An SVG-based method to support spatial analysis in Web environment

--Spatial analysis in XML/GML/SVG based WebGIS

Haosheng Huang¹ and Yan Li²

¹Institute of Geoinformation and Cartography, Vienna University of Technology
²School of Computer, South China Normal University

Abstract: This paper discusses how to provide spatial analysis for XML/GML/SVG based WebGIS. In recognition of current XML/GML/SVG based WebGIS’s insufficiency of providing spatial analysis, we focus on two important issues. First, based on the theory of spatial data modeling, we propose an SVG-based spatial information representation model to represent the needed spatial data in SVG on the browser side. Furthermore, this paper designs and implements some spatial operators and integrates them into a Spatial Extended SQL (SESQL) to support spatial query and analysis on spatial data which is represented in GML/SVG. This method can greatly improve the functions of current XML/GML/SVG based WebGIS applications, most of which have been only employed for visualization. Finally, this paper designs some case studies. The results of the case study prove that the suggested method is feasible and operable for providing spatial analysis in XML/GML/SVG based WebGIS.

Keywords: XML/GML/SVG based WebGIS, SVG-based spatial information representation, Web-based spatial analysis, Spatial Extended SQL (SESQL)

1. Introduction

With the rapid development of the Internet, web-based Geoinformation Information System (WebGIS) is an inevitable trend. WebGIS aims at provide GIS functions (such as web mapping, spatial analysis) to users through a common web browser, such as Internet Explorer, Firefox, etc. Currently, more and more researchers pay attentions to the problems of spatial information integration and sharing in WebGIS. XML/GML (Geography Markup Language)/SVG based solution is viewed as a feasible solution for solving these problems. In general, GML [1] is used as
a coding, storing and transmitting standard of multiple spatial data on the server side; while Scalable Vector Graphics (SVG) [2] is used as the representing tool for displaying and interacting with vector data on the browser side.

Currently, lots of WebGIS applications, especially XML/GML/SVG based WebGIS applications, have been employed for visualization (web mapping) only, but avoid the access to spatial analysis functions such as spatial topological query, map overlay, and buffer that are vital for spatial analysis application [2]. When performing spatial analysis task, we always need to install corresponding GIS software on our computer, and carry out the tasks in stand-alone or Local Area Network (LAN) environment. In the meaning time, these applications only support their own spatial data formats. There are also some WebGIS applications which carry out all the spatial analysis tasks on the server side and then send the result to the browser side for visualization. But this server-side solution becomes impractical, as the server can not handle a large volume of concurrent users. In the meaning time, as spatial analysis is a complex task, users always have to try different query solutions before they are satisfied with the result. As every query solution will always result in a large volume of data (intermediate results which users may not need), there will be a high transmission overload between the server side and the web browser side when using this server-side solution. When designing web-based spatial analysis, load-balancing between server side and browser side should be carefully considered.

In recognition of these limitations, this paper proposes that in order to meet the increasing need of spatial analysis in web environment, spatial analysis based on load-balancing technology should be introduced into XML/GML/SVG based WebGIS. In order to provide good performance, load-balancing spatial analysis will carry out spatial query operation on server side or browser side based on the network communication cost versus the computational cost. There are several key issues to be considered: GML (server side)/SVG (browser side) based spatial information representation, spatial operators and spatial query language for both server side (for GML) and browser side (for SVG), load-balancing middleware. The first issue focuses on using GML/SVG to represent the needed spatial data; the second issue focuses on providing spatial operators and spatial query language to carry out spatial analysis on spatial data which is represented in GML (server side) or SVG (browser side); the third issue focuses on dispensing spatial query operation to the server side or browser side.

This paper will focus on the first and the second issues. For the first issues, we introduce spatial data modeling to discuss GML/SVG based spatial information representation. For the second issues, since it is widely acknowledged that these spatial operators and SQL like languages can be used for spatial analysis, some spatial operators such as spatial topological operators, spatial directional operators are designed and integrated into a Spatial Extended SQL to support spatial query on GML and SVG.

