Toward GML-based Online GIS

GeoWeb 2006
July 24-28 Vancouver BC Canada

Dongpo Deng, Chia-Hsin Huang, Tyng-Ruey Chuang
{dongpo, jashing, trc}@iis.sinica.edu.tw
Agenda

• Motivations
• Objectives
• GML-native processors
• Performance analysis
• Prototype
• Conclusions
Motivations and Objects
Is everything connect to geospatial reference?

• A frequently repeated factoid is 80% of all digital data generated today include geospatial referencing (e.g. geographic coordinates, address, postal codes, etc.)
The increasing geospatial data

- With the proliferation of geospatial data over Internet, there is an increasing demand for efficient data integration solutions that allow interoperation of massive repositories of heterogeneous spatial data and metadata.
Geographic model will be complicated

- The nature of geographic phenomena is complicated, diversified and multi-scaled.
- Geographic model, used to abstract geographic objects from real world, is gradually sophisticated.
The limitations of relational data models

• Often geographic data models are mapped to the relational database model for data storage, operation, analysis, and visualization.

• However, because of the limitations of relational data model, we may not be able to express the intrinsic characteristics of geospatial data in a natural and effective way.
GML is a good descriptor of geographic phenomena

- GML 3 includes 28 core schemas.
- There are over 1000 tags can be identified in core schemas of GML 3.
- Over 10 thousands lines of code have been defined in core schema of GML 3.
GML acts as the important role of Online GIS

- GML has been defined to
  - transporting geospatial information.
  - modeling geospatial information.
  - store data formation of geospatial information.
The issues of using GML

- GML, providing rich vocabulary and flexible document structure, has been considered as an effective approach to express increasingly complicated geospatial data, as well as non-geospatial data.
- The exist IT methods are not always suitable for processing GML.
  - DOM and SAX
  - XQuery and XPath
Objects

• To propose two efficient GML-native query processors.
• To enhanced technique is able to produce a reduce-sized GML document fragment, using both non-geospatial and geospatial query constraints.
• To speed up query process over large GML documents.
• To build up a GML-based Web GIS, alone with a geospatial query interface and a SVG map navigator.
The GML-native processors
The conventional ways - RMDBS

- Often the conventional ways use the RDBMS as their back-end DB
  - PostgreSQL with PostGIS
  - MySQL with spatial extension
  - Oracle spatial
A novel ways – GML as back-end DB

- GeoQuery – a GML query engine
  - extend the Saxon Java XQuery processor by calling spatial functions libraries of JTS (Java Topology Suite)
- GeoSAX -- A Streaming GML Parser
  - extend the Sun’s SAX parser to support the spatial functions.
<Streams>
  <streamMembers>
    <Rivers>
      <riversMembers>
        <FootPrint id="2100100000-11">
          <name> ... </name>
          <length> ... </length>
          <multiCenterLineOf>
            <MultiCurve>
              <curveMembers>
                <LineString>
                  <coordinates>302745.435072,2442803.300075 ... 302882.866766,2442167.96039</coordinates>
                </LineString>
                </curveMembers>
              </MultiCurve>
              <multiCenterLineOf>
            </FootPrint>
          </riversMembers>
        </Streams>
      </Rivers>
      </streamMembers>
    </generalRoadMembers>
  </FootPrint id="4230904000-31">
    <name> ... </name>
    <length> ... </length>
    <multiCenterLineOf>
      <MultiCurve>
        <curveMembers>
          <LineString>
            <coordinates>301219.608538,2442423.503538 ... 306235.358961,2434884.578507</coordinates>
          </LineString>
          <MultiCurve>
            <multiCenterLineOf>
          </FootPrint>
        </generalRoadMembers>
      </Rivers>
    </generalRoad>
  </roadwaysMembers>
The result of querying

(a) A road.

(b) A river.

(c) The results of buffering the intersection of the road and the river.

