Algorithm for encoding *n*D spatial objects into GIS

D E Andrianov[1], S V Eremeev[1], Y A Kovalev[1] and K V Kuptsov[1]

[1]State University named after Alexander Grigorievich and Nikolai Grigorievich Stoletov, Gorky str. 87, Vladimir, Russia, 600000

Abstract. In the article it is proposed to apply computer topology methods to create effective data structures that will allow storing and processing spatial scenes in n-dimensional space. The basis for the representation of *n*D-geoobjects is n-dimensional simplexes. For example, the representation of spatial data of high dimension will allow us to describe the topological relationships between 3D objects in time. The mathematical foundations and software for representation and processing of spatial data of high dimension are developed. The algorithm for encoding spatial objects *n*D in GIS will provide an opportunity to solve a wide range of tasks for processing complex graphic information.

1. Introduction

The development of GIS technologies in the modern world is advancing towards the use of 3D maps, multiscale, and also in the analysis of the time component. For high dimensions, those algorithms that work well for 2D and 3D cards are inapplicable. Integration in GIS of additional characteristics, such as the third spatial dimension, time, and scale, has so far been achieved mainly through special adaptations to 2D data structures, and not by creating new ones that extend the capabilities of GIS software.

The urgency of the work is that existing algorithms for processing and analyzing spatial scenes in *n*- dimensional space require significant time-consuming, or expected loss of processing quality. The integration of dimensions raises new requirements for creating new methods and algorithms for processing spatial data. The use of topological data analysis is a promising direction in GIS. The paper proposes to apply the topology of computer techniques to create efficient data structures that will store and process spatial scene in a single *n*-dimensional spatial [1, 2, 3, 4, 5].

For the destruction of objects in the GIS, there are different sets of data structures [6, 7, 8, 9]. In work [10] approaches to storage of spatial relations in GIS, such as: spatial queries on the basis of language SQL and matrix of topological relations are considered. The main attention in the article is given to the representation of natural hierarchical structures of spatial objects. A method for storing information about spatial relationships directly in the object identifier is developed, the structure of such an identifier is given.

The paper [11] discusses the structure of the database table in IBM DB2, which is designed to store and manage spatial data in both a regular table and store spatial data in accordance with the spatial geometry of IBM DB2. DB2 not only saved all spatial information, but also guaranteed spatial data and uniformity of these attributes under any conditions. All data stored in a relational database, so it can implement GIS applications by visiting network technology. Using the method given in this article, you can save a form document or another electronic document card in one kind of relational database. The spatial data table is mainly used to store the space of the geometric data objects of the

*.shp file, each form (such as a point, line and plane) consists of a table form, each spatial object in the table is stored as IBM Geometry type DB2 Spatial Extender, field The spatial type is ST Geometry, because the design of the data table structure saves all the information in the form file and it can be well integrated between the spatial data table and the attribute data table via related properties So, you can use the space data independently of the spatial data file, we just need to read the relevant information from the relational database in order to solve the data consistency problem and the integrity problem. This method of data storage will be useful when working with vector maps mainly for their storage and convenient access to objects, but will not allow access to their topological properties.

In [12] refers to the object-oriented repository (OBS), which was developed in 1998, is he starting point of high-performance data storage. Compared to the traditional file storage of the model, the OBS model uses the object interface and storage of the unloading of the management function from application to storage device.

OBS has the following key features:

1. The object is a logical block of memory that contains object data, object attributes, and methods;
2. the interface of the object is a simple method, such as creating, deleting, opening, closing, reading, writing, etc.;
3. OBS allows intelligent devices, including device and data management, object structure and interpretation relationships, access patterns and security settings.

Advantages of OBS:

1. high performance;
2. security;
3. scalability.

Disadvantages of this method is the inability to work with *n*D objects, since the functional does not provide for storing the properties of these objects.

The article [13] describes the possibility of storing the topology from the data model of the spatial topology of Oracle. The obvious differences from other DBMSs are:

Isolated nodes. The spatial topology data model allows you to integrate isolated nodes into the topology. An important advantage of this is that isolated nodes inside can be identified through links to the database and not resorting to spatial search.

Coordination of storage of the connected nodes. There is a small amount with Oracle's spatial topology data model. Coordinates for connecting nodes are stored twice. This is because the spatial topology data model stores each edge as an Oracle Spatial SDO_Geometry data type. This SDO_Geometry for the edge includes all coordinates, including those that correspond to the start and end nodes.