The paper is arranged as follows. In section 2, we propose our GML/SVG-based spatial information representation models based on spatial data modeling. Section 3 designs some spatial operators and integrates them into a Spatial Extended SQL. In section 4, we discuss the implementation issues and design a case study to evaluate our suggested methods. Finally, the conclusion and future work are presented in section 5.
2. GML/SVG based spatial information representation

In order to carry out spatial analysis, spatial datasets which are related to the spatial analysis task should be represented in GML/SVG. GML is developed primarily as a XML-based coding standard for geographic (spatial) feature. As spatial dataset is always organizing as hierarchical structure (map - layer - spatial object), in order to represent the needed spatial dataset, a GML-based spatial representation model which takes the characteristics of spatial dataset (such as hierarchical structure) into account has to be developed. There is the similar case for SVG. In the following, we introduce spatial data modeling for solving these problems.

Spatial data modeling is the process of abstracting the real world (identifying the relevant objects and phenomena in the real world) and representing the relevant objects or phenomena in an appropriate form which can be recognized by computers [3]. As GML/SVG is one of the forms which computers use to represent relevant objects or phenomena of the real world, thus we can utilize the theory of spatial data modeling, to discuss GML/SVG-based spatial information representation model which attempts to describe and represent the relevant objects and phenomena in the real world.

There are three models during spatial data modeling: conceptual model, logical model, physical model. According to the contents and requirements of these models, spatial data modeling includes three steps [3]: 1) choosing a conceptual model which can abstract the real world most appropriately, 2) choosing an appropriate data structure to represent the conceptual model, 3) designing a file format, or the appropriate method of record or storage for the data structure in step 2.

![Figure 1. Three models in data modeling](image)

According to the above steps, we design our GML/SVG-based spatial data representation models. OGC's Geometry Object Model (GOM) [4] is one of the most popular vector models for abstracting the real world. Since GOM only defines spatial attributes of spatial entities, we extend it by adding some concepts like map, layer and non-spatial attributes, and take the extended model as our conceptual model. And then we design a spatial data structure for the above conceptual model based on object-oriented design. Finally, we use GML/SVG standard to represent the above spatial data structure, and propose our GML/SVG-based spatial information representation models. Figure 2 and figure 3 depict the proposed spatial information representation models for GML and SVG respectively.
Figure 2 depicts the GML-based spatial information representation model. We use "FeatureCollection" element to represent Map, and use boundedBy to represent the bounded area of the Map. Layer is represented as element which is inherited from "gml: Feature". In this model, if B is PART-OF A, B will be represented as child element of B. Spatial attributes and non-spatial attributes are represented as corresponding GML elements' child element.

Figure 3 depicts the SVG-based spatial information representation model. We use "svg" element to
represent Map, and use viewBox attribute to represent the bounded area of the Map. Layer is represented as "g" element. In this model, if B is PART-OF A, B will be represented as child element of B. For example, Layer is PART-OF Map, so "g" element which represents Layer is a child element of "svg" element which represents the Map. Spatial attributes and non-spatial attributes are represented as corresponding SVG elements' attributes.

With the above GML-based spatial information representation model, users can easily use GML to publish spatial information in the Web environment. At the same time, it is also very easily to translate GML document to SVG document.

