ADVANCES IN GEOGRAPHIC INFORMATION SYSTEMS

Stanley H. Collins
George C. Moon
Timothy Lehan

Collins & Moon Limited
Guelph, Ontario, Canada

ABSTRACT

Research into spatial information systems has disclosed that geographic information systems (GIS) have lagged behind important developments in computer science, information theory and interface design. Basic data structures such as vectors, strip trees and quad trees are discussed. Summaries of the properties of hierarchical, network and relational DBMS are given. Recent development of a relational spatial data base with variable-length and matrix records (RSDB) is described. It is shown how RSDB may serve as a controller for other DBMS, including the usual hierarchical models.

The enhancement of information content is seen as a new function of the DBMS, making the topology and numerical attributes of features available with their coordinates. Improvements in data management and information content are seen as necessary for achieving the final goal, the improvement of the human interface. Any mode of interaction (keyboard, cursor movements, voice etc.) should be available to the user for any application; and the choice should not be dictated by the particular query or data type.

1. INTRODUCTION

Geographic information systems (GIS) have received considerable attention recently, and were a central topic in the AUTO-CARTO SIX conference held in Ottawa in October, 1983. The ones that have been created vary in type, from mere descriptions of methods of organizing data, to complete systems of hardware and software for creating digital geographic data bases and for mapping from them. As the implementation becomes more complete and definite, the scope of the system tends to become limited. This is natural enough; the mere task of naming and classifying geographic features is a large one, and the task of creating a comprehensive geographic data base for a province or country is so formidable that it has never been fully accomplished.

Recent research by the authors has disclosed also that the designs and implementations of geographic information systems have lagged behind developments in computer science, including information theory, data base management theory, and interface design. Three basic improvements to GIS are considered in this paper. One is the design of a system of data base management systems (DBMS) and of means of navigating through them. Another is the enhancement of the information content of the basic data by including its semantics and topology; that is, by improving the data model. The third is the improvement of the human interface with the system; and when this aspect is interpreted as broadly as it should be, it is really the most important goal of GIS development.

2. THE DATA MODEL

The basic data model of most GIS is a list of feature names, with instances of the named features located by coordinates. This simple model contains no help for the user in either semantics or topology of the features.

The need for help with semantics is exemplified by the word "contour". Most cartographers will give it the meaning of a representation of a level line, but even in strictly geographical usage it may mean the outline of an areal feature or of the plane projection of a solid figure. Another example is the word "road". There are many different types of roads, even if the meaning is limited to something that carries vehicular traffic; but consider the meaning "a place, less enclosed than a harbour, where ships may ride at anchor". This takes us all the way from

topography to hydrography, and makes it evident that context must be appreciated by the system. The data model must also take topology into account, to some extent at least.

In the strict mathematical sense, the topology of geographic features and combinations of features depends only upon their 'connectivity'. For example, a sphere and a cube are topologically similar or homeomorphic, as are a contour line and the outline of a lake. These are simply-connected solids and surfaces. An example of a multiply-connected solid is a doughnut; and of a surface, a lake with islands. It will be convenient to extend the notion of topology to include the complexity of geographic shapes. A simply-connected surface (one with no holes) may still have a highly complex outline. One way of quantifying complexity of a shape is to determine the maximum number of intersections of a general straight line with the outline of the shape. We will also include under topology the properties of adjacency and intersection of like or unlike features.

2.1 Geographic Data Structures

A large class of geographic data is that which names, encodes, describes and locates topographic features. However, topography is only one branch of geography, and geographic data may belong to many other branches such as human geography, meteorology and so forth. When all such branches are included, geographic data structures and formats may take on any conceivable complexity. For the time being, however, we will consider the data structures that pertain only to information that is associated with topographic mapping, and to thematic mapping that requires a topographic base.

With this limitation, GIS must handle the following feature elements:

a) Points
b) Lines (open-ended)
c) Outlines (closed polygons)
d) Areas (groups of adjacent area elements)

The data structures consist of these elements along with the data that are required to locate, identify and describe the feature to which they pertain. The data structures may also carry some topology, the simplest type being that which relates the elements above to one another; for example, the data structure for an area may carry references to the lines which delimit it, and to the nodes in which these lines intersect.

The data structures of existing GIS are quite simple. They deal with an individual point as a name or code with a set of coordinates; a line as a coded list of coordinate sets; and an area as a coded set of lines, or as a coded set of area elements, each of which may carry a vector of properties.
There are other types of data structure, such as run-length encoding, that are used within some GIS without being shown explicitly; and there are others to be described below that do not appear to have been applied at all.

