presented at *GIS National Conference 1989*, Ottawa, March 1989

proceedings to be published

EFFICIENT PRIMITIVE GEOMETRIC OPERATIONS ON LARGE DATABASES

Wm. Randolph Franklin
Mohan Kankanhalli
Chandrasekhar Narayanaswami

Electrical, Computer, and Systems Engineering Dept.,
6026 J.E.C.,
Rensselaer Polytechnic Institute,
Troy, NY, 12180, USA,
(518) 276-6077,
Internet: wrf@ecse.rpi.edu
Fax: (518) 276-6003, Telex: 6716050 RPI TROU.

ABSTRACT

Certain common operations, such as finding all the intersections among a large set of small line segments, are the rate-limiting factor in processes like map overlay. We present the uniform grid technique, which is comparatively simple to implement, yet quite fast on large real databases. This method is suited for parallel execution, even on a SIMD or pipelined machine. These ideas have implications for computational cartography as faster machines with more memory become available to process larger databases. Some of the older data structures, such as overlay methods involving active edges, scan lines, and threaded lists, are more complicated than necessary for small databases, and simultaneously inadequate for large ones. These grid techniques are also useful for other geometric operations also. We are now implementing a complete map overlay program using the uniform grid technique, are attempting to gain access to large, real, databases and to a massively parallel machine for further tests.

INTRODUCTION

Presently there is an increasing need for the development, monitoring and inventory of natural resources. This requires improved efficiency in storage, manipulation and presentation of the collected data. In this paper we present the technique of Uniform Grid which leads to efficient algorithms for many problems such as map overlay and visualization of data.

In the classic problem of map overlay, edge intersection forms the core of the algorithm. The uniform grid is used to develop an efficient algorithm for the edge intersection problem. We are given a large number of small edges, very few of which intersect, and must determine the pairs of them that do intersect. Clearly, a naive algorithm which compares all $\binom{N}{2}$ pairs is not acceptable, especially when the databases are large, as often is the case in geographic information systems.

Useful line intersection algorithms often use sweep-line techniques, such as in Nievergelt and Preparata (Nievergelt,1982), and Preparata and Shamos (Preparata,1985). Chazelle and Edelsbrunner (Chazelle,1988) have an algorithm that finds all K intersections of N edges in time $T = \theta(K+N\log N)$. This method is optimal in the worst case. However, it is inherently sequential. Mulmuley also has an optimal randomized algorithm. But it appears to be difficult to parallelize(Mulmuley,1988).

Alternative data structures, based on hierarchical methods such as quadtrees, have also been used extensively, Samet(Samet,1984) . They are intuitively reasonable data structures to use since they subdivide to spend more time on the complicated regions of the scene. An informal criticism of their overuse in Geographic Information Systems in given in Waugh (Waugh,1986). Hillis (Hillis,1985) mentions in his thesis that many hierarchical data structures which are ideal for a sequential computer are far from efficient for a parallel computer.

Since cartographers deal with vast amounts of data, the speed and efficiency of the algorithms are of utmost importance. With the advent of parallel and supercomputers, efficient parallel algorithms which are simple enough to implement, are gaining importance. Since this field is relatively new, few implementable algorithms for geometric operations exist.

This paper concentrates on the *uniform grid* technique. Here, a flat, non-hierarchical grid is superimposed on the data. The grid adapts to the data since the number of grid cells, or resolution, is a function of some statistic of the input data, such as average edge length.

The uniform grid (in our use) was first presented in Franklin (Franklin,1978) and was later expanded by Franklin, Akman and Wu (Franklin,1980), (Franklin,1982), (Franklin,1983), (Franklin,1987), and Wu(Wu,1988) . The latter two papers used extended precision rational numbers and Prolog to implement map overlay. Geometric entities and relationships are represented in Prolog facts and algorithms are encoded in Prolog rules to perform data processing. Multiple precision rational arithmetic is used to calculate geometric intersections exactly and therefore properly identify all special cases of tangent conditions for proper handling. Thus topological consistency is guaranteed and complete stability in the computation of overlay is achieved.

