THE UNIVERSITY OF HULL

Cartographic Information Systems Research Group
C.I.S.R.G.

Hon. Co-ordinator:
Dr. M. Visvalingam
Department of Computer Science
The University
HULL HU6 7RX

Tel. (0482) 465295/465951
Telex 592530

Discussion Paper Series Editors: Dr. M. Visvalingam
Dr. M.E. Turner

C.I.S.R.G. DISCUSSION PAPER 1

From Line Geometry To Area Topology

by

P. Wade, M. Visvalingam and G.H. Kirby

July 1986

ABSTRACT

Area objects are represented on a computer by geometric or locational information (which describe the course of their boundaries), topological information (which describe the areal units to which boundaries belong), and geographic or other information (which describe the objects which map onto these areal units).

Most available systems for spatial data processing require from the outset object-related information, in the form of left/right or on-line references or area-seeds, for extracting the area topology.

This paper introduces the Disassociative Area Model (DAM) and contrasts it with other existing descriptions of areal entities. The primitive region forms the basic spatial unit for object modelling and acts as the link between the geometric and geographic components. The derivation of spatial topology focuses on the boundary, which describes one extent of a primitive region. Geographic information may be input in a variety of ways and at any convenient stage using pragmatic models derived from DAM.

CONTENTS

FIGURES

1. INTRODUCTION

One of the more challenging tasks in the development of Geographic Information Systems (GIS) is the derivation of an appropriate conceptual model for describing area entities. This paper introduces the Disassociative Area Model (DAM) and contrasts it with other existing descriptions of areal entities. Most available systems for spatial data processing require from the outset object-related information, in the form of left/right or on-line references or area-seeds, for extracting the area topology.

Conceptually, it is preferable to break down the vector encoding of polygonal areal covers into three distinct stages:

1) the geometry of the linework of the area boundaries is encoded;
2) the topology (or spatial relationships) is determined; and
3) the geography of the areal entities is encoded.

There are several advantages in separating these stages. Firstly, there can be flexibility in the order and the method by which data is supplied to the system if the geometry and the geography are considered separately. Secondly, it is possible to derive the topology by an automatic process, thus minimising the information which has to be input by the user and reducing the possibility of error. Thirdly, the separation of the geometry from the geography enables a data structure to be used which can represent complex relationships between areas (such as overlaps, hierarchies, holes and islands), and also allows several sets of areas to be held in the same database. Fourthly, if the topological description of the geometry is also held within the data structure, it is possible to use this knowledge to perform efficiently spatial operations on areas and data associated with areas; for example, the disaggregation and reassignment of attribute data from one set of areas to another. Finally, and importantly, it is possible to use the computed topology to check for completeness and consistency in the data supplied, and ensure that any rules that exist about how areas are related (such as in a hierarchy) are obeyed in the data.

This paper presents a new model for representing the spatial relationships between polygons. It disassociates the geometric and geographic components of areas for separate academic consideration. Additionally, an automatic method is given for progressing from a node-matched set of vectors, which describe the linework of area boundaries, to a complete topological description of the uncut spatial units which can be built from the boundaries. Following a discussion of some existing techniques for representing geographic areas in the next section, Section 3 describes the new model for representing the spatial relationships between polygons, which we have called the Disassociative Area Model (DAM). In Section 4 an algorithm for generating the uncut spatial units and their topology from a set of vectors is described. A description of a supporting data structure is also given. Section 5 will illustrate the utility of the methods developed in this paper by describing how they were applied in a project which was undertaken on behalf of the Ordnance Survey.

2. BACKGROUND

This section describes briefly some of the concepts and techniques used for representing polygonal areas.

The simplest approach is to encode each area object as a separate entity, without any regard for overlaps and adjacencies between areas. This leads to two immediate problems. Firstly, boundaries between adjacent areas are encoded twice, once for the area on each side, thus causing redundant information to be stored and possible 'sliver' lines to appear when plotting more than one object. Secondly, the lack of information about the spatial relationships limits the ability to manipulate areas and any associated substantive data.

One of the first systems to incorporate an explicit topological structure was the Dual Independent Map Encoding (DIME) system of the U.S. Bureau of the Census (Cooke and Maxfield, 1967). It was designed to represent a single non-overlapping set of area objects. The basic element is the straight line, uncrossed by any other, called a segment. The segment is identified by its start and end points, called nodes, and the attribute codes for the areas on either side of it.

The DIME philosophy was extended in POLYVRT (Laboratory for Computer Graphics and Spatial Analysis, 1974). In this model the chain replaced the segment as the basic element. Like a DIME segment, a chain has nodes at its two end points, it separates two areas, and is assumed to be uncrossed. It may, however, consist of many points, and it represents the whole of a continuous section of boundary between two areas. In this structure the number of points required to describe a boundary between two areas has become unimportant; it is always represented by a single chain. Thus the chain plays an important topological role, as well as forming the interface between the geometry and the area objects. POLYVRT also keeps a list of the chains which form the polygonal boundary of each area. POLYVRT first introduced the cartographic data structures which have continued to form the backbone of a variety of internal data structures used for representing segments/chains today.

A consequence of the role that chains play in the POLYVRT structure is a restriction in the types of area it can represent. For example, it is only able to record explicitly the topological relationships and boundaries of a single non-overlapping set of areas. In an attempt at greater flexibility, Peucker and Chrisman (1975) introduced GEOGRAF, which uses the Least Common Geographic Unit (LCGU) as the controlling unit of the data structure in place of the chain. The LCGU is defined as that part of the plane surface uncut by further partitioning, and its boundary is constructed as a POLYVRT polygon directly from chains. The chain now becomes the boundary between two LCGUs, and thus assumes a purely geometric role, rather than being the boundary between two area objects. Also, the uncut spatial units (LCGUs) created by the chains no longer represent the area objects directly; instead they are each given a unique identity and are used as a building block in forming the area objects.

GEOGRAF is able to model multiple sets of area objects. Thus if an administrative hierarchy of districts, counties, and countries was represented, GEOGRAF could conceive of the three levels as separate sets of areas rather than as a single set of areas (with hierarchic area codes). Each LCGU would record the attribute codes of the area it belongs to at each level. GEOGRAF can similarly represent multiple non-hierarchical data sets, such as county and parliamentary areas. The boundary between two objects of a set, for example between two counties, is described by a chain group, which is an ordered set of chains. The polygonal boundaries of these objects, in turn, consist of chain groups and will from now on be called GEOGRAF polygons.

The GIMMS segment format is an extension of POLYVRT. It allows composite area codes to be supplied for POLYVRT type chains, and from these it is able to extract GEOGRAF polygons. However, GIMMS is based on the chain (POLYVRT) rather than the LCGU, and is thus unable to combine the geometric and geographic descriptions for areas in a flexible manner.

References to objects to the left and right of segments/chains/chain groups allow the above models to cope with detached parts and holes (one or more area objects completely within another area object) without having explicit knowledge of their existence. Edwards et al (1977) proposed a hierarchical data structure (HDS) for representing areal data which has holes, holes in holes, and so on. From POLYVRT type chains which refer to area objects on either side, they form a set of directed polygonal boundaries, each of which has anti-clockwise closure and a common area code on the left or right side. Each directed polygonal boundary is then used to define an interior or exterior region. An interior region is the entire area within a boundary which has a common left area code, and an exterior region the entire area outside a boundary with a common right area code. The interior and exterior regions do not contain holes.

Area objects with holes are defined in terms of the interior and exterior regions. Each interior region corresponds to the outer extent of one disjoint part of an area object, and each exterior region corresponds to a hole in an area object. The hierarchical data structure is used to model the nesting (or containment) that exists between the interior and exterior regions. This containment hierarchy is computed automatically based on the knowledge of which area object a region includes or excludes, and also by performing spatial calculations. Once the hierarchy is formed, each disjoint part of an area object can then be defined by taking the intersection of an interior region together with all the exterior regions which are immediately below it in the hierarchy.

The advance of HDS over POLYVRT is that it uses chains to define boundaries of regions (which enclose and exclude area objects) rather than to define the area objects themselves. Modelling the hierarchy of these regions allows for the explicit representation of holes and the spatial relationships that result. The chain remains the boundary between two area objects, however, and a consequence of linking the geometric and geographic information in this manner is that the system cannot represent a set of areas which contain overlaps, or several sets of areas simultaneously. Since HDS is similar to our model further discussion of HDS is postponed until Section 3.3.

All the above systems require the linework of boundaries to be input together with left/right codes for the area objects on either side of the boundaries (or, for the LCGUs in the case of GEOGRAF). Although this expedites computer processing, it presents a poor human-computer interface and places an unnecessary burden on users. Input to the GIRAS system (Mitchell _et_ _al_, 1977) consists of _arcs_ and _polygon labels_. Arcs, unlike chains, do not carry left/right tags at input time. Along with the line data, one polygon label for each polygon of the map is entered in the system. A polygon label is an arbitrary point within each polygon with which is associated a not necessarily unique integer attribute to describe the polygon in which it is placed. The arcs are then connected by software to form polygons, and the arcs and polygons are internally cross-referenced and the attributes of the polygon labels assigned to both. The GIRAS structure is described (Mitchell _et al_, 1977, p 5) as topologically similar to that introduced by Peucker and Chrisman (1975). However, the attributes of the polygon labels need not be unique, which suggests that GIRAS does not use the GEOGRAF model as such, but that the polygons correspond directly to the detached parts of area objects as in POLYVRT.

