map consisted of 35 classes. There were 11 elevation ranges. The floodplain contained 3 regions. Quadtree building was done "bottom-up," by scanning the image and merging

nodes when all the nodes in a 2x2 block are found to be of the same type, as described in [5]. The numbers of "nodes created" ranged from about 1700 to about 14,500 (still only a small fraction of the 2**18 potential pixels in the 512x512 grid), while the number of nodes in the final quadtrees ranged from about 130 to about 13,000.

The quadtrees were stored on disk without the use of pointers. Instead, in order to conserve space, they are kept in files containing lists of the node types met in a preorder traversal of the quadtree. Some of the simpler algorithms can manipulate these files directly. Others read the file into core where it is expanded to include pointers to father nodes and son nodes.

3. REGION ANALYSIS AND MANIPULATION

A large number of quadtree algorithms were tested on the database. In particular, we were interested in obtaining timing information. This was difficult to do in the sense that precise machine execution times are hard to obtain. Thus we also measured machine independent concepts such as the number of nodes visited by the various algorithms. One of the basic operations that was measured was the connected component labeling process [6]. It was performed for each region type (land use class, elevation range). This required the computation of neighboring nodes and enabled us to do a comparison between a number of different techniques. In particular, we compared the neighbor finding techniques of Samet [7] which make use of a common ancestor; those of Hunter and Steiglitz [8] which make use of "ropes" (links between nodes of the same size that are spatially adjacent); and a new technique [1,9] which is based on modifying the quadtree traversal algorithm to pass the neighbors of each subtree's root as parameters. This required more time than using ropes (because of the need to pass the parameters), but required 40% less memory. As expected, the method of ropes was the most efficient time-wise (i.e., the average cost of 1.5 nodes being visited). However, it had the extra storage cost for the links. Interestingly, the neighbor finding techniques of Samet resulted in 3.5 nodes being visited on the average. This vindicated the theoretical analysis and the model of node configuration which served as its basis.

Programs were also written to compute the area, perimeter, coordinates of the centroid, and the enclosing upright rectangle for each of the connected components [10,11]. In addition, a program was written to determine, for a given quadtree-encoded region (not necessarily connected) and a given point (specified by its coordinates), whether or not the point lies in the region. This "point-in-polygon" task is especially efficient using quadtree encoding, since one need only move directly down the tree, as directed by the successive bits of the point's coordinates, until

a leaf is reached; the point is in the region iff this leaf is BLACK.

Finally, two types of programs for manipulating quadtree-encoded regions were rewritten. The first constructs the quadtree of the union or intersection of two regions from the quadtrees of the regions [8,10]. This is done by traversing the two trees in parallel and building the output tree in accordance with the following rules:

If current nodes are	Union	Intersection
Both BLACK	Return BLACK	Return BLACK
Both WHITE	Return WHITE	Return WHITE
Both GRAY	Recurse	Recurse
One BLACK	Return other subtree	Return BLACK
One WHITE	Return WHITE	Return other subtree

The second program constructs the quadtree of the intersection between a given region (defined by a quadtree) and a given square window (of size a power of two). This is done by finding the smallest subtree of the given quadtree that contains the window. If it coincides with the window, return the subtree; if not, split the window into quadrants and process each of them analogously with respect to the current subtree. This program proved to be rather costly since it involved building a new quadtree for a subregion of the map specified by the boundaries of the window. In both programs, on returning from each recursive call, it is necessary to check if the four leaves in a 2x2 block are all of the same color, and if so, to replace their father by a leaf of that color.

Various quadtree display techniques were attempted. In particular, a quadtree truncation method was devised which is an attempt to approximate the map without displaying the quadtree nodes at the deepest levels. Instead of uniformly treating GRAY nodes lower than a certain level as BLACK or WHITE, they were set to the appropriate color based on their weighted average - i.e., BLACK if the weighted average exceeds that of WHITE, and BLACK if vice versa. The results showed a gentle degradation when quadtree truncation was compared at different levels.

In general, it should be noted that displaying quadtree files is much faster than pixel-based files on many common CRT devices that allow specification of an entire block to be of one color. The leaves of the quadtree indicate a convenient partitioning of a map into monochrome blocks. Since it is not necessary to store the quadtree in core in order to display it, simple hardware could be designed that displayed quadtree files directly.

