International Hydrographic Review, Monaco, LXVII(1), January 1990

A DATA STRUCTURE FOR SPATIO-TEMPORAL DATABASES

by H. P. VARMA, H. BOUDREAU (*) and W. PRIME (**)

Abstract

The advantages and applications of spatial mechanisms are well documented; however, there are very few being designed. The principal hinderance to the design of spatial mechanisms is the great difficulty involved in specifying spatial problems and in interpreting spatial solutions. Similarly, the development of spatial codes to implement these techniques is held back by the lack of means to easily visualize and verify solutions, particularly in the realm of relational databases.

If spatial mechanisms are to be successful, the designer must be able to synthesize, analyse and evaluate, as well as load and extract information, using a single code representing a spatial structure. This entails the implementation of spatial relationships involving spatial data structures.

It is with this in mind that the Canadian Hydrographic Service database group embarked on the development of a new type of spatial database structure based on the quadtree concept.

INTRODUCTION

For the last five years the Canadian Hydrographic Service (CHS) has been investigating the feasibility and practicality of implementing a National Hydrographic Database. The research and studies, initiated by Ottawa, have pointed out that this is not an easy task. This is primarily because Hydrographic data is of a spatial nature, and does not fit properly into the niche provided by the Database Technologies of today.

Current Database Management Systems have been designed to provide easy, flexible access to a wide range of multifile databases involving numeric

(*) Canadian Hydrographic Service, Ottawa, Ontario, Canada.
(**) Atlantic Geoscience Centre, Bedford Institute of Oceanography, Dartmouth, Nova Scotia, Canada.

records, particularly in the domain of commercial data. However, they have been found wanting in the handling of spatial data. A database management system [1-2] is supposed to combine and use data in a variety of ways. This is conventionally done by assigning a data record to each object in the database, along with its simple attributes. Some GISs (Geographic Information Systems) have been built to manipulate spatial objects [3] in a limited two-dimensional region. Since 1977, several GIS have been built for handling spatial data, however current RDBMS (Relational Database Management Systems) cannot adequately and cost-effectively handle spatial data. This is primarily because it has been extremely difficult to establish relationships between spatial points without the explicit use of spatial mechanisms, such as polygons.

Faced with these problems, the database development group at the Bedford Institute of Oceanography realized that it would be necessary to invent a spatial data structure that could encompass spatio-temporal relationships. This data structure has been newly termed the Hyspat code (Hyperspatial code) and is based on the quadtree/octree concept.

DIFFERENCES BETWEEN D.B.M.S. AND G.I.S.

Another problem faced by the information world has been the confusion between a GIS (Geographical Information System) and a DBMS (Database Management System). A geographic information system [4] is a system designed to capture, store, manipulate, retrieve, display and locate data that are referenced to geographic locations. It is a depiction system usually written in a third generation language (eg. Fortran, C, Pascal, Ada etc.), with a graphics user interface (interactive graphic capabilities) for small subsets of the database. A DBMS, on the other hand, is an organized storage system for the entire database and utilizes fourth generation languages — SQL (Structured Query Languages) — for data manipulations and extraction. Each complements, but does not replace the other. This has caused conflicts between application programmers (3GL, GIS) and database programmers (4GL) and has resulted in disruption of the normal process of database evolution.

CURRENT STATUS OF SPATIAL DATABASES

In February 1988, the International Computer Science Institute in Berkeley, California, sponsored a two-day workshop at which 16 senior members of the database research community discussed future research topics in the DBMS area. One of the major points mentioned in the proceedings was the subject of spatial databases. To quote [5]: 'Several participants used spatial examples as hard applications. These were typically geographical databases containing encoded information currently found on maps. The problems varied from providing urban services (e.g. how does one get from X to Y efficiently on public transportation)

to conducting a military operation to environmental information systems integrating all kinds of data from under, on and over the earth's surface. There was widespread support for the importance of such applications, and the participants generally thought that this was a very good application area for extendible DBMSs. From this, it was determined that in order to utilize the true power of a relational database, the SQL libraries would have to be extended to incorporate spatial relations using the Hyspat code.

A REVIEW OF SPATIAL REPRESENTATIONS

QUADTREE

In recent years, the quadtree representation [6] has gained use as a data structure for applications in image processing, computer graphics and cartography. The quadtree is an approach to region representation that is based on the successive subdivision of an image array into quadrant (Fig. 1). Each

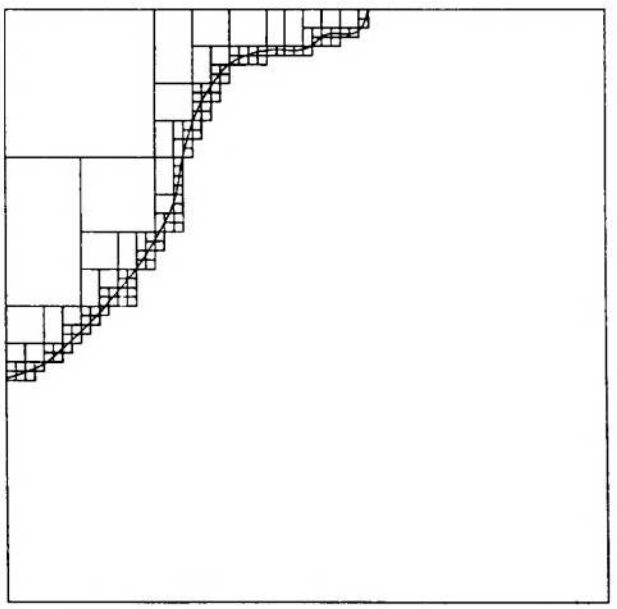
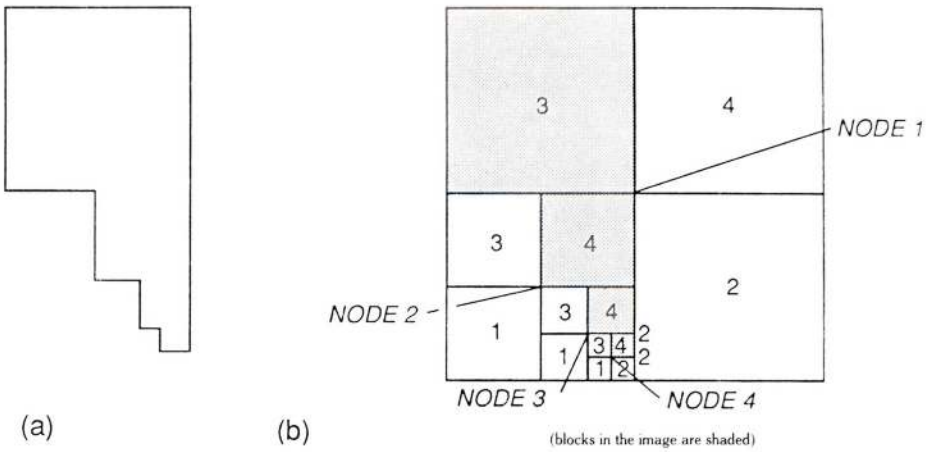


FIG. 1.— Complex areas broken down into subsquares.

node of the quadtree corresponds to a block in the original image [7]. A natural by-product of the tree-like structure (Fig. 2) is that many basic properties can be implemented as tree traversals. The differences in implementations lie in the nature of the computation at the node. Often these computations involve the examination of nodes whose corresponding blocks are adjacent to the block whose node (Fig. 3) is being processed. These nodes are called neighbors, and the process of locating them is called neighbor finding [8]. This is an important and frequent operation that is performed on spatial data sets.