Thus, for these initial and final nodes, the latitude / longitude coordinate is stored twice; once as part of the SDO_Geometry and once on the connection node record. The API for the Oracle Spatial Topology Data Model manages this data transparently for the user and keeps it consistent.

The authors of Ref. [14] describe a complex approach to constructing a hierarchical structure of a road network for a continuous multiscale representation, especially continuous selective skipping of roads in a network. In this structure, the model of the road network is constructed using a linear and areal hierarchy. A continuous multiscale view of the road network can be achieved by searching in these hierarchies.

But these algorithms are not designed for *n*D objects, including for storing information about the scale of maps and the time of changing objects on the map. Since they allow you to store either only geometric information, or specific topological features.

The basis of representation *n*D geoobjects can be put *n*-dimensional simplexes. For example, the representation of spatial data of high dimension will allow to describe topological relations between 3D objects in time. There is a need to develop mathematical bases and software for the presentation and processing of spatial data of high dimensionality. The algorithm for coding *n*D spatial objects in the GIS will make it possible to solve a wide range of tasks for processing complex graphic information.

2. Encoding algorithm *n*D objects

Vector representations of *n*D objects exclude the following problems: they are usually more compact, they can describe boundaries more accurately than raster ones, and represent object attributes directly. This makes them particularly interesting for more advanced GIS, even if they were explored as data structures. Usually, the objects used in the 2D vector GIS are not considered in other dimensions. However, there are various multidimensional geometric and topological structures that were developed in other areas and can be used for these purposes. Even with the fact that they are difficult to implement and use, and often require additional computation to support some aspects of real data. Therefore, they were almost never used in practice. It was for these objects that a data storage structure was developed based on a compressed simplex tree.

A simplex is a convex hull of $n+1$ points of an affine space (dimension n or greater) that are assumed to be affine independent (that is, they do not lie in a subspace of dimension $n-1$). These points are called the vertices of a simplex. An example of a simplex is shown in figure 1.

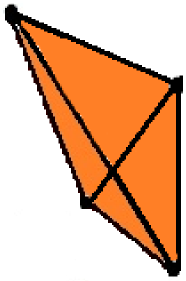

Figure 1. Simplex.

A simplicial complex is a topological space represented as a union of sets homoeomorphic to a simplex and forming a triangulation of this space such that:

1. with any of the simplexes in this set all its faces enter;
2. any two simplexes either do not have a common point at all, or intersect only along a whole face of some dimension, and only one face;
3. for any point x of the complex there is a neighbourhood U such that if it U intersects with the simplex of the complex Δ then $x \in \Delta$.

Let be $K = \{\sigma_0 \ldots \sigma_i\}$ a simplicial complex, which is shown in figure 2.

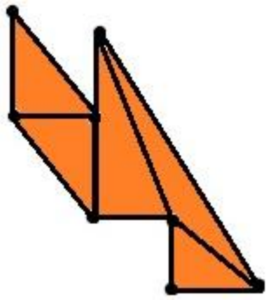

Figure 2. Simplicial complex.

In figure 2 shows a set of *n*-dimensional simplexes σ_i, where $i \in \{1 \ldots n\}$. Formally, let, $\sigma_i \in \{x_i, y_i, z_i, m_j, t_j\}$ where x_i, y_i, z_i the set of coordinates of the object in the set, $i \in \{1 \ldots n\}$, m_j - is the scale on which the object is displayed, and t_j is the time interval in the set into $j \in \{1 \ldots n\}$ which the object on the map was changed.

A simplicial tree of an undirected graph G is the spanning tree of a graph G with a distinguished root with the property that any two adjacent vertices in the graph G are related to each other by an ancestor / child relation. All search trees in depth and all the hamiltonian paths are simplex trees.

In finite graphs, although depth-first searches are inherently sequential, simplex trees can be constructed by a randomized parallel algorithm with a complexity class.

Simplicial trees can be used to determine the depth of the graph tree and as part of the planarity test for checking whether the graph is planar. The description of trees by the one-place logic of second-order graphs makes it possible to recognize the graph-dependent properties of orientation effectively for graphs with bounded tree width using the Course theorem. Not every infinite graph has a tree of simplexes and graphs, which have no such tree, can be described by forbidden minors.

A tree of simplexes exists in any graph with a countable number of vertices, even if the version of infinite depth search cannot successfully verify all vertices of the graph. In an infinite graph a tree must have exactly one infinite path for each ray of the graph and the existence of a simplicial tree characterizes graphs whose topological completions, formed by adding an infinitely distant point for each ray, are metric spaces.

