Photogrammetrie • Fernerkundung • Geoinformation 3/2008, S. 175–186, 9 figs.

Towards a G-Map Based Tool for the Modelling and Management of Topology in Multiple Representation Databases

ANDREAS THOMSEN, MARTIN BREUNIG, EDGAR BUTWILOWSKI, Osnabrück

Keywords: 3D Topology, Multi-Scale Representation, Geo-database, MRDB, LOD, Abstraction of Geoinformation, Generalized Map, Cell-Tuple Structure

Summary: The modelling and management of topology plays an increasing role in Photogrammetry, Remote Sensing and GIS. In new application fields such as 3D city models, 3D navigation systems and early warning of geological events, multi-scale topological data models are needed. In this article a unified model for topology is proposed, based on oriented d-Generalised Maps ($2 \leq d \leq 3$) to represent topology in Multi-Representation Databases (MRDB). Topological data structures and operations are presented in detail. The model can be used as a data integration platform for 2D and 3D topology, as well as for the representation and management of topology in a multi-resolution GIS. An application example, management of topology in a multi-scale land-use map based on aggregation, shows the feasibility of the new approach.

Zusammenfassung: *Auf dem Weg zu einem G-Map-basierten Werkzeug zur einheitlichen Modellierung und Verwaltung der Topologie in Multiple Representation Databases.* Die Modellierung und Verwaltung der Topologie spielt eine größer werdende Rolle in der Photogrammetrie, Fernerkundung und in GIS. In neuen Anwendungsfeldern wie 3D Stadtmodellen, 3D Navigationssystemen und Frühwarnung geologischer Ereignisse werden multiskalige topologische Datenmodelle benötigt. In diesem Artikel wird ein einheitliches Modell für die Topologie vorgeschlagen, das auf orientierten d-Generalisierten Karten ($2 \leq d \leq 3$) basiert, um die Topologie in Multi-Representation Databases (MRDB) zu verwalten. Topologische Datenstrukturen und Operationen werden im Detail vorgestellt. Das Modell kann als Datenintegrationsplattform für 2D und 3D Topologie genutzt werden. Ein Anwendungsbeispiel, die Verwaltung der Topologie in einer multiskaligen durch Aggregation entstandenen Bodennutzungskarte, zeigt die Realisierbarkeit des neuen Ansatzes.

1 Introduction and Related Work

Multiple representation databases are needed in many applications of Photogrammetry, Remote Sensing and GIS (HOPPE 1996, FRADIN et al. 2002, HAUNERT & SESTER 2005 and MEINE & KÖTHE 2005), to model geo-objects in different scales. In new 3D applications, besides geometry the importance of topology is growing, but hitherto this aspect has played a minor role in research and applications.

Exhaustive work on the modeling of topology in GIS has been published (EGENHOFER 1989, EGENHOFER et al. 1989, PIGOT 1992, CLEMENTINI & DI FELICE 1994 and GRÖGER & PLÜMER 2005). General approaches for representing topology in the context of 3D modeling have been examined by different authors. Cellular complexes, and in particular cellular partitions of d-dimensional manifolds (d-CPM) have been described to represent the topology of an extensive class of spatial objects (BRISSON

1432-8364/08/2008/175 $ 3.00

1993). Based on algebraic topology, they provide a general, less rigid framework than more specific topological representations such as simplicial complexes. The topology of d-CPM can be represented by d-dimensional *Cell-Tuple Structures* (Brisson 1993) respectively d-dimensional *Generalized Maps* (d-G-Maps – Lienhardt 1994). Lévy (1999) has shown that 3D-G-Maps have comparable space and time behavior as the well-known Doubly-Connected Edge List (DCEL) and Radial Edge structures, but can be used for a much wider range of applications, allowing for a more concise code. Lévy (1999) also introduces *hierarchical G-Maps* (HG-Maps) for the representation of nested structures. G-Maps have been used to represent the topology of land-use changes (Raza & Kainz 1999) for 3-dimensional spatial data (Mesgari 2000), and are applied, e. g., in the geoscientific 3D-Modelling software GOCAD (Mallet 2002). Fradin et al. (2002) use G-Maps to model and visualize architectural complexes in a hierarchy of multi-partitions, and an interactive graphical G-Map-based 3D-modeller (Moka 2006) has been made available by the group of graphical informatics (SIC) at Poitiers university. Own first approaches to the management of topology in Multiple Representation Databases have been shown in (Shumilov et al. 2002, Thomsen & Breunig 2007, Butwilowski 2007, and Thomsen et al. 2008).

In the remainder of this paper, we investigate how the realisation of oriented Generalized Maps and Cell-Tuple Structures based on an ORDBMS can be used to handle the topology of a digital spatial model in a generic way, supporting 2D manifolds and 3D volume models (Sections 2 and 3). In Section 4, the comparison of the G-Map with other topological models, especially with ISO 19107, is discussed. Section 5 describes some aspects of the implementation based on a object-oriented RDBMS, and Section 6 discusses different methods to handle multi-representation G-Maps and Cell-Tuple Structures. As an ongoing application example, in Section 7 we present the management of topology for a multi-scale land use map. We conclude with an outlook on our future research.