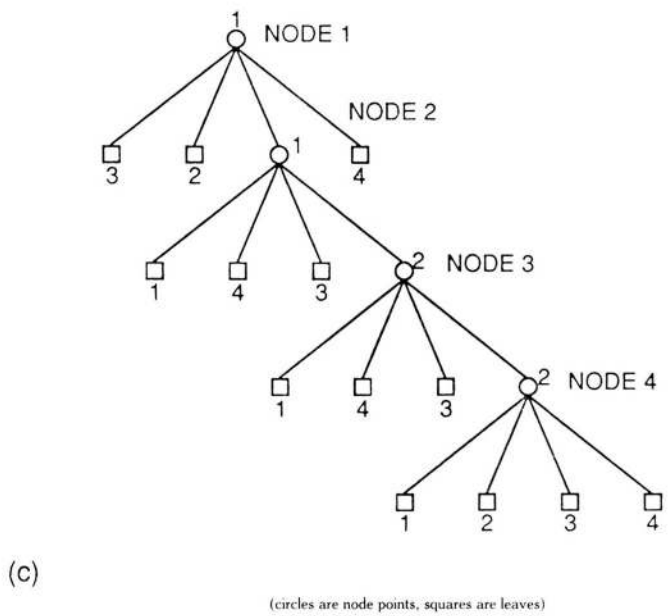


FIG. 2.— (a) region; (b) block decomposition of the region in (a); (c) quadtree representation of the blocks in (b).

OCTREES

Octrees are a natural extension of the quadtree concept [9]. In order to describe any object which we wish to represent within a given space, the best approach appears to be an initial rough approximation of the object's location, followed by successive refinements which increase the resolution of the object's details (Fig. 4). Such combinations of global and detailed specifications produce a hierarchical manifestation of the object space. For many years, this type of data

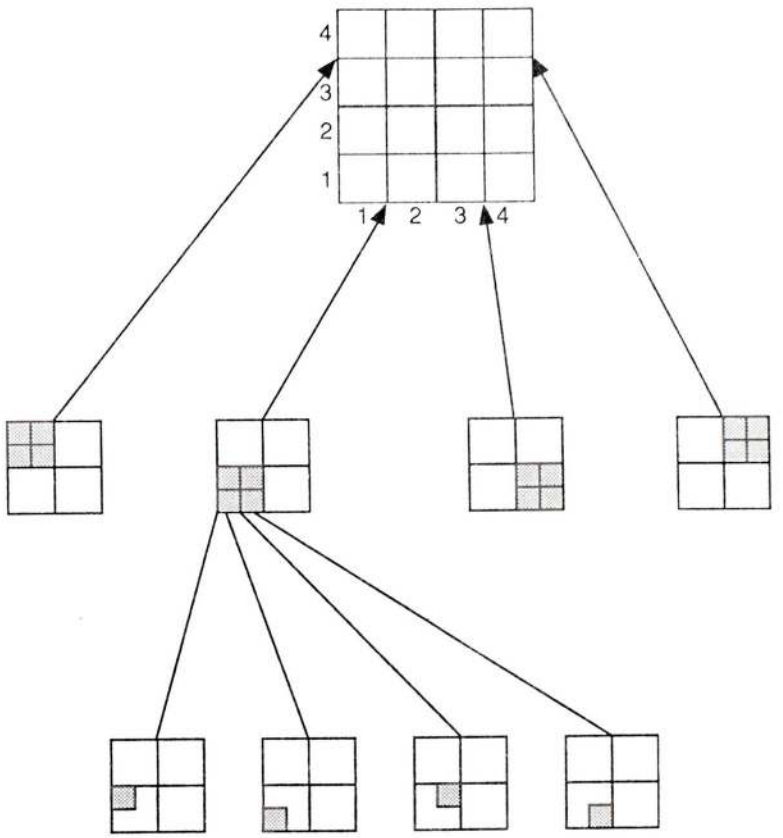


FIG. 3.— Nearest neighbor cells in quadtrees

representation was limited to pattern recognition [10]. This representation is also very useful for indexing space; that is, for specifying the location of any particular cube (Fig. 5). Because of its logical simplicity, the method of hierarchically representing three dimensional objects, called octree encoding, is exceptionally convenient for indexing 3rd space.

The octree [11] is a regular cellular breakdown of the object space (universe). The universe is subdivided into eight equal size cells. Normally, if any one of the resulting cells is homogeneous, (meaning that it lies inside or outside the object), the subdivision stops. On the other hand, if the cell is heterogenous (meaning that it is intersected by one or more of the object's bounding surfaces) the cell is divided further into eight subcells (Fig. 6). The subdivision process stops when all the leaf cells are homogeneous to some degree of precision.

The octree representation has several advantages [12]. First, any arbitrarily shaped objects (convex or concave with interior holes) can be represented to the precision of the smallest cell. Second, centre of mass, interference and volumes are easily calculated at different levels of precision. Third, because of spatial sorting and uniformity of representation, operations on octrees are simple and efficient.

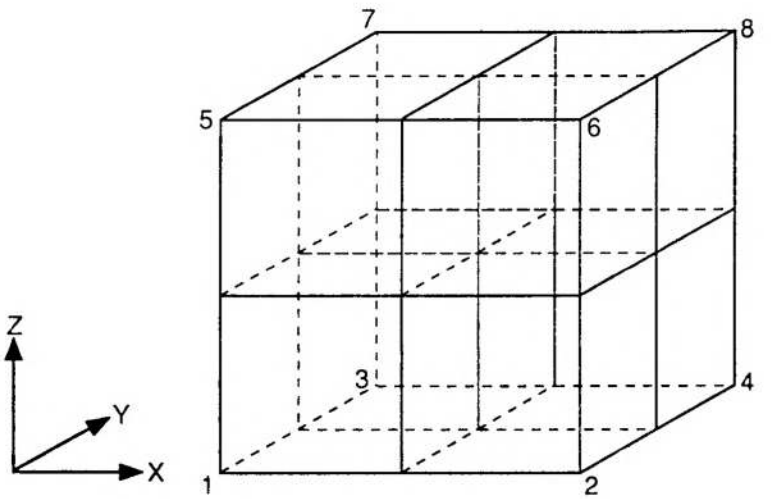


FIG. 4.— The order of octants.

VOXELS

Many approaches to data modelling are available. Most methods require some preprocessing of the input, to reduce data and to provide object representation better suited for the available display algorithms.

One representation, the voxel, corresponds closely to the format in which spatial data is collected. A voxel [13] is a rectangular volume element obtained by the division of three sets of mutually orthogonal parallel planes. The voxels which compose an object are usually the same size; that is, all the planes are equally spaced. On the other hand, the spacing of one set of planes need not be the same as the spacing of another. This will be explored in detail later.

