A Data Structure for Efficient Transmission of Generalised Vector Maps*

Min Zhou and Michela Bertolotto

Department of Computer Science, University College Dublin
Belfield, Dublin 4, Ireland
{min.zhou, michela.bertolotto}@ucd.ie

Abstract. Progressive transmission of a sequence of representations at increasing detail has been identified as a possible solution for the exchange of very large vector datasets across the Internet. In this context an important issue relates to the development of data structures to efficiently store and manipulate multiple representations. In this paper we describe a new data structure used to encode representations obtained by applying Saalfeld's modified RDP algorithm [10]. The data structure includes vertical links between different representations and therefore imposes a hierarchical organisation on the multiresolution sequence.

Keywords: Multiple representations, Progressive vector transmission, Line simplification

1 Introduction

Vector data are stored mathematically as sets of points, lines, and polygons (regions) in Geographic Information Systems (GIS). Vector data associated with location and other attribute information have been widely applied in the fields of digital mapping and location services. Spatial information sharing has become increasingly important with the new developments in communication technologies. However, since vector datasets are typically very large, users wishing to download these datasets from server to client have to suffer a long-time waiting even through faster communication links. Progressive and incremental transmission is proposed in [3-5], where a coarsest version is transmitted and displayed, then progressively substituted by subsequent finer versions until users' needs are satisfied. It ensures that a user receives no more data than desired. This approach relies on the pre-computation of a sequence of representations on the server.

In this paper, we describe a new data structure used to store different representations on the server and to support progressive transmission. The different representations are generalised by using a classical line simplification algorithm (RDP) [7, 9] improved by Saalfeld to guarantee topological consistency [10].

* The support of the Informatics Research Initiative of Enterprise Ireland is gratefully acknowledged.

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3039, pp. 948–955, 2004.

The main objectives we considered in the development of such a data structure are to:

(a) Include all vector objects in the form of points, lines and polygons
(b) Store the entire map (entities and their topological relations) only in the coarsest layer, while all subsequent layers just record newly introduced entities and increments of entities from previous layers
(c) Support hierarchical navigation, browsing and querying capabilities by maintaining vertical links between different representations of the same entities
(d) Be extendible to allow for encoding not only of geometric changes but also of topological changes

The remainder of this paper is organised as follows: In section 2 related work on progressive vector transmission is examined; section 3 describes in detail the proposed data structure for storing multiple representations and vertical links; in section 4 the implementation of the data structure used for progressive transmission in a client server architecture is described; section 5 presents some conclusions and outlines future work.

2 Related Work

Very few models have been developed for progressive vector transmission of geospatial data. So far the only implementations apply to triangular meshes used for digital terrain modelling [4]. For more generic data, two main models have been proposed in the literature. Buttenfield [5] proposes that the pre-computation of multiple representations resulted by line simplification can be performed on a server where each line is iteratively subdivided using Saalfeld's modified RDP algorithm and stored in a hierarchical strip tree. These multiple representations are transmitted progressively to clients. In the implementation, Buttenfield modified Ballard's strip tree data structure [1] to preserve the topological consistency by storing convex hulls of the subdivided lines instead of storing Minimum Bounding Rectangles of the subdivided lines. This data structure is a hierarchical tree structure, which facilitates to reconstruct linear features at a given level of detail and supports efficient progressive transmission. However, as this data structure stores the sub-polylines only, it lacks effective spatial indexing as the complete geometry of the polyline is not maintained. Spatial indexing is essential for optimizing the performance of spatial queries in a large datasets. Drawbacks of this model are: (a) only one line at a time can be simplified; (b) vertical links for efficient navigation between multiple representations are not supported; and (c) topological changes are not explicitly taken into account.

A model proposed by Bertolotto and Egenhofer [3, 4] structures a sequence of map representations at different levels of detail by applying a set of topological operators: line contraction, region contraction, region thinning, line merge, region merge, point abstraction and line abstraction [2]. The model represents a framework for progressive vector transmission in a distributed client-server architecture. This model is applicable not only to linear data but to all vector objects (i.e. points, lines and polygons). Vertical links are added to enhance the sequence of multiple

representations in a hierarchical structure. However, the model performs only topological changes on a vector map and vertical links are represented by topological operators only. Clearly, in real applications, changes in the geometry of objects also occur while generating multiple representations. For example, line simplification is a basic generalisation operation used in cartography.

In this paper we consider multiple representations generalised by line simplification operations. In order to efficiently store and transmit such representations, we have defined an appropriate data structure, which is able to manipulate multiple representations at different levels of detail organised in a hierarchical structure.

Several spatial data structures that provide some limited facilities for multiple levels of detail were described in van Oosterom's review [11]. However, none of the data structures presented in the review combines the geometric capabilities with multiple levels of detail. Furthermore they were not developed with the purpose of supporting progressive data transmission over the Internet.

