ISSN 1330-3651(Print), ISSN 1848-6339 (Online)
https://doi.org/10.17559/TV-20170601094307
Original scientific paper

A Web-GIS Online Vector Data Editing Method Based on Multi-scale Representation Data Structure

Ming LIAO, Xinlin QIAN

Abstract: From analysis of the factors affecting the service quality of Web GIS, the key factor in the efficiency of online vector data interactive editing is to control the size of the relevant data set. This paper puts forward an online vector data editing method based on the multi-scale representation data structure DBLG-tree. It designs WindowingQuery for large geometric features. Combined with SimplifyQuery, it can effectively control the vertex scale of the geometric features. At the same time, the sub-feature resulting from WindowingQuery retains the connection with the original feature and can be merged and updated. It provides DownsizeQuery which can directly control the volume of query result from the data set, so as to adapt to the performance of different browsers and mobile devices. Research shows that the execution time of DownsizeQuery under the support of multi-scale representation data structure is stable, and the size of the queried and updated data set is also stable. The upper bounded query time and query results ensure the response time of online interactive visualization and updating of the vector features.

Keyword: multi-scale representation data structure; online editing; progressive transmission; vector data simplification

1 INTRODUCTION

The bottleneck factors affecting the service quality of WebGIS application mainly include five aspects [1]: (1) the pressure of supporting frequent data access in the database server; (2) the pressure of supporting a large number of users in the GIS server; (3) the pressure of supporting the transmission of a large amount of data in the Internet; (4) limitation in the performance of the browser or mobile device; (5) expectations of non-professional GIS users. Cloud storage provides the ability to store massive data at the server and cloud computing provides the good elastic computing ability for the server-side GIS spatial analysis and data extraction (query). For performance of the data retrieval executed at the server side, as long as the query processing can be scaled horizontally to cloud servers, (1) and (2) are no longer the main factors affecting WebGIS bottleneck. Although some browsers can directly use the graphic rendering capability of the hardware, such as Chrome, Firefox, Safari which support WebGL standard, compare to the server supported by cloud computing, the bottleneck of the performance factor of network bandwidth, browser or mobile device in WebGIS application chain is more remarkable. In addition, network bandwidth varies in different network environments (such as 3G, 4G, WiFi, etc.). Different browsers also have different rendering performance of graphics and sizes of mobile device. Users have different expectations in different environments.

The basic functions provided by WebGIS application can be divided into three categories: visualization, data editing and spatial analysis. WebGIS application with visualization as the main purpose has been very popular (such as Google Map and Map world). The demand for WebGIS application with data editing as the main purpose is increasing, in which VGI application is one of the representative [2, 3]. In VGI application, the public are also the data providers. They can upload the GPS data or edit the data online with the high-resolution images as the base map through Web browsers or mobile devices.

Based on the following basic editing requirements, the users usually need to make modification on a display scale where they can clearly discern the geographical features; due to the restrictions on the size and resolution of the screen, when the vertex density of the geometric object is too close to the screen resolution, it is impossible to edit the vertex. In WebGIS, the data editing operation follows the following process: it is visualized at a scale with a larger scope and rough details; with the reduction of the observation scale, the map is gradually localized and the details of the geometric object in the viewport increase gradually until they appear in the form of full details at last; at this time, it is allowed to enter the editing state; the updates are submitted after completing editing. From the perspective of user, operations in the process of data editing application can be divided into three categories: scaling, feature editing and update submitting. From the perspective of data transmission, the whole process is the cycle of three stages: transmission of the extracted data to the client, interaction between data and users at the client, returning of the updated data to the server.

Relevant research on progressive transmission is applicable to the rapid response to the window zooming. Its basic idea is: making the multi-resolution representation of the data set at the server [4, 5] and rapidly transmitting the data at the low resolution version to the client to respond to the user operation first and then the time-consuming details of the data at the high resolution version to the client for reconstruction. During the transmission of the data at the high-resolution version, the users can do the zooming operation based on the low-resolution version. It shortens the response time of the operation. Progressive transmission needs the support of the multi-scale representation data structure.

Multi-scale representation is to form several levels of geographic features through the generalization process to represent the same geographic entity in a certain area. Multi-scale representation appears as a hierarchical structure and is usually expressed by tree in the data structure. Tree structure is suitable to express the relations of objects between different levels. Multi-scale representation can act on different granularity levels. According to the different granularities of the spatial data management unit, tree nodes can be divided into three categories: (1) when using the tree to represent single geometric feature or its clipped part, tree nodes can

correspond to the line vertices, such as Strip – tree [6], BLG-tree [7], V-tree [8]; (2) when using the tree to represent the data set index, tree nodes can correspond to geographic extent which holds geographical features, such as CIF-Quadtree [9], Gap-tree [10], tGap-tree [11], SDMR-tree [12]; (3) when using the tree to represent the tile pyramid, tree nodes can correspond to the feature set in the tiles, such as quadtree tile pyramid [13].

The construction of the multi-scale representation model can be divided into three categories according to the comprehensive method: (1) pre-calculation LoD method. This method has two advantages: first of all, the generalization process is not sensitive to the time, so the copying and computation intensive algorithms can be employed, such as optimization technique [14] and multi-agent system [4]. Secondly, human-computer interaction can be adopted to remedy the flaws in automatic generalization. (2) Real-time calculation method. During the runtime, a temporary result set is extracted from the basic data sets under the condition of keeping the topology consistency according to the selected scale. This method is characterized by no storage redundancy and strict requirement on the execution time. It needs the efficient but not too complicated generalization method, such as multi-scale database [15], radical law [16] and morphing algorithms [17]. (3) In order to improve the execution efficiency of calculation during the runtime, it is feasible to pre-process some information supporting calculation to obtain the parameters related to the scale (e.g., sampling rate) and feature associated storage, such as Reactive data structure [18].