One approach which has been widely applied to mapping systems uses the extraction of a set of 1-D primitives (contours) (Fig. 7, 8) describing the boundaries of the object on a slice-by-slice basis. Surface representations can be obtained from the contours directly, or indirectly, by tiling or spline techniques. A newer approach [14] retains 3D voxels (cubes) as primitives but achieves data compression through octree encoding, which provides an efficient object representation.

Associated with each voxel are three numeric coordinates representing its location in space, as well as attributes representing some object property at this location. The voxels can be converted to cubes by suitable interpolation. Such a dissection of object space into cubes is called a cuberille.

The general idea is that the voxels making an object are represented by a hierarchical tree structure, which achieves data compression through spatial coherence. An advantage of octree encoding is that simple operations such as union, intersection, difference of objects, translation, rotation, scaling, interference detection, binning and hidden surface removal can be accomplished by accessing

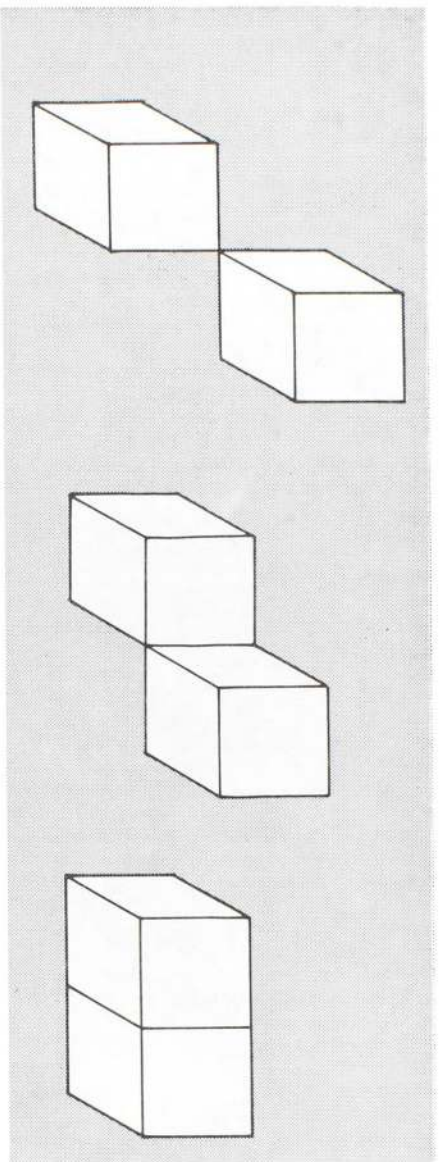


FIG. 5.— Nearest neighbors in cubes.

each node of the three only once. Furthermore, these operations require only simple arithmetic functions such as integer additions, shifts and comparisons, all of which are valid operations in a relationally based system.

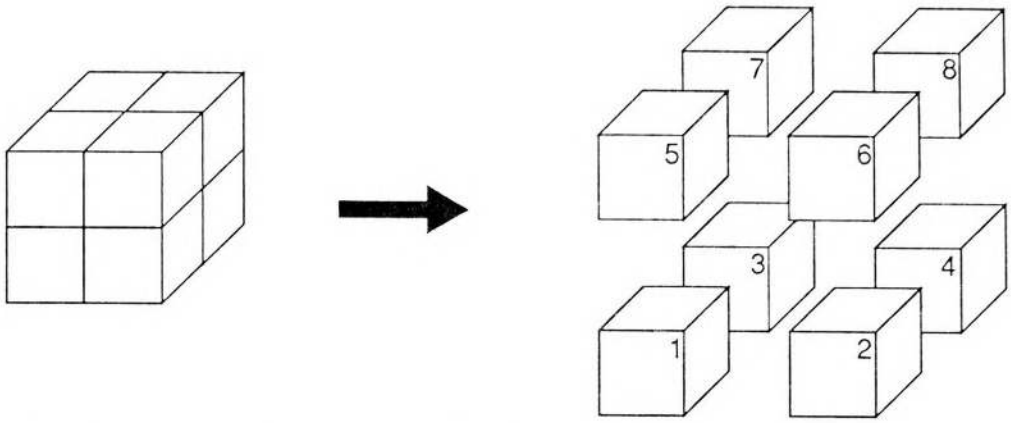


FIG. 6.— The subdivision of a rectangular cube into eight smaller cubes. On the left is the cube showing the locations of the cutting planes. On the right is an exploded view of the subdivided cubes showing the levels of the eight smaller cubes.

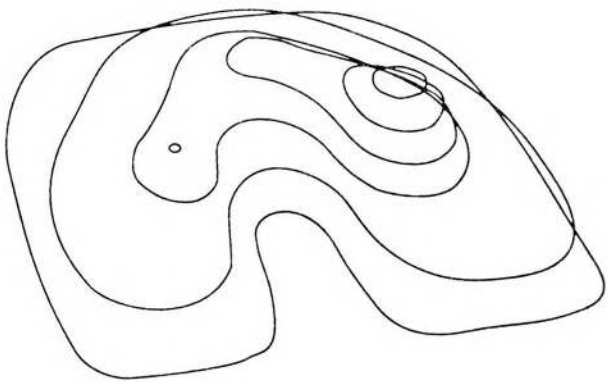


FIG. 7.— Contour lines.

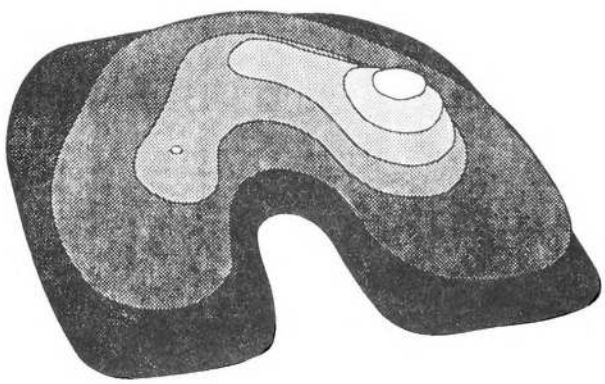


FIG. 8.— Contour lines.

TOPOLOGY

Boundary based, object based representations contain information about the surfaces of individual objects. The surface of an object is broken into one or more separate pieces, and each piece is fully described along with its own boundaries, which take the form of curves and joints.

This type of solid model must have the ability to describe how each surface piece fits together with each adjacent surface piece in the final model, so that a single, fully enclosed volume is formed. The adjacencies of these components can be derived by numerical techniques to analyze the geometric proximity of the surface pieces, though this approach is often computed intensive. However, in an evaluated representational form, such information is available explicitly.

This adjacency information is often informally referred to as the topology [15] of the solid model. The actual geometric surface description, curve descriptions and point locations are then referred to as the geometry of the solid model. The topology information can serve as a framework into which the geometric information is placed. The topology therefore serves as the 'glue' holding all the component information together.

LOGICAL SPATIAL DATABASE STRUCTURE

Two types of spatial representations are prevalent in computer technologies. One is based on the decomposition of objects [16] into their constituent parts (Fig. 9), and the other is based on the decomposition of object space (Fig. 10) into regularly shaped subspaces (quadtree, octree).