2 Explicit Modelling of d-dimensional Topology

Topological relationships between and within complex spatial objects can be described implicitly, e. g., by attaching some references to neighbours to the items of a geometry model. However, it seems more appropriate to use an explicit mathematical framework for the description and analysis of the transformation of topology during the passage between different representations in a multi-representation database. For this purpose, we use a topological model that consists of the following conceptual layers (Mallet 2002):

- the continuous *d-dimensional manifold*,
- its cellular partition which results in a finite *d-dimensional cellular complex*,
- the representation by a *d-Generalized Map* and its realisation by a *d-Cell-Tuple structure*, and
- the persistent implementation by means of tables, relationships and problem-specific functionality in an *object-relational database*.

Continuous d-manifold. A *d-dimensional manifold M in 3D*, can be roughly described as a continuous part of a space which is *locally homeomorphic* to a *d-dimensional ball in* $\boldsymbol{R}^3$. This means that for any point $p \in$ M, there exists a neighbourhood $U(p) \subseteq M$, a point $q \in \boldsymbol{R}^3$, a d-dimensional ball $B(q) \subseteq \boldsymbol{R}^3$ and a bijection φ_U: $U \leftrightarrow B$, $\varphi_U(p) = q$, continuous in both directions, that maps any point of U to a point of B. Special conditions apply to *bounded manifolds*, which are locally homeomorphic to a *half-space*. By this definition, certain singular configurations (“non-manifold situations”), e. g., “T-shaped” branchings of in 2D-manifolds, are excluded (Lévy 1999).

Cellular partition. Whereas the manifold is a continuous object comprising an infinity of points in space, for a digital representation, a discrete structure is required. This is achieved by the introduction of *cellular partitions* of continuous d-manifolds: the manifold M is decomposed into a number of *cells* c^k of dimension k ranging from *0* to d, the dimension of the manifold. Each cell c^k is homeomorphic to a k-dimensional ball $B^k \subseteq \boldsymbol{R}^k$. Such a *d-dimensional cellular partition* of a d-dimensional manifold M is a *cellular complex* consisting of a finite number of cells c^k of positive dimension $k \leq d$, and verifying the following conditions:

- For each cell c^k, its boundary ∂c^k is composed of cells $c^{k'}$ of lesser dimension $k' < k$.
- For each pair of different cells of dimension k, ${c^k}_i$, ${c^k}_j$, the open interiors ${c^k}_i$, c_j do not intersect.
- For each pair ${c^k}_i$, c_j of different cells of dimension k, the intersection of the boundaries is either void or consists of cells c^l of dimension $l < k$: $c^k \cap c^{k'} = \boldsymbol{U}_{l=0,\ldots,k-1}(\boldsymbol{U}_j\ {c^l}_j)$.

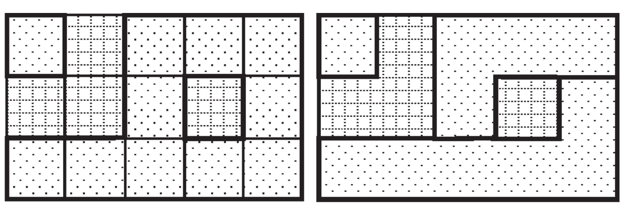
In a d-dimensional cellular complex, multiply connected objects cannot be represented as single d-cells: e. g., a 2D ring, a 3D ring (a "doughnut"), or its surface, a torus. This is different, e. g., from the spatial models provided by Oracle Spatial 11G (Kazar et al. 2008), and ISO19107, which allow, e. g., faces and solids with holes as basic elements. By using partitions of d-dimensional manifolds as discrete model of topology, we ensure that each (d-1)-cell is either part of the common boundary cell of a pair of d-cells, or is part of the outer boundary of the cellular complex. We exclude "non-manifold" configurations like, e. g., T-shaped contacts of d-cells (Lévy 1999). Mesgari (2000) proposes a solution for the modelling of some of these singularities.

Spatial objects in a cellular partition. As illustrated by the above counter-examples, certain spatial objects that occur in practical applications cannot be modelled by single d-cells, but must be represented as sets of d-cells (cf. Fig. 1), consisting of one or more connected components.

D-G-Maps and Cell-Tuple Structures. Cellular Complexes can be interpreted as a generalisation of simplicial complexes, but they lack the geometric and algebraic properties of the latter. However, the *d-dimensional Generalized Map* (Lienhardt 1994) and the *d-dimensional Cell-tuple Structure* (Brisson 1993) provide the cellular partition with the structure of an *abstract simplicial complex*. From a practical point of view, the two models can be regarded as roughly equivalent (Lienhardt 1991). Whereas the d-G-Map focuses on the algebraic structure defined by the transitions between the abstract *darts*, the Cell-Tuple Structure yields a realisation by cell-tuples and the symmetric "switch" relationships between them.

3 Generalized Maps and Cell-Tuple Structures

Following Lienhardt (1994), a *d-dimensional Generalized Map* (*d-G-Map*) consists of a finite set of objects called *darts* and a



a) b)

Fig. 1: Representation of 2D spatial objects as sets of 2-cells in a 2-G-Map – Objects that are not simply connected require more than one 2-cell for representation (left); Representation by a minimal number of cells (right).

set of bijections α_i, $i = 0, \ldots, d$ linking pairs of darts, that are *involutions*, i. e., that verify the conditions:

$$\alpha_i(\alpha_i(x)) = x \quad (1)$$

$$\text{and for all } i, 0 \leq i < i+2 \leq j \leq d, \quad (2)$$

$\alpha_i\ \alpha_j$ is an involution, i. e., $\alpha_i(\alpha_j(\alpha_i\ (\alpha_j(x)))) = x$, which implies $\alpha_j(\alpha_i(x)) = \alpha_i(\alpha_j(x))$. (3)

The G-Maps are embedded in 2D or 3D space by a mapping that to each dart associates a unique combination of a node, an edge, a face, and in 3D a solid.