The following GML codes show a map of Guangdong province in China which is based on our suggested SVG-based spatial information representation model (section 2). This map includes two layers: city and river.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<FeatureCollection >
  <boundedBy>
    <Envelope>
      <coordinates srsName="urn:EPSG:geographicCRS:4326">94928,2172873 885543,2768086</coordinates>
    </Envelope>
  </boundedBy>
  <featureMember>
    <gd_river_line >
      <ID>R1</ID>
      <length>100</length>
      <lineStringProperty>
        <LineString srsName="">
          <coordinates>725778.53,2577107.71 .....726944.96,2576656.58</coordinates>
        </LineString>
      </lineStringProperty>
    </gd_river_line >
  </featureMember>
  ......
  <featureMember>
    <gd_city_polygon >
      <ID>C1</ID>
      <population>7.5</population>
      <polygonProperty>
        <Polygon srsName="">
          <outerBoundaryIs><LinearRing><coordinates>729687,2563006 ......729687,2563006</coordinates></LinearRing></outerBoundaryIs>
        </Polygon>
      </polygonProperty>
    </gd_city_polygon >
</FeatureCollection>
```
The followings are the codes in GML schema for the above GML code.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:gml="http://www.opengis.net/gml"
    xmlns:xlink="http://www.w3.org/1999/xlink" >
    <import namespace="http://www.opengis.net/gml"
        schemaLocation="feature.xsd"/>
    <element name="FeatureCollection" type="gml:FeatureCollectionType"
        substitutionGroup="gml:_FeatureCollection"/>
    <element name="gd_river_line" type="gml: gd_river_lineType"
        substitutionGroup="gml:_Feature"/>
    <element name="gd_city_polygon" type="gml: gd_city_polygonType"
        substitutionGroup="gml:_Feature"/>
    <complexType name="FeatureCollectionType">
        <complexContent>
            <extension base="gml:AbstractFeatureCollectionType">
                <sequence>
                    <element name="boundedBy" type="gml: BoundingShapeType"
                        minOccurs="1" maxOccurs="1"/>
                    <element ref="gml: gd_river_line "/>
                    <element ref="gml: gd_city_polygon "/>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <complexType name=" gd_river_lineType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element name="ID" type="string"/>
                    <element name=" length " type="double"/>
                    <choice>
                        <element ref="gml:lineStringProperty"/>
                        <element ref="gml:multiLineStringProperty"/>
                    </choice>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
    <complexType name=" gd_city_polygonType">
        <complexContent>
            <extension base="gml:AbstractFeatureType">
                <sequence>
                    <element name="ID" type="string"/>
                    <element name=" length " type="double"/>
                    <choice>
                        <element ref="gml:lineStringProperty"/>
                        <element ref="gml:multiLineStringProperty"/>
                    </choice>
                </sequence>
            </extension>
        </complexContent>
    </complexType>
</schema>
```
The above GML document will be translated to SVG for visualization. The followings are the codes in SVG which visualize the above map.

```
<svg viewBox="94928 2172873 790615 595213" SRS="urn:EPSG:geographicCRS:4326">
  <g transform="translate(0,4940959),scale(1,-1)">
    <g id="gd_city_polygon">
      <path id="C1" population="7.5" d="..."/>
      ...
    </g>
    <g id="gd_river_line">
      <path id="R1" length="100" d="..."/>
      ...
    </g>
  </g>
</svg>
```

3 Spatial operators and GSSESQl

Since the need for spatial query languages has been identified in 1982 [5], many authors have designed their spatial extended database language [2], [6] and [7]. These SQL like languages introduce spatial data types (e.g., point, line and polygon) and spatial operators, allow users to inquire spatial features, primarily in terms of spatial relationships and metric constraints [2]. It is widely acknowledged that these spatial operators and SQL like languages can be used for spatial analysis.

According to section 2, spatial information is organized as a hierarchical structure: map - layer - spatial object. A map represented by GML/SVG can be viewed as a database, the layers as tables of the database, the attributes (spatial attributes and non-spatial attributes) of spatial entities in the layer as columns of corresponding table, spatial entities as the records of corresponding table. As a result, SQL like languages can be used to support spatial analysis on GML/SVG.
In this section, we design some spatial operators and integrate them into a spatial extended SQL, which we define as Spatial Extended SQL (SESQL). This SESQL uses the basic spatial data types discussed in section 3: Point, Linestring, Polygon, Multipoint, Multiline, Multipolygon and Multigeometry. The proposed SESQL can be used on server side or browser side for spatial query and analysis on GML/SVG.

### 3.1 Spatial Operators

In order to carry out a spatial analysis task, we always need to query the spatial object in terms of attributes, spatial topological relationships and metric constraints, and then maybe use some geometrical operations or feature fusion to handle these spatial data.

Spatial analysis basically includes buffer analysis, overlay analysis and network analysis. After analyzing their requirements, we design five kinds of operators: attribute access operators (GeometryType, Centroid, Length, Area, and Envelope), spatial topological operators (Disjoint, Touch, Crosses, Within, Overlap and Contain), spatial order operators (East, East_South, South, West_South, West, West_North, North and East_North), spatial metric operators (Max_Distance, Min_Distance and Average_Distance), and Geometrical operators (Intersection, Union and Difference). They can meet the basic requirements of spatial analysis. For network analysis, we can use Touch and Length operators to find out the touched spatial object (e.g., road) and the distance. We can use Buffer and topological operators to carry out buffer analysis. Also, we can use Intersection, Union and Difference operators to carry out overlay analysis.