For the vertex editing of polyline or polygon feature, progressive transmission has no effect of improving the response time. This is because progressive transmission shortens the response time between the rough data version and the complete data version [19, 20], but it is impossible to enter editing until the transmission of the complete version of data extraction ends. Updated data submission has the same requirement. At this time, the data need to be directly transmitted without generalization, so the data query operation should be effective, accurate and efficient. Especially when the large geometric features exist, the efficiency of data query and updating with these features becomes very low.

The key to solve the problem is that the data set of the query result is controllable. There are three reasons for it: (1) only when the data volume is controllable can it be possible to control the network transmission time and the computing and graphic rendering time of the browsers and mobile devices, so as to ensure the constant response time; (2) only when the size of the data set is adjustable can it adapt to different network states and client conditions (such as screen size and drawing performance), allow the users to improve the response time through adjusting the scale of data set (maybe at the cost of visual quality); (3) the scale of the geometric features in the data set is heterogeneous in the spatial distribution. The feature model is the discrete expression of spatial entities and the vector data expressing the feature model are composed of sampling points. For polyline and polygon, the vertices are the sampling points. The vertex density not only depends on the resolution requirement, but also has relations with the geometrical characteristics of the object. For example, under the same scale, a rectangle has a lower vertex density than a big curvature curve. The condition for the features to enter the editing state is up to the vertex density rather than the display scale. Therefore, it is more reasonable to adopt the query controlling the vertex scale.

This paper studies the Web online editing scheme of the vector data based on the multi-scale representation data structure and implements the query directly controlling the number of vertices of the data set to ensure the response time of editing and updating. Then, it makes the test evaluation on the execution performance of query and the visual effect of the final data set.

2 SELECTION OF THE MULTI-SCALE REPRESENTATION DATA STRUCTURE

The multi-scale representation data structure supporting the vector data editing needs to consider three aspects: (1) efficiency of the query related to the scale; (2) efficiency of updating the basic data set; (3) updating needs to be propagated in multiple scale representations timely. The selection of the generalization algorithm for the multi-scale representation construction cannot be constrained to the demands of the traditional cartographic generalization, because the data editing is ultimately done in the full detail scale and other scales are used only for the scaling process. The generalization algorithm focuses on high-performance real-time implementation, constant volume data transmission and scale-related incremental LoDcapacity [21]. Updating propagating requires the automatic reconstruction of the multi-scale structure. However, pre-computed LoD does not adapt to the frequent and automatic structure reconstruction, and the full real-time computing is too time-consuming. Thus, the third method can be adopted to construct the multi-scale representation, which is to pre-compute the scale parameters related to the generalization and associate the parameters with the storage of geo-features in order to improve the efficiency of real-time computing. At the same time, w.r.t. data updating efficiency granularity of the data access needs to be considered.

2.1 Problems of Large Geometric Features

With the construction of the global or regional database, the data sets created from map collection need seamless splicing and feature merging during database building [22]. When the geographical features crossing a large area appear in the geographic database and the vertices contained in these features may be hundreds of thousands, they are called large geometric features. Typical examples are boundary, Mainstream River, large lake, contour, highway, etc. Two samples of boundary data are listed here as examples: (1) the global high-resolution coastline data (GSHHS), polygon number: 188, 571, total number of vertices: 10,348,493, number of vertices of the maximum polygon: 1,181,126; (2) municipal administrative region boundary data in Jiangxi Province, China at the scale of 1:10000, polygon number: 11, number of vertices: 3,401,603, number of vertices of the maximum polygon: 445,123. Large geometric features exert great influence on data query, updating and co-editing.

2.1.1 Query Problem

Windowing query with geometrical features as the basic granularity selects features whose vertices fall into the query window. Query with this kind of granularity cannot effectively control the data scale. The irrelevant vertices outside the target window (Window Outer) as shown in Fig. 1 are 25 times more than vertices in the window (Window Inner).

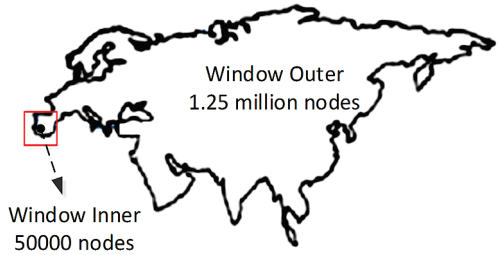

Figure 1 Spatial query based on intersection

If clip operation is done on the feature for data extraction, it may lose the structural information between the whole and its parts and cause ambiguity when the client restores the state [11], as shown in Fig. 2.

