DUAL HALF EDGE DATA STRUCTURE IN DATABASE FOR BIG DATA IN GIS

M. Goudarzi [a] , M. Asghari [a], P. Boguslawski [b], A. Abdul Rahman [a,c]

[a] Faculty of Geoinformation and Real Estate, University Teknologi Malaysia - (m.j.t.goudarzi, ma.asghari)@gmail.com
[b] Faculty of Environment and Technology, University of the West of England - pawel.boguslawski@uwe.ac.uk
[c] alias@utm.my

KEY WORDS: Data Structures, DBMS, GIS

ABSTRACT:

In GIS, different types of data structures have been proposed in order to represent 3D models and examining the relationship between spatial objects. The Dual Half-Edge (DHE) is a data structure that permits the simultaneous representation of the geometry and topology of models with a special focus on building interiors. In this paper, from the storage cost point of view, the G-Maps model is analyzed and compared with the DHE model, since they have some features in common and also G-Maps is used widely in GIS. The primary result shows that the DHE is more efficient than the G-Maps with regard to the storage cost.

1 INTRODUCTION

Nowadays, in the GIS field, 3D models are used more frequently in order to meet requirements of advanced applications such as model construction, representation, and analysis (Boguslawski, 2011). In many cases the geometry and topology need to be stored together (Van Oosterom *et al.*, 2002). 3D models in CAD, for example, represent geometrical data and do not manage the topology. In Building Information Modeling systems (BIM), models include topological information but spatial relations among objects are not explicitly stored. Topological models are essential for finding adjacent features, coincident boundaries and navigating through the model. Relational DBMSs are able to store big topological data, but do not cover all aspects such as large data storage, data updating, querying and analyzing are only partially supported. With having DHE implemented in database, the model benefits from storing and retrieving data at any time. This expands the applications of the model such as multi-access to data and work on a part of a big model while the rest of the model is stored in the database. DHE in database (DHE-DB) makes analyzing and querying big models easy for users. It is important that how topological data is integrated with geometrical data in order to be used on application or DBMS side. One of the important issues in integrating geometry and topology is the complexity of operations.

Topological models are used to perform overlap, intersection and other topological operations (Penninga, 2004). Particularly, graph-based topological models overcome the issue of local neighborhood when the real geometry is not important (Lee and Zlatanova, 2008). In this paper, we discuss a new representation of topological data in a database environment using the duality concept.

The remaining sections of the paper as follows: Section 2 discusses a literature review of two data structures. Section 3 and Section 4 include DHE-DB architecture and the topology representation method, construction and modification operators. Section 5 summarizes the primary result achieved and highlights future works.

2 LITERATURE REVIEW

According to Oosterom et al. (2002) in order to implement topology in DBMS a full operational solution that covers all possible topological structure should be assessed. For this reason, the Dual Half-Edge (DHE) model considered as a case study, which fully supports topology through primal and dual graphs. To compare DHE data structure with models in DBMS the Generalised Maps (G-maps) is taken into account.

2.1 G-Maps

The d-Generalized Map (d-G-Map), is an abstract model of the topology of a d-dimensional cellular complex based on algebraic topology (Lienhardt, 1989). The d-G-Map is composed of darts and a set of applications ($\propto_0, \propto_1, \ldots, \propto_n$). A dart is a half-edge that constitutes an edge with connecting to another dart, Figure 1. In other words, in one dimensional space an edge consists of two darts, distinct half-edge, that gathered together by a "tie" $\propto_0$. The "tie" $\propto_1$represents the connection of two edges to a same vertex, Figure 2. Two connected faces, and two connected closed 3-dimensional objects are defined by $\propto_2$ and $\propto_3$ respectively, Figure 3 and Figure 4.

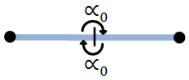

Figure 1. Example of two darts tied together and formed an edge, $\propto_0$ is a subdivision of n-dimensional space and represents darts in a model.

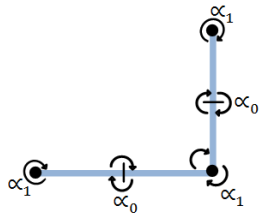

Figure 2. Example of two edges are tied together. $\propto_1$ is a set of distinct edges that share same vertex.

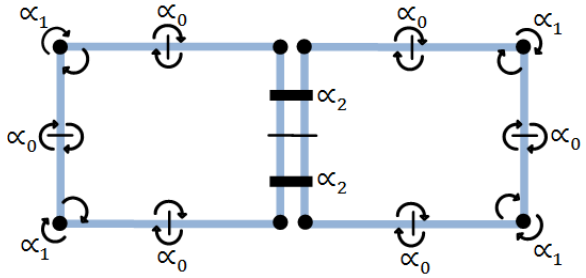

Figure 3. Example of gathering two faces in 2D model by $\propto_2$.