GIRAS also explicitly recognises _islands_ (the GIRAS term for holes) and _compound islands_ (islands which are subdivided) by defining the extent of each polygonal area by the arcs of its outer perimeter in a clockwise direction and the arcs of any interior islands in an anti-clockwise direction. The outer and inner extents of a complex polygonal area are fixed using the position of the polygon label (see Mitchell _et al_, 1977, p 11). Thus GIRAS, like HDS, uses directed polygonal boundaries, although the various boundaries which define a polygonal area do not each have a separate identity in GIRAS. The geometric hierarchy of holes within holes remains implicit in GIRAS. However, it is possible to encode a geographic hierarchy between area objects (such as that between administrative areas) by associating a composite feature code with the polygon label. GIRAS therefore uses a number of ad hoc procedures to circumvent problems in spatial data processing without seeking to accommodate them within an underlying conceptual model of areal entities.

The Level 3 Digital Line Graph (DLG-3) format (Allder and Elassal, 1984), as the name implies, uses a link and node structure to encode the area, line, and point features of a map together with their spatial relationships. The attributes of lines and points are associated with the links and nodes, and the links also contain explicit information about the topological relationships. When areas are represented, each closed part of the plane surface created by the linework of the boundaries is given a unique identity, in a similar way to the LCGUs in GEOGRAF. The links encode the adjacency relationships between LCGUs by referring to the LCGU on either side. The attributes of each LCGU, that is the area objects to which they belong, are associated with a digitised point (which need not be contained within the LCGU). Attributes are not encoded if they can be derived from relationships to adjacent elements. For example, a link is known to be a boundary between two area objects if the attribute codes for the LCGUs on either side of the link are different for that class of area object. The links which form the polygonal boundaries of LCGUs and area objects are not explicitly stored in the structure - further processing is required to derive this information.

ARC/INFO is a modular geographic information system with facilities for input, analysis, data management, and output of geographic data. Two data models are used within ARC/INFO. Cartographic data (locational and topological) are represented using a topological data model said to be based on DLG (ESRI, 1985, p 2.9). Thematic data (or attribute data) are represented using a relational model. In the name ARC/INFO, ARC and INFO refer to these two structures respectively, and ARC/INFO refers to the composite data model and the commands provided to manage and manipulate data held in these structures.

The input of polygonal data to ARC/INFO is similar to GIRAS in that it consists of arcs, without left/right tags, and polygon labels. Moreover, ARC/INFO allows even greater freedom in how they are digitised. Arcs can be digitised as 'spaghetti' without the user having to identify intersections (nodes), as the system can identify intersections automatically prior to polygon building. Also, the entry of polygon labels is optional, as the system does not use them to fix the outer and inner extents of complex polygons when automatically deriving the topology. A not necessarily unique attribute code is supplied by the user for each arc and polygon label at the time of digitising. The system also assigns to each a unique internal sequence number.

When the polygon topology is generated, each polygonal area formed is also assigned a unique internal sequence number by the system, together with the attribute code supplied by the user for the polygon label contained within the polygon (or zero if there is no label within the polygon). The system also creates a polygon attribute table and an arc attribute table which can be accessed by the user through the INFO relational database system. These tables contain one record for each polygon or arc, and initially hold the system number and user attribute code, and the area and perimeter of polygons and the topology and length of arcs (ESRI, 1985, pp 2.12 - 2.16). The user may store additional attributes in these tables, or in additional attribute tables which can be associated with these tables using relational operations.

The data structures and the commands provided by ARC/INFO give the user considerable flexibility in how area objects are represented. For example, the user could assign a unique code to each polygon label (and thus each polygon), and regard the polygons as being similar to LCGUs (as in GEOGRAF and DLG-3). Several overlapping sets of area objects could then be represented in the same data set by holding a list of area codes for each polygon in the polygon attribute table. The attributes of each set of area objects could be held in separate attribute tables.

An alternative approach would be to use a separate data set for each set of non-overlapping areas, and assign their area codes to the polygon labels. The polygons would then correspond directly to the detached parts of area objects (as in POLYVRT and GIRAS), and the attributes of the area objects could be held in the polygon attribute table and/or in additional attribute tables. ARC/INFO functions for spatial manipulation support either approach, although the organisation of the file system encourages the use of coverages (separate data sets) for each kind of area object. With this latter approach, an individual object within a coverage is represented by its corresponding polygon instead of being unduly fragmented.

ARC/INFO automatically establishes the existence of holes when deriving the polygon topology, and defines the spatial extent of each polygon by the arcs of its outer perimeter followed by the arcs of any interior holes as in GIRAS. However, this information is held within the ARC structure, and is not accessible by the user. It is possible to discover that holes exist in an area object from the topological information about the polygons and arcs held in the INFO structure, but it is not always possible to discover which are the outer and inner extents of the polygon. Thus it may not be possible to answer questions about the containment relationships between area objects, such as whether one area object totally surrounds another.

The Working Group on Terms and Definitions of the American National Committee for Digital Cartographic Data Standards held that 'holes in cartographic objects constitute a gap in our knowledge' (Moellering, 1984, p 24). HDS and GIRAS addressed this problem but both rely on the input of object-related information for extracting the relationships between boundaries. ARC/INFO and DLG-3 also recognise the existence of holes. However, only HDS makes explicit the hierarchy of geometric polygons.

3. DISASSOCIATIVE AREA MODEL (DAM)

This model disassociates the geometric and geographic components of areas with holes for separate academic consideration. This section briefly outlines the essential features of this model and then compares it with its predecessors.

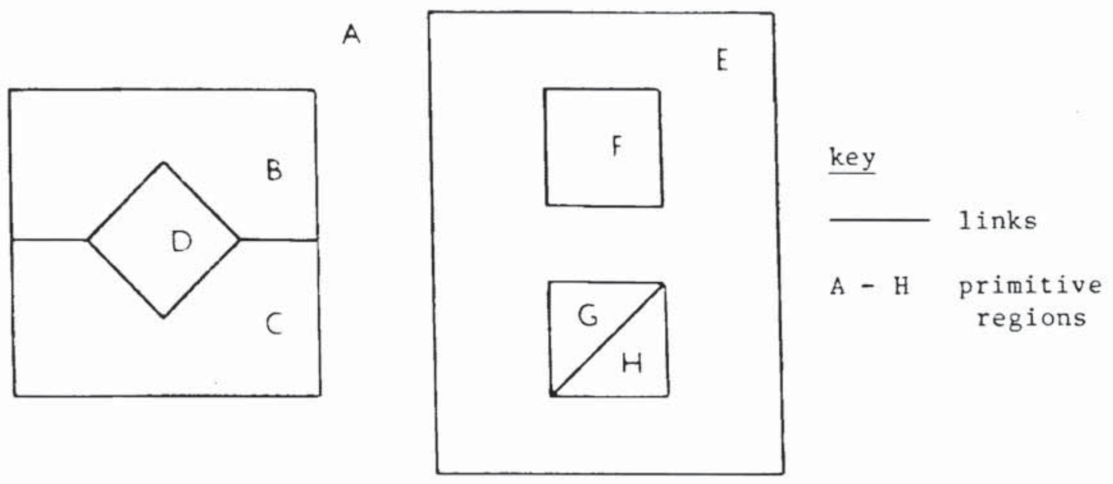
3.1 Geometry

Using only the geometry of the bounding lines, the areal map can be dissected into a set of non-overlapping parts, which we call primitive regions. For example, the linework of Figure 1a results in eight primitive regions (A - H), including one (A) which surrounds all the other primitive regions. The primitive region is similar to the LCGUs of GEOGRAF.