In Brisson's (1993) terminology, *Cell-tuple Structures* consist of a set of *cell-tuples (node, edge, face[, solid])* attached to the corresponding spatial objects. The cell-tuples are pairwise linked by "switches" defined by the exchange of exactly one component, and corresponding to Lienhardt's involutions:

$$\begin{aligned} &\alpha_0: (n, e, f, s) \leftrightarrow (n', e, f, s),\\ &\alpha_1: (n, e, f, s) \leftrightarrow (n, e', f, s),\\ &\alpha_2: (n, e, f, s) \leftrightarrow (n, e, f', s),\\ &\alpha_3: (n, e, f, s) \leftrightarrow (n, e, f, s') \end{aligned} \quad (4)$$

A d-G-Map can be represented as a graph with cell-tuples as nodes, and edges defined by the involution operations (cf. Fig. 2).

Orientation. We require all d-manifolds to be *orientable*, and the corresponding G-Maps and Cell-Tuple Structures to be *oriented*: the set of darts / cell-tuples can be divided into two parts of equal size carrying opposite sign, and each α_i transition links a pair of items of opposite sign. Whereas Lienhardt's definition permits darts at the boundary of a G-Map, that do not have a counterpart for the α_d transition, such a situation is excluded in our model (Pigot 1992). The G-Maps here are formally considered as *unbounded*, by introducing an outside "universe" that is not a standard cell.

Navigation on G-Maps. The use of the G-Map structure for topological queries and operations, is supported by methods for the *navigation* on the G-Map graph, and for the retrieval of solids, faces, edges and nodes, represented by volume cells, surface patches, curves and points.

a) All navigation on a G-Map relies on the α_i transitions, i. e., the passage from one cell-tuple to its neighbour by the exchange of one cell of dimension *i*.
b) A sequence of cell-tuples starting with ct_0, finishing with ct_N and linked by α_i transitions with varying index *i* defines a *path* on the G-Map graph, or a *loop*, if it is closed.

Paths and loops are determined by the sequence of transitions $\alpha_{i_1} \ldots \alpha_{iN}$ or shorter by the indices $i_1 \ldots i_N$, and can be noted $Path_{i1 \ldots iN}(ct_0)$ and $Loop_{i1 \ldots iN}(ct_0)$.

c) The subset of all cell-tuples that can be reached from a given cell-tuple ct_0 by any combination of transitions $\alpha_{i_1} \ldots \alpha_{i_N}$ is called an *orbit* and is noted $Orbit_{i1 \ldots iN}(ct_0)$. To avoid ambiguities, sometimes the dimension of the G-Map is also noted: $Orbit^d_{i1 \ldots iN}(ct0)$.

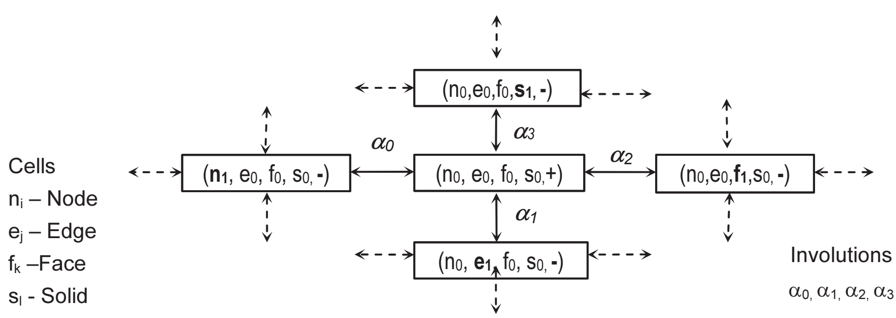


Fig. 2: Graph representation of an oriented 3-G-Map $G\,(D, \alpha_0, \ldots, \alpha_d)$ with cell-tuples as darts and α_i transitions.

Certain orbits describe loops of fixed or variable length, certain loops are defined by orbits, but in some cases the cell-tuples of an orbit may not be arranged in a continuuos loop (LÉVY 1999).

Transformations of d-G-Maps. There are two classes of operations: the *Euler operations* (MÄNTYLÄ 1988) that preserve the overall topological properties of the cellular complex described by the *Euler characteristic* $X = no.\ of\ nodes - no.\ of\ edges + no.\ of\ faces$ [$-$ *no. of solids*], and the *non-Euler* operations. The basic topological operations consist in the division of k-dimensional cells ($k > 0$) by the insertion of $k-1$-dimensional boundary cells, the dual operations – duplication of k-dimensional cells ($k < d$) by insertion of $k+1$-dimensional coboundary cells, and the inverse merging (resp. collapsing) operations by the deletion of a boundary (coboundary) cell (cf. Fig. 3) (THOMSEN & BREUNIG 2007). Note that if the boundary or coboundary cell to be removed is part of the outer boundary of a cellular complex, the deletion operation is not always admissible.

Basic non-Euler Operations like the creation or destruction of an isolated cell or of a connected component, or the "sewing" of two hitherto disconnected components into one, and the inverse operation, affect the overall topological properties of the model. Both the Euler and non-Euler operations are implemented using relational database operations comprised in a transaction. In some of these procedures, paths, loops or orbits are used to identify a sequence of cell-tuples to be updated, e. g., for the splitting of a solid by the introduction of a new face along a closed loop on the inside of the boundary of the solid (cf. Fig. 3).