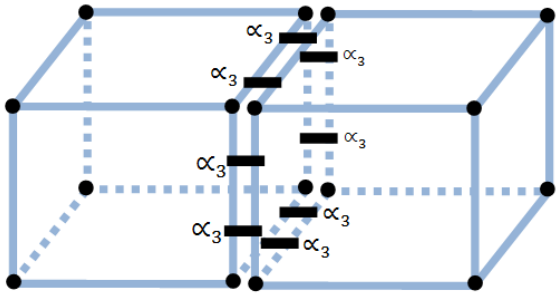

Figure 4. Example of gathering two cubes along their faces by $\propto_3$.

A possible representation of a d-G-Map is a d-celltuple structure. The main shortcoming of G-Maps is their extreme verbosity. The problem of representing a general finite topological space by some data structure is known to be in O(n^2) and this complexity bound is optimal. The optimal complexity O(n^2) is exceeded only by the number of darts in G-Maps: the storage-space complexity is not efficient (Bradley and Paul, 2014). It is worse when the storage of the involutions is taken into account. Thus, storing darts and involutions of a G-map can become very costly if no limitation is imposed on dimension (Bradley and Paul, 2014). An involution determines a subdivision of a set. For example, consider the set $\propto_0$({1,2},{3,4}), regarding to the involution concept the involution of $\propto_0$(1) = 2, and in reverse the involution of $\propto_0$(2) = 1. Using involution, the path from one point to a destination and then back to the original path can be achieved. For example, consider the path (∂, c, A), Figure 5, which is correspondent to the set $(\propto_0, \propto_1, \propto_2)$. It starts with vertex ∂, steps up to the incident edge c and ends with the incident area A.

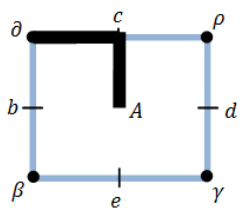

Figure 5. Illustrates a path, (∂, c, A), from a vertex to face A.

If there is a unique replacement for this path's elements to get the same sequence, then one could also fix a position i for all these paths and then every incidence path has at most one "bypass" at its i^{th} position. Thus, there is a function $\propto_i$ which either sends an incidence path to such bypass at position i or leaves the path unchanged if that bypass does not exist. Applying $\propto_i$ again returns the original path. Such a function, where $\propto_i$ ($\propto_i$ (x)) = x holds for all elements x, is called an involution. If such involutions $\propto_i$ exist for every dimension number i ranging from zero to the dimension n of the given space, they generate permutation groups that uniquely determine the space and, hence, can also be used as a topological data model for arbitrary dimension. Permutation is the combinations of elements based on the order of elements, $(\propto_0, \propto_1, \propto_2, \ldots, \propto_n)$, that allows to find all possible path in the structure (Thomsen and Breunig, 2007; Bradley and Paul, 2014).

In spatial data, modelling the dimension of space particularly is much less than the number of stored elements. It means, not only the storage space is unnecessarily wasted, but also means that higher dimensional data may have many redundancies when stored as G-maps. G-maps particularly is designed for manifold data that makes storage of non-manifold topological data unnatural if not possible (Bradley and Paul, 2014).

2.2 Cell-Tuple

The cell-tuple and G-Map are very similar. Brisson (1989) introduced the notion of cell-tuple structure. The cell-tuple structure consist of a set of cell-tuples (node, edge, face, solid) attached to the corresponding spatial objects: the abstract nodes n, e, f, s belong to 4 classes distinguished by different dimensions, whereas all nodes of a simplicial complex belong to the same finite set of vertices in space (van Oosterom et al., 2008). The abstract nodes n, e, f, s, corresponds to different dimensional cell. Four types of primitive entities can be classified according to their underlying dimensions: A vertex corresponds to a 0-cell, an edge corresponds to a 1-cell, a face corresponds to a 2-cell, and a region corresponds to a 3-cell, Figure 6.

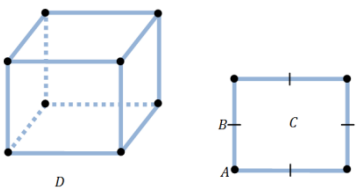

Figure 6. Example of corresponding entities to dimensional cell: A(node) = 0-cell, B(edge) = 1-cell, C(face) = 2-cell, and D(solid) = 3-cell.