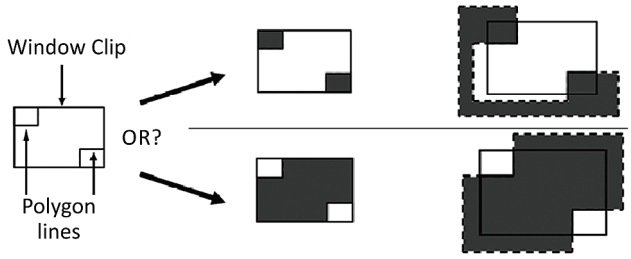

Figure 2 Ambiguity when clipped result restores polygon

V-Tree [8] is designed to be a data structure storing the long coordinate point strings and support the effective extraction. It is used for the local storage and extraction of long geographical objects under the large scale. Its idea about vertices extraction is effective. However, it is still an extension of B-tree essentially. Each *K* points belong to a MBR division and simplification operation adopts the strategy of selecting one point every *K* vertices (that is, taking one point in each MBR), vertex selection is casual and random, which cannot ensure the selected ones to have the significant geometric characteristics or meet the multi-scale representation requirement of GIS. Thus, it needs WindowingQuery which can clip and extract large geometric features. Different from clip operation, this kind of query needs to retain the structural relationship between the whole and its parts without losing the characteristic of multi-scale representation

2.1.2 Updating Problem

At present, there are three solutions for modeling of the spatial data: 1) topological information first; 2) geographic features first; 3) hybrid structure. The second one is the mainstream solution. Its representatives are OGCSFS (Simple Feature Specification) and ESRI's Geodatabase. A record in the dataset corresponds to a Feature and each Feature contains Geometry and several attributes. The topological information is not stored explicitly. The basic unit of data reading and writing is Feature or Geometry. As shown in Fig.1, however, for each user, the edited vertices of a large geometric feature actually accounts for only a tiny part of vertices of the whole feature. Therefore, it is inefficient to read all vertices of a geometric feature, make local modification and then write them back.

2.1.3 Co-editing Problem

Multi-user co-processing is an issue of technology and workflow. The co-processing of the database has two main strategies: version strategy and locking strategy. The first one is suitable for long transactions or offline mode. Rather than locking the database, it exports the copy or snapshot of the original data for each user, edits each copy and then do the multi-channel integration based on region of the changed dataset. Its examples are ESRI's SDE's Version mode and OSM's offline editor mode.

For the updates conflict coordination of large geometric features, if database takes the geometric feature as the basic unit, how to distinguish between differences and merge when multiple users do the local editing of the same object and submit them? It needs to examine two situations: 1) if the editing parts of the users are in the same area on the space, only the overwrite strategy can be adopted (according to the priority or timestamp). At this time, the updates conflict coordination with features as the basic unit is reasonable. 2) If the editing areas of different users are disjointed, the cover strategy is unsuitable. At this time, the updates conflict coordination with features as the basic unit is unreasonable. Instead, it needs to merge the edited parts of the geometric features in different areas.

In short, to solve the problems in query, updating and co-editing in large geometric objects, it needs the multi-scale data structure support the simplification and windowing as well. The results of windowing query can retain the structural information between the whole and its parts, so as to implement the local updating of features. The clipped sub-feature maintains the semantic information to facilitate the whole feature recovery at the client. Compatible sub-features can be merged in database to support multiple users make updating in different parts.

2.2 DBLG-tree Structure

BLG-tree [7] selects the vertices by the DP algorithm to construct the multi-scale representation of the line object. BLG-tree has three properties:

1. Locality. It is an imbalanced binary tree. Each tree corresponds to an independent line object. Any subtree corresponds to a sub-line of the line object. Any node corresponds to the inner vertex of the line object and stores its sequence number and *e* value (the geometric significance of a vertex refers to the deviation distance of the line object to the original one after simplification).
2. Order. It is a sorted binary tree. The sequence number of the tree node (vertex) is greater than that of the vertex represented by its left child node and smaller than that of the vertex represented by its right child node.
3. Duality. The structure of a BLG-tree corresponds to the operation result of the DP algorithm. Any subtree corresponds to the operation result of the DP algorithm of

the subline.

DBLG-tree is the improvement of BLG-tree. The differences are mainly in two aspects:

1. Relative Number Used by the Node

As shown in Fig. 3, (a) is a line object and (b) is the BLG-tree representation of (a). The node uses the absolute sequence number of the vertex in the line. In (c), the absolute sequence number converts to the relative number. The conversion from relative number to absolute sequence number is to sum the relative numbers of all left ancestors along the line from the node to the root node, in which A is left ancestor of B when B is the right descendant of A.

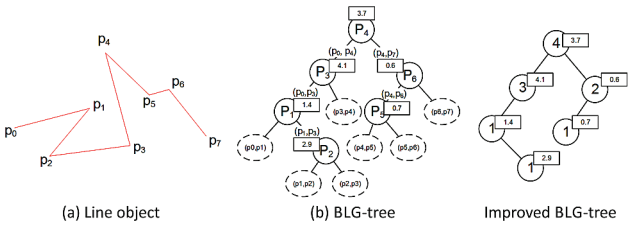

Figure 3 BLG-tree and its sequence number improvement

2. In addition to store the relative number and e value of the node, tree nodes also attach a MBR.

The node MBR represents the minimum bounding rectangle of all its descendants. As shown in Fig.4, the yellow box in (a) represents that MBR of p_2 node is the minimum bounding rectangle of the point set composed by $\{p_1, p_2\}$; the blue box in (a) represents that MBR of p_3 node is the minimum bounding rectangle of the point set composed by $\{p_1, p_2, p_3\}$. Later, the paper will describe how MBR is used to accelerate the windowing query. A relaxed alternative of MBR is also described below.