2.1.1 Run-length encoding

Run-length encoding is a means of condensing the information carried by a raster image or a rasterized thematic map (see Figure 1). When a particular type of area element or pixel is first encountered in a row-by-row traverse of the image, that pixel is given an opening flag. If a number of pixels with identical properties are then encountered sequentially, they are counted; and a closing flag is given to the last of them. Only the positions and codes of the flagged pixels and the counts are stored. The storage and the reproduction of an image or thematic map is greatly simplified by this technique, if the homogeneous areas are mostly large compared to the size of the pixel. The flagged pixels are also available for drawing line maps from an image, and a converse process can be used to create images from line maps. This and the following methods involve overhead which may not always be justified.

2.1.2 Strip Trees

The strip tree is a data structure for storing linear features. Consider a line (Figure 2a) that is made up of many points, and assume that it contains many more points than are required to define the line to the desired accuracy (this situation arises often with stream-digitized lines). A rectangle is computed that encloses all the points of the line with its long dimension parallel to the straight line joining the end-points. The farthest distance of a point from this line is calculated, and if it is within the required tolerance for locating the line, the job is finished and the straight line is taken as the representative of the original. If not, two rectangles are computed on the sections of the line defined by the end-points and the point that lies farthest from their join (Figure 2b). The process is repeated until the width of the enclosing rectangle is within the tolerance for defining the line.

The resulting data structure is called a strip tree because it can be stored as a tree structure. Different levels of generalization correspond to the different levels in the tree,

making it easy to generalize at different levels from the same data source.

2.1.3 Quad trees

The Quad tree offers a similar method of condensing the information in raster images (Figure 3). The image is first divided into sections with numbers of rows and columns that are integral powers of two. A section is then quartered, and each quarter is tested for homogeneity of the pixels it contains. If a quarter is homogeneous, it is stored as a unit. If not, it is quartered again. The process is continued until all quarters are homogeneous or until they become individual pixels. Once again, a B-tree may be used to store the image, and the economy of storage and recovery may be great. A processed Landsat image is ideal for this type of manipulation, while a raw image will ordinarily not justify it.

2.2 Semantics of Geographic Feature Representations

Specific geographic features are represented in GIS by symbols that may be words, letters, numerals or graphics. The semantics of these symbols refers to the meanings that they convey, and is therefore of the greatest importance to the user of geographic information. Individual symbols should convey unequivocal information to broad classes of users. Groups of symbols may convey more information than is contained in the individuals; one example of this being the information on slope and aspect that is carried by sets of contour lines. Another is the idea of population density and land use practices that are given by the clustering of house symbols. These examples show how the semantic value of symbols is related to the experience and understanding of the user.

The user of a data base must be able to enter symbols to initiate data base action, and must be able to understand the symbols that are returned from data base query processes. The accuracy of this two-way communication is crucial, and careful data base design is required to minimize the confusion that can arise from misunderstanding of the symbols, whether they represent commands, data input, or returned data or graphics. If words are being used to make queries concerning geographic features, the words must have exact meanings in the context of the specific data base.

The design of a high-level query procedure for a GIS should include some artificial intelligence to resolve simple context problems such as these. Large amounts of processing overhead are required to accomplish this, using natural language queries, and

the user must still understand the effects of the semantic structures of his queries. A simpler solution is to require the user to determine the type of object desired and to formulate a specific request for it.

To assist in the identification of the specific object desired within a data base, the object is often assigned attributes. The attributes can themselves be objects, but they serve the purpose of descriptive adjectives. The use of attributes narrows the definition of the object and allows for the selection of the specific information required. For example, the user can specify the exact type of 'road' that is required. Otherwise, the entire range of objects covered by the term 'road' may be selected, and their attributes can be examined separately.

2.3 Topology of Geographic Features

Present geographic information systems provide only the very simplest of topological information, if they provide any at all. It may be possible to derive the topology of features from their coordinate data, but for many important and simple topologies the derivation may be expensive or even impossible. In such cases the topology must be derived at the time of feature digitization, and it must be carried with the coordinate data in the data base. For ease of recovery of information, the topology should carry the coordinates.

2.3.1 Common Topological Relationships

The following is a list of important topological relationships, more or less in order of their complexity:

a) Connectivity of vectors.

Present GIS often carry contours, roads and other vectors as disconnected and unrelated strings of point coordinate sets. For example, a given road has connection with near and distant roads and places. Such information has inherent connectivity that can be important, and it is difficult to recover once it has been lost.

b) Connectivity and adjacency of areas

Areas are stored as sets of raster points, or more commonly as closed polygonal outlines. In the latter case the unity of an area is assured by the data format, but in neither case is it known from the basic data format whether two areas touch one another. Some polygonal data structures, such as the SDTC data

transfer format which has been adopted by several Canadian government agencies, carry this information by labelling each section of a polygon with the codes of the areas that it separates. It is also possible to label each node of a polygonal structure with the codes of all the areas that meet at that node.

c) Intersection

Intersection of features is an important property that is not presently provided. One example is the intersection of linear features such as roads and streams. This information would allow mapping all culverts and bridges on a road system. Another example is the intersection of areas, such as wetlands with proposed construction zones. There are many examples of interesting intersections that are now determined laboriously or not at all.

d) Proximity

Proximity of linear and areal features is more difficult to determine than adjacency, but may be just as important. Two examples will be given. In forestry, we might have the query "What areas of mature timber lie within one kilometre of a given proposed road system?". The military might ask "What areas of tree cover lie within 2000m of a proposed line of advance?". It is clear that the answers to many such queries cannot be included in the basic data structures, but the data base management system should provide the answers at reasonable cost in time and money.

