A RELATIONAL APPROACH TO VECTOR DATA STRUCTURE CONVERSION

Jan W. van Roessel* and Eugene A. Fosnight*
Technicolor Government Services, Inc.
Sioux Falls, South Dakota 57198

BIOGRAPHICAL SKETCH

Jan W. van Roessel is currently employed as a Principal Scientist by Technicolor Government Services, Inc. at the EROS Data Center in Sioux Falls, South Dakota. He received a PhD in Wildland Resource Science and an MS in Engineering Science from the University of California, and an MF from the University of Washington.

BIOGRAPHICAL SKETCH

Eugene A. Fosnight is currently employed as a Senior Scientist by Technicolor Government Services, Inc. at the EROS Data Center in Sioux Falls, South Dakota. He received a diploma in Cartography from the University of Wales, Swansea, United Kingdom.

ABSTRACT

The proliferation of geographic information systems and digital data bases is creating a need for efficient methods to convert data from one spatial data structure to another. One approach is to create ad hoc interfaces, with a potential of N(N-1) interfaces for N data structures. Using an intermediate data structure, at most 2N interfaces are required. An intermediate relational data structure is therefore proposed that takes the form of a set of normalized relations stored in a relational information management system. The advantages of this approach are found in the simplicity of the relational approach, and the availability of relational operators to be used as higher level tools, to convert from and to the relational data structure. The Relational Information Management System (RIM) is used for the ongoing research. In conjunction with the relational data structure, another higher level tool has been developed to cope with linked lists, tree structures, vectors, and matrices, which are not otherwise easily reduced. This data tracking system is programmable at a higher level, in a syntax that allows a concise expression of the desired restructuring. Output from this system can be further operated on by relational operators to arrive at the desired intermediate data structure. This, and other topology verification and checking tools, are visualized as part of a core system dedicated to the conversion and collation of spatial data from diverse origins.

*Work performed under U.S. Geological Survey contract no. 14-08-0001-20129.

INTRODUCTION

With a growing number of geographic information systems and digital spatial data sets, a capability to adapt digital spatial data from one system to another has become much sought after. Yet such adaptations are not as easily accomplished as is sometimes believed because of the special characteristics of spatial data. Non-spatial digital data have two main aspects with respect to their organization, physical and logical (Martin, 1977), but spatial data can be thought of as having two additional traits, spatial-logical (topological) and locational (coordinate system, projection, measurement units).

Adaptation is therefore not a simple reformatting problem, as with other data sources, but rather it is a far more complex proposition with spatial implications that cannot be readily deduced by an initial inspection of the data. It may even be impossible to convert certain types of data structures, short of a major processing effort. However, data structures of similiar types, such as an arc-node organization, can be converted from one system to another with a reasonable amount of effort.

One of the obvious first steps for developing a conversion methodology is to obtain a consistent description of the vector data structures involved, and to characterize them in order to anticipate possible conversion problems.

For this purpose a Backus Naur Form (BNF) type of notation was developed to describe the data organization concisely and consistently. The use of BNF for describing the syntax of geographic data was earlier suggested by Cox, Aldred, and Rhind (1980).

A second step in the description process has been to cull the most important characteristics of each data structure. The five major spatial data aspects--physical, logical, logical-spatial (topological), locational, and attribute--were further reduced to constituent components, and this process was repeated until a satisfactory level of detail was reached. The overall breakdown was then used as a checklist against each BNF description, as well as other documentation, to obtain a set of hierarchical characteristics of the data structure under consideration.

INTERFACING MODELS

One approach to the interfacing of N data structures is to match each one with every other one, thereby creating N(N-1) connections. Alternately, one may implement a "ring" concept whereby each system interfaces to one and only one other system. This yields the minimum number of N interfaces. Its drawback is that data may have to pass through N-1 other systems before a target system is reached, with proportional processing time and potential for errors. A third approach uses an intermediate data structure to which each original structure must be converted before it is transformed to the target configura-

tion. This method requires 2N interfaces and is the one most commonly adopted; it is also the approach used by the authors. Quite often this method is associated with an intermediate tape format. Our concern is to define an intermediate data structure maintained in a database system, which can in turn be readily transferred between systems.