The first representation dates back to the 1963 Sutherland's Sketchpad System. Since then, many vector and raster graphics systems have provided both software and hardware support for structured spatial data. These hierarchical representations have been proven to be convenient for positioning objects and their components in space, and for moving objects relative to one another. In addition, they offer considerable memory savings when objects and object components occur several times in a scene. Objects and object components need to be defined only once, and can subsequently be found by the application of linear transformations in the hierarchy.

The second kind of spatial representation, the decomposition of object space, has been the focus of much recent research. In this case, the entire object space is divided repeatedly into cells or cubes, resulting in a tree structure. The leaf nodes do not contain primitives, such as edges and polygons, but approximate the object components by the cells or cubes to some degree of precision. This type of decomposition does not provide the memory savings offered by object decomposition. However, the spatial decomposition provides a robust representation applicable to a wide class of objects, and it allows fast computation of geometrical properties.

The traditional spatial data structures have been broadly classified as either topological or grid [17, 18]. Each has different advantages for representing

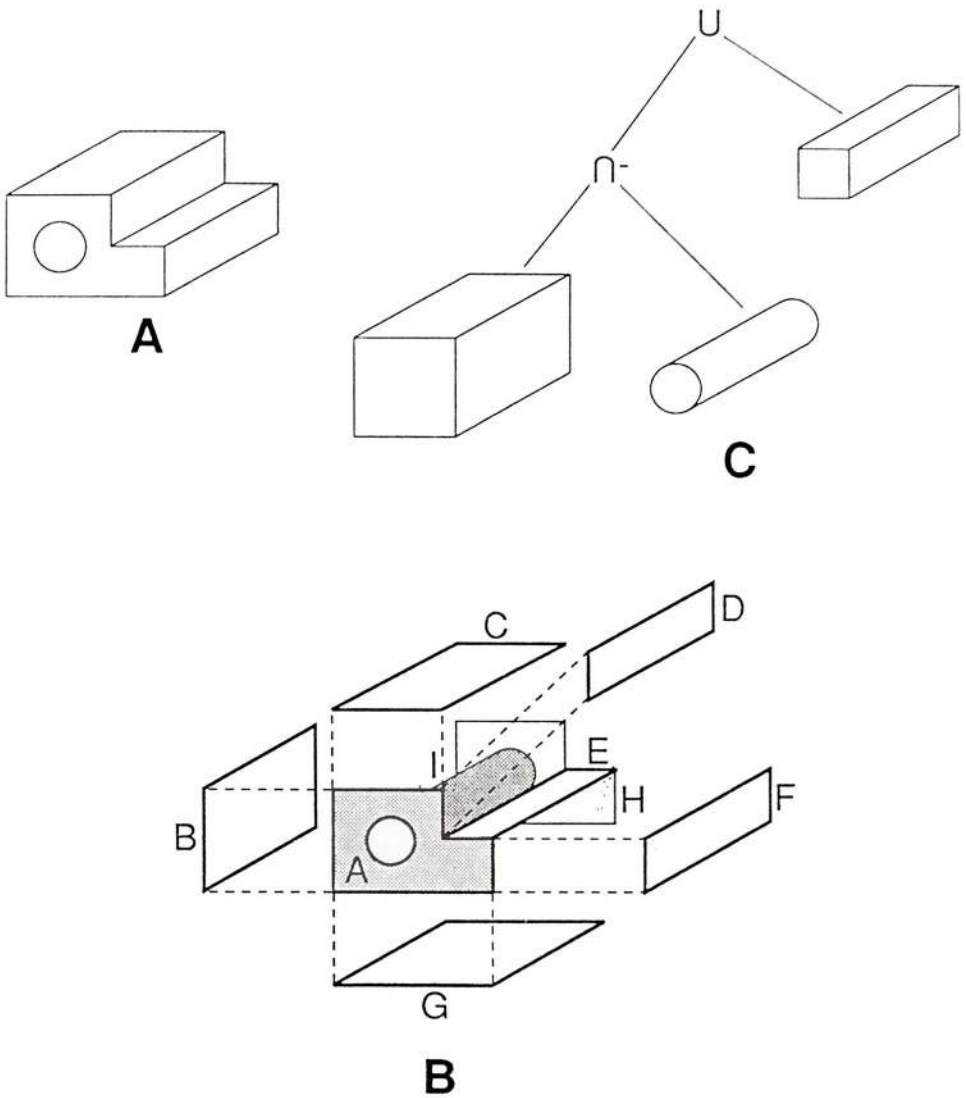


FIG. 9.— (a) A 3D object, (b) its boundary model, and (c) its component tree.

certain types of data and supporting spatial data processing operations. Topological data structures (Fig. 11) are ordered point sets such as isolated points, line segments (point pairs) and lists of points which outline geographical features. Gridded structures [19] subdivide the area of interest by a fine rectangular mesh (Fig. 12). For image computations, two-dimensional data representation have been accuracy dependent on resolution and data structure category.

Topological structures have traditionally provided better representation of legally defined and smaller-than-resolution-level objects. Such objects include:

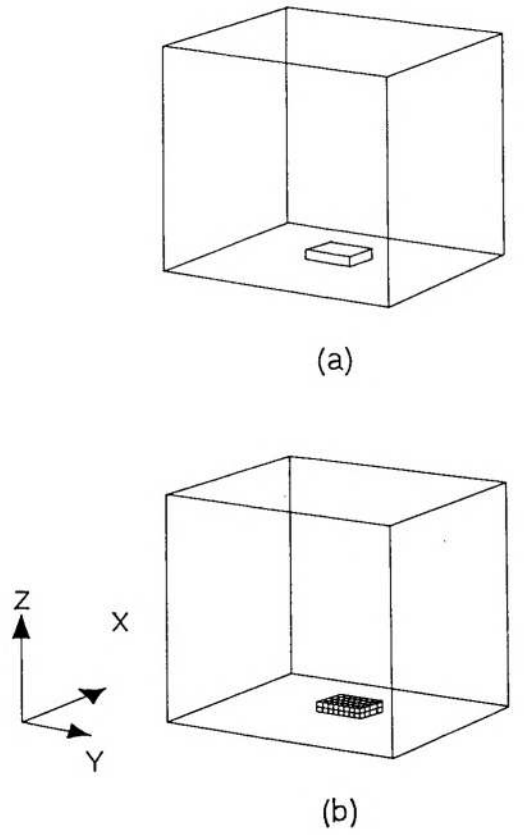


FIG. 10.— (a) A parallelepiped and (b) its corresponding octree.

- counties
- real estate parcels
- shoreline
- straight line boundaries between surveyors' landmarks
- cities/rivers (on small-scale maps) with measurements (area, width deviation) smaller than computer representation resolution (so that they appear as points or lines)

Gridded systems, on the other hand, were better in handling fuzzy boundaries, such as contours. However, within the last ten years, image analysis techniques have developed new data structures based on quadtree and octree concepts. These new structures are radically changing the traditional concepts of spatial datasets.