3 The Proposed Data Structure

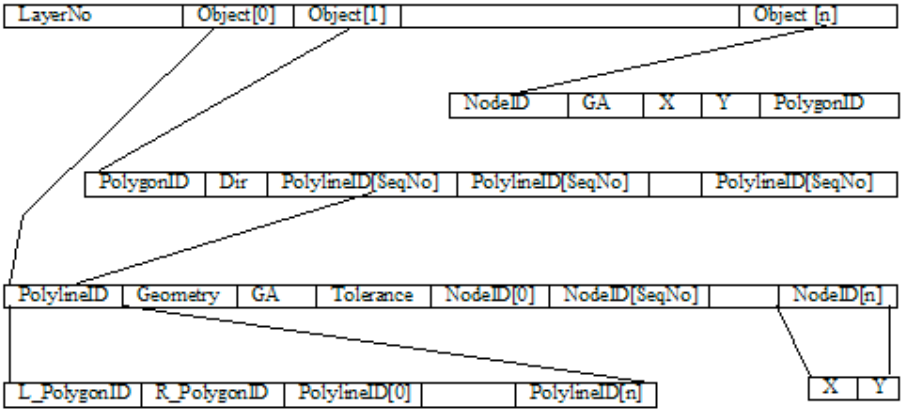
A new data structure, EXDCEL, is developed to implement a mechanism for dynamically interpreting vertical links and compose the maps transmitted incrementally. The increments downloaded progressively are then merged into a complete map.

Our data structure represents a hierarchical extension of the Doubly-Connected Edge List data structure (DCEL), a topological data structure for plane vector maps [8]. The version of DCEL used here is an extension of classical DCEL including information about isolated features entities [6]. Such a data structure stores:

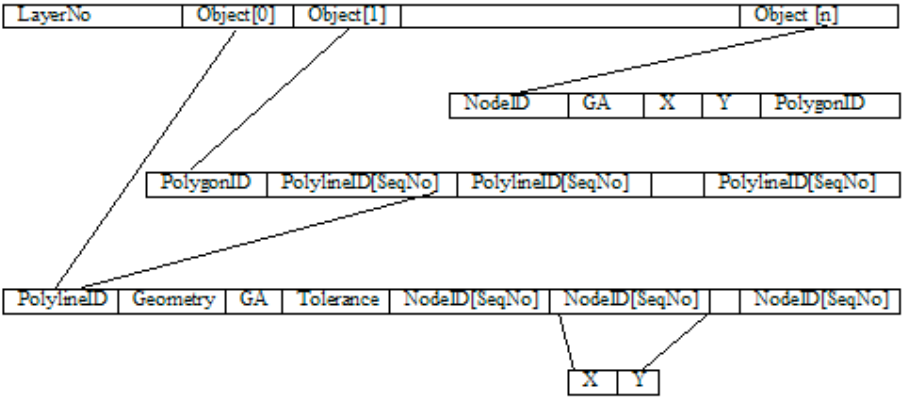
- For each point: its coordinates, a reference to an edge incident to it, and the region containing it if it is a point feature.
- For each line: its geometry, its endpoints, reference to the left and to the right polygon, and 4 lines sharing an endpoint with it.
- For each region: a list of references to edges forming the outer boundary and possible inner boundaries, a list of lines composing linear features, and a list of point features.

The following information is added to the above structure to form the EXDCEL data structure:

- An identifier is attached to each entity. The identifier for a layer is added. The points of a polyline and the polylines of a polygon are ordered by sequence number. Vertical links that consist of these identifiers and sequence number, enhance a sequence of multiple representations with a hierarchical structure.
- A tolerance value is a pre-determined threshold distance, which is used by RDP algorithm to control data reduction. It is added as an attribute of an entity. Multiple tolerance values are applied for sectors of a polyline or polylines to produce a nearly constant number of objects stored on layers.
- Graphical and semantic attributes can be associated with entities. These are used to classify an entity as a geographical feature such as a road, railroad, river, lake etc.



(A) EXDCEL data structure used for storing spatial entities and their spatial relations at the coarsest layer



(B) EXDCEL data structure in the case of storing increments and vertical links only at the subsequent layers.