4. CONCLUDING REMARKS

The experiments described in this paper provide a quantitative assessment of the efficiency of quadtrees as a means of representing regions in a cartographic database. The quadtree representation is significantly more compact than the binary array representation. Efficient algorithms exist for region property computation and manipulation using the quadtree representation; the expected time required by these algorithms is proportional to the sizes of quadtrees involved as established on theoretical grounds in [5-7,11,12]. The point-in-polygon and set-theoretic operations on regions are especially efficient. Truncation of quadtrees can be used to generate approximations to regions that are even more compact. We have not given timing results in this paper (see [1]) because of their dependence on the operating system and on the available memory. The quadtree algorithms are easy to implement using structured programming languages; in our experiments, the C language was used. In conclusion, our experiments confirm that quadtrees constitute a viable method of region representation which is quite suitable for use in geographic information systems.

REFERENCES

1. Rosenfeld, A., Samet, H., Shaffer, C., and Webber, R. E., Application of hierarchical data structures to geographic information systems, Computer Science TR-1197, University of Maryland, College Park, MD, June 1982.

2. Klinger, A., Patterns and search statistics in Optimizing Methods in Statistics, J. S. Rustagi, Ed., Academic Press, New York, 1971.

3. Rosenfeld, A., Quadtrees and pyramids: hierarchical representation of images, Computer Science TR-1171, University of Maryland, College Park, MD, May 1982.

4. Samet, H., Hierarchical data structures for representing geographical information, Computer Science TR-1208, University of Maryland, College Park, MD, August 1982.

5. Samet, H., Region representation: quadtrees from binary arrays, Computer Graphics and Image Processing 13, 1980, 88-93.

6. Samet, H., Connected component labeling using quadtrees, Journal of the ACM, 1981, 487-501.

7. Samet, H., Neighbor finding techniques for images represented by quadtrees, Computer Graphics and Image Processing 18, 1982, 37-57.

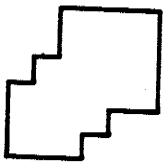
8. Hunter, G. M. and Steiglitz, K., Operations on images using quadtrees, IEEE Transactions on Pattern Analysis and Machine Intelligence 1, 1979, 145-153.

9. Jackins, C. and Tanimoto, S. L., Quad-trees, Oct-trees, and K-trees: a generalized approach to recursive decomposition of Euclidean space, Department of Computer Science, Technical Report 82-02-02, University of Washington, Seattle, Washington, 1982, to appear in IEEE Transactions on Pattern Analysis and Machine Intelligence.

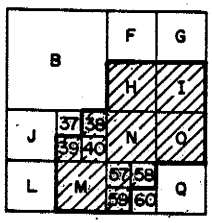
10. Shneier, M., Calculations of geometric properties using quadtrees, Computer Graphics and Image Processing 16, 1981, 296-302.

11. Samet, H., Computing perimeters of images represented by quadtrees, IEEE Transactions on Pattern Analysis and Machine Intelligence 3, 1981, 683-687.

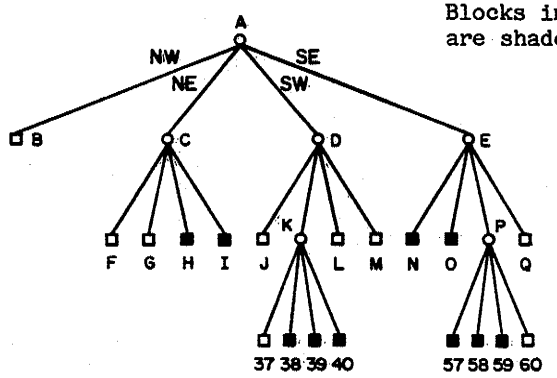
12. Samet, H., Algorithms for the conversion of quadtrees to rasters, Computer Science TR-979, University of Maryland, College Park, MD, November 1980.



(a) Region



(b) Block decomposition of the region in (a). Blocks in the image are shaded.



(c) Quadtree representation of the blocks in (b).

Figure 1. A region, its maximal blocks, and the corresponding quadtree.