THE CHOICE OF DATA MODELS

Global mapping models (ie. mosaics of polygons) are universally accepted as the key to efficiently organizing spatial data sets. Regular recursive elemental polygons with implicit spatial relationships (eg. adjacencies) have been shown to

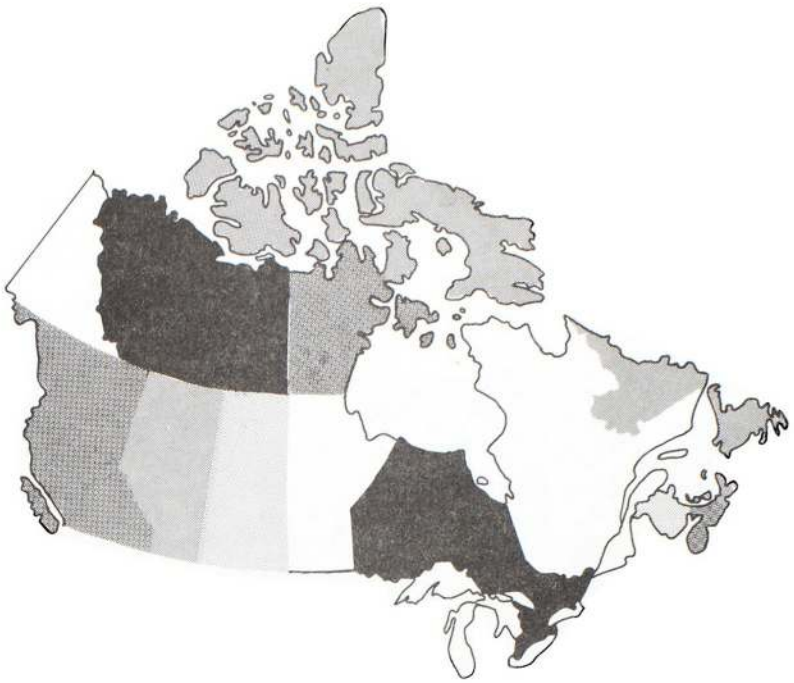


FIG. 11.— Cartographic polygons.

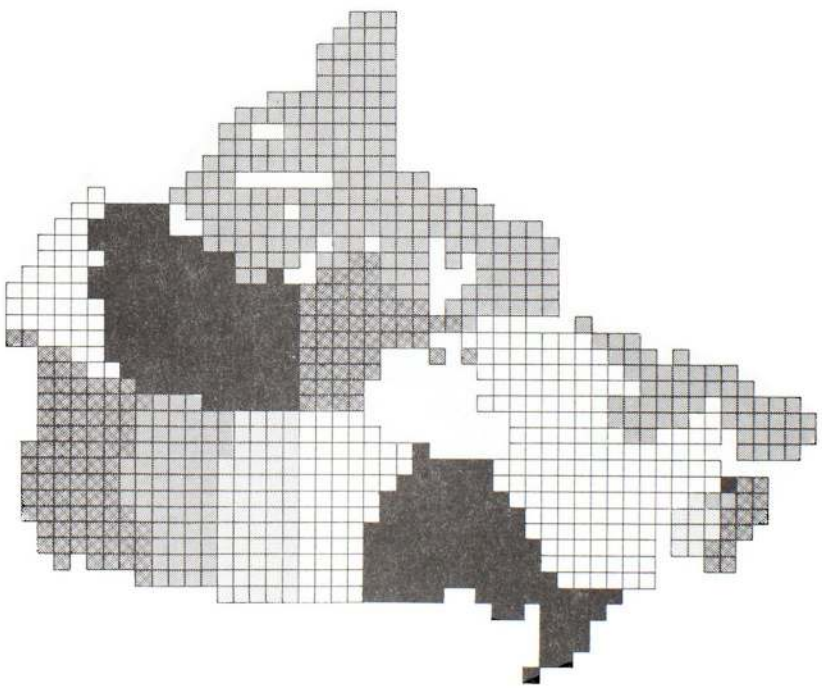


FIG. 12.— Gridded polygons.

serve as effective data models, upon which conceptual and logical implementations can be built.

Of the three practical mosaics of squares, triangles and hexagons, none appear to enjoy a distinct advantage in terms of computational complexity or storage efficiency [20]. The simplest is the square; that is, a contiguous non-overlapping mosaic of squares. It is also the most widely used and researched [21].

The specific programmable implementation of the data model is a data structure. In this case, the implementation data structure is based on successive subdivision of squares, which is conceptually defined as a quadtree.

THE HYDROGRAPHIC HYPERSPATIAL CODE

The Hydrographic Hyperspatial Code (Hyspat Code) that is being introduced is a variant on the quadtree/octree structure. It is initialized as a single square world coordinate system, based on (0,0) at the south pole to (360,180) at the north pole (Fig. 13). It is an information driven system, in that a limit is placed on the number of data points permitted to reside within one cell. Whenever a cell overflows in terms of data volume, (for initial prototyping there reside 100 000 points per cell) it is subdivided into four smaller quadrants (Fig. 14) and the process is repeated as the database is loaded.

Each quadrant in database terminology is, in fact, a table. Each table, when subdivided, becomes a node point, whose identity and implicit spatial attributes are stored on a coded string (ie. the table's name). Each node point on the string signifies a division by two of the limits of the quadrant. With this structure, one can see an orderly breakdown of space into smaller quadrants with respect to cell size. In other words, one can continue subdivision until a single point resides in a single cell.

The model differs from traditional models in that there is no attempt made either to maintain equality of geographic areas among cells or to approximate the spheroidal shape of the earth. It is simply a subdivision of geographic coordinates.

Fully functionality of this data structure can be achieved by computing each latitude and longitude down to the 30th level (Fig. 15) on the nodal string, which gives a final resolution of 1.8×1.8 centimetres on the spheroid. This nodal string (Hyspat code) is stored in the database as an attribute attached to every spatial data point. For the prototype implementation, this was chosen as 30 characters, coinciding with the Oracle limit on the table name length.

The Hyspat code is a character string which fully describes the subdivision limits. The power of this structure can be realized as the implied spatial relationships are achieved through partitioning the data into manageable-sized cells, while spatial operations are done by pattern matches on strings defining cellular boundaries. The length of the string defines the size of the cell, while the nodal points on the string define the location of the cell. The final nodal point, along with a pattern match on the string, can also determine the nearest neighbor cells. Thus, by using substrings and key attributes, one can dynamically establish spatial relationships between areas or points without the use of polygons. With this capability, one can generate topologies using 4th generation SQL queries.