This paper presents the uniform grid as an efficient means of finding intersections between edges in real world data. The uniform grid is similar to a quadtree is the same sense that a relational database schema is similar to a hierarchical schema.

The uniform grid data structure is ideally suited to execution on a parallel machine because of the simple data structures. Also, it is more numerically robust than sweep-line algorithms. This is of importance in the cartographic domain because numerical instability can easily introduce topological inconsistencies which tend to be difficult to rectify.

The uniform grid technique is fairly general and can be used on a variety of geometric problems such as computing Voronoi diagrams, convex hull determination, hidden surface removal, Boolean combinations of polygons and the mass properties of the union of a set of polygons.

UNIFORM GRID TECHNIQUE FOR EDGE INTERSECTION

Assume that we have N edges of length L independently and identically distributed (i.i.d.) in a 1×1 screen. We place a $G \times G$ grid over the screen. Thus each grid cell is of size $\frac{1}{G} \times \frac{1}{G}$. The grid cells partition the screen without any overlaps or omissions. The intersection algorithm proceeds as follows.

1. For each edge, determine which cells it passes through and write ordered pairs *(cell number, edge number)*.

2. Sort the list of ordered pairs by the cell number and collect the numbers of all the edges that pass through each cell.

3. For each cell, compare all the edges in it, pair by pair, to test for intersections. If the edges are *a priori* known to be either *vertical* or *horizontal,* the vertical edges are compared with the horizontal edges only. To determine if a pair of edges intersect, we test each edge's endpoints against the equation of the other edge. We ignore calculated intersections that fall outside the current cell. This handles the case of some pair

of edges occurring together in more than one cell. It has been shown that the optimal grid size can be determined using $G = \min\left(\delta\sqrt{N}, \frac{\pi}{4L}\right)$ where δ is a constant(Franklin,1989).

RESULTS

Edge Intersection

For various data sets we tried many values of G to learn the variation of time with G. Results from intersecting the 116896 edges in all the 4 overlays of the Chikamauga DLG (Figure 1b) showed 144,666 intersections in all, and the best time for computing them was 37.15 seconds on a Sun 4/280 with a 325×325 grid. The average edge length was 0.0022 and the standard deviation 0.0115, so the edge lengths were quite variable. The time was within 50% of this for grids from 175×175 up to 1000×1000, which shows the extreme insensitivity of the time to the grid size. This is why real scenes with dense and sparse areas can be accommodated efficiently.

For the USA state boundaries (Figure 1a) shifted and overlaid on themselves, the execution time was within 20% of the optimum from about $G = 40$ to $G = 400$ and was within a factor of two of the optimum from about $G = 20$ to $G = 700$. Outside these limits, the execution time started to rise quickly. Other results are listed in Franklin et. al(Franklin,1988).

The economy of the grid structure was shown by the fact that the number of comparisons between pairs of edges needed to isolate the intersections is about twice the number of the edges when using the optimal grid resolution. This behavior was also observed in hidden surface algorithm described in earlier publications. There is not much room for further improvement by a hierarchical method.

One of our examples consisted of 1,819,064 edges, with an average length of 0.0012, forming a complete VLSI chip design. We found all 6,941,110 intersections in 178 seconds. In this case, the program was optimized to use the orthogonality of the edges. The edges' lengths were quite variable, with the standard deviation being over 30 times the mean. This example illustrates the generality of this method and its applicability to other areas besides cartography.