A cell-tuple corresponds to a path in the incidence graph (Timpf and Laube, 2012). Graph theory can be used to construct a graph, where vertices represent cells and edges represent incidence relationships between cells. Such a graph is called an incidence graph (Le *et al.*, 2013).

In the cell-tuple, involutions are implemented in two steps: first, from a given cell-tuple entry, create a new cell-tuple key by exchanging exactly one cell_id. Second, use this key to retrieve the corresponding complete entry from the database. The involution operators define the neighbourhood relationships between the abstract simplexes (van Oosterom et al., 2008).

The two main limitations of the Brisson's cell-tuple is that it can only represent a very regular class of structures and that the test for membership in this class is not decidable in higher dimensions (Cardoze, et al., 2006). Thus, the G-Maps and cell-tuple share same problems: both are limited to manifold objects (Lienhardt 1991; Rezaei Mahdiraji et al. 2013). The cell-tuple representation with combination of involutions as shown in Figure 7.

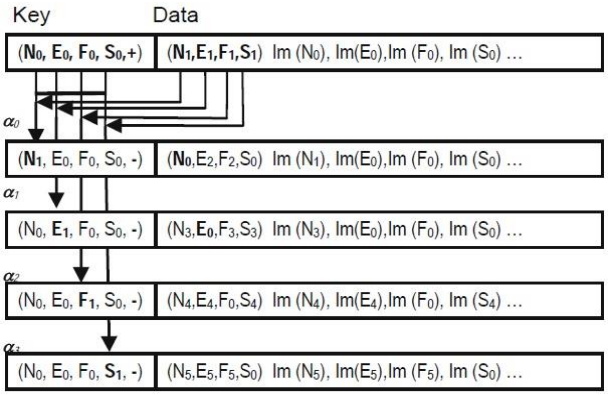

Figure 7. Cell-tuple representation with combination of involutions (Van Oosterom et al., 2008).

One another drawback of this data structure draws attentions when it comes to be applied in database. The implementation of Cell-Tuple without restriction on dimension aspect would be very costly in terms of storage space. There are five primitives Cell-Tuple, node, edge, face, and solid that must be considered in database. Furthermore, objects such as polyhedron, polygon, arc, and vertex are extended from corresponding primitives. This database model, Figure 8, is only for 3D modelling. However, the further dimension requires extra objects and consequently more space.

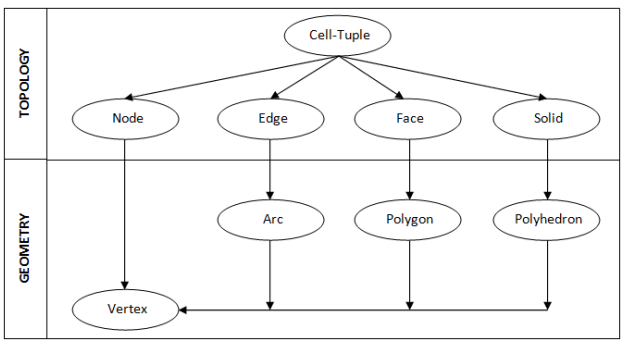

Figure 8. Cell-Tuple data structure's model in database.

2.3 Dual Half-Edge

The Dual Half-Edge (DHE) (Boguslawski, 2011) is related to the quad-edge (Guibas and Stolfi, 1985), facet-edge (Dobkin and Laszlo, 1987), radial-edge (Weiler, 1988) and half-edge (Mäntylä, 1988) data structures. Entities present in this data structure are: cells, faces, edges, and vertices. A cell is bounded by faces that form a closed shell. Faces are convex or concave polygons – there is no restriction on triangle faces, which applies to many GIS applications. There are two spaces (two structures): the primal and the dual, and they are represented as graphs. Each entity (volume, face, edge, and vertex) in one space is matched by another entity in the dual space. It conforms to the 3D Poincaré duality rules (Boguslawski, 2011). For 3D model construction in DHE, for example a 3D model of the building interiors for emergency management system, the geometry is considered for visualization and the topology to represent connections between rooms inside the building. The model stores them simultaneously in primal and dual graphs. Another important issue is the ability of assigning attributes to the object and storing semantic.

3 RESEARCH METHODOLOGY

3D city modelling is abounded with unsolved challenges such as processing big 3D models, analyzing the model, and many more. Most of 3D models represent only exterior which allows 3D visualisation but, as a matter of fact, they are 2.5D models. However, interior modelling is important from disaster or emergency management systems' viewpoint.