4 Comparison with Other Topological Models

Ordered topological models. LIENHARDT (1991) compares G-Maps with different "ordered topological models": e. g., the winged-edge (BAUMGART 1975), the radial-edge (WEILER 1988), the quad-edge (GUIBAS & STOLFI 1985) and the cell-tuple structures. He concludes that "order models are based on the same ideas, and … it is possible to show that these models are equivalent (with respect to dimension and orientability)". Cell-tuple structures are equivalent to G-Maps without boundaries. However, some of the other models permit non-manifold configurations, which must be explicitly excluded.

ISO 19107 and GML 3. QUAK & DE VRIES (2005) compare the winged-edge structure to the ISO 19107 model (OPEN GEOSPATIAL CONSORTIUM 2007) and conclude that "it is possible to losslessly map to the ISO19107 model and back", but propose to extend the ISO model to reduce cost.

A detailed comparison of the proposed approach to the modelling of topology and the ISO 19107 and GML 3 model is beyond the scope of this article, therefore we only make some preliminary observations: The ISO 19107 and GML 3 model support oriented 2D and 3D topological models, specifying topological complexes consisting of nodes, edges, faces and topological solids as primitives, and the corresponding directed primitives as building elements for 1D, 2D, and 3D topological complexes, as well as for boundaries, coboundaries, and for "topological collections", i. e., topological curves, surfaces, volumes. In accordance with the object-oriented approach, relation-

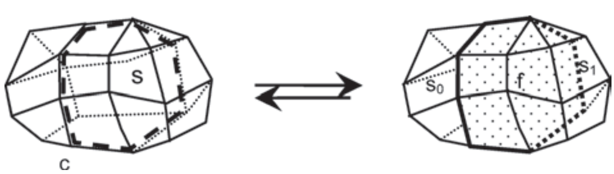


Fig. 3: A 3D Euler operation: splitting a solid *s* into solids s_0 and s_1 by the insertion of a 2D face *f*, and the inverse merge operation. The location of the contact between the face *f* and the boundary of the solid *s* is defined by the loop *c*.

ships between topological, geometrical and other entities are encoded with the topological objects, whereas the G-Map and the Cell-Tuple Structure use a separate system of entities and 1:1 relationships, namely darts/cell-tuples and involutions/switches to represent the topological structure. Thus the incidence, adjacency and order relationships in ISO 19107 have to be translated into sets of darts/cell-tuples, involution transitions and orbits.

The ISO 19107 model is less restrictive than G-Maps and Cell-Tuple Structures, and in consequence, the range of topological configurations that can be represented is larger, admitting, e. g., dangling edges, inner loops, curves that meet the interior of a face etc, which are precluded in G-Maps. On the other hand, topological operations on the ISO 19107 model may be more complicated, as numerous special cases have to be handled or excluded. Provided that the strict requirements of the G-Map are met, we expect no fundamental problems when translating the topological primitives from GML 3 to the corresponding cells in the G-Map model, and of topological complexes into cellular complexes represented by G-Maps and vice versa.

5 Object-relational Database Implementation

Implementation in transient storage. The in-memory-implementation of the graph-representation of a d-G-Map is straightforward (Lévy 1999): for each dart object, the α_j transitions are implemented by $d+1$ references to other darts, additional references to geometric objects and to thematic properties realise the geometric representation, and a set of flags is used to mark darts that have been traversed, the next α_i transition to take etc. The α_j references may be extended into objects as well, with methods to check the symmetry of the reference, and for permitting/barring the use of a given transition (Fradin et al. 2002). In order to support the navigation on the G-Map, for each cell of dimension from *0* to *d* a reference to a *starting dart* may serve as an "entry point" into an orbit describing the topological relationships within the cell and outside.

Object-Relational database implementation. For a persistent implementation based on an object- relational DBMS, the cell-tuple structure is more appropriate (cf. Fig. 4). Instead of references to locations, the transitions between cell-tuples are controlled by keyed access using one of three *search patterns* (cf. Fig. 5). In a G-Map without boundary, there is always exactly one corresponding cell-tuple to be retrieved.

Here the tuple (*node, edge, face, cell, solid*) acts as a key, which is augmented by the *sign*, if positive and negative cell-tuples are stored in the same table. For example, the following SQL query, for the cell-tuple $ct(c0, c1, \mathbf{c2}, c3, \sigma, \ldots)$ retrieves the cell-tuple $ct'(c0, c1, \mathbf{c2'}, c3, -\sigma, \ldots)$ corresponding to ct by an α_2-transition, i. e., by an exchange of faces:

SELECT * FROM celltuples WHERE node = c0 AND edge = c1 AND solid = c3 AND sign != σ;

Using hierarchical indexing of the cell-tuple relation, we expect the access time for a single transition to grow like $O(\log N)$,

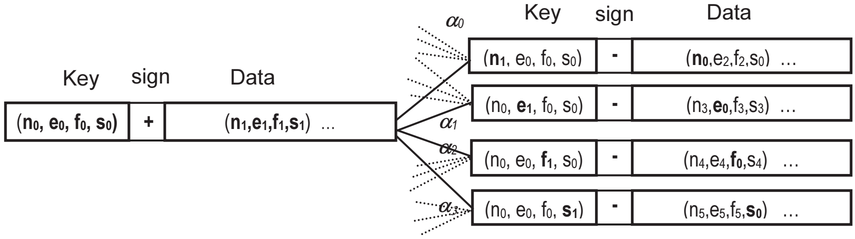


Fig. 4: An oriented 3-G-Map as a pair of relations $(c_0, \ldots, \boldsymbol{c_i}, \ldots c_d, +) \leftrightarrow (c_0, \ldots, \boldsymbol{c_i'}, \ldots c_d, -)$.