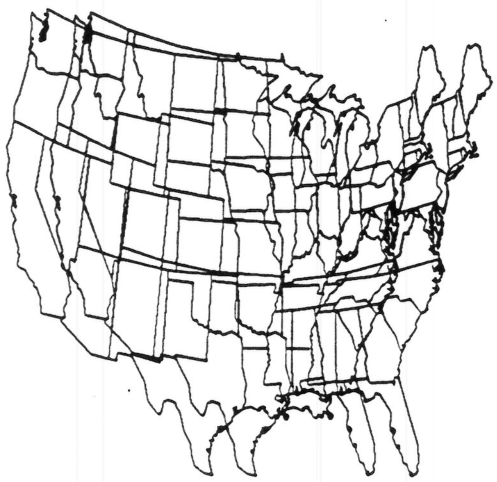

Fig 1(a). USA Map - Shifted and Overlaid on itself

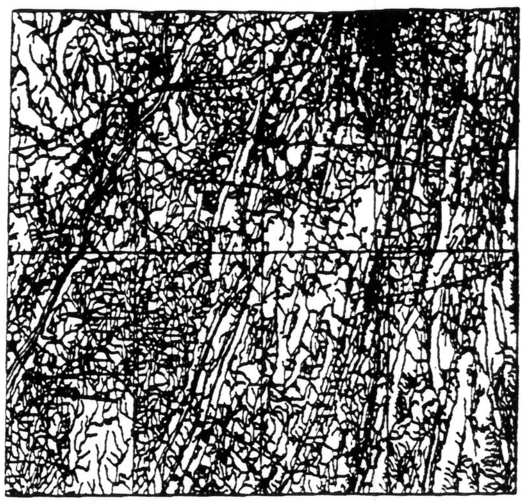

Fig 1(b). Chickamauga Area - All 4 overlays

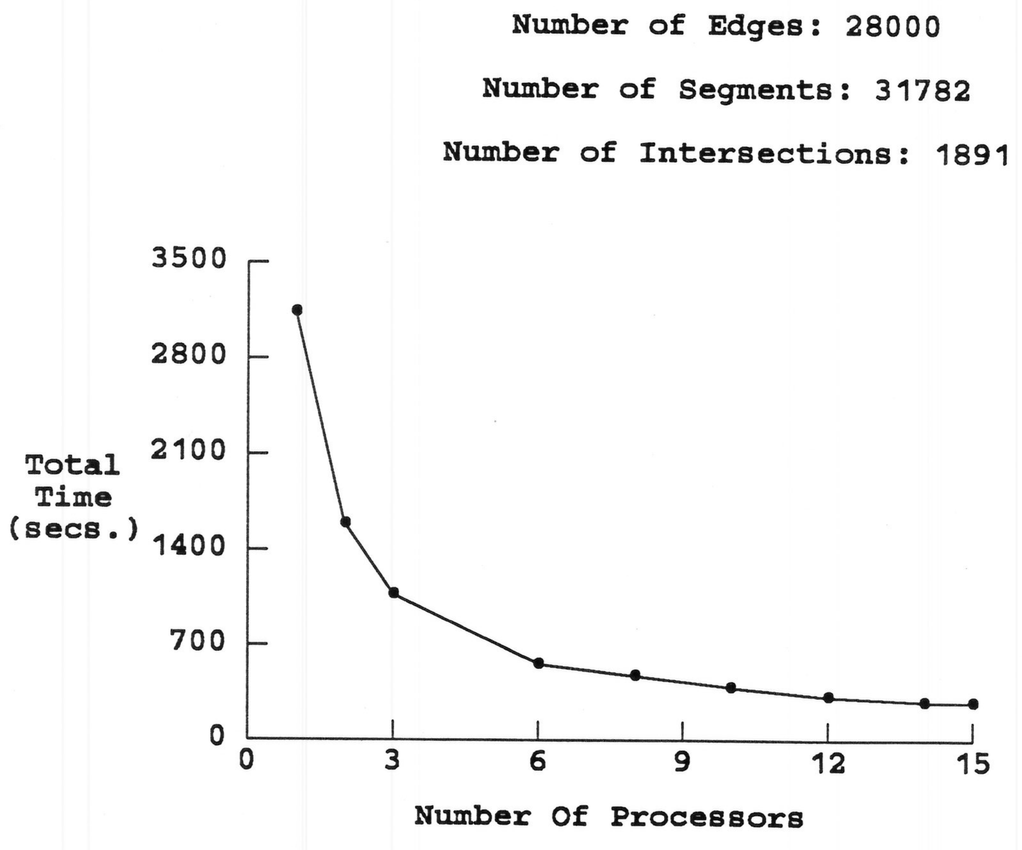

Figure 2: Timing of the boolean combination algorithm

Edge Intersection in Parallel

The uniform grid method is ideally suited to execution on a parallel machine since it mostly consists of two types of operations that run well in parallel: applying a function independently to each element of a set to generate a new set, and sorting. Determining which cells each edge passes through is an example of the former operation.