(d) Combine and recolor (a), (b), and (c) in a SVG map.
XQuery expression

```
1 declare namespace my="http://www.sinica.edu.tw/";
2 declare namespace gml="java:GML.XQGeoExtensions";
3 declare namespace svg="java:GML.XQSVGExtensions";

5 declare function my:get_geo1() as element() {
6   for $var1 in doc("lanyu.xml")//Rivers//FootPrint[@id="2100100000-11"]
7     return <result1>{$var1}</result1> ;

9 declare function my:get_geo2() as element() {
10    for $var1 in doc("lanyu.xml")//Roadways//FootPrint[@id="4230904000-31"]
11      return <result1>{$var1}</result1> ;

13 svg:GML2SVG(gml:Buffer(gml:Intersection(
14    my:get_geo1(), my:get_geo2()), 50))

16 {: This is a comment section. Instructions down below select all buildings bounded by a viewBox.
17    declare function my:get_building() as element() {
18      <result1>{
19        for $var1 in doc("lanyu.xml")//Buildings//FootPrint
20          return $var1
21      } </result1> ;
22
23    svg:GML2SVG(gml:queryByBox(299000, 2430000, 312000, 2445000, my:get_buildings()))
24  The end of the comment section. :)
25```
The problems will occur in the GML solution

- The processor will not be work if the GML document is large in size.
Prefiltering Framework - Idea

- **Rapidly** and **approximately** execute the query to extract candidate fragments.
  - Rapidly: Consider only certain subset of the steps in the given query
  - Approximately: Consider only easy-to-test structural properties of the chosen steps, namely relationships between tag positions.
  - See details in http://www.iis.sinica.edu.tw/~jashing/prefiltering
An Example

- XPath expression: //A//E
- Answer: sub-trees rooted by \((E_8, E_{15})\) and \((E_9, E_{14})\)
The components of the GML-based Online GIS

- Indexing
  - Geospatial Plug-in Module (GPM)
  - XML Pre-filtering Technique (XPT)
  - Spatial extension

- Query Engine
  - XQuery
  - SAX

- Data storage
  - XML

- DOM

- Streaming

Geo-enhanced GML processor

Institute of Information Science, Academia Sinica
Geospatial Plug-in Module (GPM) for the XML Pre-filter

- Geospatial Plug-in Module (GPM) is developed for the XML pre-filtering technique to perform geospatial filtering functionality.
- *GPM* can index the boundary of each geographical feature in the documents and provides an intersection operation to query indexed features.
GPXQuery

- **GPXQuery** is an efficient geospatial supported query engine for manipulating high voluminous GML documents by integrating the *GeoXQuery* with the XML prefiltering technique containing a geospatial plugin module.
- The XML prefiltering technique is aimed to be designed as a non-intrusive software module.
- That means that existing XML tools/applications are able to integrate the prefiltering technique with minimal modifications and need not aware of its mechanisms.
The XML prefiltering technique provides an interactive XML streaming parser that is able to parse an XML documents in a random access manner.

Different to the GPXQuery query engine, GPSAX is a GML-native streaming parser.

Users need to employ a query algorithm to calculate exact answers. That is, GPSAX is more suitable for answering simple queries, which contain only reverse axes free XPath expressions and/or a viewbox constrains.
The system architecture of the GML-based Online GIS

- In the **GPXQuery**, the XQuery processor only sends XQuery expressions to the XML prefiltering module for generating candidate-set GML documents that are actually used to evaluate exact matched geographic features for the SVG-based map navigator.

- Note that, the XQuery processor also transforms the matched features into SVG elements by calling an external SVG transformation library.
Performance analysis
The Execution Environment and Datasets

- Four GML processors
  - GeoXQuery
  - GPXQuery
  - GeoSAX
  - GPSAX

- Two datasets
  - 1.2 GB GML document (the Taipei city)
  - a 152 MB GML document (the Sinyi area)

- Setup
  - an Intel Pentium-4 PC running at 2.53 GHz with 1GB DDR-RAM,
  - a 120GB EIDE hard disk,
  - the MS Windows 2000 server OS.
  - Java 2 (Standard Edition V.1.4.2).
The Dataset

Large dataset—Taipei, 1,147 MB

Small dataset – Shinyi, 152 MB
Querying by a feature ID (small dataset)

<table>
<thead>
<tr>
<th></th>
<th>Run Time (s)</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>GXQ</td>
<td>42</td>
<td>1065</td>
</tr>
<tr>
<td>GSAX</td>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td>gPFXQ</td>