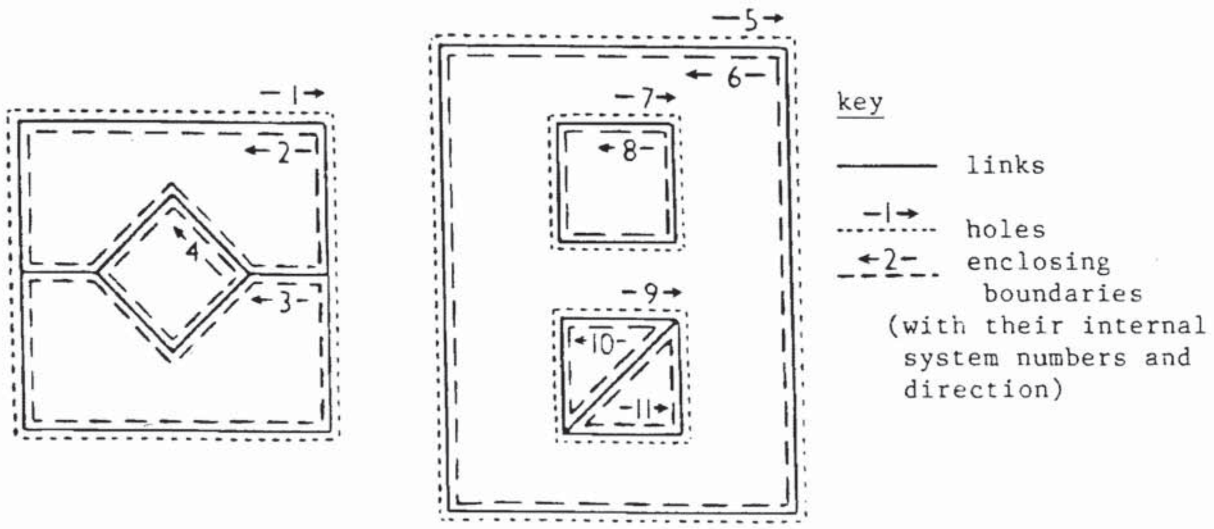
At this stage, primitive regions exist only in concept. The representation of the topology hinges on the boundary. Figure 1b shows the set of boundaries required to describe the primitive regions of Figure 1a. Each boundary is a closed loop, with direction, which defines one extent of a primitive region. A boundary can be sub-classified as being either an enclosing boundary or a hole. The outer boundary of a primitive region is known as an enclosing boundary, and any inner boundaries are known as holes. Thus each boundary forms an extent of one, and only one, primitive region and each primitive region is bounded by one enclosing boundary and zero or more holes. There is one exception to this - the outermost primitive region has no enclosing boundary but just one or more holes (in Figure 1b the outermost primitive region is described by holes 1 and 5). This primitive region forms the complement of the union of all the other primitive regions on the plane surface.

FIGURE 1

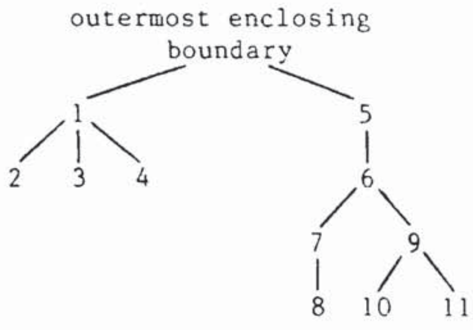
Representation of the geometry of area objects



a) The primitive regions formed by the linework of a map.



b) The set of boundaries which describe the above primitive regions.



	level in tree	type of boundary
....	0	enclosing
....	1	hole
....	2	enclosing
....	3	hole
....	4	enclosing

c) The rooted tree representing the containment hierarchy of the boundaries.

Each boundary has a separate existence within our intermediate data structure. Boundaries are equivalent to a closed loop in geometry, but they also have an associated direction to distinguish enclosing boundaries from holes. Where one primitive region completely surrounds another primitive region, the hole in the outer primitive region and the enclosing boundary of the inner primitive region are identical in shape. The two boundaries, however, remain unique since they have opposite direction (for example see boundaries 7 and 8 in Figure 1b). The direction of a boundary thus relates it to one specific primitive region.

Boundaries are composed of links which are similar to arcs. DAM is unconcerned as to how the link geometry is represented. It does, however, assume that the links do not cross and that they are node-matched in order to extract the boundaries. It would therefore be necessary to pre-process spaghetti digitising into a link and node structure. An algorithm for forming the boundaries from a set of links is given in Section 4.1.

When primitive regions all occur at the same level, that is, when there are no nested holes, there is a one-to-one correspondence between boundaries and primitive regions. For example, if the outermost primitive region, A, is excluded from consideration, there is a one-to-one correspondence between B, C, D and 2, 3, 4 respectively. Thus each primitive region may also be assigned the identity of its enclosing boundary. Links provide the adjacency relationships between boundaries and their corresponding primitive regions as in GEOGRAF.

When primitive regions (and therefore holes) are nested, there is a many-to-one correspondence between boundaries and those primitive regions containing holes. The links continue to provide the adjacency relationships between boundaries, and also between clusters of adjacent primitive regions. The problem is that the identity of primitive regions containing holes remains unknown. What exists, instead, is the distinct references to separate boundaries at the outer and inner extent of such primitive regions, and no single reference to the primitive regions themselves. For example, boundaries 6, 7, and 9 remain as separate references and there is no knowledge at this stage that they all belong to E.

The containment relationships between a complete set of boundaries can be viewed as forming a hierarchy which can be represented by a rooted tree. The root of the tree consists of a nominal reference to the enclosing boundary of the outermost primitive region, that is, the part of the plane surface which surrounds all the other boundaries. Each boundary is enclosed spatially by every boundary which precedes it in the tree but no other. When two boundaries are identical in shape, the one which is a hole is considered as surrounding the one which is an enclosing boundary. Thus the level of each boundary in the tree is equal to the number of other boundaries which surround it. Also, boundaries at even levels of the tree will be enclosing boundaries and those at odd levels will be holes. Figure 1c illustrates the general case of the rooted tree. Note that the tree is not an ordered rooted tree as there is no set order to the edges leaving each vertex of the tree.

This hierarchical system can be applied to any set of boundaries irrespective of their complexity. For example, currently the area within hole number 7 is uncut and forms a single primitive region. It could feasibly be cut into a number of primitive regions, as is the area within hole number 9. Some of these primitive regions may in turn contain further holes. However, this would all be represented by suitable branches starting from boundary 7 in place of the one shown.

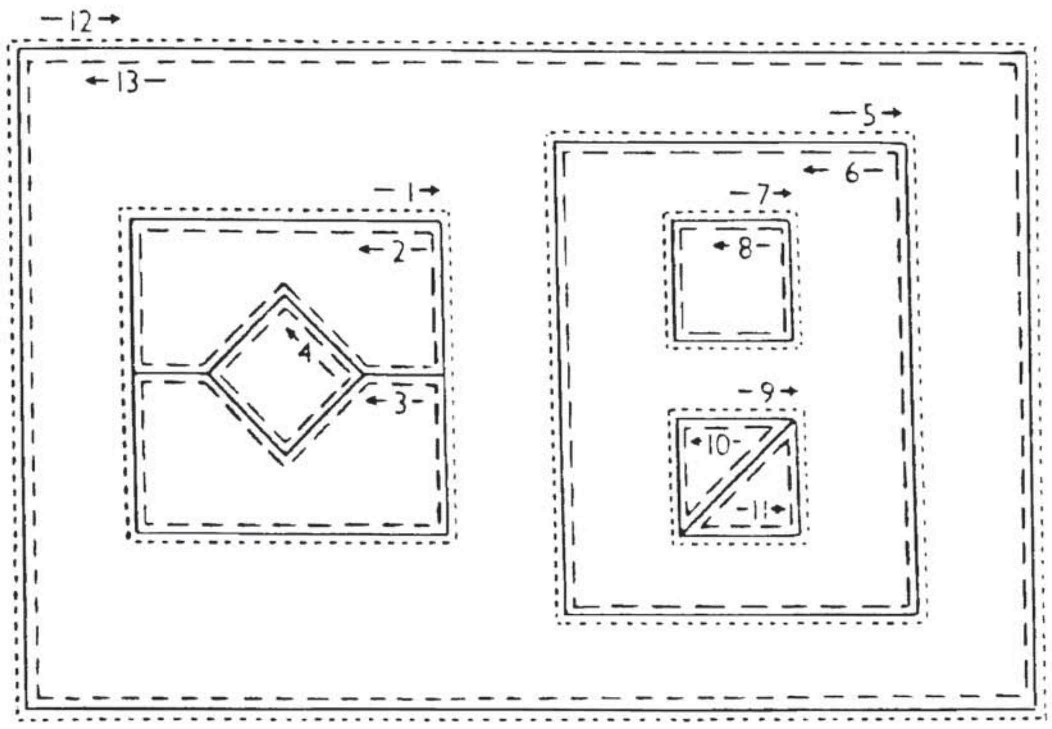
It is also possible that the map frame will be digitised around all the boundaries (see Figure 2a). This would have the effect of creating additional boundaries and primitive regions. If the map frame does not touch any of the other boundaries (as in the figure), then the hole and enclosing boundary created by the map frame would be at levels 1 and 2 respectively in the tree, and all the other boundaries would be moved down by two levels. Figure 2b shows the revised tree.

The derivation of such a tree fully resolves the containment relationships between the boundaries - and thus also the spatial extent of the primitive regions - since the holes within each primitive region immediately follow the enclosing boundary for that primitive region in the tree. The set of holes at level 1 of the tree describe the holes in the outermost primitive region, whose enclosing boundary is undefined. The tree also makes the nesting of primitive regions within holes in other primitive regions explicit. The derivation of this tree for a set of boundaries is a one-off process requiring an exhaustive spatial search. Forerunners to DAM were constrained by the inability in practice to resolve fully the relationships between the boundaries. An algorithm which attempts to minimise the computation required is given in Section 4.2.