To represent a compressed tree of simplexes K, we store the corresponding numbers in the tree that satisfy the following properties:

1. The nodes of a tree are in a bijection with simplexes of all dimensions of the complex. The first level is associated with an empty face.

2. Each node of the tree, except the first, preserves the label of the vertex. In particular, the node N associated with the simplex σ retains the label of the extreme vertex (σ).

3. Vertexes, whose labels occur along the path from the first level of the tree to the node N, are connected with the simplex σ, are vertices σ. The labels are sorted in ascending order along the given path, and each label appears only once. This data structure is called a simplicial tree K [15].

Consider the simplex tree in figure 3, which contains the following sublevels:

1. the card;
2. scale;
3. The object;
4. coordinates of the object;
5. Time;

6-7. other sublevels

It should be noted that in this figure an area is created, beginning with sublevel 2. These objects are compressed, identical and are on different scales.

The purpose of compression is to identify the common parts of the object and save them only once. More specifically, if the same subtree is located on two different nodes of the simplex tree, then the subtree is stored only once, and the two root nodes now point to a unique copy of the subtree. As a consequence, the nodes are no longer in a bijection with the nodes of the complex, but we still have the property that the paths from the root are in a bijection with simplexes.

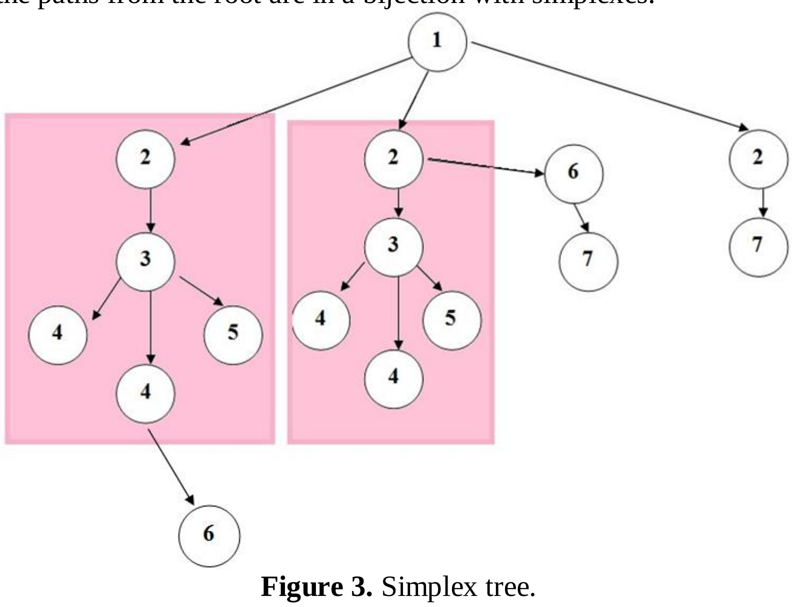

Figure 3. Simplex tree.

Access to all the heirs of a compressed simplex tree can be realized similarly to the usual simplex tree. Allowing an ascending traversal in the tree is also possible (with additional pointers from descendants to ancestors), and this improves the efficiency of some operations, such as surface searches. However, in the compressed simplex tree, the ancestors are not unique. To accommodate this, we mark the ancestors that were available, and use this to return in the upward direction.

3. The results of the algorithm

Figure 4 shows the images of the object on different scales, which changed over time.

The main search takes place on the largest scale, on the remaining scales of the maps, a search is performed and selection of previously found objects by the identifier. Detailed search only occurs in the corresponding buffer zones, and not across the entire map, which allows you to significantly reduce the time to find the right objects.

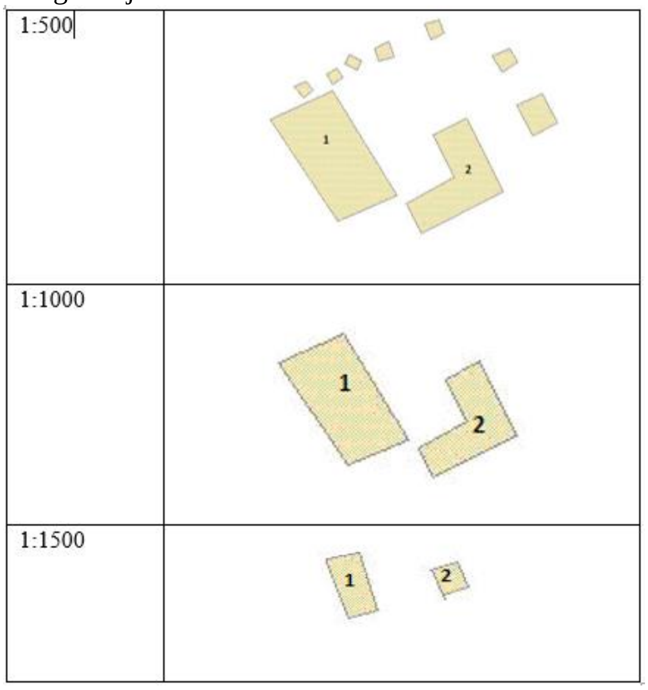

Figure 4. Displaying an object on different scales.