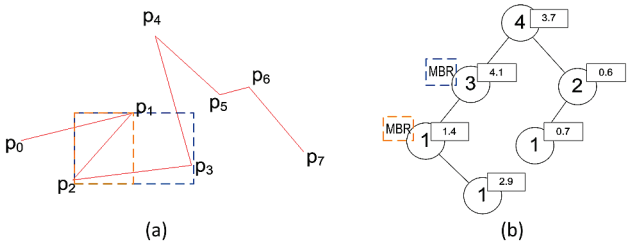

Figure 4 Node associated MBR

Considering the characteristics of the DP algorithm, as shown in Fig. 5, p_1 is the child node of p_3. It represents that for p_0p_3 baseline, the offset e_1 of p_1 is the maximum in the $\{p_0, p_1,\ldots, p_3\}$ point sets. Then, MBR_{loose}, formed by the external expansion of e1 based on MBR of $\{p_0, p_1, p_3\}$ contains $\{p_0,..., p_3\}$. If query window W does not intersect MBR_{loose}, $\{p_0,\ldots, p_3\}$ can be pruned securely.

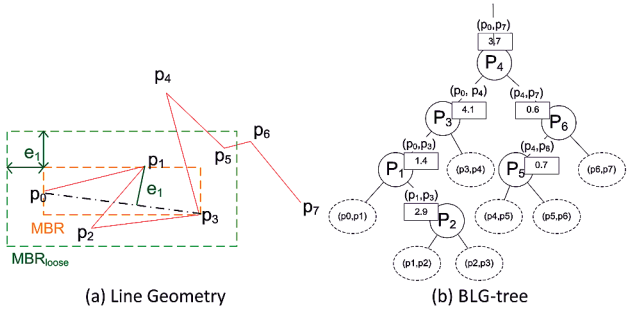

Figure 5 Loose MBR

Generally, for a node of DBLG-tree, it is feasible to judge whether all sub-nodes under it are out of the query window w through a loose bounding MBR_{loose} formed by the external expansion of e of the minimum bounding rectangle composed of three points, namely, the node, its lowest left ancestor and lowest right ancestor nodes (the starting and ending vertex of the corresponding baseline in DP). MBR_{loose} is obtained through real-time calculation. It can replace the MBR value stored in nodes.

2.3 Windowing Query, Fragment Merger and Updating

2.3.1 Definition of Windowing Query

The query method that extracts the geometrical feature according to the window w is called windowing query, written as WindowingQuery. Each geometric feature can be represented as a BLG-tree. The result of WindowingQuery is still a BLG-tree sub-tree which is composed of the nodes in the window w and their ancestors along the BLG-tree. The results of WindowingQuery are shown in Fig. 6. (a) represents a complete line object. $w1$, $w2$, $w3$ and $w4$ represent the query windows. (b) is the WindowingQuery result with the condition w1 and it is a subtree ST_{w1} of T. (c) is ST_{w2}, (d) is ST_{w3} and (e) is ST_{w4}.

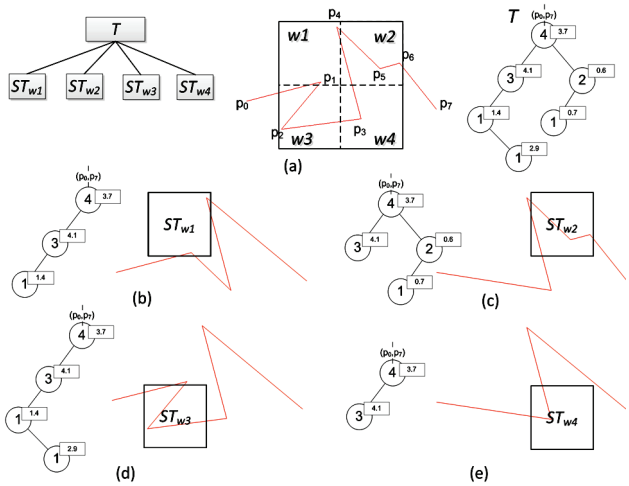

Figure 6 WindowingQuery of a DBLG-tree

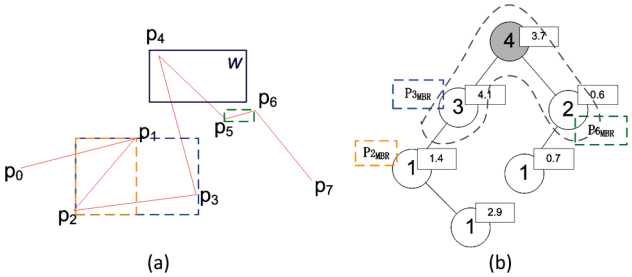

Figure 7 Node traversing under the WindowingQuery algorithm

2.3.2 Efficiency of Windowing Query

Each node of a DBLG-tree associates with a MBR value. The MBR can be used to accelerate WindowingQuery since the MBR of the ancestor contains that of its descendants. If the MBR of the ancestor does not meet w, all descendants will not be in w. Thus, it is unnecessary to search further. In Fig. 7, (b) is to query the point set $\{p_4\}$ in w. The query process is completed by

traversing three nodes $\{p_3, p_4, p_6\}$, which is the most optimistic. The most pessimistic situation is that it needs to traverse the whole tree (equivalent to running N times of the array method). The average of the most optimistic and most pessimistic situations can be regarded as $\log_2(N)$, where N is the number of vertices in a feature.

2.3.3 Fragment Merger

The data state recovery of the client includes two categories: the first is the data recovery from the rough state to the full detail state in window zooming, which is generally realized through progressive transmission; the other is the recovery of the result set of WindowingQuery from fragmentation to the overall state.