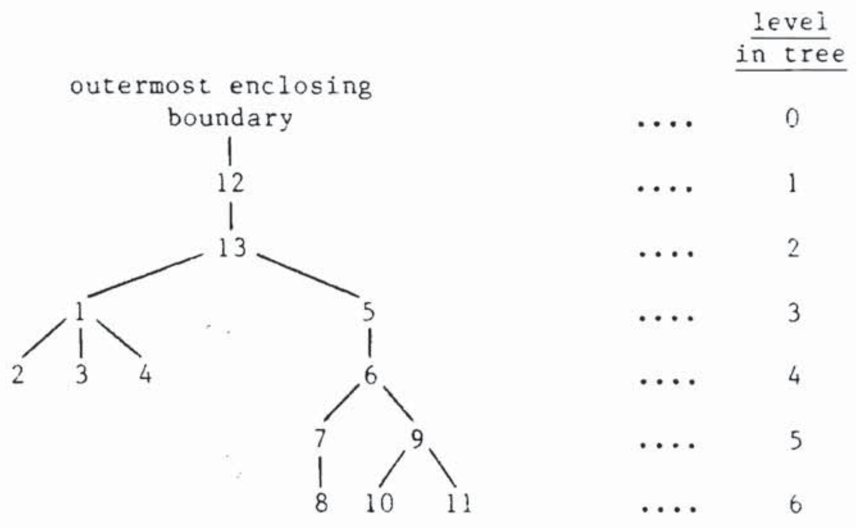
Once a primitive region assumes an identity, it becomes the basic building block for area modelling and forms the pivot between the geometry and the geography as in GEOGRAF and its derivatives.

FIGURE 2

The implications of including a map frame



a) Boundaries 12 and 13 are the hole and enclosing boundary formed by a map frame which does not touch any of the other boundaries.



b) The revised tree of boundaries.

3.2 Geography

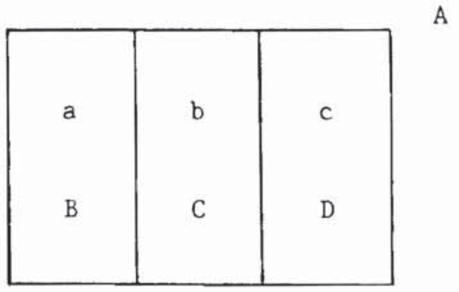
Area objects are specific instances of different types of geographic phenomena. Some applications require only simple objects, such as a description of the administrative areas at a specified level. Others require complex objects, such as a hierarchy of administrative areas. Each application may define its own set of types of simple and complex objects.

The spatial descriptions and other attributes of area objects may be provided in different formats by various sources; and applications may require access to some or all of these descriptions. DAM provides an application independent model of geometric entities and a framework for integrating the geographic and geometric descriptions where necessary. At the bottom-most level, there may be a one-to-one mapping between objects and primitive regions in the simplest case (Figure 3a). Where there are disjoint parts, there is a one-to-many relationship between objects and primitive regions (Figure 3b). DAM allows a variable number of objects to be associated with each primitive region, that is, a many-to-one mapping (Figure 3c). Thus it copes with both hierarchic objects (administrative hierarchies) as well as objects which overlap at any one level (broadcasting areas). Most other models cannot cope with the latter case.

DAM is not a universal model of topographic, let alone geographic, phenomena. It does, however, provide a framework for the flexible input of geographic area information in a variety of formats. For example the data about area objects might be associated with the links, or with digitised points which lie within the objects, or both. Alternatively, the user could point to the primitive regions which belong to each area object once the digitising of the links has been completed. In addition, information may be available from other sources,

FIGURE 3

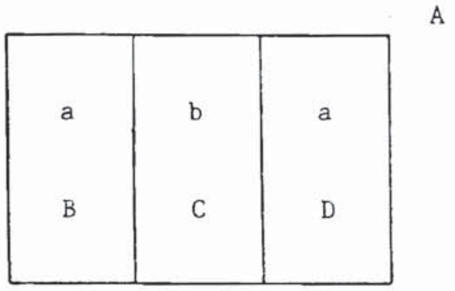
Relating area objects to the primitive regions



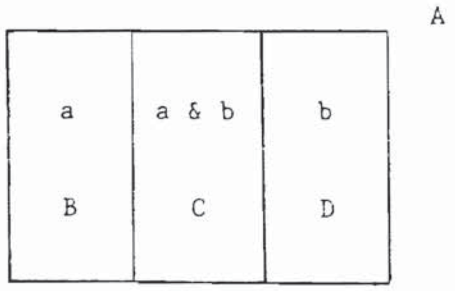
key

A - D	primitive regions
a - c	area objects

a) Simple area objects have a one-to-one relationship to primitive regions.



b) Area objects with disjoint parts have a one-to-many relationship to primitive regions.



c) Overlapping area objects have a many-to-one relationship to primitive regions.

such as a table which defines how a single level of objects form a hierarchy. Since phenomena under consideration and the input format are both variable, further development of DAM requires an interface to a rule-processing facility. Ad hoc processes, based on a given rule-set, must otherwise be provided for integrating the geography with the geometry and for data validation and automatic editing.

DAM adopts a 'human' view of the geometry of area entities in that, given the links, it is possible to extract a unique identity for each primitive region, to establish adjacency relationships and also to extract the information about the nesting of primitive regions. Since the geometric and geographic relationships are each processed separately and then related via the primitive region, DAM also provides a capability for cross-checking the geometric and geographic information (this is illustrated in Section 5 below).

Once the object data has been assigned to the primitive regions, the boundaries of geographic objects can be computed. This is done by forming the union of the primitive regions belonging to each object and discarding any internal lines. Each geographic object comprises one or more disjoint parts, and each disjoint part can be described by one outer boundary and zero or more inner boundaries in a similar way to the geometric primitive regions. The boundaries are stored as an ordered list of links. Through the links, the adjacent primitive regions, and thus adjacent areas, to this area can be found. In addition, if a list of the constituent primitive regions is kept for each object and a list of objects is kept for each primitive region, spatial relationships between areas can easily be determined.

3.3 Comparison of DAM with predecessors

Most modern data structures which represent areas in vector form use a POLYVRT type chain and node structure for holding the geometrical extent of area objects. However, whilst the chain and the polygonal boundary are also used in POLYVRT as a direct interface to the topology and the spatial extent of area objects respectively, successors to POLYVRT have introduced intermediary constructs in an attempt at greater flexibility.

GEOGRAF uses the LCGU as a building block for forming area objects. The chain now assumes a geometric role, and holds the topology of the LCGUs. The topology of the area objects is represented by the chain group. However, the LCGU (unlike DAMs primitive region) is assumed to be simply connected, and so there is a one-to-one correspondence between the boundary and the LCGU. This prevents the explicit representation of holes.

In HDS and GIRAS the chain once again has the topological role of separating two area objects (as in POLYVRT). The role of the polygonal boundary is altered, however. Instead of assuming that there is a one-to-one correspondence between polygonal boundaries and area objects, these structures define the spatial extent of an area object by one or more directed polygonal boundaries, thus allowing the representation of objects with holes. Both systems automatically establish the connection between the polygonal boundaries at the outer and inner extents of an area object; HDS goes further and makes explicit the full containment hierarchy between the enclosing boundaries and holes in a similar way to DAM, although Edwards et al (1977) conceived of them as interior and exterior regions respectively. Importantly, the areas formed by intersecting interior with the appropriate exterior regions in HDS may in some circumstances consist of more than one primitive region. This is due to the method used for boundary formation, which could result in a non-unique set of boundaries. Consequently, the interior and exterior regions may not form a hierarchy, let alone a unique one.

GIRAS differs from DAM in that, as in HDS, the directed polygonal boundaries are the boundaries of geographic objects, rather than geometric primitive regions. Both utilise information about the area objects when computing the containment between polygonal boundaries, instead of resolving it from the locational (or geometric) information alone - some reasons for considering the geometric and geographic information in isolation and only connecting it once the topology is formed were given in Section 1.

ARC/INFO, with its data structures and commands, allows geometric and geographic information to be considered separately and the topological relationships to be computed from only the geometric details. However, whilst it discovers the existence of holes, it defines the topology of the polygons formed only in terms of their adjacencies and does not record the containment relationships. Within ARC, the spatial extent of each polygonal area is described by a list of the arcs needed to create its boundaries; the arcs of the enclosing boundary being the first in the list. However, neither this list nor the coordinates of the arcs are accessible through INFO. It is difficult therefore, and sometimes impossible, to answer questions about the containment relationships between area objects. For example, a user would find it difficult to establish whether A contains B or B contains A, or to determine the areas contained by A.

The GIRAS model included some explicit information on containment. Fegeas et al (1983, p 13) described one reason for this, namely that this would make it easy for small polygonal areas to be eliminated or combined with larger surrounding areas, for example for cartographic and information generalisation. We also anticipate that the hierarchic structure of DAM, taken in conjunction with other properties, could be of value for the automatic recognition of area objects. The inclusion of containment relationships in the polygon attribute table of ARC/INFO would greatly extend its usefulness at least for the semi-automatic verification of attribute data. Alternatively, the topological information in ARC could be represented in a relational form and made (read-only) accessible to users through INFO. Others (van Roessel and Fosnight, 1985, and Kirby et al, 1986) have established that the topological information can be held and manipulated in relational form.