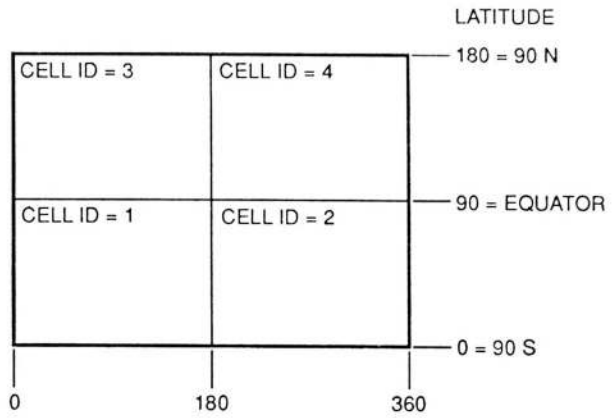
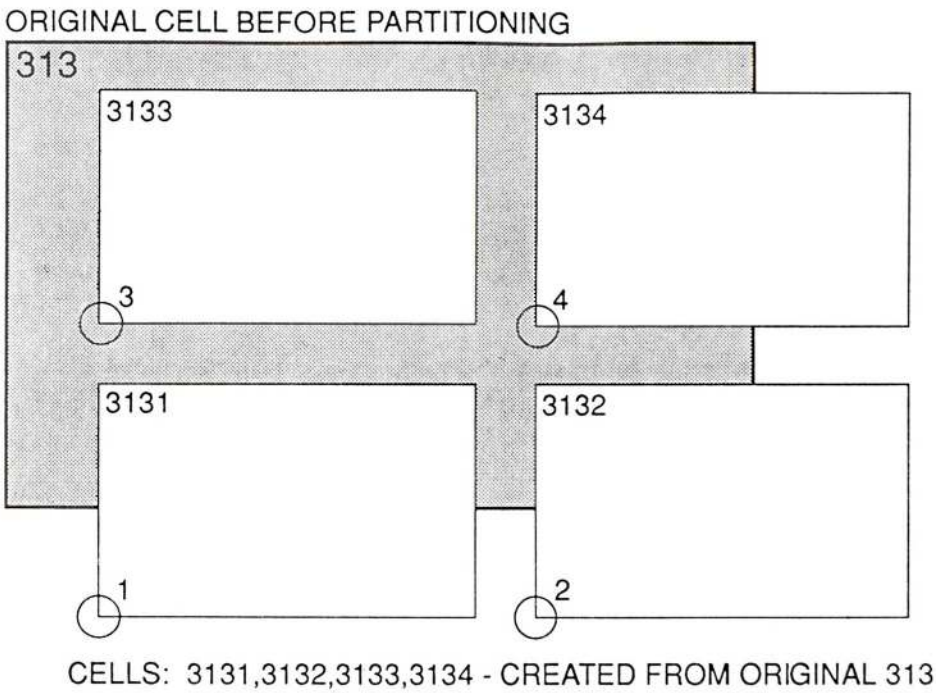


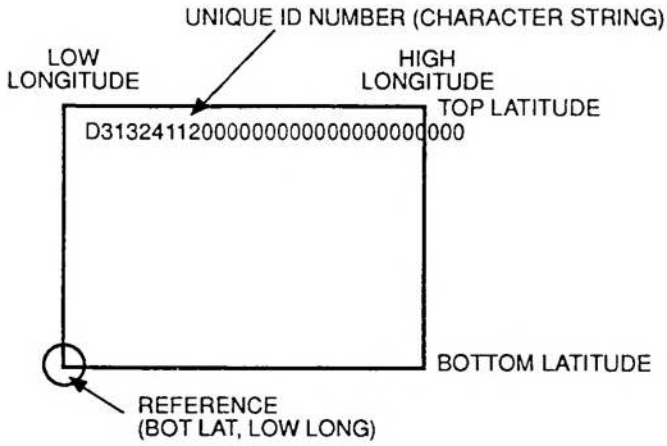
FIG. 13.— Cell initialization at startup.



NB: ON PARTITION COMPLETION CELL 313 IS DROPPED (DESTROYED).
NUMBER OF CHARACTERS IN CELL ID IS IMMEDIATE COUNT OF THAT CELL'S LEVEL OF REPARTITIONING.

FIG. 14.— New references generated:

1 — Ref-lat 3131 = Ref-lat 313+0, Ref-lon 3131 = Ref-lon 313+0;
2 — Ref-lat 3132 = Ref-lat 313+0, Ref-lon 3132 = Ref-lon 313+1/2 (delta-lon)0;
3 — Ref-lat 3133 = Ref-lat 313+1/2 (delta-lat), Ref-lon 3133 = Ref-lon 313+0;
4 — Ref-lat 3134 = Ref-lat 313+1/2 (delta-lat), Ref-lon 3134 = Ref-lat 313+1/2 (delta-lon).



LATITUDE = REAL WORLD COORDINATES +90
RANGE (0-180)

LONGITUDE = REAL WORLD COORDINATES
RANGE (0-360)

ATTRIBUTES:

ID	UNIQUE 30 CHARACTER STRING FROM WHICH REAL WORLD COORDINATES CAN BE DERIVED
COUNT	NUMBER OF DATA POINTS IN THE CELL
SMAX	NUMBER OF DATA POINTS ALLOWABLE BEFORE DYNAMIC REPARTITIONING TAKES PLACE
ONLINE	FLAG INDICATING THE CELL IS ON OR OFF-LINE CURRENTLY

FIG. 15.— One cell (partition).

TEMPORAL APPLICATION

The explanation here has been two dimensional. If the third dimension is applied to the structure, the logic does not change. Instead, the patterns become like octrees and the cells become cubes that are subdivided according to the same logic, until a single point resides in a cube. This puts a three dimensional coordinate system into a single key. Another aspect is to make TIME the third dimension, with the z values defined as functions of time (Fig. 16). This in effect can put a four dimensional key into a single string, which represents the translocation of a three dimensional object or voxel through time (t,x,y,z), This time data elements is newly termed as a Toxel. This same key can be represented as a binary string, which can facilitate searches by orders of magnitude simply by doing Exclusive ORs or bit masking, thereby greatly speeding up the overall search process of pattern matching. Such types of structures and methodologies can greatly enhance the techniques of trend analysis and temporal analysis. Hence the Hyspat code provides a new way to depict spatio-temporal information, which has so far eluded conventional database technologies.

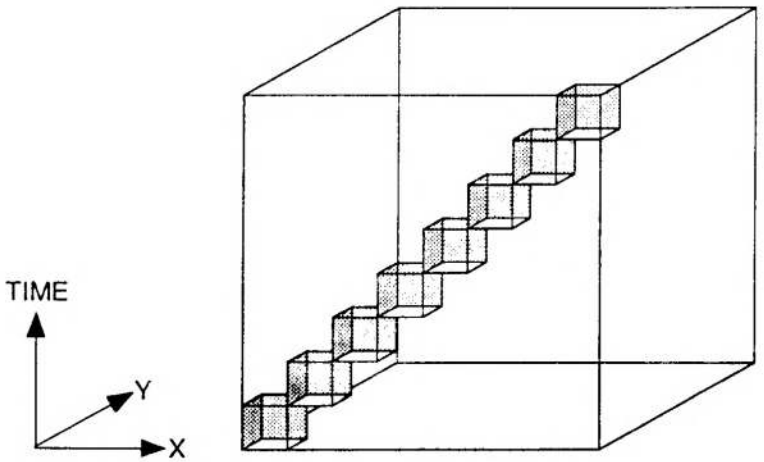


FIG. 16.— Hypercubes moving through time.

FUNCTIONALITY

The quadtree representation was chosen for the detailed explanation of the Hyspat code in order to facilitate the understanding of the structure by the reader. This was done because the author recently gleaned the fact that people are three dimensional entities who tend to think in two dimensions.

The temporal relationships can be implemented by going to octree encoding of one to eight, rather than the quadtree encoding of one to four. The logic applied is exactly the same.