### 3.2 SESQL and Some Query Examples

As SESQL is designed for spatial query, there is no need to consider data insert, update and delete. All we should do is integrating the above spatial operators to the original SELECT clause of SQL. The following EBNF (Extended Backus-Naur Form) is the grammar of SELECT clause of SESQL.

```
<select_state> ::= SELECT <selectexpression> FROM <fromexpression> WHERE <whereexpression> GROUP BY <groupbyexpression>
<selectexpression> ::= <select_list> | <Derivated_att> | <Derivated_att>*
<Derivated_att> ::= <spatial_operator> AS <Att_name> | <Derivated_att>*
<spatial_operator> ::= <att_operator> | <topo_operator> | <order_operator> | <dist_operator> | <geometrical_operator>
<att_operator> ::= <att_operator_v> | <att_operator_g>
<att_operator_v> ::= <Att_function1>(<att_geom>)
<Att_function1> ::= GeometryType | Length | Area
<att_geom> ::= expression | <analysis_operator> | <att_operator_g>
<att_operator_g> ::= <Att_function2>(<att_geom>)
<Att_function2> ::= Centroid | Envelope
<topo_operator> ::= <topo_function>(<att_geom>, <att_geom>)
<topo_function> ::= Disjoint | Touch | Crosses | Within | Overlap | Contain
$order_operator> ::= <order_function>(<att_geom>, <att_geom>)
```
We will carry out some spatial queries on the GML/SVG map described in section 2.

The above GML/SVG map includes two layers (gd_city_polygon and gd_river_line), which can be viewed as the following tables: gd_city_polygon (id, population, d) and gd_river_line (id, length, d). In the following there are some query examples.

1) Query example 1: List the cities which are the neighbors of city "C1".

```
SELECT A.id AS "Neighbors_of_C1" FROM gd_city_polygon A, gd_city_polygon B WHERE Touch (A.d, B.d) = True AND B.id="C1"
```

2) Query example 2: List the cities which are crossed by the river "R1".

```
SELECT c.id AS cid FROM gd_river_line r, gd_city_polygon c WHERE r.id="R1" AND Crosses(r.d, c.d) = True
```

3) Query example 3: River "R1" can supply water for the cities, which are 40KM around the river, List this kind of cities.

```
SELECT cy.id FROM gd_city_polygon cy, gd_river_line r WHERE Overlap(cy.d, Buffer(r.d, 40)) = True AND r.id="R1"
```

### 3.3 Workflow of spatial analysis and a spatial analysis example: site selection

According to [8], the workflow of spatial analysis includes four steps: 1) defining the goal and evaluation criteria; 2) representing the needed geospatial data; 3) carrying out spatial query and analysis with GIS tools; 4) result appraisal and explanation. Step1 and step 4 need domain knowledge and are mainly carried out by experts. Step 2 and step 3 need GIS tools to support human-computer interaction. The proposed method can be used to assist step 2 and step 3.

In this section, we take site selection as a spatial analysis example, and use GML/SVG to carry out
this task according to the workflow described in section 2.

Suppose city “Z” wants to choose a site for a theme park. According to the above workflow, we carry out this task based on our proposed method:

Step 1, define the goal and evaluation criteria. The selection criteria are: 1) Site must lie in unused land; 2) Site must be beyond 250 meters of the roads; 3) Site must be within 1200 meters of rivers.

Step 2, use the method discussed in section 2 to represent the needed geospatial data (for this task, including road, river, and landuse) in GML.

The following is the needed spatial data represented in GML/SVG. It includes three layers which can be viewed as the following tables: landuse (id, type, d), road (id, d), and river (id, d).

```xml
<?xml version="1.0" encoding="UTF-8"?>
<FeatureCollection>
  <boundedBy>
    <Envelope>
      <coordinates srsName="...">... </coordinates>
    </Envelope>
  </boundedBy>
  <featureMember>
    <landuse>
      <ID>A</ID>
      <type>unused</type>
      <polygonProperty>
        <Polygon srsName="">
          <outerBoundaryIs><LinearRing><coordinates>...</coordinates></LinearRing>
        </outerBoundaryIs>
      </Polygon>
    </landuse>
  </featureMember>
...
  <featureMember>
    <road>
      <ID>C1</ID>
      <lineStringProperty>
        <LineString srsName="">
          <coordinates>....</coordinates>
        </LineString>
      </lineStringProperty>
    </road>
  </featureMember>
...
  <featureMember>
```
<river>
  <ID>R1</ID>
  <lineStringProperty>
    <LineString srsName="">
      <coordinates>...</coordinates>
    </LineString>
  </lineStringProperty>
</river>

And the corresponding SVG codes are as follow.