$$
\begin{array}{ccc}
\text{starting cell-tuple} & \text{search pattern} & \text{retrieved cell-tuple} \\
\boldsymbol{ct_0} & & \boldsymbol{\alpha_j(ct_0)} \\
(c_0,\ldots,\boldsymbol{c_j},\ldots,c_d,sign,\ldots) \rightarrow & \left\{\begin{array}{l} (1)\ \text{“}c_0,\ldots,*,\ldots,c_d,-sign\text{”} \\ (2)\ \text{“}c_0,\ldots,\neg\boldsymbol{c_j},\ldots,c_d\text{”} \\ (3)\ \text{“}c_0,\ldots,\boldsymbol{c_j'},\ldots,c_d\text{”} \end{array}\right\} & \leftrightarrow (c_0,\ldots,\boldsymbol{c_j'},\ldots,c_d,-sign,\ldots)
\end{array}
$$

Fig. 5: Search patterns for α_j transitions. The symbols "¬" and "*" denote "not" and "any".

where N is the total number of cell-tuples. Using hash indexes, even better behaviour can be achieved. This access method, at the cost of some overhead, makes the transition between cell-tuples independent from any particular storage or object identifier details and thus can be used transparently both with persistent storage in a client-server configuration, and in transient storage and easily cope with update operations.

Orbits and loops vs. ordered subsets. In a cellular partition of a d-dimensional manifold, spatial objects are represented by collections of cells, i. e., subsets of the underlying sets of k-cells, $k = 0, \ldots, d$. Therefore, we can always deduce corresponding queries on the associated cell-tuples. Information on the connectivity of the resulting subsets of cell-tuples, however, requires methods that systematically explore the adjacency and incidence relationships between cells, represented by the α_i-transitions, in particular orbits and loops.

Whereas the implementation of involutions and orbits in transient storage is simple, it is more intricate in the context of the relational model. In an RDBMS, the retrieval of subsets by conditions imposed on attribute values is well supported, as are basic sorting operations on resulting subsets. Using appropriate indexing, also a small finite number of links between relations by foreign keys or by joins poses no particular problem, even if some overhead is involved. Orbits, paths, and loops, however, may involve an undetermined and potentially large number of links between cell-tuples, possibly defined by a recursive formula. Although recursion is supported by ANSI SQL, it is not yet implemented in all widespread relational DBMS (PostgreSQL .org 2006) – or is not implemented in the same way. Without SQL recursion, the use

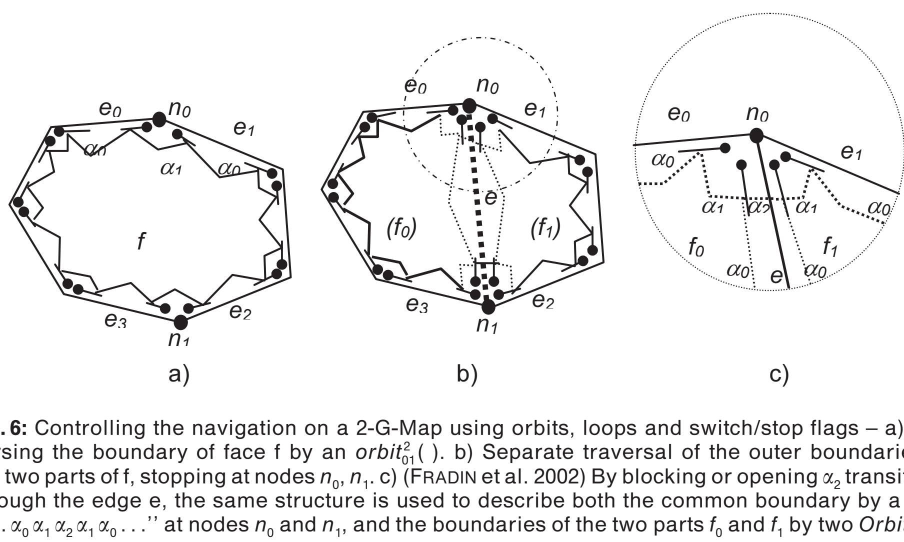


Fig. 6: Controlling the navigation on a 2-G-Map using orbits, loops and switch/stop flags – a) Traversing the boundary of face f by an $orbit^2_{01}(\ )$. b) Separate traversal of the outer boundaries of the two parts of f, stopping at nodes n_0, n_1. c) (Fradin et al. 2002) By blocking or opening α_2 transitions through the edge e, the same structure is used to describe both the common boundary by a loop "$\ldots \alpha_0\,\alpha_1\,\alpha_2\,\alpha_1\,\alpha_0 \ldots$" at nodes n_0 and n_1, and the boundaries of the two parts f_0 and f_1 by two $Orbit^2_{01}(\ )$ loops.

of a programming language is required for coding the loop or the recursive procedure, which in turn issues an SQL command for each step. Even if we reduce the overhead incurred using optimisation methods like prepared statements, this remains a clumsy way to solve a simple task. In order to handle this problem, we first have to find out which topological operations can be implemented using simple (ordered) subset retrievals, and for which operations paths, loops or orbits are essential.