2.3.2 Topology and Graph Theory

The following sections will just introduce the idea of handling topological relationships by means of graph theory. Actual physical objects have area, volume and shape, but their map representations are points, lines and polygons which can be considered as purely geometrical and topological objects. If topological questions are to be answered, however, the topology of the features must be carried in the data base. For example, if the task is to find a route between A and B by following roads, representing roads as a linear network allows efficient solutions. It is also possible to take into account such factors as bearing strength, width and surface status as constraints in the route selection. It is vital, however, that the network be connected: it would be difficult or impossible to find a route from A to B if the roads connections were not made, or if they were interrupted by bridges.

The topology of features that can be considered as linear can be modelled by connectivity graphs. The graph may be a map-like

representation, perhaps with the scale distorted (like a subway-car station map), or may be a computer-stored abstraction of the network. The nodes of the graph are intersections and connections of roads to bridges, ferries and the like. The arcs of the graph represent sections of the road and the connecting features. Note that for pure topology the coordinates of the nodes and the lengths of the arcs need not be specified.

The topology for areal (polygonal) features can be modelled by adjacency and incidence graphs. An incidence graph is topologically homeomorphic with the line-drawn map of the features. That is, the arcs represent the boundaries between adjacent polygons, and the nodes or vertices represent the points where three or more polygons meet. The degree of the vertex is the number of polygons that meet there, and, at the data capture level, the question of whether all the polygons are closed can be answered by examining the degrees of the vertices.

An adjacency graph looks like the incidence graph turned inside out. A vertex represents a polygonal area, and an arc represents a pair of adjacent polygons. Computer queries of a digital adjacency graph can determine whether a possible path exists across terrain which has been classified into passable and impassable areas, for example. For simple maps the question can be answered at a glance, but the adjacency-graph approach is required for analysis of detailed representations of complex terrain.

In all the cases above, the topology may carry positional and descriptive information. The vertex of an incidence graph may carry the coordinates of the point where the polygons meet, and the arcs may carry information about the adjacent polygons. Left and right identifiers may be carried also. The great virtue of the graph representation is that graph theory may be applied to derive new and important relationships of features.

3. INTERFACES

A GIS interface consists of the hardware and software which allow the user to interact with the geographic data base. Ease of user interaction is one of the major goals of GIS design, so that the interface must provide a simple conceptual model of the interaction. It must also contain an English-like query language that is easy to learn, and integrity checks on the user's actions to ensure smooth interaction. One goal of interface design is to make it unnecessary for the user to have an intimate knowledge of what the data base contains or how it is structured. Most GIS offer some on-line help, but the interfaces tend to be designed

for particular applications by a particular class of user. The following sections give a brief description of the components and requirements of a graphical interface for entering, retrieving and editing geographic data.

3.1 The User's Actions

The actions to be carried out by the user are:

- entering and editing graphical entities (points, lines, polygons, symbols, groups, text, etc.).
- data base interaction (retrieval, update, create, delete, etc.).
- saving images, printing images.
- modifying the display space.

Lines, symbols, objects and all other stored items have to be referenced so that the user can identify them without ambiguity.

3.2 Control Objects

The interactive interface programs are controlled with the assistance of control objects, including the following:

- a spatial window for displaying spatial information.
- a text window for displaying text data such as error messages, for information selection by interaction with the data base, and for requests by the user.
- a menu space for different menus.

When using a pointing device a cursor should be displayed to indicate position on the display. When the user moves from one control object to another, the cursor should change appearance to indicate this transition.

A scale indicator should always be visible on the display. This may take the form of a displayed scale fraction or of a divided and annotated scale line.

It is understood that the user will receive some form of feedback to all actions. In many cases, this will consist of the movement of a cursor on the screen. In other cases it may be in the form of a text message.

3.3 Command Modes of the Graphics Interface

The graphical interface includes a workspace, which contains a small subset of the data base at any one time. Five explicit modes of operation are suggested for the graphical interface. Three of the modes are interactive, the other two are used when leaving the editor:

1. Edit - add to or delete from the graphical information in the display state. This will involve an update to the workspace.