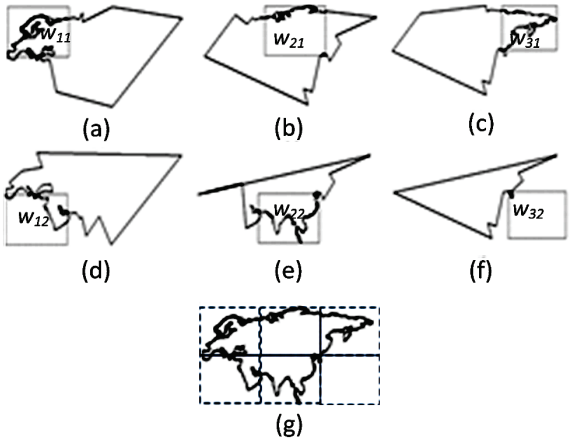

Figure 8 Fragment merger for vector tiles

Fragment merger has two applications. First of all, it is used in the vector tiles which are produced by WindowingQuery on DBLG-tree. At the client, the data state can be recovered through the fragment subtree merging. Since all WindowingQuery results are from the same DBLG-tree whose subtrees have common ancestor nodes and are compatible in the structure, there will be no ambiguity during merging operation. Fig. 8 presents the partition of the large geometric object according to the rule of 3×2, in which the rectangular boxes represent the non-overlapped windows w_{11}, w_{21}, w_{31}, w_{12}, w_{22} and w_{32}, respectively and (g) represents a complete Eurasia polygon merged by all fragments at the browser.

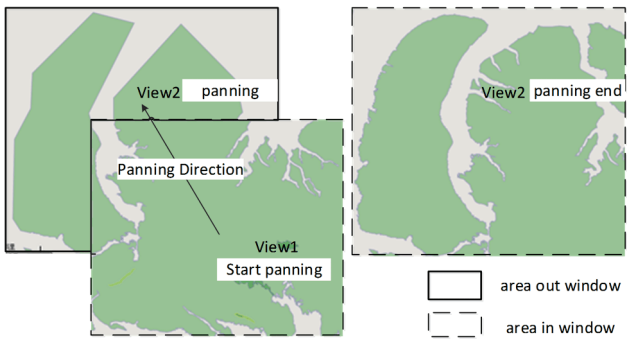

Figure 9 Fragment merger for screen moving compensation

Secondly, it is used for the simplified area compensation in the window screen moving, as shown in Fig. 9. The spatial region at this time is irregular. The difference between the front and back screen areas can be supplemented for details through the WindowingQuery result.

2.3.4 Fragment Updating

Since DBLG-tree represents a vertex sequence, and the vertex sequence number changes with the insertion and deletion of the vertices, it is inconvenient to use the sequence number to identify the nodes needing updating. They are represented by the subtrees of DBLG-tree, known as the path subtree. The path subtree contains only all ancestors of the target node. Any node in DBLG-tree can be represented by the only path subtree. If vertex updating is to insert, it is represented by the subtree of the inserted node's immediate predecessor and successor. If vertex updating is to delete or change, it is directly represented by the path subtree of the target node.

It is observed that any updating action, whether inserting, moving or deleting the vertex, makes the result of the DP algorithm of a subline rather than the whole line to change. The subline on which the result of the DP algorithm changes is tagged as the affected subline. The corresponding subtree to the subline is known as the invalid subtree. The invalid subtree loses the structural property of DBLG-tree due to the updating operation, so it needs to be reconstructed according to the result of the DP algorithm on the affected subline at the appropriate time. It is not difficult to find the invalid subtree caused by an updating action. Take inserting new vertices as an example. Its process is to start from the root node of DBLG-tree to look for the target node according to the searching method of the binary search tree. The target node is the corresponding tree node to the vertex after the inserted point. In the search process, for any node coming across, as long as it is not the target one, it is needed to calculate the distance between the new node and the simplified line corresponding to the node. If the value is greater than the round-off error, the subtree with this node as the root is invalid.

If the structural property is not maintained in the modification process of the fragment data, it means that the DBLG-tree of the fragment data is equivalent to the binary search tree with the relative vertex number. Thus, vertex modification, inserting and deletion all adopt the same treatment as the binary search tree. Inserting is to directly become the child node of its immediate predecessor or successor; deletion is to find the target node and delete it; modification is to find the target node and then replace the deleted one with the immediate predecessor.

Each update operation will mark an invalid subtree. If the invalid subtree is the descendant or ancestor of an existing invalid subtree, the smaller subtree will be merged into the bigger one.

When it needs to submit updating, the invalid subtrees are read overall and the DP algorithm is executed (there may be multiple independent invalid subtrees) to generate a new tree structure and replace the original structure with the subtree in the original tree.

2.4 SimplifyQuery Limiting the Number of Vertices

The query for a geo-feature with condition of the upper limit number of vertices I is called SimplifyQueryI. The inputs of the query algorithm are the root nodes n of

the target feature and *i*, which the limit number of vertices contained in simplified feature. The query result is a subtree of *n*, representing the simplified line composed of *i* vertices.

The algorithm uses the weighted breadth first search (WBFS) which outputs the tree nodes with the round-off error monotonously decreasing to construct a priority queue with the round-off error as the weight. The priority queue has good characteristics. It can help to implement progressive transmission and other advanced functions. The weighted breadth first searches of tree structure are similar to regular BFS with the only difference of changing the queue to the priority queue.