The Workgroup on Data Organization of the National Committee for Digital Cartographic Data Standards (Nyerges, 1984) has proposed a model for digital cartographic data conversion to an intermediate structure as follows:

$$S \xrightarrow{T1} I1 \xrightarrow{T2} I2 \qquad (1)$$

The above model is only for the conversion from source to intermediate; the conversion from intermediate to target is similar. In the above, S represents the source data structure, T1 is a transformation producing an "intermediate" intermediate structure I1, and T2 is a transformation producing the desired standard interchange form. The Workgroup mentions the importance of the data schema for this model. The schema defines the logical aspect of the data structure, and forms a link with the physical representation. A sobering note is the complete absence of any kind of explicit logical or topological schema in any of the existing data structures considered. However, for our approach, the model proves to be useful for considering the different transformation stages in the data structure conversion.

THE RELATIONAL APPROACH

Other investigators have applied the relational model or relational concepts to geographic data handling. At least one prototype geographic information system (GEOQUEL) has been constructed using the relational database management system INGRES (Go, Stonebraker and Williams, 1975). Shapiro and Haralick (1980) have proposed spatial data structures based on relations in relation trees. Garret and Foley (1982) have reported an attempt to build an experimental graphics system using relations, where events trigger continuously evaluated qualified updates, through which the consistency between relations is always maintained. A graphics application (IMPS) based on a relational data base (IMS) was developed by IBM in the United Kingdom (IBM, 1979).

We have selected the relational data model for the intermediate data structures for a number of reasons. One of the main arguments for our choice is the elegance and simplicity of the data representation, resulting from the use of "flat" files and a complete absence of pointers.

A second consideration is the availability of the relational algebra and its unique relational operators. To minimize the effort associated with creating a number of transformation funcitons for different input models, it

follows logically that one should attempt to develop higher level tools that can be reused for each conversion. The aim of our study is to investigate whether relational operators fall in this category.

A third reason for considering the relational model is the availability of a number of different software systems, including many operating in the microcomputer environment, as well as hardware implementations in the form of back-end database machines. Research for this paper was performed using the Relational Information Management system (RIM).

Normalization is one of the important considerations in relational database theory. Five normal forms are recognized (Kent, 1983). The first normal form is the most crucial and states that "all occurrences of a record type must contain the same number of fields." This is not an option, but a matter of definition: relational database theory does not deal with variable length records or vector or matrix type attributes. Our proposed intermediate data structure is strictly of first normal form, which allows us to use the full range of relational operators.

Given the simplicity of flat files, the database schema also becomes very simple, and consists of a description of the individual attributes and the composition of the relations in terms of these attributes. Although much can be learned about the contents of a database by inspection of the schemas, the schema is still inadequate for describing the spatial organization of the data.

RELATIONAL DATA STRUCTURE

The type of spatial data structure selected is of the arc-node type. A full complement of spatial elements is represented: regions, polygons, arcs, nodes, and vertex points (line and point type elements are not further considered in this paper). We define a polygon as a closed figure bounded by connected straight line segments, a region as consisting of an outer polygon and zero or more island polygons, and nodes as arc termination points.

The mutual referencing of the elements is resolved as follows (the relational theory excludes direct pointers, so all referencing is in terms of element identifiers). Given the hierarchy obtained by defining more complex elements in terms of their component parts--regions, polygons, arcs, nodes and vertex points--the most austere organization is obtained by having each element reference its subordinated elements. For example, by defining the element identifier attributes regnum, polnum, arcnum, nodenum, strtnode, endnode, and pointnum, as well as the coordinate attributes x, y, we can propose to maintain the following relations:

regpol: regnum, polnum

polarc: polnum, arcnum

arcnode: arcnum, strtnode, endnode (2)

arcpoint: arcnum, pointnum

nodepoint: nodenum, pointnum

pointxy: pointnum, x, y

where strtnode and endnode refer to nodenum in the nodepoint relation.

The above scheme has downward referencing only. Reverse references can be established with relational operators. For instance, to compute an "arcpol" relation containing polygon numbers by arc, one can isolate the arc column from the polarc relation using the project operator, remove duplicate arc numbers (if not included in the project), and then join this temporary relation with the polarc relation using arcnum as the join item, while maintaining the derived arc ordering.