The fundamental logic of the Hyspat code is that it is a dynamic stacked image database structure. If one sorts the data using the Hyspat code as the key, one finds that the data is effectively clustered in space/time. This permits:

- explicit relationships of entities over the same area
- efficient data retrieval
- efficient input and storage of large data sets
- user ease in adding two dimensional information

The dynamically stacked Hyspat code data structure consists of tables, with the table name being the matched substring of the Hyspat code string attribute, thereby providing the implicit spatial relationships. All Hyspat codes having the same substring are grouped into these tables. When a table exceeds the user dictated size level, (100,000 points), it subdivides into four subsections. The table names increase by one level by storing the nodal point on the string.

example:

Table D413121130000000000000000000000 overflows

This table is subdivided into

D413121131000000000000000000000
D413121132000000000000000000000
D413121133000000000000000000000
D413121134000000000000000000000

The information is poured from table

D4131211300000000000000000000

into the above tables and the originating table is dropped.

The digital data are 'stacked' by sorting on the Hyspat code. Then all information values for each cell are stored in a single table. This type of data structure allows a choice of data types related to space. It yields simple sequential uniform format files that can be used by virtually any computer system. Thus, data of any region, taken at different times, can be combined and added to symbolic information about the same area at any time. This dynamic storage allocation offers:

- simplified and efficient grouping of data due to string matches between Hyspat codes
- effective geographic windowing of data where each table is named as a substring of Hyspat code
- ease of access to offline storage, where entire tables are exported to individual media, such as magnetic tapes or optical storage.

The quadtree data model, as opposed to a topological representation, achieves:

- superior representation of the subtle variations in spatial data
- versatility in data manipulation
- storage efficiency

A significant point is that one can easily compute relative distances between cells. For example, one can do an Hyspat code string match between two points, and if they match up to the 30th level then the data points are approximately 1.8 cm apart. This resolution is achieved at the worst case on the equator. If the string length is increased to 35 levels, the final resolution of the spatial cells of the world is 5.7 $\times$ 5.7 mm. With the limits of the temporal hypercube set at 0 to 1024 years, and utilizing the same spatial limits of 0 to 360 and 0 to 180 degrees, a final resolution of the temporal cell is 5.7 $\times$ 5.7 mm $\times$ 1 second. If one further extends the string to 70 characters the toxel resolution becomes .ooo8 seconds $\times$ 1.65*10 to the power -10 mm $\times$ 1.65*10 to the power -10 mm for the entire world for a time period of 8,673, 820,672 years. These 70 characters can be represented using octal binary string (Fig. 17) representation of 210 bits.

Each level on the string signifies a division by two of the cell's boundaries. In this manner, one can have a length associated with the latitude. This would provide approximate distances without doing any major computation. The significance is that this type of proximity determination among attributes, or groups of attributes, or even different entities, can be combined with user defined functions (e.g. statistics of groupings, binning, Krigging, depth contouring) without compute intensive geographic calculations. The polygon structure cannot handle such variations. If contours are available, the determination of minimum distance between them would be exceedingly expensive, but with the Hyspat code, spatial relationships can be dictated between points without the use of polygons. Therefore, the generation of topologies from the data becomes a possibility [22].

Yet another interesting facet of this structure is that one can compute string length with respect to positional error. In this manner, implicit error statistics for

TOXELS 35 CHARACTERS LONG REPRESENT

THE WORLD 0 - 180 LAT 0 -360 LONG FOR 1034 YEARS

RESOLUTION

5.7MM X 5.7MM X 1 SECOND

BINARY OCTAL REP 105 BITS

TOXELS 70 CHARACTERS LONG REPRESENT

THE WORLD 0 - 180 LAT 0 -360 LONG FOR 8,673,820,672 YEARS

RESOLUTION

1.65*10EXP-10MM X 1.65*10EXP-10MM X .0008 SECONDS

BINARY OCTAL REP 210 BITS

FIG. 17.— Toxel representations.

the data can become inherent in the data string by zeroing the least significant nodal points.

example:

the total nodal string is

D4131211321134222133211142143

With a positional uncertainty of + 0.64 metre, the string becomes

D4131211321134222133211100000

This establishes confidence levels without adding additional error attributes to the database.

CANADIAN HYDROGRAPHIC SERVICE DATABASE

A major problem with conventional processing packages is that they are too rigid, not allowing new manifestations of the data without major overhauling of the software packages. Simple flat file management is used at the cost of storage wasted in unused blank fields. New attributes are required to conform in name, type and length to the predefined fields. System expansion is limited by the number of spare fields available for new data. However, in a DBMS, the attribute structure is variable rather than fixed, permitting new fields to be added at any time.

The CHS DBMS [23] differ from other conventional database systems since it is largely a spatio-temporal database. It differs in that most queries or operations require a large degree of spatial manipulation. This data manipulation is done by data selection subject to scale, such as shallow-biased and deep-biased overplot removal [24], and by superceding data (temporal data) by a more recent dataset. The CHS data manipulation capabilities use basic algorithms that can be varied to perform a variety of operations on the digital data. These algorithms

can be successfully implemented on the database because the stacked data structure is highly effective in supporting these operations. The basic methodology is to move a window over the stacked data. At each stop, all data from each cell inside the window can be read and the data in any active cell can be modified. The user then can specify the general type of action to be taken, such as:

- add, replace, delete, extract or supercede data
- localize the logical definition of the active cells (for which processing will take place)
- define logical and arithmetic operations to be performed within each cell
- define the nature of any new data to be created

The primary keys to interrelate project numbers, charts, vessels, establishments, collection platforms and surveys are space and time. These elements will reside in catalog tables which point spatially, using Hyspat codes, to the relevant cells within the data base. Each cell, in reality, is a geographic window (containing data) which can interrelate spatially and temporally with various sources. The source tables will have time stamps (which gives temporal aspects to the data base), indicating which agencies were in a particular geographic area during a finite period of time

example:

vehicle/start year/julian day/end year/julian day/Hyspat code
Baffin/1987/126/1987/220/H1121314211300000000000000000

CHARTS AND AREA DEPICTION USING JOIN TABLES

A previously stored image (graphic portrayal) is retrieved by accessing image tables, which contain unique pointers using time elements and source id (source identification). These pointers are join tables that extract image data from the whole data set. The user obtains these image tables by specifying window limits, using latitude, longitude, boundaries as well as scale. A conventional GIS then is used to perform subjective decisions on the data, such as interactive editing and overplot removal (data thinning). The time elements and source id are then obtained from the GIS and loaded into image tables within the Database. The time elements and source id serve as pointers to subsets of the database. In this manner, several image tables can access the same data points without adding redundancy to the database. Thus, image overlaps and scale changes become trivial to overcome.

The time elements and source id for each selected data element is always unique, because no single collection platform can be in two places at the same time. This is one of the reasons it is advocated that time element fields should be filled, using system time at the moment of digitizing historical graphical documents, such as linen backs or field sheets. The cursor, similar to a collection vehicle, cannot be at two places at the same time. The source id, which is the name of the collection vehicle or digitizer table, determines whether or not the time is valid, in that the time represents actual collection or pseudo time obtained from the computer system at the moment of digitization. From this, the time elements along with source id can be seen to provide a unique key to perform joins.