Loops and orbits are also necessary for the implementation of division and merge operations (cf. Fig. 6). In order to ensure that the result f of a merging operation on two d-cells f_0 and f_1 is a d-cell, we must ensure that the boundary $e = f_0 \cap f_1$ is connected – otherwise situations like (cf. Fig. 7 c, d) may occur. If e is simply connected, we may first merge it into a single edge e', which then in turn is removed. The inverse procedure, namely the division of a d-cell, also involves the use of an orbit. Within a transaction, we proceed as follows:

1. Mark the two nodes n_0 and n_1 where the dividing edge joins the boundary of c.
2. Insert the dividing edge e into the table of edges.
3. Insert two new faces f_0 and f_1 into the face table.
4. Insert four cell-tuples $(n_0, e, f_0, -s)$, $(n_0, e, \mathrm{f}_1, +s)$, $(n_1, e, f_1, -s)$, $(n_1, e, f_0, +s)$. The sign s being chosen to match the signs of the existing cell-tuples.
5. Update the existing cell-tuples (n_0, e_0, f, s), $(n_0, e_1, f, -s)$, (n_1, e_2, f, s), $(n_1, e_3, f, -s)$ using two new face identifiers, resulting in (n_0, e_0, f_0, s), $(n_0, e_1, f_1, -s)$, (n_1, e_2, f_1, s), $(n_1, e_3, f_0, -s)$.

Note that from step 4 onward until step 7 below, the model is temporarily inconsistent!

6. Update all remaining cell-tuples on the boundary of f_0 such that f is replaced by f_0.
7. Update all remaining cell-tuples on the boundary of f_1 such that f is replaced by f_1.

Steps 6. and 7. require that we determine the two sides of the former boundary of face f before they are explicitly marked by f_0 and f_1. This is not possible by a subset query, but it can be done using two orbits starting, e. g., at cell-tuples $ct_0(n_0, e, f_0, -s)$ and $ct_1(n_0, e, f_1, +s)$, if the alpha-transitions are stored explicitly, or by using two paths starting at ct_0 and ct_1, and stopping, as soon as $ct_2(n_1, e, f_0, +s)$ and $ct_3(n_1, e, f_1, -s)$ are encountered.

Considering the division of a solid s by a newly introduced face f in a 3-GMap (cf. Fig. 3), we note that a closed loop is required to define the *seam* c where the dividing face f meets the boundary of s. For the merging of two neighbouring solids s_1 and s_2, we require their common boundary b to be 2-dimensional and simply connected.

The given examples show that some queries and operations require loops or orbits. Therefore we propose the following 2-step approach to the implementation of topological queries on a RDBMS-based d-GMap: first, retrieve an appropriate subset of the cell-tuples by standard RDBMS methods, and second, generate orbits, loops or paths in transient memory whenever necessary.

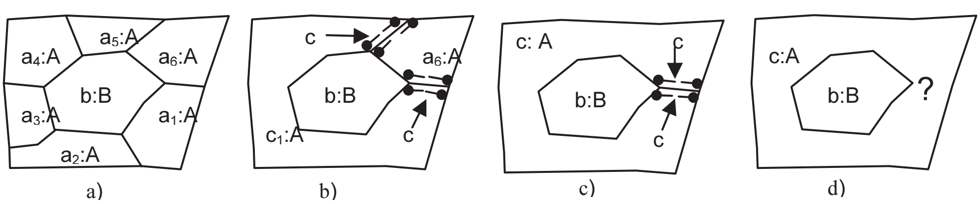


Fig. 7: Problems encountered during aggregation. (a) A face b of class B is completely surrounded by faces a_i of a different class A. (b) Stepwise merging all cells of class A results in a bridge configuration (c) and finally in a ring (d).

6 Aggregation Methods for G-Maps and Cell-Tuple Structures

Hierarchical G-Maps. LEVY (1999) proposes to use hierarchical structures for the construction of complex geological subsurface models, both in volume and in boundary representation. These *hierarchical G-Maps* are obtained in the following way: We start with a coarse 3D G-Map. By subdividing its 3D cells (solids), a structure is obtained that consists of *frames* – the coarse cells, and finer G-Maps that fit into the frames. Whereas the darts of the subdivisions possess a geometrical representation in 3D space, the darts of the coarse frame G-Map have no separate embedding. Instead, *embedding by delegation* is used: its darts are associated with a subset of the darts of the fine G-Maps, and use the geometric representation of the subdivision as embedding. Applying a similar approach to the 2D faces of the coarse model, a boundary representation of the coarse model is obtained. Whereas the construction of the topological hierarchy by subdivision proceeds top-down, the delegated geometrical representation is propagated bottom up.

Multiple grouping. For the topological modelling of buildings, FRADIN et al. (2002) use *multiple groupings*: for each grouping of cells, $d+1$ flags are associated with the references representing α_i-transitions, that indicate whether the given cell boundary may be ignored during navigation on the G-Map (cf. Fig. 6c). This approach is economic in memory space, as the same fine G-Map, augmented by the space required for the additional flags, is used for several groupings at the same time, and higher-level G-Maps do not require a separate representation. On the other hand, the possibility of reducing processing time for coarser and hence smaller representations is lost. Note that both methods support lower resolution models obtained by aggregation of cells, but not by simplification of boundaries or by displacement.