Where
LayerNo: Layer Identifier. It points to the layer where a spatial object is allocated
Object: an array of objects on a layer. Object can be a point feature, polyline, polygon and line features of polygon.
NodeID: point feature Identifier
PolylineID: Polyline Identifier
PolygonID: Polygon Identifier
GA: Graphic Attributes such as line type, width, colour, etc. to clarify entities as roads , rivers
Tolerance: a parameter used by RDP algorithm to control the data reduction and is applicable to polyline and polygon
SeqNo: the sequence number of a node in its original polyline or the sequence number of a polyline in its original polygon
L_PolygonID: PolygonID on the left side of polyline
R_PolygonID: PolygonID on the right side of polyline
Dir: Direction such as 1 represents an outer polygon and 0 represents an inner polygon
X: X-coordinate of a point
Y: Y-coordinate of a point

Fig. 1. Examples of EXDCEL data structure

In order to avoid redundancies, we follow an approach similar to the one described in [3, 4] for storing a sequence of multiple representations. Only the coarsest layer is stored completely with endpoints of polylines and their topological relations. The subsequent layers store increments (new points of polylines) and vertical links only. Note that spatial relations need not be stored on the subsequent layers (i.e. only the coarsest layer is encoded by means of a DCEL data structure). Only endpoints of a polyline are necessary to completely reconstruct the topology of the polyline itself. Topological relations of polylines are not changed during refinement of lines by merging new points between the endpoints. Fig. 1 (A) shows the data structure used to encode the coarsest layer (entities and their spatial relations), while Fig. 1. (B) shows the data structure in the case of encoding the increments and vertical links only to transmit efficiently.

4 Implementation Details

We have implemented our progressive transmission system as a distributed 4-tier client-server architecture including a client, a map server, an application server and a web server, using the Java programming language. As Java is platform independent, our application can be easily run on any platform without any changes or recompilation required. Different clients are connected to the web server via Internet simultaneously. All communications between a client and the map server are conducted through the web server and the application server.

4.1 Map Server

The map server supports multiple representations of spatial datasets and relies on an Oracle9i Spatial Database Management System. Multiple representations at different levels of detail are pre-computed on the server side and stored in a hierarchical structure. Client applications provide a user interface for transmission requests and data display.

All information is stored in the database based on a topological data structure using relational tables, from which geometric entities are extracted as object views. Most of the spatial functionality offered by Oracle Spatial such as spatial indexing and spatial querying can be exploited using these geometric objects.

Even though the RDP simplification algorithm has become a default simplification routine in many GIS packages, the algorithm treats individual polylines as isolated features. Therefore it can generate self-intersections, intersections between polylines and other topological inconsistencies. Saalfeld indicated conflicts could only occur with vertices of other polylines that lie within the closed convex hull of the original polyline. He proposed a modified RDP algorithm by adding extra checks for the external points in a dynamic convex hull data structure, which can ensure that topological consistency is preserved [10]. In our system, the modified RDP algorithm was implemented in Java to simplify different polylines simultaneously. Multiple representations generalised by the algorithm using different tolerance values preserve

topological consistency. This resolves the critical issue of manipulating multiple representations.

Tolerance is used to obtain a roughly constant number of objects stored on layers. Better output is achieved by using different tolerances in different sections of a polyline or for different polylines. There is no reliable procedure for automatically selecting tolerance values. In our implementation, tolerance-setting can be changed on a case-by-case basis. In doing so, the number of layers and a roughly constant number of objects stored on layers can be optimised by predetermined tolerance values on the server side.

For example, we consider the polyline in Fig. 2 consisting of 28 points. The coarsest level (Layer 0) stores at least two end points of the polyline (point 1 and point 28) and is created by applying the modified RDP algorithm with the largest predetermined tolerance value. The next layer (e.g. Layer 1) is created by storing new points, which are selected by applying the algorithm with a smaller tolerance value. All new points, which lie in between two points (point 1 and 28) of Layer 0, with regard to their original sequence number, are stored. This process is repeated until the tolerance value equals 0. Fig. 2 shows multiple representations of a polyline by performing the modified RDP algorithm with different tolerance values. In such a way, all points of the polyline are grouped into different layers.

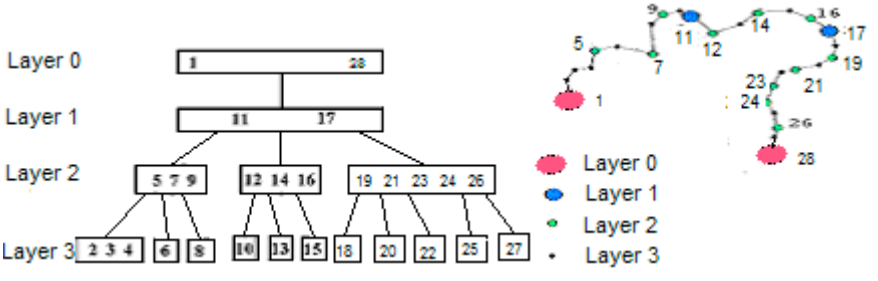


Fig. 2. Points allocated in every layer on server side

4.2 Vertical Links

Every point identified by its NodeID is ordered by its original SeqNo in a polyline allocated into a layer on the server. A pointer is made from the root to the new points consisting of LayerNo, PolylineID (or PolygonID), NodeID and SeqNo. This allows us to know which points are added to the subsequent layers and which ones are preserved. Fig. 3 shows the link from layer 0 to layer 1 consists of LayerNo (1), PolylineID (177267165), NodeID (new point: 32772) and sequenceNo (new point 17). Furthermore, different representations of a polyline at different levels of detail can be linked in a hierarchical structure i.e. layers::polygons::polylines::nodes. Vertical links allow for hierarchical navigation and browsing across levels and facilitate to reconstruct a map at a given level of detail (for more detail about this see section 4.3).