We implemented several versions of the algorithm on a Sequent Balance 21000 computer, which contains 16 National Semiconductor 32000 processors(Sequent,1986), (Kallstrom,1988) and compared the elapsed time when up to 15 processors were used to the time for only one processor(Kankanhalli,1988) . We used the 'data partitioning' paradigm of parallel programming which involves creating multiple, identical processes and assigning a portion of the data to each process. The edges are distributed among the processors to determine the grid cells to which each edge belongs and then the cells are distributed among the processors to compute the intersections. Since the Sequent Balance 21000 is a shared memory parallel computer, shared data structures is the communication mechanism for the processors. The synchronization of the processors is achieved by using locks.

The speedup ratios obtained range from 8 to 13 using 15 processors. The results from processing 3 overlays of the United State Geological Survey Digital Line Graph, totaling 62,045 edges found 81,373 intersections. The time for one processor was 273 seconds, and for 15 processors was 28 seconds, for a speedup of about 10.

The speedup achieved for any parallel algorithm is dependent on the amount of inherently sequential computation in the algorithm, the hardware contention imposed by the competing processors, the overhead in creating multiple processes and the overhead in synchronization & communication among the multiple processes. We believe that the first factor is not dominant when using the uniform grid technique. The large speedups achieved show that the other three factors also do not affect the performance significantly. Finally, the speedup, as a function of the number of processors, was still rising smoothly at 15 processors. This means that we should achieve an even bigger speedup on a more parallel machine.

Polygon Intersection in Parallel

Polygon intersection is another frequently encountered problem. The uniform grid can be used to solve this problem efficiently. The main steps in the algorithm are to compute the points of intersection between the edges of the two polygons and then to form independent segments (segments which do not intersect with any other segments). The next step is to classify these segments based on which polygon the segment belongs to and on its location, relative to

We have implemented the algorithm partly on a Sun 3/50 and part on a Sun 4/280. Testing all 3660 edges in both input maps to find intersections takes only 1.73 seconds on the Sun 4.

FUTURE DIRECTIONS

Our long term goal is to develop efficient techniques for primitive geometric operations. Parallel computers are our tools to apply them effectively on huge databases. We propose to expand the preliminary concepts described here from 15 processors to massively parallel machines. Recently we have obtained access to a Connection Machine and an NCUBE multiprocessor. The Connection machine is a SIMD machine and has upto 65,536 processors with excellent interprocessor communication mechanisms. The Connection machine is well suited for fine grained parallelism. The NCUBE is a MIMD machine where the nodes are connected in a *hypercube* fashion. The NCUBE has upto 1024 processors and is suitable for coarse grained parallelism. We want to investigate the suitability of both types of machines for geometric problems. We propose to implement more high level algorithms in parallel using our techniques for efficient geometric operations. Hidden-surface removal is a fundamental problem in visualization of map data. We propose to develop an efficient parallel algorithm for this problem. Boolean combinations of polygons and polyhedra is another relevant problem for which we are developing efficient algorithms. We feel that this technique will be useful in the automated generation of relief charts too. We believe that our techniques could be used in a geographic information system built on a parallel machine.

CONCLUSION

Our technique has been successfully used for many important problems which occur in cartography including map overlay. The results indicate that this is a very robust general technique which is fast and easily implementable. It is evident from this research that simple solutions are often faster than theoretically efficient but convoluted and complicated methods. Our algorithm is parallelizable and shows very good speedup with minimal auxiliary data structures.

We are investigating other problems where the uniform grid technique may be applied for inventing parallel algorithms. We feel that the uniform grid technique is a good technique for parallel geometric computation.

ACKNOWLEDGEMENTS

This work was supported by the National Science Foundation under PYI grant no. CCR-8351942. It used the facilities of the Computer Science Department funded by a DoD equipment grant, and the Rensselaer Design Research Center.