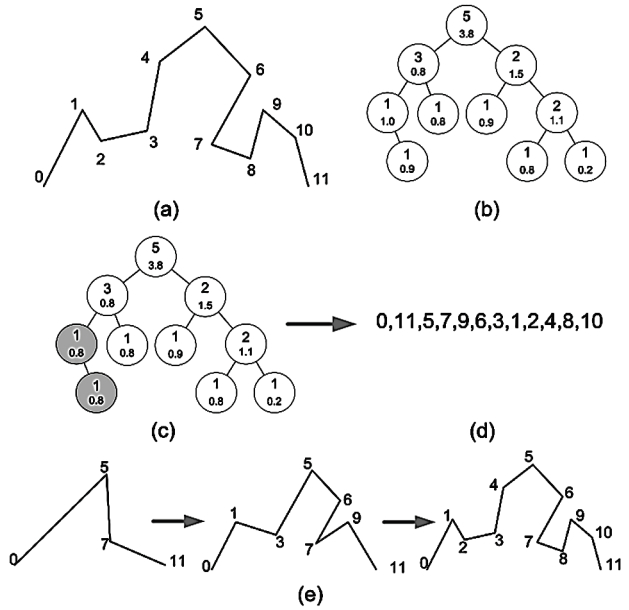

Figure 10 Application of DBLG-tree for the query limiting the number of vertices of the geometric feature

Fig. 10 shows the method of applying DBLG-tree for the query limiting the number of vertices of a single geometric feature. (a) represents the original line object. (b) is DBLG-tree of (a). (c) shows monotonization of DBLG-tree. Grey nodes represent that the error value of the node takes that of the parent node after monotonization. (d) represents the output result after the weighted breadth first search on DBLG-tree. (e) is the result set under the conditions of $i \leq 4$, $i \leq 8$, $i \leq 16$, respectively.

2.5 Control of the Vertex Number of the Dataset

It can be thought that the number of vertices is a direct measurement of the data set size. The control of the vertex number of the data set is carried out in two aspects: controlling the maximum vertex number *i* of a single geometric feature first and then the vertex number *I* of the whole data set.

The vertex number of a single geometric feature can be limited through SimplifyQuery and WindowingQuery. SimplifyQuery samples the vertices of the whole geometric feature according to a unified requirement. WindowingQuery is to further simplify geometric features in the area outside the target window. SimplifyQuery can reduce the number of vertices of features in a dataset homogeneous on the whole spatial extent. WindowingQuery of geometric features has the effect of simplifying geometric features heterogeneously over the space. The sampling rate within the query window is homogeneous, and the sampling rate outside the query window is heterogeneous, which means linear features outside the window will be simplified in various degrees, namely the farther from the window the more details will be simplified from the line feature, as shown in Fig. 11.

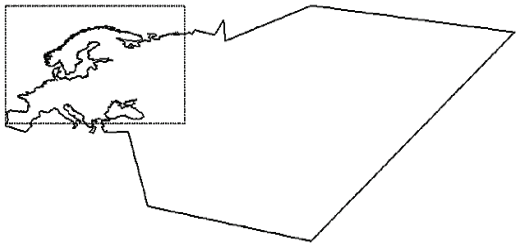

Figure 11 Simplifying effect of WindowingQuery on large geometric feature

SimplifyQuery solves the vertex scale problem of the general geometric features and WindowingQuery further solves the vertex scale problem of large geometric features. Through the results of above two types of queries, *i* is effectively and accurately controlled.

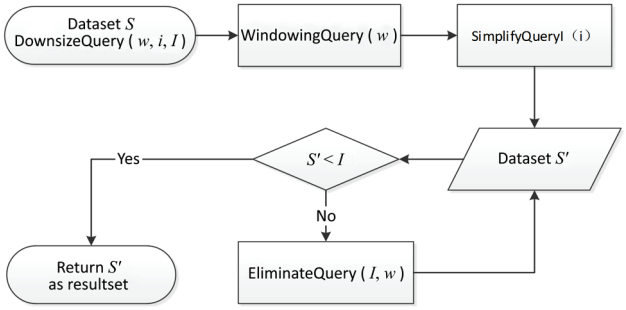

Figure 12 Processing of DownsizeQuery

DownsizeQuery is designed on the basis of controlling the vertex scale of a single geometric feature. It adopts the query window *w*, the upper limit *i* of vertices of a single geometric feature and the upper limit of vertices of the result set *I* to be query conditions together to control the vertex number of the result set, expressed as DownsizeQuery (*w, i, I*). DownsizeQuery can be implemented by WindowingQuery (*w*), SimplifyQueryI (*i*) and EliminateQuery (*I, w*). Its process is shown in Fig. 12.

The implementation of WindowingQuery and SimplifyQueryI has been given in Section 2.3 and 2.4. A vertex histogram based selection algorithm given in literature [23] can be used as EliminateQuery. The algorithm selects the important geometric features and controls the scale of the result set and the uniformity of the feature distribution at the same time based on the vertex histogram.

Considering the distinguishable visibility of the vertices, the recommended value of *i* can be set to be between 0.5 and 2 times of the number of pixel in the screen diagonal line of the display device. For example, for 1280×1024 display screen, i can be set to 1500.

For the setting of *I* value, obviously $i < I$; if $i \geq I$, a result set with only one geometric feature will appear.

Empirical study has shown that $I = i \times 10$ or so and obvious small area elements will not be eliminated. For example, when $i = 1500$, I can be set to 20000.

3 EXPERIMENT AND PERFORMANCE ANALYSIS

The response time of the real-time editing process of the entire WebGIS vector data can be broken down into three main parts: execution time of the server, data transmission time, vector drawing time at the client.