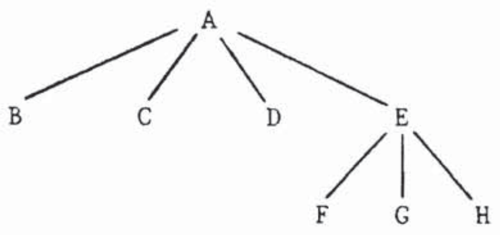
ARC/INFO and GIRAS do not maintain a separate identity for each boundary, but instead just give an ordered list of arcs which describe the outer and inner extents of each polygon. Thus if these systems were to make explicit the containment hierarchy in a similar way to HDS and DAM, it would be a hierarchy of polygons (that is, primitive regions) rather than a hierarchy of boundaries. Whilst it would give the containment of polygons within other polygons, it would not indicate the clusters of polygons which lie within each hole of another polygon. Thus some of the information concerning nesting is lost (see Figure 4).

DAM provides a synthesis of its predecessors and identifies the essential functions of the unit line, unit boundary, unit of space, and unit object of given area types. Within a specific pragmatic model, the functions of these various parts may be replicated or transferred to other units for efficient computer handling or for improving the user interface.

It is possible to deduce and use any pragmatic model, which is consistent with DAM, to specify data formats for various tasks, for example for data input, output, or transfer. As a corollary, DAM can be used to evaluate whether a complete and coherent description of area objects can be derived from a proposed pragmatic model. DAM was in fact constructed precisely for that purpose as described in Section 5.

FIGURE 4

Implications of storing the hierarchy of primitive regions rather than of boundaries



The rooted tree representing the containment hierarchy of the primitive regions in Figure 1a. There is no information about the clusters of adjacent primitive regions within each hole in a surrounding primitive region (cf. Figure 1c).

4. DERIVING THE TOPOLOGY

This section describes algorithms for deriving the topology of a set of primitive regions from the geometry of a set of node-matched links. This operation has two distinct stages: first, a set of boundaries are extracted from the links; then the containment hierarchy between the boundaries is computed. The data structure used during these operations is also given.

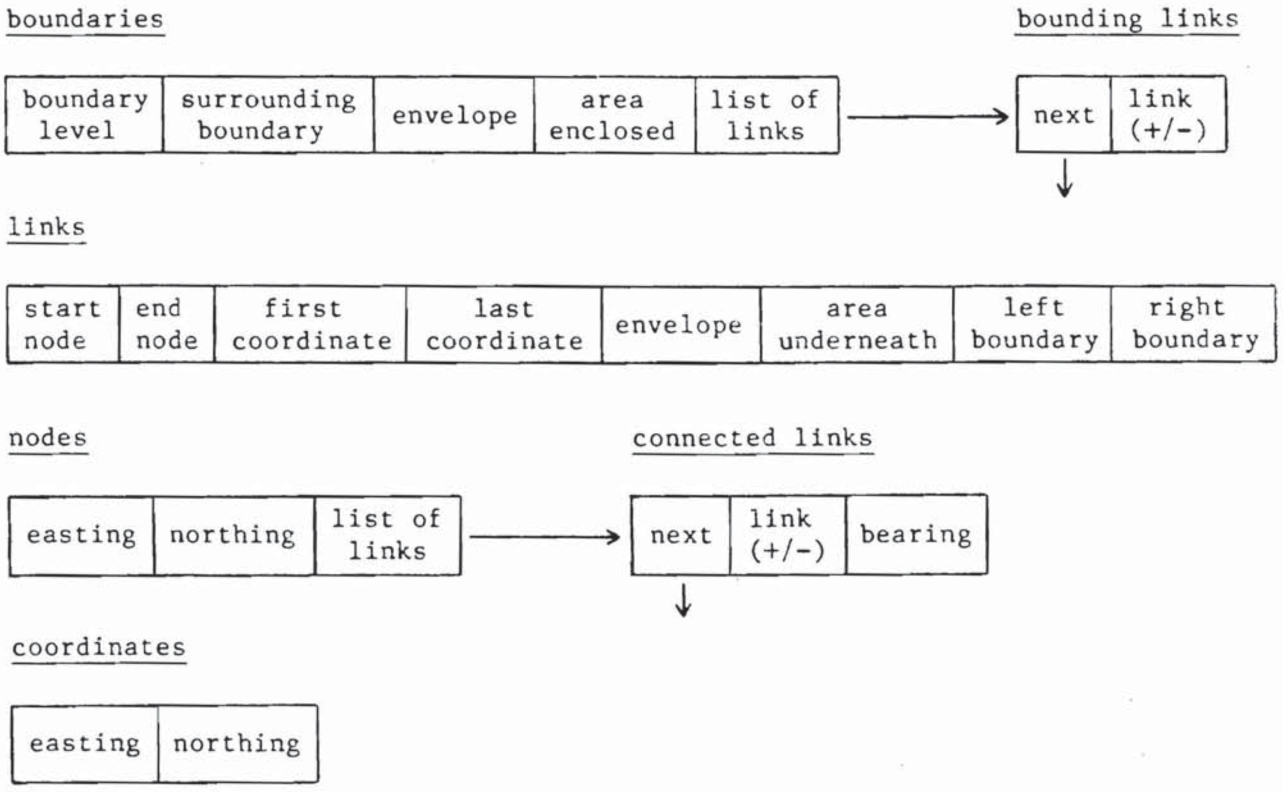
4.1 Extracting the Boundaries

This process connects together a set of links to form a set of closed boundaries. The process assumes that the links have been topologically structured into a link and node structure; if necessary, the data must be pre-processed to convert it to this form.

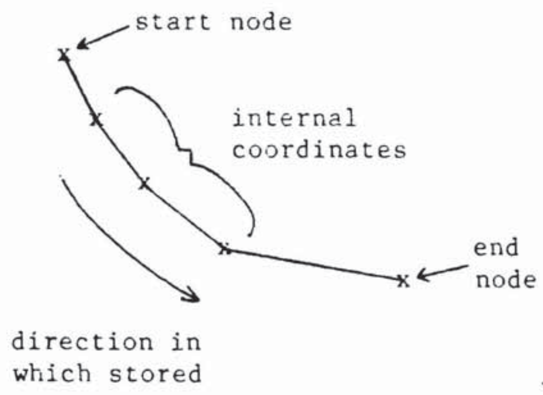
The links are encoded using the link, node, and coordinate records shown in Figure 5. These records are similar to the structure used by the POLYVRT system. One link record is used for holding the details of each link. The coordinates of the two end points of the link are stored in node records, whose record numbers are kept in the link record. The internal points of the link (if any) are stored separately in a contiguous block of records in the coordinates file. The record numbers of the first and last coordinates are again kept in the links record. There need be no restriction on the number of internal points in a link, although it may be desirable to set an upper limit for implementation purposes, in which case long links can be split into two or more. The envelope (that is enclosing rectangle) and area underneath a link are calculated and held in the link record, as these values are each required several times later. The area is obtained by summing the area of the trapezium underneath each

FIGURE 5

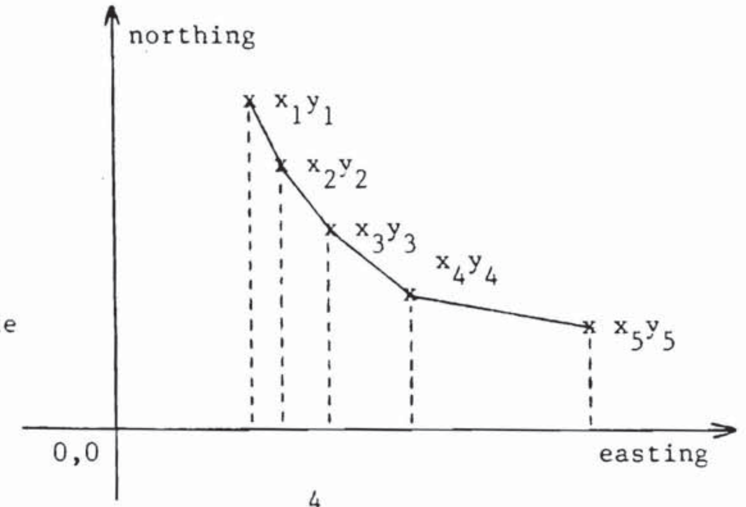
Data structure used for holding the geometry and topology of the linework of area boundaries



Example of a link



Calculation of area underneath a link



$$\text{area underneath} = \sum_{i=1}^{4} \tfrac{1}{2}(x_i - x_{i+1})(y_i + y_{i+1})$$

consecutive pair of points in the link using a standard formula (see Figure 5) - this yields a negative area for some links, which is used to advantage later. For example, this formula would yield a negative area for the example link in Figure 5; however, if the link was stored in the opposite direction (that is x_5y_5 is x_1y_1 and so on) then the area obtained would have the same magnitude but be positive.