Spatial references, such as establishing which regions are to the left and which are to the right of the arc, cannot be so easily established in this manner. Considering also the frequency with which this type of information is used, one is led to consider storing some higher level information at the lower levels. In particular, the arcnode relation can be renamed archdr, and be given two additional attributes, lftreg and rgtreg, as follows:

archdr: arcnum, strtnode, endnode, leftreg, rgtreg. (3)

Also, for network tracking, a nodearc relation is desirable which arcs emanating from each node are stored:

nodearc: nodenum, arcnum. (4)

The above considerations, and the removal of the pointxy relation, lead to our proposed intermediate data structure I2:

regpol: regnum, polnum

polarc: polnum, arcnum

archdr: arcnum, strtnode, endnode, lftreg, rgtreg (5)

arcxy: arcnum, x, y

nodearc: nodenum, arcnum

nodexy: nodenum, x, y

One further decision to be made is how to cope with complex islands. The union of the polygons of a complex island constitute the hole in the outer polygon. Not all

arcs of the island polygon are therefore island arcs. We have solved this problem by defining the outer boundary of the complex island to be a separate polygon. This gives the advantage of being able to join the regpol relation with polarc and arcxy relations to directly obtain a regxy relation in which each region is explicitly represented in sequence, complete with all its islands, including complex ones. In this way, the topology is defined in a simpler, clearer organization. Thus, our approach is to organize the topological information implicitly by position, rather than explicitly.

Position is critical within most relations. The vertex point coordinates in the arcxy relation are obviously stored in positional sequence. Less obviously, the arcs in the nodearc relation are stored in a clockwise rotation around the node. The arcs in the polarc relation are stored in clockwise order for normal polygons and counter-clockwise for island polygons. Finally, in regpol, the first polygon for each region is the outside polygon.

When constructing polygon or region boundary lists, one must keep in mind that arcs must be traversed in different directions for the regions to the left and the right of the arc. The traversal direction is stored implicitly in the archdr relation. If the polygon currently being traversed corresponds to the right region in the archdr relation, the arc is rotating clockwise. The converse is true when it matches the left region.

Other non-spatial relations are defined in addition to the above core of spatial relations. Most important are those for attribute data. They are visualized as a nested set, each consisting of one primary and a number of secondary relations, namely:

attprime: elnum, att1, att2, att3...etc. (6)

This relation links the primary attributes to the spatial elements through the common element identifier. In addition, each attribute in the primary relation can be a key for an entry in secondary relation, in which that attribute is further detailed. For instance, one may have:

attsec: att1, satt1, satt2, satt3...etc. (7)

This allows new attribute relations to be created in which the attribute detail can be varied at will, using join operators.

The overall interfacing effort is reduced not only by the total number of required interfaces, but also by the amount of time required to construct each interface. This time is reduced further when suitable higher level tools are available.

According to the Workgroup model, these tools can be assigned to T1 and T2 categories. The database system used is perhaps an exception; it can be assigned to both

categories because it allows the data structures I1 and I2 to exist.

The scope of T1 is the following. The scheme being tested makes use of the fact that the input data structures can be cast in the form of files with fixed- or variable-length records. These files can be translated on a one-to-one basis to unnormalized (with respect to the first degree) records in the RIM system, storing one record per relation row. One would not expect this capability in an RDBMS, but RIM permits records with variable-length attributes, although relational operations cannot be performed on variable-length items. With the transfer, a schema is developed for the unnormalized data structure in the relational database, and I2 represents the normalized intermediate structure presented earlier.

In the T2 category, a breakdown can be made considering tools that allow data to be restructured from T1 to T2, and those to be used to repair errors, perform checks, and make I2 more complete.

RESTRUCTURING TOOLS

The transformation from the unnormalized form to a normalized representation is accomplished by a specially developed program (RIMNET) that allows one to track data

