Efficient Data Transmission Using Map Generalization On Client-side Web GIS

Chen LIANG Chung-Ho LEE Zu-Kuan WEI Hae-Young BAE
Department of Computer Science and Engineering, INHA University, Inchon, 402-751, KOREA
E-Mail: liangchen76@hotmail.com

ABSTRACT

Recently researches have been made on the client-side Web GIS, which can lessen the load of a server and provide users with interactive geographic information. However, the initial delay is the main drawback because of a high volume of geographic data and because the server does not associate spatial features with the map scale. Even when a complex spatial object is too small to be distinguished from a point by the naked eyes, its complete data will be transmitted.

This paper proposes a new efficient schema to reduce the response time and increase transmission efficiency. Briefly speaking, "Transmit what can be seen" is the main idea. By exploiting the generalization algorithm, the proposed method allows the server to extract readable features from objects according to the display scale. Meanwhile, increasingly detailed map will be cached on the client. Therefore this method will contribute to the transmission efficiency of Web GISs.

1. Introduction

Internet is a global network of computers connected with communication devices. It is a means for GIS users to exchange GIS data, conduct GIS analysis and present GIS output[7].

Today server-side GISs and client-side GISs are two major kinds of Web GISs[6]. In server-side GISs a server performs all functions, in which the server will be a bottleneck for all users because clients can not share the processing load. In client-side GISs all functions are performed by a client. A server becomes only a data provider so that it will not become a bottleneck in the whole system. The client-side GIS has been more suited for the development of Web GISs.

It is extremely important for GIS to present features on the map as clearly as possible. However, it does not guarantee that at all time all features on a map are readable to users. " As scale decreases, lines should become less irregular."[1]. So features are associated with scales. It is normal that some features can not be readable at a certain scale. And not all geographic data transmitted across Internet are helpful for visual quality. For example, objects in the building layer are represented by complex objects in database. At a smaller scale, it looks no more than like a point or a simple object no matter how accurately this object is drawn. If server transmits the complete data of spatial objects to client without any consideration about the display scale on client side, the superfluous data increases the transmission cost. To eliminate those data can contribute to the transmission efficiency.

This paper proposes a "transmit what seen" method to reduce the size of a query result for an individual operation and improve the transmission efficiency. For every query, servers generalize complex objects according to the display scale on the client and transmit only those data that can upgrade visual quality.

The remaining of this paper is organized as follows. Section 2 describes the "transmit what can be seen" method, the data structures and the generalization algorithm. Section 3 discusses the processing on basic spatial operations. Section 4 discusses how to refine the method and states conclusions.

The research is supported by Ministry of Information and Communication under work of university S/W research center.

2. The "Transmit What Seen" Method

2.1 "What seen, What transmitted"

This section proposes a display-scale dependent generalization method. To improve the transmission efficiency, the method associates object features with the map scale. To maximize the transmission efficiency, it is favorable to transmit only those data which can contribute to the visual quality of the map on the client, that is, transmit only readable features to the client. By using the generalization method, those data, which can not improve visual fidelity, will be eliminated. Therefore, the method exploits generalization to reduce the transmission cost without degradation on visual fidelity. Figure 1 describes the processing procedure of a system where this method is applied.

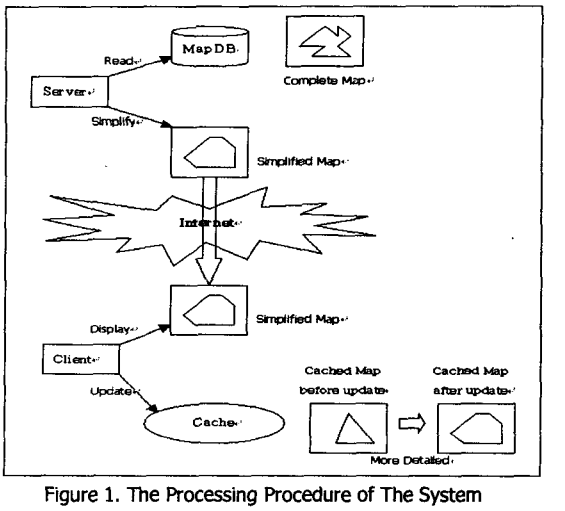

Figure 1. The Processing Procedure of The System

Geographic objects are commonly represented by geometric data such as points, lines and regions. A vertex consists of x and y coordinates. The significant vertices refer to those which are regarded as noticeable in shapes and locations of objects in a map[3]. At a scale, a simplified object should both preserve the main element of shape, and recognize topological properties. Although not all features appear at a certain scale, users would

feel comfortable when the recognizable features are preserved[1].

From the comparison between two maps in figure 2, a lot of detailed features in the left map do not appear in the other map. Therefore, the size of spatial objects is much less. If only the simplified objects are transmitted, the transmission cost will decrease if the bandwidth of network is fixed.