There is one node record for each location where one or more links start and end. With each node is kept a list of the record numbers of the links which meet there. The record number is signed to indicate whether the link starts (positive) or ends (negative) at the node. The bearing at which each link leaves or enters the node is also held, and the lists are sorted on this field before the boundaries are formed so that the links are referred to in a clockwise order around the node.

Once all the links have been added to a database, the process of boundary formation can take place. One boundary record is used to hold the details of each boundary. An ordered list of the links which describe each boundary is formed, and the envelope and area enclosed are calculated. The algorithm used to form each boundary works by initially selecting one link, and then taking a succession of leftmost links radiating from the previously selected link until the beginning of the first link is reached again. This is now the preferred method used by several systems, as it only uses the geometric information about the links. Many earlier systems (such as POLYVRT and HDS) used information about the area objects on the left and right of the links when chaining the boundaries. Such a method is not suitable for systems which have moved away from the input of links together with left/right area codes (as in GIRAS and ARC/INFO).

The method for forming the boundaries is given by the following pseudo-code. The names of variables are given in upper case, and references to the database have the form 'RECORD_NAME [RECORD_NUMBER].FIELD_NAME', for example 'LINKS [I].START_NODE' refers to the value held in the 'start node' field of the i^{th} 'links' record. Comments to explain the logic are enclosed by curly brackets.

Algorithm to form the set of boundaries of a set of links

```
let BOUND = 0 {a counter for the number of boundaries formed}
for each LINK
    if LINKS [LINK].LEFT_BOUNDARY is still unknown
    then {form the boundary on the left of this link}
        let BOUND = BOUND + 1
        call FORM_BOUNDARY (BOUND, LINK)
    end if
    if LINKS [LINK].RIGHT_BOUNDARY is still unknown
    then {form the boundary on the right of this link}
        let BOUND = BOUND + 1
        call FORM_BOUNDARY (BOUND, -LINK)
    end if
end for
{BOUND now holds the number of boundaries formed}
```

Procedure to form the boundary on one side of a specified link

```
procedure FORM_BOUNDARY (BOUND, START_LINK)

   let LINK = START_LINK

   let BOUNDARIES [BOUND].ENVELOPE = LINKS [abs (LINK)].ENVELOPE

   let BOUNDARIES [BOUND].AREA_ENCLOSED = 0.0

   repeat {chain round the links which describe this boundary}

      if LINK > 0

      then {the boundary on the left of the link is being formed, so traverse
            the link in its stored direction, that is from its start node to
            its end node}

         let LINKS [LINK].LEFT_BOUNDARY = BOUND

         let BOUNDARIES [BOUND].AREA_ENCLOSED =
             BOUNDARIES [BOUND].AREA_ENCLOSED + LINKS [LINK].AREA_UNDERNEATH

         let DEST_NODE = LINKS [LINK].END_NODE

      else {the boundary on the right of the link is being formed, so
            traverse the link in reverse direction, that is from its end node
            to its start node}

         let LINKS [-LINK].RIGHT_BOUNDARY = BOUND

         let BOUNDARIES [BOUND].AREA_ENCLOSED =
             BOUNDARIES [BOUND].AREA_ENCLOSED - LINKS [-LINK].AREA_UNDERNEATH

         let DEST_NODE = LINKS [-LINK].START_NODE

      end if

      let BOUNDARIES [BOUND].ENVELOPE = the maximum extents of the union of
                                        BOUNDARIES [BOUND].ENVELOPE and
                                        LINKS [abs (LINK)].ENVELOPE

      append LINK to the list of links of BOUND

      {find the entry in the list of links of DEST_NODE for LINK entering the
       node (or leaving it if LINK is being followed in reverse direction)}

      let POINTER = NODES [DEST_NODE].LIST_OF_LINKS

      while CONNECTED_LINKS [POINTER].LINK <> -LINK

         let POINTER = CONNECTED_LINKS [POINTER].NEXT

      end while
```

```
        {assign the next link in the list of links of DEST_NODE to LINK. This
         is the next link to leave or enter DEST_NODE when moving round the
         node in a clockwise direction, and will be the next link followed to
         continue the formation of the boundary}

        if CONNECTED_LINKS [POINTER].NEXT = nil

        then {the entry for -LINK is the last in the list of links of
              DEST_NODE, so loop round to the first entry in the list}

           let LINK = CONNECTED_LINKS [NODES [DEST_NODE].LIST_OF_LINKS].LINK

        else

           let LINK = CONNECTED_LINKS [CONNECTED_LINKS [POINTER].NEXT].LINK

        end if

     until LINK = START_LINK

  end procedure
```

At the end of this process each link is part of two boundaries, where the boundary on its left has the same direction as the link, and the boundary on its right has the reverse direction. As a result of the method used for chaining the boundaries, it is automatic that enclosing boundaries have an anti-clockwise direction and holes have a clockwise direction. A further consequence is that enclosing boundaries have a positive area and holes have a negative area. This is the area enclosed by the boundary, and not that of the corresponding primitive region. The boundary does not have a separate identity in the ARC/INFO and GIRAS data structures, and so they only record the area of the primitive region. Note that the area of a primitive region can be derived by summing the areas of all its boundaries in DAM, whereas the converse (obtaining the area enclosed by a boundary) is not possible in ARC/INFO and GIRAS.

Each boundary has a separate existence within our data structure. The adjacencies between boundaries are given by the link records. However, where a primitive region contains holes, there is as yet no connection between the enclosing boundary of the primitive region and the boundaries which describe its holes. For example, there would be no connection between enclosing boundary 6 and holes 7 and 9 in Figure 1b. The set of holes belonging to the outermost primitive region have also not been identified. These relationships need to be resolved, and an algorithm for computing this will be described next.

4.2 Forming the Containment Hierarchy

As mentioned in Section 3.1, forming the containment hierarchy of a set of boundaries fully resolves the relationships between the boundaries, and thus the topology of the primitive regions. This hierarchy can be represented by a rooted tree, and the derivation of this tree for a set of boundaries is a one-off process. Edwards et al (1977, pp 20 - 22) describe the algorithm used in their HDS system; it utilises information about the area objects, and lacks the efficiency of the method given below. Information about area objects (the position of the polygon label) is also used in the GIRAS system in order to determine the outer and inner extents of a primitive region (Mitchell et al, 1977, p 11). However, the description of the algorithm used is not sufficiently detailed to allow comments on its efficiency. In ARC/INFO, no geographic information is used in resolving the primitive regions, as the supply of polygon labels by the user is optional. The ARC/INFO Users Manual (ESRI, 1985), however, does not explain the methods used.

A novel algorithm for forming the containment hierarchy between a set of boundaries follows. It does not require any geographical information, and it attempts to minimise the computation required. During the process, the first two fields of the boundary records are completed (Figure 5); that is the level of each boundary in the tree and also which boundary immediately precedes each boundary in the tree are ascertained. This information gives a complete description of the rooted tree.

The first part of this process is to find the immediate descendants of each hole in the tree:

Algorithm to discover the enclosing boundaries which immediately follow each hole

```
for each HOLE {that is, those boundaries with a negative area}
   push HOLE onto an empty stack
   repeat
      pop an item off the stack into BOUND
      {loop for each link in the list of links of BOUND}
      let POINTER = BOUNDARIES [BOUND].LIST_OF_LINKS
      while POINTER <> nil
         let LINK = BOUNDING_LINKS [POINTER].LINK
         if LINK > 0
         then {BOUND is on the left side of LINK}
            let ADJACENT_BOUND = LINKS [LINK].RIGHT_BOUNDARY
         else {BOUND is on the right side of LINK}
            let ADJACENT_BOUND = LINKS [-LINK].LEFT_BOUNDARY
         end if
         if ADJACENT_BOUND <> HOLE and
            BOUNDARIES [ADJACENT_BOUND].SURROUNDING_BOUNDARY is still unknown
         then
            push ADJACENT_BOUND onto the stack
            let BOUNDARIES [ADJACENT_BOUND].SURROUNDING_BOUNDARY = HOLE
         end if
```

```
        let POINTER = BOUNDING_LINKS [POINTER].NEXT
      end while
    until the stack is empty
  end for
```

Applying this to Figure 1b, when following the links of hole 1, enclosing boundaries 2 and 3 will be identified as being surrounded by hole 1, and they will be pushed onto the stack. When following the links of the next boundary taken from the stack (either 2 or 3), enclosing boundary 4 will also be identified as being surrounded by hole 1, and it too will be pushed onto the stack. The stack will then be emptied by the next iterations as there are no further immediate descendants of hole 1. Note that the algorithm identifies all immediate descendants of a hole, even if they do not share any common links with it (for example enclosing boundary 4 was correctly identified as following hole 1).

The same process would take place for holes 5 and 7, thus identifying the surrounding hole for enclosing boundaries 6 and 8 respectively. Hole 9 would also be found to surround enclosing boundaries 10 and 11.