<svg viewBox="..." SRS="...">
  <g transform="...">
    <g id="landuse">
      <path id="A" type="unused" d="..."/>
    </g>
    <g id="road">
      <path id="C1" d="..."/>
    </g>
    <g id="river">
      <path id="R1" d="..."/>
    </g>
  </g>
</svg>

Step 3, utilize SESQL to carry out spatial analysis. First, we utilize "Buffer" operator to create buffer for road and river, then use "Intersection" and "Disjoint" operators to carry out overlay operation of landuse, buffer of road and buffer of river. The query sentences are as follows:

CREATE VIEW ribuf AS SELECT id, Buffer(d, 1200) AS buf FROM river;
CREATE VIEW robuf AS SELECT id, Buffer(d, 250) AS buf FROM road;
CREATE VIEW landriver AS
  SELECT Intersection(landuse.d, ribuf.buf) AS temp1 FROM ribuf, landuse
  WHERE landuse.type="unused" AND Disjoint(landuse.d, ribuf.buf)=False;
SELECT Intersection(landriver.temp1, robuf.buf) AS site FROM landriver, robuf
  WHERE Disjoint(landriver.temp1, robuf.buf)=False;
Step 4, result appraisal and explanation. After executing the above sentences, the appropriate site will be highlighted on the SVG map. If users are not satisfied with the result, they can return to step 1.

4. Implementation and case studies

4.1 Implementation

Algorithms of computational geometry can be utilized to develop our spatial operators. As Java has provided some basic computational geometry APIs (Java 2D API), we implement the spatial operators with algorithms of computational geometry and Java 2D API. For the SESQL, we develop a compiler to carry out syntax, sentence, and semantic analysis for SESQL sentences. For the server side implementation (for GML), spatial operators and SESQL compiler are designed as Java servlets. For browser side implementation (for SVG), spatial operators and SESQL compiler are developed as Java applets, and embedded in HTML; DOM APIs of HTML and SVG are used to access SVG document and invoke the Java applets; a user interface is embedded in HTML for inputting SSESQL query sentences (Fig. 6).

In the following section, we design two case studies to evaluate the suggested methods on GML/SVG-based spatial information representation and analysis. As load-balancing between server side and browser side is not our research focus in this paper, we carry out spatial analysis based on the browser side implementation (Java applets).

4.2 case study

In this section, we design a case study to evaluate the suggested tool. The cultivated lands are sensitive resources in a highly populated country like China, especially, in the most quickly developing province of Guangdong. People may consider whether the cultivated lands change with the growth of transportation network. We choose one issue to analyze and discuss: what has happened to cultivated lands with the increasing of the road density (railway, highway, and province-level road). First, we utilize the "intersection" and "length" operators to calculate the total road length for every city, and use "area" operator to calculate every city's area, and then calculate every city's road density. Figure 4 depicts the result of this issue. The result lists the name of cities and their road density at the right low box. Every city is marked with different color according to its road density, and with a statistic bar graph on it.
Figure 4. The cultivated lands change along the railway and main highway in 15km extent

Legend for bar graphs in Figure 4: dark orange for cultivated lands increasing; dark green for cultivated lands decreasing; middle orange for unused land increasing; middle green for unused land decreasing; light orange for residential area increasing; dark grey for residential area decreasing

5. Conclusion

This paper tries to introduce spatial analysis functions to XML/GML/SVG based WebGIS. We focus on two issues: GML (server side)/SVG (browser side) based spatial information representation, spatial operators and spatial query language for both server side (for GML) and browser side (for SVG). For the first issue, based on spatial data modeling, we propose a GML/SVG-based spatial information representation model to represent the needed spatial data in GML/SVG on the server/browser side. Furthermore, we design and implement some spatial operators, and integrate them into SESQL language, and then discuss how to use SESQL for spatial analysis in Web environment. In order to evaluate our suggested methods, this paper designs and implements two case studies. The results of the case study show that our proposed methods are feasible and operable to support spatial information analysis in Web environment.

Our next step is to consider the load-balancing technology between server side and browser side for SVG-based spatial analysis. We are also interested in the problem of completeness of the
spatial operators for spatial analysis task.

Acknowledgements

Huang's work is supported by supported by the SemWay project (funded by Austrian BMVIT) and the UCPNavi project (funded by Austrian FWF). We also would like to give acknowledgement to the support of the project, by Science and Technology Department, Guangdong Province, for it continued financial support in 3 stages (2002B32101, 2004B32501001, 2005B30801006) of this research project.

Bibliography