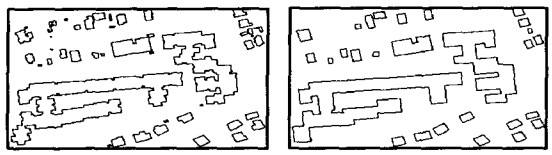

Figure 2. The Building Layer at Different Scales

2.2 Map Generalization Algorithm

Over-detailed data reduce the transmission efficiency, so a generalization algorithm is employed to eliminate those data which can not help with the improvement on visual quality and extracts significant features from an object. According to the display scale of the map on the client, the algorithm generalizes the map to preserve the features which are recognizable at a certain scale.

Map generalization or simplification has been studied for many years. Some simple algorithms are purely based on mathematical models and geometry, but more advanced algorithms take into account the characteristics of features such as spatial relationships and patterns[1]. The possibly most well known algorithm proposed by Douglas and Peucker in 1973 focuses on line simplification. The simplification algorithm first joins the beginning and ending vertices of a line feature by a straight line, and then examines perpendicular distances to the individual vertices. Those that are closer than a selected threshold distance can be removed. The point furthest away is selected as a new end point for repetition of the process until there are no points closer to a line than the threshold[2].

2.3 The Structures for Transmission and Cache

The proposed method defines structures for transmission and cache. Features can be represented by the significant vertices like the turning points[3].

The transmission structure consists of a group of spatial objects. An object is composed of *Object Identifier*(OID), *Graphic Attribute*(GA), *Number Of Vertices* and *Vertex*. *Object Identifier* is an identifier of a spatial object. *Graphic Attribute* is attributes which are displayed such as line width, color, style and so on. *Number Of Vertices* are the number of vertices that transmitted from server to client. A *Vertex* is composed X and Y coordinates. Moreover, an object has a *flag* to say whether it is simplified or not.

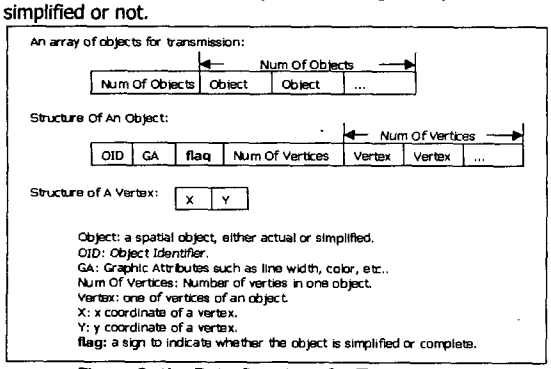

Figure 3. the Data Structure for Transmission

Compared with the object structure for transmission, the structure for cache has a one more field — **Scale**, with which features of spatial objects are associate with the scale. Figure 4 describes the complete structure of an object for cache.

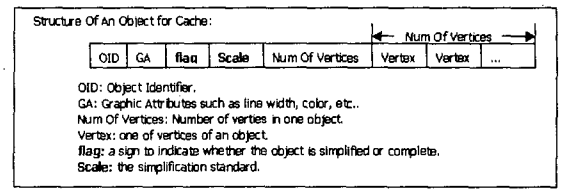

Figure 4. the Data Structure for Cache

2.4 The "Transmit What Can Be Seen" Method

A scale is defined as the number of pixels to describe 1 unit distance in the map. When scale decreases, features will become less. Therefore, at a certain scale some features are not readable. When operations are ZOOMIN or ZOOMOUT or the client's window is resized, the scale will change. At that time, more or less features should be extracted from objects. The display scale is calculated with the following equation.

Scale = Max (ClientWidth / MapWidth,
ClientHeight / MapHeight)

Where, MapWidth and MapHeight are the width and height of the map to be read, and ClientWidth and ClientHeight are the size of the client window in pixel.

When a user makes a query, the client first examines which spatial object is cached and complete, and requests those objects which have not been cached. For those cached simplified objects, if new scale is reduced, they have to be simplified again on the client side. If users demand more detailed map, generalization has to be performed on complete objects by the server. If the scale does not change, they are certainly still meaningful, for example, when users pan the map. The following algorithm describes the client's procedure for making a query.

Algorithm 1. QueryObjs (QueryMBR, ClientWindow, CurrentScale)

Input: QueryMBR: the user's query MBR.
ClientWindow: the size of the client window.
CurrentScale: the current display scale.
Output: a set of object IDs to be requested from server — RequestedOIDs.

```
01: NewScale = Max ( QueryMBR.Width / ClientWindow.Width,
      QueryMBR.Height / ClientWindow.Height )
02: for "all objects overlapping with QueryMBR"
03:     if "object has not been cached" then
04:         Add object.OID into RequestedOIDs;
05:     else if "object.flag == 1 and
                CurrentScale != NewScale" then
06:         if "NewScale > object.scale" then
07:             Delete object from Cache;
08:             Add object.OID into RequestedOIDs;
09:         end if
10:     end if
11: end for
12: Send RequestedOIDs to server;
```

End Of Algorithm 1

The server tries to generalize the requested spatial objects according to the display scale. If simplification damages an object's visual fidelity, its complete spatial data have to be transmitted. Otherwise, the simplified objects will be sent to the

client. Algorithm 2 describes the server's response to the client's query.