The Dual Half-Edge data structure has been considered as a base for our implementation which includes 3D geometry, topology and semantic information, Figure 9. Requirements for 3D models, functionality of topological operators, and lack of support for external data structures, such as the DHE, provided by Oracle Spatial and PostGIS make these platforms unsuitable for our research. Therefore, we decided to develop operators for 3D model construction and querying, such as finding intersections of polygons or cross section of a model.

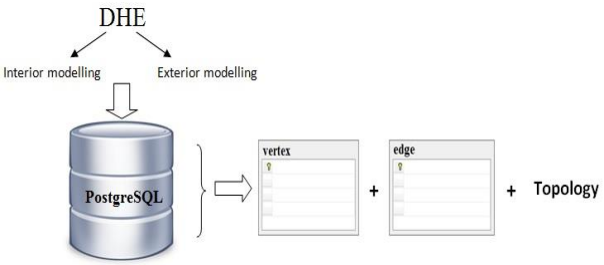

Figure 9. The structure of the proposed model.

4 MANAGING TOPOLOGY IN DBMS: DHE-DB

To develop the DHE-DB it was necessary to explore what functionality should be implemented and where to locate an interface: in front-end application or in database. According to Zlatanova and Stoter (2006) since some of the operations such as spatial analysis, inserting new data and visualization might be very complex and time consuming therefore complex spatial functions should be implemented within the front-end applications to avoid overloading of the server while generic spatial functions should be implemented in DBMS. Meanwhile, having all functionality implemented within DBMS will prevent reimplementation of those functions in different applications and it would guarantee the consistency. Moreover, there would be no unnecessary data transfer between DBMS and front end application (Van Oosterom *et al.*, 2002). On the other hand, the solution must have the possibility of extending by users in order to develop applications without conflicting with the database. Therefore all operators were implemented within the database. Our model entities use standard data types thus any geometrical primitives such as line or polygon are not required to represent. The PstgreSQL9.3 is used for implementation phase.

4.1 Topology Representation

One of the key aspects of this research is managing topology in a database without having additional topological primitives. The

only primitives used in our model are vertex and edge, see Figure 10.

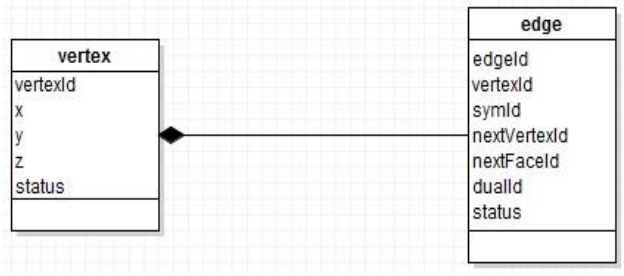

Figure 10. Tables representing two main primitives.

Other elements, such as faces or volumes, are retrievable through stored procedures, figure 11. This will decrease the cost of storage in our model since there is no need to have extra tables to store all topological primitives. The topological relations are all embedded in the DHE data structure and are retrievable with queries whenever they are needed. Edge attribute and vertex attribute are two entities of our model that attach labels to edges or vertices.

Figure 11. Examples of the stored procedures.

4.2 Operators Construction and Modification

For model construction, normally automatic bulk model construction is needed, and construction operators to modify or update the model. However, in the original DHE implementation the model is built by incremental addition of individual segments. Therefore, modified Euler operators were developed as a set of standard operators within the database - stored procedures (functions in PostgreSQL).

There are four levels of operators, from level 0 up to level 3 which is the highest level, see Table 1. Functions in DHE-DB, implements these operators that permit users to construct and modify a 3D model. Level 0, for example, create primal and dual elements and connect them together while the higher levels prepare and organize parameters for lower-level operators. In addition, navigation operators, such as Sym, Next_f, Next_v, provide users the navigation ability whereby the user can navigate from one part of the model to another part (Boguslawski, 2011). Users can communicate with these operators through the front-end interface.

For example, in order to create a simple cube, three high level operators: MEVVFS, MEV, and MEF should be used. These operators, stored procedures in DBMS, are called by the user through the GUI. Then, the corresponding lower level operators are called to fulfill the user command: function MEVVFS calls the function MakeComplexEdge. Afterwards, in the function MakeComplexEdge two lower level functions MakeEdge and Snap are called. Finally, the function MakeEdge calls the function MakeHalfEdge and function Connect HalfEdges. Likewise, the function Snap calls the function ConnectHalfEdges. In the Graphical User Interface, user only has access to the highest level of operators along with navigation operators.