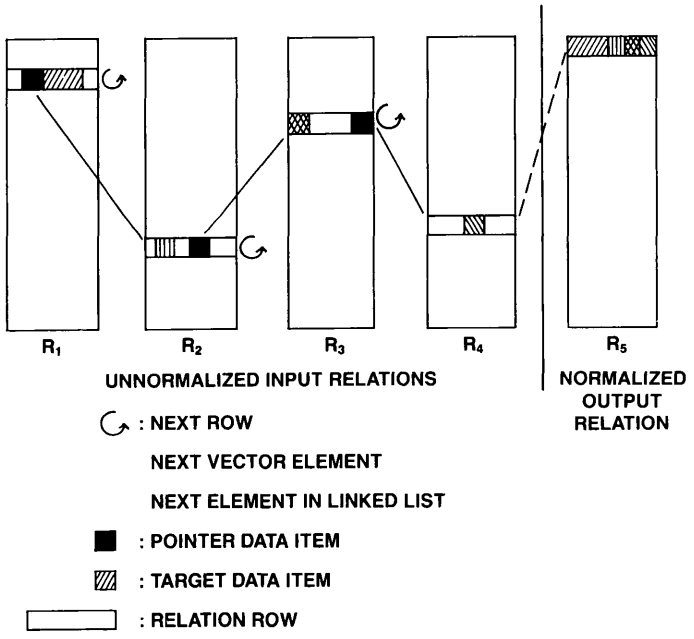

Figure 2.--Schematic of RIMNET operation.

items along predefined access paths in the unnormalized representation. This program is one of the main conversion tools. Using relational operators, the normalized forms are then further reduced to the essential relations in the intermediate relational format (I2).

RIMNET has been designed to serve as a normalization tool, but also to cope with such data organizations as linked lists, which cannot easily be unscrambled with relational operators. The principal idea behind the program is demonstrated in Figure 2.

The desired tracking sequence can be defined by a concise free-format syntax in which the relations, groups, and types of increments are specified. The basic element in the syntax is the group specification. In a group the pointer data item name is separated from the target data item names by a colon. Either pointer or target item names can be omitted. The entire group specification can be enclosed in brackets with following meaning: {} row increment; [] linked list; <> vector increment. Brackets can be nested. The following are examples of valid group and increment specifications:

```
{POINTER:TARGET}

[POINTER:]                                    (8)

<:TARGET>

{<:TARGET>}
```

For a relation, group specifications can be concatenated. To indicate the relevant relation, groups are preceded by the relation name, as follows:

```
REL1 = {POINTER:TARGET}                       (9)
```

When more than one relation is involved, specifications are strung together, as in the following example:

```
REL1 = {POINTER:TARGET} =                     (10)

REL2 = <:VECTOR>
```

Here, POINTER points to a row of REL2. The output relation will contain two data items per row: TARGET, and an element of VECTOR.

An example of the programmable tree-traversing that is possible with RIMNET is found in a simple situation in which the rows of the unnormalized relation REL contain a vector attribute, VECTOR, and this relation must be normalized by assigning one vector element to each row of the output relation. The normalizing operation is then specified as follows:

```
REL = {<:VECTOR>}                             (11)
```

where: VECTOR is the right hand side of the group specification, VECTOR is the target item, and an explicit pointer is not needed because the <> brackets imply a next vector element pointer, while the {} brackets imply a next row pointer. REL is the input relation for which VECTOR is defined as an attribute in the schema.

Execution of the syntax starts at the first row and causes each vector element to be transferred, proceeding then to the second vector of the second row, and so on.

Slightly more complex is an operation in which a variable number of elements must be retrieved, as indicated by another data item TOTEL stored in the rows of the input relation. The syntax is then:

REL={TOTEL}<:VECTOR(1-TOTEL)> (12)

If, in addition, the elements may be further spread out over more than one record, and the continuation record is indicated by an item NEXTREC, the syntax becomes:

REL= {:TOTEL}[NEXTREC:]<:VECTOR(1-TOTEL)> (13)

The square brackets indicate a linked list for which NEXTREC is the pointer item.

RELATIONAL OPERATORS

Once the initial relations (I1) are normalized, relational operators can then be applied to the extent possible to convert them to the desired intermediate structure (I2). Thus, we are testing the concept that these operators can be an important set of higher level tools.

So far, the following RIM operators have been used frequently: JOIN, PROJECT with WHERE clause, INTERSECT, CHANGE with WHERE clause and DELETE DUPLICATES. Especially the INTERSECT command was found to be a powerful operator, with advantages over a JOIN in many instances.

TOPOLOGICAL TOOLS

One of the objectives of the intermediate data structure is to serve as a way-station where improvements and quality checks can be made. Therefore, we have adopted the philosophy of "weak input". This means that we will be able to accept data deficient in topology, because we will make necessary fixes and enhancements with our tools. One of the significant advantages of a relational information management system such as RIM is that the data structure can be inspected, partially copied, changed, and otherwise manipulated in any desirable manner, interactively. One therefore enjoys total access to the data.