The result of the operation of the *n*D coding algorithm is shown in figure 5 as a graph showing the objects on scales of 1:1000 and 1:1500.

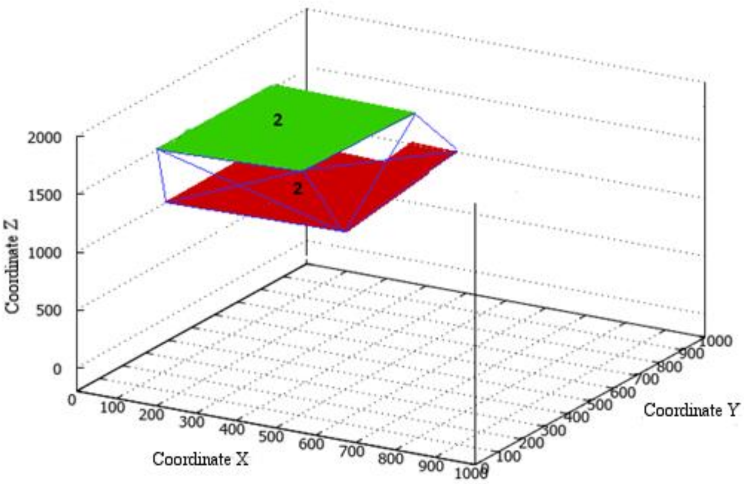

Figure 5. Graph of found objects on different scales. Where X and Y are the coordinates of the object, and the axis Z is the scale of the map (1:500, 1:1000 and 1:1500).

Green on the chart highlighted the object "2" on a scale of 1:1500, red highlighted the object "2" on a scale of 1:1000, the lines that unite them form simplexes.

The work of the algorithm was tested when searching for objects on different scales of the map. To search for objects, an algorithm was used to search for spatial objects based on specified criteria based on buffer zones in multi-scale GIS [16]. The result is displayed in the graph of figure 6.

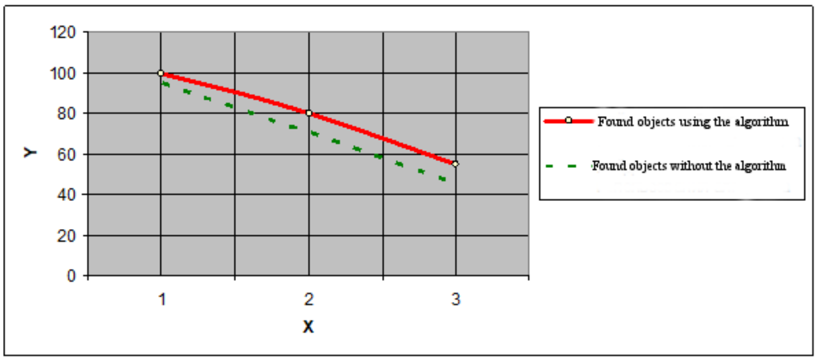

Figure 6. The plot of found objects on different scales.

The graph in Figure 6 shows the search for objects on different scales. Where the axis X is the scale of the map (1:500, 1:1000 and 1:1500), Y is the number of objects on the map. On the basis of the work, a graph was constructed in which an increase in the number of found objects using the developed algorithm by 10% is observed with respect to the search for objects without using it.

Also a graph was constructed based on the search of objects for a certain period of time.

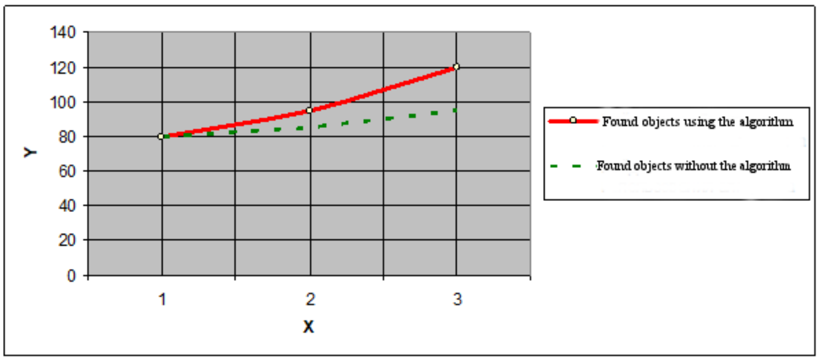

Figure 7. The schedule of the found objects in different time interval.