Table 1. Construction operators in DHE-DB (Boguslawski, 2011).

Levels	Operators
L3	MEVVFS, MEVFFS, MEV, MEF, MZEV, MVE, Join/MergebyFace, Join/MergebyEdge, Join/MergebyVertex
L2	MakeComplexEdge, ComplexSplice, MakeFace, Sew
L1	MakeEdge, Snap, Splice, MakeDegenerateEdge
L0	MakeHalfEdge, ConnectHalfEdges, HalfSplice

5 CONCLUSION AND FUTURE WORK

Based on the literature the number of primary and secondary entities of each reviewed model as follows: DHE-DB includes five entities of vertex, edge, complex, edge attribute and vertex attribute: and cell-tuple model includes nine entities of cell-tuple, node, edge, face, solid, vertex, line, polygon and polyhedron.

Despite the fact that DHE-DB contains the lower number of entities in contrast with other comparable models, it is fully functional for both topological and geometrical queries.

We intend to add intersection and navigation operators for expanding the applications of our work. Thus, by using intersection operators, our model would be able to detect overlaps, gaps, and normal connection between two objects.

Acknowledgement
The authors wish to thank Ministry of Science, Technology and Innovation in Malaysia, for providing necessary financial assistance. The authors also like to thank the partial support from Universiti Teknologi Malaysia, to support us with eScience grant, vote no. 4S049, UTM. The authors would like to thank anonymous reviewers who gave valuable suggestion that has helped to improve the quality of the manuscript as well.

References

Boguslawski, P., 2011. Modelling and analysing 3d building interiors with the dual half-edge data structure, pp. University of Glamorgan.

Bradley, P. E., and N. Paul, 2014. Comparing G-maps with other topological data structures. Geoinformatica 18**:** 595-620.

Cardoze, D. E., G. L. Miller and T. Phillips, 2006. Representing topological structures using cell-chains, in *Geometric Modeling and Processing-GMP 2006*. Springer, pp. 248-266.

Dobkin, D. P., and M. J. Laszlo, 1987. Primitives for the manipulation of three-dimensional subdivisions, pp. 86-99 in *Proceedings of the third annual symposium on Computational geometry*. ACM.

Guibas, L., and J. Stolfi, 1985. Primitives for the manipulation of general subdivisions and the computation of Voronoi. ACM transactions on graphics (TOG) 4**:** 74-123.

Le, H. H., P. Gabriel, J. Gietzel and H. Schaeben, 2013. An object-relational spatio-temporal geoscience data model. Computers & Geosciences 57**:** 104-115.

Lee, J., and S. Zlatanova, 2008. A 3D data model and topological analyses for emergency response in urban areas. Geospatial information technology for emergency response 143**:** C168.

Lienhardt, P., 1989. Subdivisions of n-dimensional spaces and n-dimensional generalized maps, pp. 228-236 in *Proceedings of the fifth annual symposium on Computational geometry*. ACM.

Lienhardt, P., 1991. Topological models for boundary representation: a comparison with n-dimensional generalized maps. Computer-aided design 23**:** 59-82.

Mäntylä, M., 1988. An introduction to solid modeling.

Penninga, F., 2004. Oracle 10g Topology; Testing Oracle 10g Topology using cadastral data GISt Report No. 26, Delft, 2004, 48 p, pp.

Rezaei Mahdiraji, A., P. Baumann and G. Berti, 2013. Img-complex: graph data model for topology of unstructured meshes, pp. 1619-1624 in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM.

Thomsen, A., and M. Breunig, 2007. Some remarks on topological abstraction in multi representation databases, in *Information Fusion and Geographic Information Systems*. Springer, pp. 234-251.

Timpf, S., and P. Laube, 2012. *Advances in Spatial Data Handling: Geospatial Dynamics, Geosimulation and Exploratory Visualization*. Springer Science & Business Media.

Van Oosterom, P., J. Stoter, W. Quak and S. Zlatanova, 2002. The balance between geometry and topology, in *Advances in Spatial Data Handling*. Springer, pp. 209-224.

Van Oosterom, P. J. M., S. Zlatanova, F. Penninga and E. Fendel, 2008. *Advances in 3D geoinformation systems*. Springer.

Weiler, K., 1988. The radial edge structure: a topological representation for non-manifold geometric boundary modeling. Geometric modeling for CAD applications**:** 3-36.

Zlatanova, S., and J. Stoter, 2006. The role of DBMS in the new generation GIS architecture, in *Frontiers of geographic information technology*. Springer, pp. 155-180.