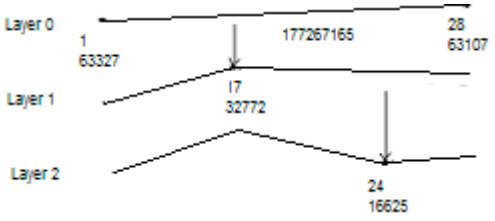


Fig. 3. Vertical links joining different representations of a polyline in a multiple map representation sequence

4.3 EXDCEL on Client Side

On the server, only the coarsest layer is completely stored; increments and vertical links are stored on the subsequent layers. On the client, the dataset corresponding to an intermediate representation must be reconstructed.

Vertical links are transmitted with increments and used to reconstruct the complete topological data structure of intermediate layers. Multiple representations progressively downloaded from server are organised as a hierarchical structure (see fig. 2 as a reference): a polyline with the coarsest level of detail can be created initially and a finer polyline will be created by only merging new points according to its sequence number. A complete representation corresponding to an intermediate level of detail can be obtained by merging the coarsest layer with increments that are progressive transmitted by following vertical links in a hierarchy until the users requests are satisfied.

Reconstructing the topology on the client makes data management locally possible: locally available entities and their spatial relations can be manipulated without having to reconnect to the server. This saves communication across networks and users waiting time.

5 Conclusions

In this paper, we describe the development and implementation of a new data structure for progressive transmission of a sequence of multiple representations generalised by applying a line simplification algorithm. Such a data structure supports hierarchical navigation, topological consistency of multiple representations and spatial indexing.

Our data structure is a hierarchical extension of a topological data structure for plane maps and therefore can be used to store not only linear data but generic vector datasets. Furthermore it can be extended to include the vertical links corresponding to topological changes such as the ones performed by operators described in [2]. We are working towards a complete integration of topological changes and geometric changes within the same data structure. This is essential in real map generalisation applications.

We are also currently testing our system to evaluate the efficiency of the transmission process.

References

1. Ballard. D. 1981. Strip Trees: A Hierarchical Representation for Curves. Communication of the Association for Computing Machinery, vol. 14: 310-321
2. Bertolotto, M.: Geometric Modelling of Spatial Entities at Multiple Levels of Resolution. Ph.D. Thesis, Department of Computer and Information Sciences, University of Genova, Italy (1998).
3. Bertolotto, M., Egenhofer, M.: Progressive Vector Transmission. Proceedings, 7th International Symposium on Advances in Geographic Information Systems, Kansas City, MO: (1999) 152-157.
4. Bertolotto, M., Egenhofer, M.: Progressive Transmission of Vector Map Data over the World Wide Web, GeoInformatica - An International Journal on Advances of Computer Science for Geographic Information Systems, Vol. 5 (4), Kluwer Academic Publishers (2001) 345-373.
5. Buttenfield, B.P.: Transmitting Vector Geospatial Data across the Internet, Proceedings GIScience 2002, Lecture Notes in Computer Science, Vol. 2478. Springer-Verlag, Berlin (2002) 51-64.
6. De Floriani, L., Marzano, P., Puppo, E.: Spatial quieries and data models, in Spatial Information Theory – A theoretical basis for GIS, A.U. Frank, I. Campari (eds.), Lecture Notes in Computer Science 716, Springer-Verlag (1993) 113-138.
7. Douglas, D.H., Peucker, T.K.: Algorithms for the reduction of the number of points required representing a digitised line or its character. The Canadian Cartographer, Vol. 10 (2), (1973) 112-123.
8. Preparata, F.P., Shamos, M.I.: Computational Geometry: an Introduction, Springer-Verlag (1985).
9. Ramer, U.: An Iterative procedure for the polygonal approximation of plane curves, Computer Vision Graphic and Image Processing, Vol. 1, (1972) 244-256.
10. Saalfeld, A.: Topologically consistent line simplification with the Douglas-Peucker algorithm. Cartography and GIS, Vol. 26 (1) (1999).
11. van Oosterom, P:. Reactive Data Structures for Geographic Information Systems. PhD-thesis Department of Computer Science, Leiden University, (1990).