Figure 7 shows a graph with the search for objects in different periods of time. Where the axis X is the time period in years (2009, 2013 and 2016), Y is the number of objects on the map. The graph shows an increase in the number of found objects using the developed algorithm with an efficiency of 15% relative to the search for objects without using it.

4. Conclusion

The algorithm of encoding of *n*D spatial objects in GIS is developed in the article. A data structure based on a simplex tree was also developed and analogs of data structures for multi-scale GIS were considered.

This algorithm can be used in construction, for example, when testing soil changes in recent years. Also, the algorithm can be used to bypass obstacles, for example a quadcopter, which will analyze the presence of an obstacle in its path on the basis of the map.

5. References

[1] Eremeev S V, Andrianov D E and Komkov V A 2013 Algorithms for the formation of a graph model of urban territory in the GIS *Geoinformatics* **4** 19-24 (in Russian)

[2] Bulaev A V 2008 A formal model for establishing topological relations with objects that contain curvilinear segments *Algorithms, Methods and Systems of Data Processing* **13** 16-24 (in Russian)
[3] Sharapov R V and Varlamov A D 2013 Basic metrics that assess the quality of the work of image search systems *Algorithms, Methods and Systems of Data Processing* **2** 3-11 (in Russian)
[4] Gashnikov M V, Glumov N I, Kuznetsov A V, Mitekin V A, Myasnikov V V and Sergeev V V 2016 Hyperspectral remote sensing data compression and protection *Computer Optics* **40(5)** 689-712 DOI: 10.18287/2412-6179-2016-40-5-689-712
[5] Kopenkov V N and Myasnikov V V 2016 Development of an algorithm for automatic construction of a computational procedure of local image processing, based on the hierarchical regression *Computer Optics* **40(5)** 713-720 DOI: 10.18287/2412-6179-2016-40-5-713-720
[6] Fursov V A, Goshin Ye V and Kotov A P 2016 The hybrid CPU/GPU implementation of the computational procedure for digital terrain models generation from satellite images *Computer Optics* **40(5)** 721-728 DOI: 10.18287/2412-6179-2016-40-5-721-728
[7] Myasnikov E V 2017 Hyperspectral image segmentation using dimensionality reduction and classical segmentation approaches *Computer Optics* **41(4)** 564-572 DOI: 10.18287/2412-6179-2017-41-4-564-572
[8] Sadykov S S, Bulanova Y A, Kanunova E E and Zakharova E A 2014 Development of concepts for constructing an information system for diagnosing neoplasms on mammograms *Information Technologies* **10** 51-56 (in Russian)
[9] Teryokhin A V 2014 The concept of recognition of arbitrarily located three-dimensional objects from two images of projections *Algorithms, Methods and Systems of Data Processing* **2** 29-40 (in Russian)
[10] Eremeev S V and Filimonov M M 2014 Algorithm for coding spatial identifiers in hierarchical topological systems *Algorithms, Methods and Systems of Data Processing* **4(29)** 50-58 (in Russian)
[11] Zhou Yan and Zhu Qing 2006 A scalable distributed spatial data storage mode *International Society for Photogrammetry and Remote Sensing* **XXXVI** 855-859
[12] Zhu Bin and Wang Anbao 2011 The Storage Technology for GIS Data Realization *Journal of Computers* **6(10)** 2229-2236
[13] Zhilin Li and Qi Zhou 2012 Integration of linear and areal hierarchies for continuous multi-scale representation of road networks *Intern. J. of Geographical Information Science* **26** 855-880
[14] Galdi D E 2005 Spatial Data Storage and Topology in the Redesigned MAF *TIGER System* **3** 103-112
[15] Boissonnat J-D, Srikanta K C and Tavenas S 2015 Building Efficient and Compact Data Structures for Simplicial Complexe *An extended abstract appeared in the proceedings of SoCG* **32** 24-31
[16] Kovalev Y A and Eremeev S V 2016 An algorithm for finding spatial objects based on given criteria based on buffer zones in multiscale GIS *Proceedings of the 26th International Scientific Conference "GRAPHICON"* **19** 414-416 (in Russian)

Acknowledgments
The reported study was funded by RFBR and Vladimir region according to the research project № 17-47-330387 and № 326 of 29.09.2017.