Classification tables. It is possible to translate hierarchical G-Maps into Cell-Tuple Structures by representing references using foreign keys. Multiple grouping however, is tightly associated with the in-memory implementation of α_i -transitions, orbits and loops using references. The explicit representation of the cells within the cell-tuples leads to a different approach: starting from a high resolution model, successive lower levels of detail are obtained by aggregation as follows. A classification of the d-dimensional cells at high resolution is represented by a N:1-relation that to each cell a associates its class A. In a copy of the original cell-tuples, the d-cells are replaced by their class identifiers. Then all pairs of cell-tuples $(c_0, c_1, \ldots, A, +)$, $(c_0, c_1, \ldots, A, -)$ – the fixed points of the α_d-transitions – are removed, while all pairs $(c_0, c_1, \ldots, A, +)$, $(c_0, c_1, \ldots, B, -)$ with $B \neq A$ are kept.

Separate embeddings at different levels of detail. If generalisation is restricted to aggregation, we can apply embedding by delegation, representing each aggregated cell by a collection of finer cells. As the new Cell-Tuple structure comprises a copy of the old one, however, it is also possible to create a different geometrical embedding, and apply

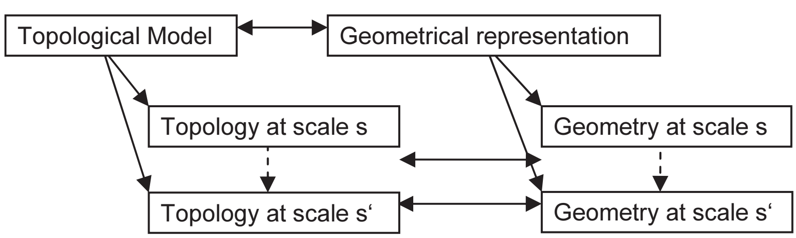


Fig. 8: Topological and geometrical changes must stay consistent during generalisation.

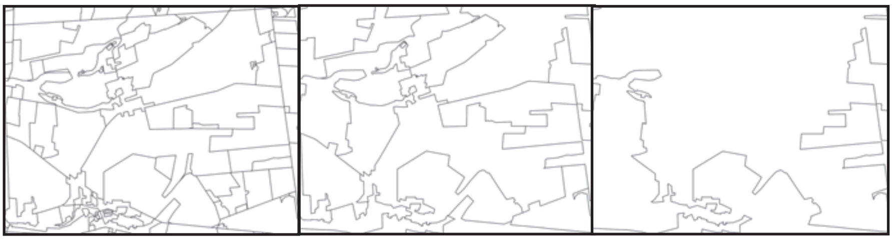


Fig. 9: Application example: a section of ca. 2% of a digital map on land-use at three different scales (by courtesy of J. HAUNERT, IKG Hannover University).

simplification and displacement to it. While this approach is more expensive in storage space, it is more flexible than the previous ones, and navigation on the smaller aggregated representations may be considerably faster. Here, however, a consistency problem arises: if we apply generalisation methods separately to the topological model and its geometrical embedding, then we must take care, that the resulting generalised models are consistent with each other (cf. Fig. 8). This can of course be achieved by having either geometrical operations control the changes in topology or vice versa. However, this depends on the particular application.

7 Applications of the Approach

Topology of a multi-resolution map of land-use. In an ongoing study, the methods presented here are applied to model the topology of a land-use map at three different scales, namely 1: 50.000, 1: 250.000, and 1: 1.000.000. The land use maps were provided by JAN HAUNERT and MONIKA SESTER, IKG, Leibniz University Hannover as Arc GIS shape files. The smaller scale maps were produced by aggregation starting from scale 1: 50.000, according to tables defining classes of similar land use. We first construct the topology for the largest scale map and then use the class tables to control the process of aggregation of cells for the 1: 250.000 and 1: 1.000.000 maps. During this process, the construction of inconsistent cells with holes, or of disconnected entities has to be avoided. Because no displacements occurred during the geometrical generalisation, we can apply the classification table method outlined above, to generate a distinct 2-G-Map at each Level of Detail, and provide links between corresponding cells and cell-tuples using the classification table and the vertex locations.

In another ongoing study, we use the Cell-Tuple structure to study the integration of a 3D city model and building plans with a digital 2D cadastral map (THOMSEN et al. 2008).

8 Conclusions and Outlook

The oriented d-G-Map and the d-Cell-Tuple structure can be employed with an ORDBMS to yield a simple, flexible, and scalable representation of the topology of spatial models based on cellular partitions. After analysis of the topological operations to be used, it can be used for the management of topology in a multi-representation database, in particular for the integration of the topology of models of different dimension and scale. We are currently extending our d-G-Map implementation into a topological access tool for the ORDBMS-based GIS PostGis (POSTGIS.ORG 2006), and for our OODBMS-based spatio-temporal database db3d, and plan to extend the approach to a time-dependent topology model.

Acknowledgements

We thank the Deutsche Forschungsgemeinschaft (DFG) for supporting this work within the DFG joint project "Abstraction of Geoinformation", grant no. BR 2128/6-1.

References

BAUMGART, B., 1975: A polyhedron representation for computer vision. – American Federation of Information Processing Societies (AFIPS) Nat. Conf. **44:** 589–596.

BRISSON, E., 1993: Representing Geometric Structures in d Dimensions: Topology and Order. – Discrete & Computational Geometry **9:** 387–426.

BUTWILOWSKI, E., 2007: Topologische Fragestellungen bei der Kombination von 3D-Stadtmodellen mit 2D-Karten in einer räumlichen Datenbank – Diploma thesis, University of Osnabrück: 116p.

CLEMENTINI, E. & DI FELICE, P., 1994: A comparison of methods for representing topological relationships. – Information Sciences **80:** 1–30.

