A GENERAL SPATIAL DATA STRUCTURE

Linda Shapiro and Robert Haralick
Machine Vision International
Ann Arbor, Michigan 48104

The basic kinds of data found in maps are points, lines and areas. This data can be stored in many forms: grid cells, lists of points, lists of line segments, and polygons are common examples. The best storage representation is the one that can be accessed most efficiently for a given application. Unfortunately, if everyone uses a different data structure, very little sharing of software or data can take place. In this section we define a general spatial data structure that can be used to represent any spatial or relational data.

An atom is a unit of data that will not be further broken down. Integers and character strings are common examples of atoms. An attribute-value table D/V is a set of pairs D/V = {ai, vi} where ai is an attribute and vi is the value associated with attribute ai. Both a and vi may be atoms or more complex structures. For example, in an attribute-value table associated with a structure representing a person, the attribute AGE would have a numeric value, and the attribute MOTHER might have as its value a structure representing another person.

A spatial data structure D is a set D = (R1, ..., RN) of relations. Each relation Rk has a dimension Nk and a sequence of domain sets S1(k), ..., SNk(k). That is for each k = 1, ..., K, Rk ε S1(k) x ... x SNk(k). The elements of the domain sets may be atoms or spatial data structures. Since the spatial data structure is defined in terms of relations whose elements may themselves be spatial data structures, we call it a recursive structure. This indicates 1) that the spatial data structure is defined with a recursive definition, and 2) that it will often be possible to describe operations on the structure by simple recursive algorithms.

A spatial data structure represents a geographic entity. The entity might be as simple as a point or as complex as a whole map. An entity has global properties, component parts, and related geographic entities. Each spatial data structure will have one distinguished binary relation containing the global properties of the entity that the structure represents. The distinguished relation is an attribute-value table that will generally be referred to as the D/V relation. When a geographic entity is made up of parts, we may need to know how the parts are organized. Or, we may wish to store a list of other geographic entities that are in a particular relation to the one we are describing. Such a list is just a unary relation, and the interrelationships among parts are n-ary relations.

For example, we may represent the state of Virginia by a spatial data structure. In this case, the D/V relation would contain global attributes of the state such as population, area, boundary, major crop, and so on. The values of most of these attributes (population, area, major crop) are atoms. The value of the boundary attribute is a spatial data structure defining the boundary. A list of counties could be included as one of the relations, or it might be more valuable to store the counties in a region adjacency relation, a binary relation associating each region (county) with another region (county) that neighbors it. Counties, of course, would also be represented by spatial data structures.

Some other geographic entities that are related to a state are its highways, railroads, lakes, rivers, and mountains. Some of these entities will be wholly contained in the state and others will cross its boundaries. One way to represent this phenomenon is to use a binary relation where the first element of each pair is a geographic entity, and the second element is a code indicating whether the entity is wholly contained in the state. The spatial data structures representing the geographic entities themselves would contain more specific information about their locations.

User queries take one of two forms. First, the user may request the value of an attribute such as POPULATION. Since this attribute is stored in the database, query processing is simple and fast. On the other hand, the user may call upon the intelligence aspect of the system to produce a list of rivers within 20 miles of Roanoke, Virginia. To meet this demand, the system must then be capable of reviewing a list of 20 miles of Roanoke, Virginia. To meet this demand, the system must then be capable of reviewing a list of 20 miles of the coordinates associated with the data structure for the city of Roanoke. Both types of queries, as well as data base creation and maintenance, demand physical storage structures that perform the following tasks rapidly and efficiently:

1. Creation of a new spatial data structure
2. Deletion of a spatial data structure
3. Addition of a relation to a spatial data structure
4. Deletion of a relation from a spatial data structure
5. Addition of an n-tuple to an existing relation
6. Deletion of an n-tuple from a relation
7. Changing the value of an attribute
8. Changing one or more components of an n-tuple

REFERENCES


CH2079-2/84/0000/0082$01.00 © 1984 IEEE

82