The experimental data are the global high resolution coastline data (GSHHS). The total vertex number of the data set is 10,348,493. The vertex number of a single polygon with the maximum number of vertices is 1,181,126.

The hardware of the computer is configured as below: Intel core i3 2.53 GHz processor, 4Gb DDR3 1067 MHZ memory, NvidiaGeforce 310 MX display card (512M graphics memory), screen window size 1280 × 1024 pixel and SVG for browser side drawing.

The conditions of DownsizeQuery are set as follows: the upper limit number i of vertices for a single element is 1500 and the upper limit number I of vertices for the data set is 20000.

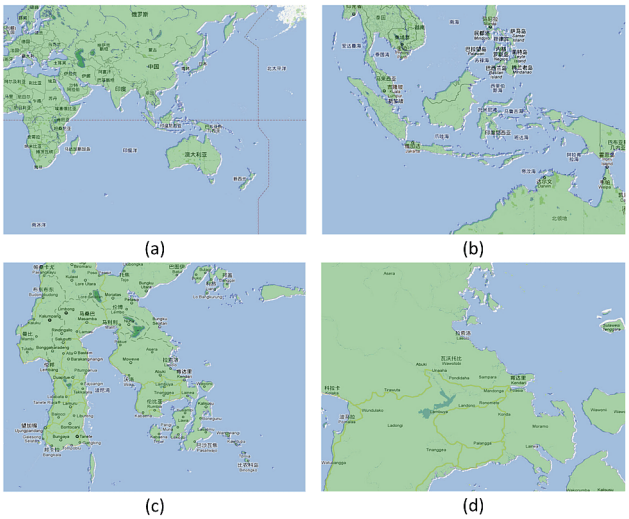

Figure 13 Visualization Effect of DownsizeQuery Result

3.1 Visualization Effect

To examine the visualization effect of DownsizeQuery, Google Map is taken as a base (white) to be superimposed over the DownsizeQuery result of GHHSS data set (light green). If there is no white polygon exposed, it indicates that the downsizing effect is good. In Fig. 13, (a), (b), (c) and (d) are the display effect of the map at level 2, level 4, level 6 and level 8, respectively. Light green basically covers white without obvious area feature missing. It indicates that provided that DownsizeQuery controls the data set within the constant data size, the visualization quality of the data set is not significantly reduced and its influence on people's cognition is little.

3.2 Execution Performance

When executing zooming or moving operation, the geographical extent w is corresponding to the screen window changes. The execution time of DownsizeQuery is shown in Fig. 14. Thus, with the change of query window, the execution time of DownsizeQuery has a slight change and the absolute execution time is at the level of 10 milliseconds. It can be considered that the execution performance of the scheme owns the characteristics of query window insensitive, that the running time of the data query processing has nothing to do with the query window scale. It can obtain the stable response time.

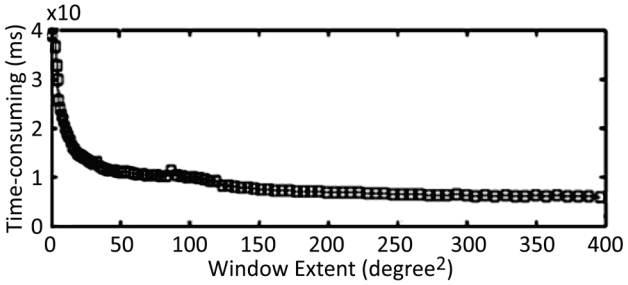

Figure 14 Execution time of DownsizeQuery

3.3 Drawing Performance

At the browser side, Firefox, Safari, Chrome and IE9 all provide native support for SVG. Due to the inferior rendering efficiency on SVG, Firefox cannot be interactively used. Safari, Chrome and IE9 have good rendering performance on SVG. Empirical study shows that if the number of vertices is limited to less than 20000, the drawing time of SVG under Chrome is about 1 s. The evaluation on the drawing performance of several Web vector drawing technologies like SVG, Flash and Canvas in the browsers by some literature shows that when other conditions are the same, the difference in the drawing time among different browsers is up to 5 times. It is only one of the factors behind the user's visualization difference. It also demonstrates the necessity to customize the size of the data set by the users according to their own visualization environment from another aspect.

4 CONCLUSION

The key to improve the Web online editing performance of the vector data and the response time of operation is to effectively, accurately and efficiently control the size of the transmission data set. Progressive transmission well resolves the timely response in window zooming operation, but it can do nothing about data editing and updating submittal under the condition of requiring the complete detail version of data. This paper uses DBLG-tree as a multi-scale representation data structure to organize vertices of geometric features and propose WindowingQuery and Simplifyquery to limit vertex number of features with consideration of spatial query and geometric characteristics preserving.

DBLG-tree supports fragment merger and fragment updating and effectively solves the query and updating efficiency problem brought by large geometric features. It can be used to process DownsizeQuery by controlling the vertex number of the data set on the basis of WindowingQuery, SimplifyQuery and EliminateQuery, so that the users can control the transmission size of the data

according to the network speed and the performance of the client. Experiments have shown that the execution time of DownsizeQuery owns the characteristic of insensitive to the size or location of the query window. In the condition of setting the appropriate upper limit vertex number of the data set, DownsizeQuery can effectively control the data size of the queried result set while ensuring the visualization effect of different display levels. The stability of DownsizeQuery and the controllability of the data set scale can provide the stable response time for the users' online editing of the vector data.