Algorithm 2. MakeResult (RequestedOIDs, Scale)
Input: RequestedOIDs: a set of object IDs to be requested from server.
Scale: the map scale on a certain client.
Output: ResultObjs: a set of spatial objects, either actual or simplified.

```
01: for "all OIDs in RequestedOIDs"
02:     Get object by the OID;
        /* Simplify a single object according to Scale */
03:     Clear object.flag with 0;
04:     Do
05:       for " all vertices of object"
06:           Draw a virtual line Baseline from Vertecies[I-1]
                to Vertecies[I+1];
07:           if " the Distance from Vertecies[I] to BaseLine is
                smaller than Scale " then
08:               delete Vertecies[I] from object;
                  Set flag with 1 /* simplified */
09:       end for
10:   Until "No Vertex with over-threshold distance";
11:   Add object into ResultObjs;
12: end for
13: Send ResultObjs to client;
```

End Of Algorithm 2

After receiving the query result the client caches both complete objects and simplified objects. Consequently, in the cache on the client side there are two kinds of these spatial objects. When the display scale is reduced, the map is brief. Accordingly, the number of simplified objects is large and that of complete objects is small. With the increase in the display scale, the map is getting more detailed. Moreover, client will replace objects with less features with the corresponding objects with more ones. So the number of simplified objects decreases while that of complete objects increases. Gradually, objects on the client have the increasingly high scale and rich features. When the briefer map is demanded, enough features of objects can be extracted from cached objects with higher scale to fit in the map at lower scale. Of course, these complete objects can be exploited repeatedly once they are cached.

3. Query Processing

The section discusses the query processing. Spatial operations can be classified into two groups. One is the basic operation such as ZOOMIN, ZOOMOUT, MOVE while distance and area measure, polygon overlap and buffers fall into the other group — spatial analysis.

If the client has cached enough information, some basic operations can be performed on the client side. Otherwise, they have to be performed by the server. For example,

- ZOOMIN. When a more detailed map is demanded, it is necessary to simplify complete objects again. After this operation, the client will replace objects at a smaller scale with ones at a larger scale.
- ZOOMOUT. Before this operation is offered, objects at larger scale have been cached on the client, the less detailed objects will be simplified locally from these more detailed ones having been cached, even if these cached objects are not complete.
- MOVE (map). The characteristic of this operation is that the scale does not change. The client will request the objects having not been cached on the client, and then the server simplifies the requested objects according to the current scale on the client and then returns the result.

After generalization, some features will disappear on a map. Analysis performed on generalized objects is not correct or accurate. Therefore, it is favorable to perform such operations on complete objects in order to gain the correct and precise results. For example,

- Distance and Area Measure. If the relevant objects are not complete on the client, the kind of operations has to be processed by the server. In such a case, only the result will be returned to the client rather then these relevant objects. Of course, if these complete objects have been cached on the client, there are enough data for the client to perform these operations.
- Polygon Overlap. In the brief map, some objects with missing features are distorted actually. It is possible for the spatial relationships to change, especially when a unsophisticated generalization tool is employed. To avoid this problem, making this operation be performed on complete objects is a better choice.
- Buffer and Skeleton Zones. The creation of boundaries, inside or outside an existing polygon offset by a certain distance, and parallel to the boundary needs precision and accuracy. Therefore they have to be computed with complete objects rather than simplified ones.

4. Conclusions and Discussion

The proposed method applies generalization into Web GISs to reduce the transmission cost. Data structures for this method are described. The detailed procedure for spatial data transmission between server and client is discussed. This method can accommodate many algorithms flexibly. Therefore, this method has improved transmission efficiency and the response time.

The high-performance map generalization will contribute to the efficiency of the proposed method. Admittedly, a high efficient algorithm is not enough to speed up and sharpen map generalization. Without sufficient information about the features, relying on algorithms and computation to solve the problems can lead to expensive processing and inaccurate results. Therefore, enriching GIS databases with appropriate feature attributes and structures to support automated generalization has become necessary[1]. To take advantage of the latest development in map generalization, this method is given sufficient flexibility to accommodate a generalization procedure.

Reference

[1] Esri White Paper, "Map Generalization in GIS: Practical Solutions with Workstation ArcInfo Software", ESRI Inc. July 2000.

[2] Robert Laurini and Derek Thompson, "Fundamentals of Spatial Information Systems", Academic Press. 1992

[3] Young-Hwan, Oh, "Advanced Progressive Transmission for Spatial Data in Web based GIS", The Second AEARU Workshop on Web Technology. October 1999

[4] Ralf Hartmut Guting, "An Introduction to Spatial Database Systems", VLDB Journal, Vol.2, No.4, 1994.

[5] Esri White Paper, "The Future of GIS on the Internet", ESRI Inc. June 1997.

[6] Luis Manuel Callejas, "Web-Based GIS: A brief History", http://blaze.trentu.ca/~ermlc/web-gis.html

[7] Z.R. Peng, "An Assessment of the Development of Internet GIS", Proc. Of the 1997 ESRI User Conference, 1999