The second part in the process is to find the immediate descendants of each enclosing boundary in the tree:

Algorithm to discover the holes which immediately follow each enclosing boundary

```
for each HOLE
  initialise an empty list for the possible surrounding boundaries of HOLE
  for each ENCLOSING_BOUNDary

     {perform an envelope and area test}

     if BOUNDARIES [HOLE].ENVELOPE is properly contained in
           BOUNDARIES [ENCLOSING_BOUND].ENVELOPE   and
        -(BOUNDARIES [HOLE].AREA_ENCLOSED) <
           BOUNDARIES [ENCLOSING_BOUND].AREA_ENCLOSED

     then

        add ENCLOSING_BOUND to the list of possible surrounding boundaries
         of HOLE

     end if

  end for

  let FOUND = false

  obtain the coordinates of an arbitrary point on the polygon of HOLE

  while the list is not empty and not FOUND

     remove the ENCLOSING_BOUNDary with the smallest area from the list

     if the point from the polygon of HOLE is enclosed by the polygon of
      ENCLOSING_BOUND {perform a point-in-polygon test}

     then {HOLE is an immediate descendant of ENCLOSING_BOUND in the tree}

        let FOUND = true

        let BOUNDARIES [HOLE].SURROUNDING_BOUNDARY = ENCLOSING_BOUND

     end if

  end while

  if not FOUND

  then {there is no boundary which surrounds HOLE, and thus the hole is
        at level 1 of the tree}

     let BOUNDARIES [HOLE].SURROUNDING_BOUNDARY = 0 {a rogue value}

     let BOUNDARIES [HOLE].BOUNDARY_LEVEL = 1

  end if

end for
```

Returning to Figure 1b, no enclosing boundaries will be found which surround holes 1 and 5, so a surrounding boundary of 0 is recorded. In addition, their level will be recorded as 1. The envelope and area test will suggest that hole 7 immediately follows enclosing boundary 6 in the tree, and this will be confirmed by the point-in-polygon test. Similarly hole 9 will be connected to enclosing boundary 6.

The final part of the process is to determine the level of each boundary. This is straightforward - the holes at level 1 are known, and so the enclosing boundaries which are surrounded by these holes are found and recorded as being at level 2. The holes surrounded by these enclosing boundaries can then be found and recorded as being at level 3, and so on, and this continues down the hierarchy. This completes the formation of the tree.

5. AN APPLICATION OF DAM

This section briefly describes how DAM was used in a project we undertook to derive hierarchic area objects from an Ordnance Survey (OS) database of feature coded vectors.

5.1 Background to the OS 1:625,000 Data

The 1:625,000 database was established by the Ordnance Survey for purposes of experimentation by themselves and others. The aim was to provide positive evidence towards the design of the 1:50,000 database (Haywood, 1984). The structure of the database was recognised as a crucial factor influencing not only the usefulness of a small-scale database but also its feasibility. The structure has implications for the cost of initial data capture and subsequent maintenance of the database.

It should be emphasised that the data we received was transferred from a database that the OS did not regard as final in terms of design nor complete as far as the data content was concerned. A systematic search for errors and omissions had not been carried out; these were known to exist, but their extent was completely unknown.

The OS database design considered the cost effective capture of data from maps to be one of the most important influences. We undertook to evaluate whether the OS design for representing the hierarchy of administrative areas (districts, counties, and countries) was adequate in concept, structure, and content for other purposes. The OS database does not provide an explicit description of the boundaries of the areas, nor does it give the hierarchical and spatial relationships between them (for example which districts belong to a county, or which counties are adjacent to one another). Instead, it just holds the essential topographic details of area objects using 'links' and 'area-seeds'.

Each link is a line (described by an ordered list of Cartesian coordinates) with an associated feature code which indicates the type of administrative boundary it represents. Only one link exists for each part of a boundary, and the feature code corresponds to the highest level of administrative area to which the boundary belongs. Since the administrative units form a hierarchy, it could be inferred that the boundaries of high-level objects also form a boundary of objects below them in the hierarchy and that the coastline could form all other boundaries.

Each detached part of an administrative unit is indicated by an area-seed, a representative point within the polygon enclosing that part of the object. This polygon is similar to a GEOGRAF polygon. The area-seeds carry the name of the area, and also have an associated feature code to identify the type of area to which they relate. For the administrative areas there are codes for national, county, metropolitan county, and district, and also a further set of codes for the seaward extensions of the administrative areas.

This pragmatic data model is extremely convenient and cost-effective for data capture since it records once, and only once, each explicitly recorded map detail. However, a consequence is that the extraction of area objects is quite complex.

5.2 Extraction of the Geometric Topology

Our first task was to extract the links and area-seeds relating to the administrative areas from the remainder of the OS data. The links were stored in a POLYVRT type structure - this conversion was straightforward, as the links for the administrative boundaries do not cross one another and only meet at their end points. This stage was only concerned with the geometry, or course, of the links; however, the link feature codes and the area-seeds were also stored for later use.

The OS data is divided into units of 100 x 100 km. squares, based upon the National Grid. These units are referred to as 100 km. libraries. We were supplied with four such libraries which together made a 200 km. square whose south-west corner had the grid reference SN 000000. The four libraries were amalgamated into a single data base to ease subsequent processing. However, areas along the edge of the 200 km. square remained cut and their links remained as dangling lines. Extra links which corresponded to the edge of the 200 km. square were therefore inserted into the database so that the boundaries of the incomplete areas became closed.

The last part of the geometric processing was to form the boundaries of the primitive regions and calculate their containment hierarchy, as described in Sections 4.1 and 4.2 respectively.

5.3 Identification of Area Objects

In the OS scheme, the construction of an area object begins with the retrieval of all its area-seeds. From each area-seed a spatial search takes place according to a set of rules to find a node, that is, a link end point, which lies on the outer boundary of that area. Links are then chained from this starting node, according to further rules, to form a complete polygon. This process identifies the outer boundary of each disjoint part of the area object. Further checks and processing are required to establish whether there are any holes within the area, and if so to determine their boundaries.

In our scheme, the aim is to classify each unit of space (that is, primitive region), rather than identify the linework of the area boundaries. This involves finding the name and type of district, county, and country of each primitive region. By focusing on the primitive region it is possible to provide a more coherent representation of area objects. For example, the representation of the containment relationships between primitive regions makes it unnecessary for repeated spatial searches to establish whether area objects contain holes. Also, the knowledge of the spatial relationships between the primitive regions enables checks to be carried out to ensure that the relationships between area objects are consistent.

Figure 6 illustrates the techniques that were used to name the primitive regions. The various line styles indicate the feature codes that were supplied with the links. Firstly, each area-seed was allocated to a primitive region by using a point-in-polygon test to find the primitive region containing it, and its associated attributes (the type of the area-seed and the name of the administrative unit) were assigned to the primitive region. Figure 6a has the area-seeds marked upon it.

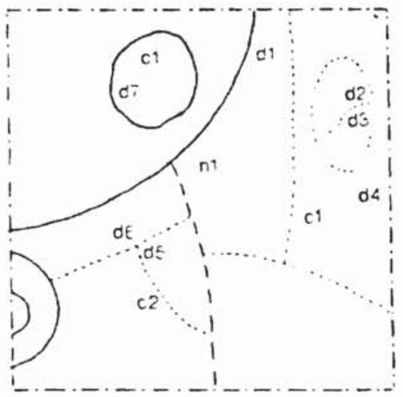
The next process was to search outwards and inwards from these named primitive regions to find neighbouring primitive regions which belong to the same object,

FIGURE 6

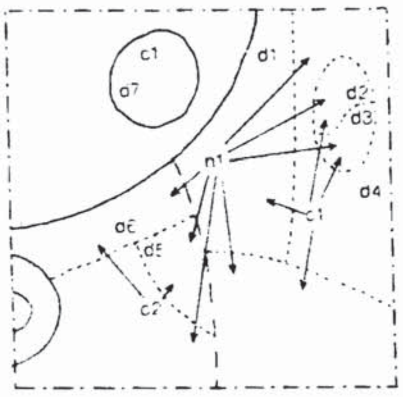
The naming of primitive regions

key

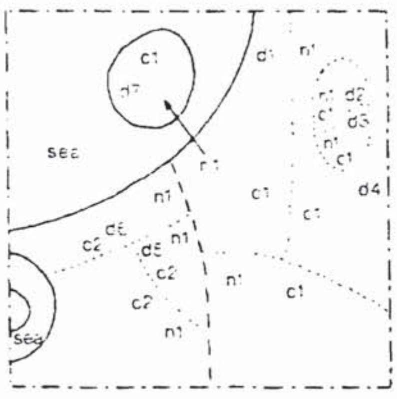
———	coast links
- - - - -	county links
··········	district links
—·—·—·	links corresponding to the map frame added by our software
n1	national area-seed
c1,c2	county area-seeds
d1-d7	district area-seeds



a) The attributes of the area-seeds are assigned to the primitive regions containing them.



b) The attributes of named primitive regions are assigned to unnamed neighbours which belong to the same administrative unit.



c) Names are deduced where area-seeds are intentionally omitted in the OS data, and the sea is identified.

that is, a named administrative unit. The knowledge of the spatial relationships between the primitive regions made this a thorough and efficient ptocess. Each search started from a seeded primitive region. If the feature code of a link which bounds the primitive region was below the level of the seed, the name was also given to the adjoining primitive region if it was still unnamed. The extent of this search was limited by a link at an equal or higher level than the seed. Figure 6b illustrates this process.