Acknowledgements

This work was supported by the Young Academic and Technical Leaders Research Foundation of Key Laboratory of Geospatial Information Engineering, NASG of China (201511), the Open Foundation of Key Laboratory of Water Information Collaborative Perception and Intelligent Processing, Jiangxi Province of China (2016WICSIP009), and the National Key Technology R&D Program of China (2015BAH50F01).

5 REFERENCES

[1] Fu, P. & Sun, J. (2010). *Web GIS: Principles and Applications*. ESRI press. ISBN: 9781589482456. 312p.
[2] Turner, A. (2006). *Introduction to Neogeography*. O'Reilly Media, 54p.
[3] Goodchild, M. F. (2007). Citizens as Voluntary Sensors: Spatial Data Infrastructure in the World of Web 2.0. *International Journal of Spatial Data Infrastructures Research, 2*, 24-32.
[4] Barrault, M., Regnauld, N., Duchene, C., et al. (2001). Integrating multi agent, object oriented and algorithmic techniques for improved automated map generalisation. *Proceedings of the 20th International Cartographic Conference*.
[5] Yang, B. (2007). Efficient transmission of vector data over the Internet. *International Journal of Geographical Information Science, 21*(2), 215-237. https://doi.org/10.1080/13658810600894281
[6] Ballard, D. H. (1981). *Strip trees: a hierarchical representation for curves*. Commun ACM. https://doi.org/10.1145/358645.358661
[7] Oosterom, P. V. & Bos, J. V. D. (1989). *An object-oriented approach to the design of geographic information systems*. Springer Berlin Heidelberg. https://doi.org/10.1016/0097-8493(89)90002-2
[8] Mediano, M. R., Casanova, M. A. & Dreux, M. (1994). V-Trees - A Storage Method for Long Vector Data. *International Conference on Very Large Data Bases - Vldb'94* September 12-15, 1994, Santiago De Chile, Chile.
[9] Kedem, G. (1982). The Quad-CIF Tree: A Data Structure for Hierarchical On-Line Algorithms, 352-357.
[10] Oosterom, P. V. (1995). The GAP-tree, an approach to 'on the fly' map generalization of an area partitioning.
[11] Antoniou, V., Morley, J. & Haklay, M. (2009). *Tiled Vectors: A Method for Vector Transmission over the Web*. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-10601-9_5
[12] Deng. H. Y. (2009). R-Tree Index Structure for Multi-Scale Representation of Spatial Data. *Chinese Journal of Computers, 32*(1), 177-184. https://doi.org/10.3724/SP.J.1016.2009.00177
[13] Lu F. (2003). A Multi-Granularity Approach for Adaptive Mass Vector Dataset Access and Transmission. *Annals of GIS, 9*(1-2), 29-34. https://doi.org/10.1080/10824000309480585
[14] Graepel, T., Burger, M. & Obermayer, K. (1998). Self-organizing maps: Generalizations and new optimization techniques. *Neurocomputing, 21*(1-3), 173-190. https://doi.org/10.1016/S0925-2312(98)00035-6
[15] Cecconi, A., Weibel, R. & Barrault, M. (2012). *Improving Automated Generalisation for On-Demand Web Mapping by Multiscale Databases*. Springer Berlin Heidelberg.
[16] Pillewizer, W. & Pillewizer, W. (1966). The Principles of Selection, a Means of Cartographic Generalization. *Cartographic Journal*.
[17] Sederberg, T. W. & Greenwood E. (1992). A physically based approach to 2-D shape blending. *Acm Siggraph Computer Graphics, 26*(2), 25-34. https://doi.org/10.1145/142920.134001
[18] Oosterom, P. V. (1991). *The reactive-tree: A storage structure for a seamless, scale less geographic database*. Auto-carto.
[19] Wen, F., He, Z., Dai, Z. & Yang, X. (2014). Characteristics of Investors' Risk Preference for Stock Markets. *Economic Computation & Economic Cybernetics Studies & Research, 48*(3), 235-254.
[20] Yang, B. (2005). A multi-resolution model of vector map data for rapid transmission over the Internet. *Computers & Geosciences, 31*(5), 569-578. https://doi.org/10.1016/j.cageo.2004.11.011
[21] Yang, B. & Weibel, R. (2009). Editorial: Some thoughts on progressive transmission of spatial datasets in the web environment. *Computers & Geosciences, 35*(11), 2175-2176. https://doi.org/10.1016/j.cageo.2009.07.001
[22] Jiang, J., Huangm W., Wei-Huam L. U., et al. (2009). Research on Entity-based Data Modeling for National Geo-spatial Information Service Platform. *Geomatics World*.
[23] Li, D. & Qian, X. (2010). A Brief Introduction of Data Management for Volunteered Geographic Information. *Geomatics and Information Science of Wuhan University, 35*(4), 379-383.

Contact information:

Ming LIAO, PhD
1) School of Remote Sensing and Information Engineering, Wuhan University, China 430079
2) Jiangxi Province Key Laboratory of Water Information Cooperative Sensing and Intelligent Processing, Nanchang, China, 330209
3) Jiangxi Provincial Geomatics Center, Nanchang, China, 330209
Room 918, No.2166 Fanghu Road, Jiangxi Provincial Geomatics Center, Nanchang County, Jiangxi Province, China
E-mail: liaomingjxgc@foxmail.com

Xinlin QIAN, PhD
Chinese Academy of Surveying & Mapping, Beijing, China, 100083
28 Lianhuachixi Road, Haidian district, Beijing
E-mail: xinlinqian@vip.qq.com