REFERENCES

Chazelle, Bernard and Edelsbrunner, Herbert (Chazelle,1988) "An Optimal Algorithm for Intersecting Line Segments in the Plane," *Foundations of Computer Science - 29th Annual Symposium*, White Plains (October 1988).

Franklin, W. Randolph (Franklin,1978) *Combinatorics of Hidden Surface Algorithms*, Center for Research in Computing Technology, Harvard University (June 1978). Ph.D. thesis

Franklin, Wm. Randolph (Franklin,1980) "A Linear Time Exact Hidden Surface Algorithm," *ACM Computer Graphics* **14**(3), pp. 117-123, Proceedings of SIGGRAPH'80 (July 1980).

Franklin, Wm. Randolph (Franklin,1982) "Efficient Polyhedron Intersection and Union," *Proc. Graphics Interface'82*, Toronto, pp. 73-80 (19-21 May 1982).

Franklin, Wm. Randolph (Franklin,1983) "A Simplified Map Overlay Algorithm," *Harvard Computer Graphics Conference*, Cambridge, MA (31 July - 4 August 1983). sponsored by the Lab for Computer Graphics and Spatial Analysis, Graduate School of Design, Harvard University.

Franklin, Wm. Randolph and Wu, Peter YF (Franklin,1987) "A Polygon Overlay System in Prolog," *Autocarto 8: Proceedings of the Eighth International Symposium on Computer-Assisted Cartography*, Baltimore, pp. 97-106 (March 29— April 3, 1987).

Franklin, Wm. Randolph, Chandrasekhar, Narayanaswami, Kankanhalli, Mohan, Seshan, Manoj, and Akman, Varol (Franklin,1988) "Efficiency of Uniform Grids for Intersection Detection on Serial and Parallel Machines," *Computer Graphics International*, Geneva (May 1988).

Franklin, Wm. Randolph, Chandrasekhar, Narayanaswami, Kankanhalli, Mohan, Sun, David, Zhou, Meng-Chu, and Wu, Peter YF (Franklin,1989) "Uniform Grids: A Technique for Intersection Detection on Serial and Parallel Machines," *10th International Symposium on Computer-Assisted*

Cartography (to be presented), Baltimore (April 1989).

Hillis, W. Daniel (Hillis,1985) *The Connection Machine*, MIT Press, Cambridge, MA (1985).

Kallstrom, Marta and Thakkar, Shreekant (Kallstrom,1988) "Programming Three Parallel Computers," *IEEE Software* **5**(1), pp. 11-22 (January 1988).

Kankanhalli, Mohan "The Uniform Grid Technique for Fast Line Intersection on Parallel Machines," (Kankanhalli,1988) M.S. Thesis, Electrical,Computer & Systems Eng. Dept., Rensselaer Polytechnic Institute, Troy, NY (April 1988).

Mulmuley, Ketan (Mulmuley,1988) "A Fast Planar Partition Algorithm, 1," *Proc. 29th Symp. on Foundations on Computer Science*, White Plains, pp. 580-589 (October 1988).

Nievergelt, J. and Preparata, F.P. (Nievergelt,1982) "Plane-Sweep Algorithms for Intersecting Geometric Figures," *Comm. ACM* **25**(10), pp. 739-747 (October 1982).

Preparata, Franco P. and Shamos, Michael Ian (Preparata,1985) *Computational Geometry An Introduction*, Springer-Verlag (1985).

Samet, H. (Samet,1984) "The Quadtree and Related Hierarchical Data Structures," *ACM Computing Surveys* **16**(2), pp. 187-260 (June 1984).

Sequent,1986. Sequent Computer Systems Inc.,, *Balance Technical Summary*, 1986.

Waugh, T.C. (Waugh,1986) "A Response to Recent Papers and Articles on the Use of Quadtrees for Geographic Information Systems," *Proceedings of the Second International Symposium on Geographic Information Systems*, Seattle, Wash. USA, pp. 33-37 (5-10 July 1986).

Wu, Peter Y.F. and Franklin, Wm. Randolph (Wu,1988) "A Logic Programming Approach to Cartographic Map Overlay," *International Computer Science Conference*, Hong Kong (December 1988).