EGENHOFER, M. J., 1989: A formal definition of binary topological relationships. – Int. Conf. Foundations of Data Organisation and Algorithms (FODO) 1989, Lecture Notes in Computer Science 367, Springer, Berlin: 457–472.

EGENHOFER, M.J., FRANK, A. U. & JACKSON, J. P., 1989: A topological data model for spatial databases. – 1st Symposium on Large Spatial Databases, Lecture Notes in Computer Science 409, Springer: 271–286.

FRADIN, D., MENEVEAUX, D. & LIENHARDT P., 2002: Partition de l'espace et hiérarchie de cartes généralisées. – 15e Journées de l'Association Française d'Informatique Graphique (AFIG): 12p.

GUIBAS, L. & STOLFI, J., 1985: Primitives for the Manipulation of General Subdivisions and the Computation of Voronoi Diagrams. – ACM Transactions on Graphics **4** (2): 74–123.

GRÖGER, G. & PLÜMER, L., 2005: How to Get 3-D for the Price of 2-D-Topology and Consistency of 3-D Urban GIS. – Geoinformatica **9** (2): 139–158.

HAUNERT, J.-H. & SESTER, M., 2005: Propagating updates between linked datasets of different scales. – XXII Int. Cartographic Conference.

HOPPE, H., 1996: Progressive meshes. – 23rd Int. Conf. on Computer Graphics and Interactive Techniques (ACM SIGGRAPH): 99–108.

KAZAR, B.M., KOTHURI, R., VAN OOSTEROM, P. & RIVADA, S., 2008: On Valid and Invalid Three-Dimensional Geometries. – Lecture Notes in Geoinformation and Cartography (LNG&C), Springer: 19–46.

LÉVY, B., 1999: Topologie Algorithmique – Combinatoire et Plongement. – PhD Thesis, Nancy: 202p.

LIENHARDT, P., 1991: Topological models for boundary representation: a comparison with n-dimensional generalized maps. – Computer-Aided Design **23** (1): 59–82.

LIENHARDT, P., 1994: N-dimensional generalized combinatorial maps and cellular quasi-manifolds. – Int. Journal of Computational Geometry and applications **4** (3): 275–324.

MALLET, J. L., 2002: Geomodelling. – Oxford University Press: 599 p.

MÄNTYLÄ M., 1988: An Introduction to Solid Modelling. – Computer Science Press: 401 p.

MESGARI, M. S., 2000: Topological Cell-Tuple Structures for Three-Dimensional Spatial Data. – ITC Dissertation 74, ITC, Enschede: 200 p.

MEINE, H. & KÖTHE, U., 2005: The GeoMap: A Unified Representation for Topology and Geometry. – 5th Int. Workshop on Graph-based Representations in Pattern Recognition 2005. Lecture Notes in Computer Science 3434, Springer, Berlin: 132–141.

MOKA, 2006: Modeleur de Cartes. – www.sic.sp2mi.univ-poitiers.fr/moka/ (accessed 21. 03. 2007).

OPEN GEOSPATIAL CONSORTIUM, 2007: OpenGIS®R Geography Markup Language (GML) Encoding Standard. – OpenGIS®R Standard OGC 07-036, Version: 3.2.1, editor: Clemens Portele. www.opengeospatial.org/standards/gml (accessed: 11. 3. 2008)

QUAK, C.W. & DE VRIES, M. E., 2005: Topological and temporal modelling in GML. – Topology and spatial databases workshop: 1–8.

PIGOT, S., 1992: A topological model for a 3D spatial information system. – 5th Symposium on Spatial Data handling (SDH): 344–360.

POSTGIS.ORG, 2006: PostGIS documentation. – postgis.refractions.net/documentation (accessed 21. 03. 2007).

POSTGRESQL. ORG, 2006: PostgreSQL 8.3.0 Documentation. – www.postgresql.org/docs (accessed 21. 03. 2007).

RAZA A. & KAINZ, W., 1999: An Object-Oriented Approach for Modelling Urban Land-Use Changes. – Urban and Regional Information Systems Association (URISA) Journal **14** N° 1: 37–55.

SHUMILOV, S., THOMSEN, A., CREMERS, A.B. & KOOS B., 2002: Management and visualisation of large, complex and time-dependent 3D objects in distributed GIS. – 10th International Symposium on Advances in Geographic Information Systems (ACM-GIS): 113–118.

THOMSEN, A. & BREUNIG, M., 2007: Some remarks on topological abstraction in multi representation databases. – Int. Workshop on Information Fusion and Geographical Information Systems (IF&GIS'07). Lecture Notes in Geoinformation and Cartography (LNG&C), Springer: 234–251.

Thomsen, A., Breunig, M., Butwilowski, E. & Broscheit, B., 2008: Modelling and Managing Topology in 3D Geoinformation Systems. – Advances in 3D Geoinformation Systems. Lecture Notes in Geoinformation and Cartography (LNG&C), Springer: 229–246.

Weiler, K., 1988: The radial edge structure: a topological representation for non-manifold geometric boundary modelling. – In: GeometricModeling for CAD Applications, Elsevier: 3–36.

Address of the Authors:

Dipl.-Math. Andreas Thomsen, Prof. Dr. rer. nat. Martin Breunig, Dipl.-Geogr. Edgar Butwilowski, Universität Osnabrück, Institut für Geoinformatik und Fernerkundung, D-49069 Osnabrück, Tel.: +49-541-969-4064, Fax: +49-541-969-4061, e-mail: andthoms | mbreunig | ebut wilo@uos.de

Manuskript eingereicht: Dezember 2007
Angenommen: März 2008