In another sense "weak input" means that only certain basic relations need to be extracted from I1; others can be built from these initial relations. The fully defined topology of I2 can actually be created given a "weak"

input of two tables. One table must contain the arcnumbers and vertex point coordinates (arcxy), while the other must contain the adjacent regions for each arc (regardless of order -- a weak archdr). A topological restructuring program that uses these two input relations has been written and tested.

It is possible to visualize even weaker input, such as merely a set of connected arcs, or even arbitrary spaghetti. Given a data structure, the question is: how much of its topology should be salvaged. We envision the eventual use of a "generalized vector processor" that will be able to reduce vector data to basic line segment components from which it can recreate the desired topology.

CONVERSION EXPERIENCE TO DATE

The first system for which a data structure has been converted to the relational format is the Geographic Entry System (GES) of Electromagnetic Systems Laboratory's (ESL) Interactive Digital Image Manipulation System (IDIMS). From the relational format GES data have been transferred to the ARC/INFO system of the Environmental Systems Research Institute (ESRI), and AGIS/GRAM of Interactive Systems Corporation (ISC).

A second system for which data structures were converted is the Analytical Mapping System (AMS) used by the U.S. Fish and Wildlife Service. This system is the vector input system to its companion Map Overlay and Statistical System (MOSS).

Subsequent systems for which data were transferred to the relational format were ICARAS (a raster-to-vector conversion system developed by D. Nichols of the Jet Propulsion Laboratory), as well as data from an Intergraph digitizing system (EDCIDS) being developed at the EROS Data Center, and the Unified Cartographic Line Graph Encoding System (UCLGES) of the U.S. Geological Survey (used to structure digital line graph (DLG) data).

Tools such as RIMNET have been successfully used in at least two interfaces (GES and AMS). In the more polished interfaces, the various tools are tied together with the VAX/VMS DCL language.

CONCLUDING REMARKS

Although we are currently at the beginning of our interfacing efforts, the relational approach with the use of higher level tools shows promise for leading towards a conversion model that will be closely aligned with the one proposed by the Workgroup on Data Organization of the National Committee for Digital Cartographic Data Standards.

The simplicity and elegance of the relational approach is reflected in the simple relations of our intermediate spatial data structure, such that the spatial operations can take full advantage of the relational method.

The development of a good set of interfacing and diagnostic and improvement tools will provide us with an excellent capability to tackle diverse interfacing problems.

REFERENCES

Breithart, Y. J., and Hartweg, L. R., 1984, "RIM as an implementation tool for a distributed heterogeneous database", NASA Integrated Program for Aerospace - Vehicle Design Symposium II, Denver, Colorado.

Codd, E. F., 1970, "A relational model of data for large shared data banks," Communications of the ACM, v. 13, no. 6, p. 377-387.

Codd, E. F., 1971, "Relational completeness of data base sublanguages", R. Rustin (Ed.), Data Base Systems (Courant Computer Science Symposia VI), Prentice-Hall, Englewood Cliff, New Jersey.

Cox, N. J., Aldred, B. K., and Rhind, D. W., 1980, "A relational data base system and a proposal for a geographical data type", Geo-Processing, v. 1, no. 3, p. 217-229.

Garrett, M. T., and Foley, J. D., 1982, "Graphics programming using a database system with dependency declarations", ACM Transactions on Graphics, v. 1, no. 2, p. 109-128.

Go, A., Stonebraker M., and Williams, C., 1975, "An approach to implementing a geo-data system". Memorandum No. ERL-M259, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, California.

IBM, 1979, IBM interactive management and planning system under IMS/VS. User guide: SB11-5220. IBM European Publications Center, Uithoorn, The Netherlands.

Kent, W., 1983, "A simple guide to five normal forms in relational database theory". Communications of the ACM, v. 26, no. 2, p. 120-125.

Martin, J., 1977, Computer Data-Base Organization, O. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Nyerges, T., 1984, Editor, "Digital Cartographic Data Standards: Alternatives in Data Organization", Progress Report of the Workgroup on Data Organization of the National Committee for Digital Cartographic Data Standards, Report 4, January.

Shapiro, L. G., and Haralick, R. M., 1980, "A spatial data structure", Geo-Processing, v. 1, no. 3,

Shapiro, L. G., 1980, "Design of a spatial information system in map data processing", edited by H. Freeman and G. G. Pieroni, Academic Press.