Following this there were still some primitive regions which were not named at all levels. This arises because:

1) The area-seeds for some area objects, which were cut by the map edge, occurred outside the map edge and were thus missing from the subset we received. This type of omission could not be overcome and so these areas could not be named. The existence of these unnamed objects was established however.

2) Some omissions in the OS data were intentional. Islands do not contain national area-seeds, and seaward extensions do not contain national and sometimes county area-seeds. It was possible to deduce the names in such cases from other information. For an island, its nationality was taken from a land counterpart which had the same district or county name. If missing, the nationality and county of a seaward extension was similarly taken from a land counterpart with the same district name.

There was one further set of unnamed primitive regions which arose from the system of identifying each unit of space: those which represent the sea. It was possible to name these primitive regions as such by starting from any one primitive region whose land/sea status was known and expanding outwards and inwards. The land/sea state changes of course every time a link with the coastline feature code is crossed.

Figure 6c shows the naming of the primitive regions after all the above processing. There is one primitive region whose county is unknown and three whose district is unknown due to these areas being truncated by the map edge. There is also one on the left edge of the map whose nationality has not been supplied and cannot be deduced. It is known to be land, however, due to the wider context made explicit by the processing.

It was then possible to build the hierarchy of administrative areas within the restrictions caused by the lack of data. This involved:

1) the allocation of districts to counties, and counties to countries; and
2) the extraction of the boundaries of the areas by aggregating their primitive regions and eliminating internal boundaries.

The area objects do not yet exist as such within the system; instead, they are held as attributes of the primitive regions. However, there is the knowledge within the system for the extraction of the entire hierarchy of area objects in a variety of forms, including the formats required for transfer to other systems (such as GIMMS and DLG-3).

The use of an intermediate data structure, based on DAM, which focussed on the geometric topology of the primitive regions rather than on the topology of the links, made it possible to validate partially the OS data, for example:

1) that the disjoint parts of area objects at each level of the administrative hierarchy each contain just one area-seed for that level;

2) that each area object is completely bounded by links with the appropriate feature codes;
3) that the administrative areas form a proper hierarchy;
4) that the type of each area is appropriate to its status as land or sea; and
5) that all areas are named, except those on the edge of the map.

We found just one coding error; the links which form the northern seaward extension of the England/Wales boundary had been feature coded as a county seaward extension. This violated the rule that the feature code of a boundary should correspond to its highest level in the administrative hierarchy.

5.4 Summary of the Project

The project confirmed that it is possible to derive transformations of the OS 1:625,000 experimental data for manipulating hierarchically related objects and their polygonal descriptions. The object hierarchy had to be inferred from fragments of information, using the scattered clues and the nesting rules for the hierarchic link and area-seed feature codes. The concept of the primitive region, and the knowledge of the geometric topology, provided a convenient framework for solving the puzzle. Further details of this project can be found in Visvalingam et al (1985).

A Geographic Information System should be capable of deriving a complete description of hierarchically related and/or overlapping area objects from feature-coded map details, that is, fragments of information. In theory, ARC/INFO appears to be capable of this. However, the verification of the resulting information base would involve the re-formulation of relatively simple rules in terms of possibly complex relational operators. Moreover, in some cases such verification may be impossible if the user does not have access to all information, especially information supplied by the user (for example, multiple polygon labels within one polygon).

6. CONCLUSION

This paper has made a number of contributions and raised several issues. These can be summarised as follows:

1) The paper presented a critical review of the major academic ideas on cartographic data structures to date. The functions of the unit line, unit boundary and unit of space were articulated in POLYVRT, HDS and GEOGRAF respectively. In ARC/INFO the geometric and geographic descriptions of area objects were divorced and connected through the unit of space.

2) We then described the Disassociative Area Model (DAM), which provides a synthesis of the above concepts. DAM is a conceptual description of the geometric components of polygonal area covers. The model identifies the essential functions of the unit line, unit boundary, unit of space and unit object of given data types. The unit of space (primitive region) continues to connect the geometric and geographic descriptions as in ARC/INFO. However, we believe that all descriptions can and should be made accessible by users through use of a single data model in ARC/INFO.

   DAM's description of area topology utilises the concept of a tree of boundaries as proposed in HDS. This tree describes the containment relationships between the boundaries, and identifies the boundaries which define each primitive region. Boundaries in turn are composed of links (that is, node matched lines). However, unlike in HDS, GIRAS and ARC/INFO, the links refer to the boundaries on either side and not to the objects or primitive regions directly. The boundary thus becomes the connection between the unit line and unit of space.

3) The paper also outlined some data structures and algorithms for the automatic extraction of boundaries, the tree of boundaries, and thus the primitive regions, using only the geometric information. The algorithms and data structures could be adapted to suit specific requirements. For example, static and variable data sets would require slightly different representations.

4) DAM provides a convenient framework for the flexible input of geographic area information in a variety of formats and allows a partial verification of geometric and geographic components against each other. This was demonstrated by a case study in which we extracted a complete and coherent description, compatible with the DAM model, of hierarchically organised administrative areas from feature coded map details, that is, from fragments of information.

This was effected through ad hoc processes for integrating the geography with the geometry and for data validation. However, we believe that this should be (since it could be) effected through a general purpose GIS. In state-of-the-art GIS, which tend to be based on the relational model, rule-processing would be effected with relational operators. This may be an acceptable solution in the short term, but we believe that the relational model neither provides a natural user interface for rule-processing, nor an ideal data model for cartographic phenomena.

We therefore believe that while existing GIS serve a useful role, their availability should not stifle the quest for fresh ideas.

ACKNOWLEDGEMENTS

We are grateful to a number of organisations and individuals. The Science and Engineering Research Council granted the CASE studentship held by P. Wade, with the Market Analysis Division of CACI as the collaborating body. The Ordnance Survey provided a small grant and access to their data and internal documentation. The University of Hull provided the research base.

We are also grateful to John Griffiths of the C.I.S.R.G. and David Jordan of the Department of Pure Mathematics, University of Hull, for discussions on containment tests and terminology in area topology respectively.

REFERENCES

Allder, W.R. and Elassal, A.A. (1984) USGS Digital Cartographic Data Standards. Digital Line Graphs from 1:24,000-scale Maps. U.S. Geological Survey Circular 895-C.

Cooke, D.F. and Maxfield, W.H. (1967) The Development of a Geographic Base File and its Uses for Mapping. Papers from the Fifth Annual Conference of the Urban and Regional Information Systems Association, 207 - 219.

Edwards, R.G., Durfee, R.C. and Coleman, P.R. (1977) Definition of a Hierarchical Polygonal Data Structure and the Associated Conversion of a Geographic Base File from Boundary Segment Format. An Advanced Study Symposium on Topological Data Structures for Geographic Information Systems, Harvard University, Cambridge, Massachusetts.

ESRI (1985) ARC/INFO Users Manual - Version 3. Environmental Systems Research Institute, Redlands, California.

Fegeas, R.G., Claire, R.W., Guptill, S.C., Anderson, K.E. and Hallam, C.A. (1983) USGS Digital Cartographic Data Standards. Land Use and Land Cover Digital Data. U.S. Geological Survey Circular 895-E.

Haywood, P.E. (1984) The Ordnance Survey 1:625,000 Database: General Principles and Data Structure. Ordnance Survey Internal Report.

Kirby, G.H., Wade, P. and Visvalingam, M. (1986) Storage and Retrieval of Topographic Data using a Relational Database Management System. Stage 2

Report, Ordnance Survey (OS) contract on Computer Handling of OS 1:625,000 Digital Maps.

Laboratory for Computer Graphics and Spatial Analysis (1974) POLYVRT Manual. Harvard University, Cambridge, Massachusetts.

Mitchell, W.B., Guptill, S.C., Anderson, K.E., Fegeas, R.G. and Hallam, C.A. (1977) GIRAS: A Geographic Information Retrieval and Analysis System for Handling Land Use and Land Cover Data. U.S. Geological Survey Professional Paper 1059.

Moellering, H. (1984) A Working Bibliography for Digital Cartographic Data Standards. Issues in Digital Cartographic Data Standards, Report No. 5, National Committee for Digital Cartographic Data Standards, Columbus, Ohio.

Peucker, T.K. and Chrisman, N. (1975) Cartographic Data Structures. The American Cartographer, 2(1), 55 - 69.

van Roessel, J.W. and Fosnight, E.A. (1985) A Relational Approach to Vector Data Structure Conversion. Auto-Carto 7 Proceedings (American Society of Photogrammetry and American Congress on Surveying and Mapping, Washington D.C.), 541 - 551.

Visvalingam, M., Kirby, G.H. and Wade, P. (1985) Extraction of a Complete Description of Hierarchically Related Area Objects from Feature-coded Map Details. Stage 1 Report, Ordnance Survey (OS) contract on Computer Handling of OS 1:625,000 Digital Maps. (This document is available as an OS Technical Paper).