<td>6.39</td>
<td>179</td>
</tr>
<tr>
<td>gPFSAX</td>
<td>5.09</td>
<td>23</td>
</tr>
<tr>
<td>xPFXQ</td>
<td>4.89</td>
<td>305</td>
</tr>
<tr>
<td>xPFSAX</td>
<td>3.54</td>
<td>39</td>
</tr>
</tbody>
</table>

Shinyi-Query a feature
Querying by a feature ID (large dataset)

Run Time (s)  Memory

<table>
<thead>
<tr>
<th></th>
<th>Run Time (s)</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>GXQ</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>GSAX</td>
<td>233</td>
<td>2</td>
</tr>
<tr>
<td>gPFXQ</td>
<td>38</td>
<td>179</td>
</tr>
<tr>
<td>gPFSAX</td>
<td>36.18</td>
<td>298</td>
</tr>
<tr>
<td>xPFXQ</td>
<td>32</td>
<td>305</td>
</tr>
<tr>
<td>xPFSAX</td>
<td>29.12</td>
<td>305</td>
</tr>
</tbody>
</table>
Query by a viewbox (small dataset)

<table>
<thead>
<tr>
<th></th>
<th>Run Time (s)</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>GXQ</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>GSAX</td>
<td>158</td>
<td>18</td>
</tr>
<tr>
<td>gPFXQ</td>
<td>196</td>
<td>958</td>
</tr>
<tr>
<td>gPFSAX</td>
<td>512</td>
<td>57</td>
</tr>
<tr>
<td>xPFXQ</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>xPFSAX</td>
<td>1343</td>
<td>57</td>
</tr>
</tbody>
</table>
Query by a viewbox (large dataset)

<table>
<thead>
<tr>
<th></th>
<th>Run Time (s)</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>GXQ</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>GSAX</td>
<td>1008</td>
<td>243</td>
</tr>
<tr>
<td>gPFXQ</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>gPFSAX</td>
<td>5968</td>
<td>767</td>
</tr>
<tr>
<td>xPFXQ</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>xPFSAX</td>
<td>13802</td>
<td>489</td>
</tr>
</tbody>
</table>

Taipei - Query by a viewbox

Run Time (s) vs Memory
Querying by a layer and a viewbox (small dataset)

<table>
<thead>
<tr>
<th></th>
<th>Run Time (s)</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>GXQ</td>
<td>21</td>
<td>655</td>
</tr>
<tr>
<td>GSAX</td>
<td>57</td>
<td>2</td>
</tr>
<tr>
<td>gPFXQ</td>
<td>2.37</td>
<td>39</td>
</tr>
<tr>
<td>gPFSAX</td>
<td>1.57</td>
<td>39</td>
</tr>
<tr>
<td>xPFXQ</td>
<td>2.09</td>
<td>39</td>
</tr>
<tr>
<td>xPFSAX</td>
<td>1.28</td>
<td>39</td>
</tr>
</tbody>
</table>
Querying by a layer and a viewbox (small dataset)

<table>
<thead>
<tr>
<th></th>
<th>Run Time (s)</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>GXQ</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>GSAX</td>
<td>242</td>
<td>2</td>
</tr>
<tr>
<td>gPFXQ</td>
<td>19</td>
<td>298</td>
</tr>
<tr>
<td>gPFSAX</td>
<td>11.39</td>
<td>298</td>
</tr>
<tr>
<td>xPFXQ</td>
<td>9.89</td>
<td>305</td>
</tr>
<tr>
<td>xPFSAX</td>
<td>8.937</td>
<td>305</td>
</tr>
</tbody>
</table>
Performance Characteristics Analysis

• SAX-based methods (*GeoSAX* and *GPSAX*) are more suitable for dealing with simple queries against huge documents since they consume relatively less memory as compared to XQuery-based methods (*GeoXQuery* and *GPXQuery*).

• The geospatial plugin module of the prefiltering technique can efficiently filter out uninteresting geographic features with a little additional effort.

• The prefiltering technique may lower resource consumption, particularly about memory space usage and disk access time.
Prototype
The snapshots of the GML-based Web GIS
The snapshots of the GML-based Web GIS
CONCLUSIONS

- A prefiltering technique for efficient XML document processing is applied to speed up geospatial operations over large GML documents.
- The *GPXQuery* and *GPSAX* processors have been employed to develop a GML-based Online GIS, along with a geospatial query interface and a SVG map navigator.
- Experimental results show that the XML prefiltering technique significantly improves the performance of both the *GPXQuery* and *GPSAX* processors. They either lower query execution time or reduce memory space consumption.
CONCLUSIONS

• The GML-based GIS reported is still impractical for dealing with gigabyte-sized geospatial datasets. We suggest the following areas for improvement.
  – A more efficient index management system or data structure would be useful to lower the cost of retrieving the indexes.
  – There is no efficient way to update XML documents incrementally.
  – As GML can represent both geospatial and non-geospatial information in a single document very easily, it does raise the issue of how better to process documents of this nature.
Thank you for your attention!!

Dongpo Deng
dongpo@iis.sinica.edu.tw