2. Show - involves querying the workspace. If the information is not found in the workspace, the data base is searched and the required material is transferred to the workspace for graphical or textual display. The default mode upon first initiating the graphical editor will be 'show'. The reason for this is that upon intitialization the display screen is blank. It is therefore probable that the graphical editor was invoked to show spatial objects, which may then be edited or just viewed.

3. Select - involves querying the workspace or the data base (if the required information is not in the workspace) and presenting the data to the text window. Depending on the amount of data being returned, it may be necessary to increase the text window size. Redirection of the data to another device is also possible. It will be possible to transfer the results of a 'select' into a 'show' request without repeating the query under show mode.

4. Stop - exits from the graphical editor. The changes made in the workspace are incorporated into the data base.

5. Abandon - stop the graphical editor. The changes in the workspace are abandoned and do not affect the data base.

In general, the command language should consist of as few explicit modes as possible. An explicit mode is one that must be invoked by the user, by changing from another mode with an explicit command. For example, to delete from the screen of the graphical device, the user could be required to enter a DELETE mode. Options within DELETE mode would likely be presented to the user at that time. To exit DELETE mode, the user would be required to explicitly abandon the mode and enter another one.

The user must always be aware of what mode is in use, and it is easy to become confused if too many explicit modes are available. On the other hand, if there are very few modes, the

number of commands on the screen at one time become excessive. A good compromise must be reached in grouping the commands into modes.

4. DATA BASE MANAGEMENT SYSTEMS

4.1 Hierarchical Data Bases

The most common type of data base for GIS is the hierarchical data base, illustrated in Figure 4a. It is most suitable for data that occur naturally in hierarchical form; for example, data organized by municipality within county within province within country. Data from any one of the root nodes may be recovered rapidly by a search down through the branches. In many GIS, the depth of the hierarchy is not great, and large files of data can be recovered quite rapidly.

Hierarchical systems break down when it becomes necessary to recover related information from many leaf nodes. While the complete data set for any one census tract, for example, may be readily available, it is quite a different matter to answer such a query as "What is the proportion of home owners to renters in all census tracts within four given counties?". Each leaf node representing a census tract must be located and queried individually; a very slow process.

4.2 Network Data Bases

A network data base is shown diagrammatically in Figure 4b. Such a base makes it easy and fast to recover related information, if the relationships have been built into the network. The recovery of specific large blocks of data may also be fast, but once again the paths of navigation through the base must be known. Network data bases are not easy to modify and are not easily understood by the user, but they might well be used as components of a complete GIS, at the command of a master data base system.

4.3 Relational Data Bases (RDB)

A diagram of a relational data base is given in Figure 4c. The RDB looks simple in the diagram, and it looks simple to the user also. The simplicity is deceptive because a highly developed relational data base can embody all the complexities of a set of hierarchical and network data bases.

The elements of the RDB are relations, domains and tuples. In many cases the relation represents a type of object, the domain an attribute of the object, and the tuple an instance or example. In the simple tabular representation, the relation is the table, the attribute is the column, and the tuple is the row. The more general terminology is necessary because the domains and tuples, in relational data base practise, may be numbers or symbols unrelated to any physical quantities. The strength of the RDB is that it may be indexed to link relations to one another in ways that are easy to visualize. There are so many useful examples of RDB structures that it is difficult to pick one that is not too narrow in scope to illustrate the power of the system. However, one example is the relation Roads, with attributes such as Expressways, Dual Highways, County Roads, etc. Each attribute will have a number of instances in a province or region. An individual expressway in this relation may be indexed to point to its own relation, with domains such as Sections, Municipalities, Accident Statistics and so on.

It is theoretically possible in an RDB to index every item to everything else, but it is obviously impractical to carry the indexing too far. Once a given system of indexing is set up, however, the recovery of related information is very convenient and fast.

4.4 RSDB - A Master Relational Data Base

The common relational data bases used in business contain records of fixed length. This is satisfactory for systems of moderate size because it does not take too many characters to specify any employee's name or to enter any sum of money, for example. In large business systems, and particularly in geographic systems, data often occur in large masses which must be recovered all at once. A topographic map may contain several million data, with one or more millions in the contour file alone. A Landsat image with perhaps 15 million pixels, each carrying four bands of colour data, is another example that must be recovered quickly.

Collins & Moon Limited has recently developed the central system of a Relational Spatial Data Base (RSDB) with variable-length and matrix records, to handle problems of the types above. This capability makes it possible, within the relational data base, to simulate network data bases with their fast recovery of related material, and hierarchical data bases with their easy addressing and fast recovery of large blocks of data.

5. CONCLUSIONS

Future geographic information systems will take advantage of modern developments in interface technology, in information theory, and in data base management systems. This paper has stressed the consideration of topology and semantics, and the need for new interface design. The most important recommendation is the creation of a comprehensive relational data base to respond to user queries, with the capability of addressing subsidiary hierarchical and network data bases as well as simple data files.