TABLE ARCHIVAL

Very Large Databases are primarily in static mode, resting on secondary storage media such as magnetic tapes or optical disks. Most of the operations involved on the database are updates and extractions of secular partitions of the database. This logical user access method will be supported by export/import capabilities. The access method will also be chosen by the system on the basis of available memory size and the quantity of data needed simultaneously to fulfill the user request. This is based on the assumption that the total database cannot reside simultaneously on the system, due to lack of disk storage or memory. This fits in nicely with this type of schema, as only subsections of the database are placed in dynamic mode on the system at user request.

Most large digital databases are stored on tape, a strictly sequential medium. However, with the advent of read/write capabilities on optical disks, this medium could rapidly go the same route as vinyl long playing records (with the advent of compact disks). The CHS database schema can work on both mediums. However, the spatial structure will be more efficient with random access capability available on the optical disk. This could drastically improve access time. If the user accesses more than one cell at a time, and if all the cells of the required data cannot be stored on the system, the system can read the data from secondary storage devices. This would give it the capability to perform operations on the cells sequentially. Also, the updated records can be written out in parallel with the input process. In this manner, a divide and conquer principle can be adhered to without causing a system overflow.

CONCLUSION

The dynamically stacked Hydrographic Hyperspatial database structure, and the accompanying data manipulation facilities, effectively cover the types of data storage and operations required by a spatial database.

The database structure permits direct comparison of variables over the same area. It can store and retrieve data efficiently, as well as combine information without loss of data which occurs in other systems. This type of capability allows the user much greater ease and flexibility in data manipulation than is available in most other current GIS or database systems.

The data accessing approach in this system consists of the ability to retrieve, add, replace, delete, supercede and perform operations on the multi-variable data included in the window. The data manipulation operations can be applied to data in each cell. Various methods of physical storage and searching of the stacked data are possible.

This system illustrates the great flexibility and generality of the data structure. Relational Spatial analysis can be done directly on the data, without the implementation of polygonal windowing that is currently in use in other systems. Finally, with the advent of spatio-temporal structures such as the Hyspat code,

the existing database technologies can blaze a new trail into the information world.

References

[1] Gio WIEDERHOLD: Databases, IEEE Computer, October 1984, pp. 211-222.

[2] Richard SHEY, Gio WIEDERHOLD: Data Engineering and Information Systems, Stanford University, IEEE Computer January 1986, pp. 18-30.

[3] H.P. VARMA: An Interactive Editor for Hydrography, *Canadian Hydrographic Service Lighthouse*, March 1985.

[4] C.L. MacDONALD, I.K. CRAIN: Applied Computer Graphics in a Geographic Information System: Problems and Successes, Canada Land Data Systems Environment Canada, IEEE Computer Graphics and Design, October 1985, pp. 34-39.

[5] E. NEUHOLD, M. STONEBRAKER: Future Directions in DBMS Research, TR-88-001 International Computer Science Institute, May 1988, 947 Center St. suite 600 Berkeley, California, 94704-1105.

[6] H. SAMET: A top-Down Quadtree Traversal Algorithm, University of Maryland IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-7 no.1, January 1985, pp. 94-97.

[7] H. SAMET, M. TAMMINEN: Computing Geometric Properties of Images Represented by Linear Quadtrees, University of Maryland, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-7 no. 2, March 1985, pp. 717-720.

[8] H. SAMET, C.A. SHAFFER: A Model for the Analysis of Neighbor finding in Pointer Based Quadtrees, University of Maryland, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-7 no. 1, November 1985, pp. 717-720

[9] H. NOBORIO, S. FUKUDA, S. ARIMOTO: Construction of the octree Approximating Three Dimensional Objects by Using Multiple Views, University of Toyonaka, Osaka, Japan, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 10 no. 6, November 1988, pp. 769-781.

[10] B.B. CHAUDHURI: Applications of Quadtree, Octree, and Binary Tree Decomposition Techniques to Shape Analysis and Pattern Recognition, Indian Statistical Institute India, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-7 no. 6, November 1985, pp. 652-660.

[11] I. CARLBOM, I. CHAKRAVARTY: A Hierarchical Data Structure for Representing the Spatial Decomposition of 3-D Objects, Schlumberger-Doll Research, Ridgefield, Connecticut, IEEE Computer Graphics and Applications, April 1985.

[12] H. SAMET, R.E. WEBBER: Hierarchical Data Structures and Algorithms for Computer Graphics, University of Maryland, IEEE Computer Graphics and Applications, July 1988.

[13] I. GARGANTINI, T.R. WALSH, O.L. WU: Viewing Transformations of Voxel-Based Objects Via Linear Octrees, University of Western Ontario, London, Canada, IEEE Computer Graphics and Applications, October 1986.

[14] A. KAUFMAN, R. BAKALASH: Memory and Processing Architecture for 3-D Voxel-Based Imagery, State University of New York, Stony Brook, N.Y., IEEE Computer Graphics and Application, November 1988.

[15] M. AGOSTON: Algebraic Topology, Marcel Dekker, New York, 1976.

[16] T.C. WOO: Interfacing Solid Modeling to CAD and CAM: Data Structures and Algorithms for Decomposing a Solid, University of Michigan, IEEE Computer, December 1984, pp. 44-49.

[17] D.C. GOSSARD, R.P. ZUFFANTE, H. SAKURAI: Representing Dimensions, Tolerances and Features in MCAE Systems, Massachusetts Institute of Technology, IEEE Computer Graphics and Applications, March 1988, pp. 51-59.

[18] D. LAURENT, S. MOTET: Geomatic: A 3-D Graphic Relief Simulation System, University of Paris VII, France, IEEE Computer, December 1984.

[19] H.P. VARMA, H. BOUDREAU: Probability of Detecting Errors in Dense Digital Bathymetric Data Sets by Using 3-D Graphics Combined with Statistical Techniques, Canadian Hydrographic Service Proceedings, CISM Conference, June 1989.

[20] N. AJUA: On Approaches to Polygonal Decomposition for Hierarchical Image Decomposition, Computer Vision Graphics and Image Processing, 1983.

[21] W. TOBLER, Z. CHEN: A Quadtree for Global Information Storage, Geographical Analysis, 1986.

[22] T.L. KUNII, T. SATOH, K. YAMAGUCHI: Generation of Topological Boundary Representations from Octree Encoding, University of Tokyo, Japan, IEEE Computer Graphics and Applications, March 1985.

[23] H.P. VARMA, A.J. KERR: Hydrography and the Digital Era, Canadian Hydrographic Service, *International Hydrographic Review,* 1987, Monaco.

[24] H.P. VARMA, M. JAY: Histerics in Hydrography, Canadian Hydrographic Service, In-House Publication, 1987.

Acknowledgements

Dr. D. Wells	University of New Brunswick
Dr. Y.C. Lee	University of New Brunswick
Dr. D. Gregory	Bedford Institute of Oceanography
Mr. R. MacNab	Bedford Institute of Oceanography
Mr. A. Sherin	Bedford Institute of Oceanography
Dr. B. Loncaravic	Bedford Institute of Oceanography
Mr. P. Bellemare	Bedford Institute of Oceanography
Dr. J. Verhoef	Bedford Institute of Oceanography