Multilevel Updating Method of Three-Dimensional Spatial Database

Presented By: Tristram Taylor
SE521
Written By: Yangting Liu, Gang Liu, Zhenwen He, Zhengping Weng

From: China University of Geosciences

For: 2010 18th International Conference on Geoinformatics
Contents

• What is a Spatial Database?
• Introduction
• The Analysis of Features of 3D Spatial Data Update
• Multilevel Data Updating Method and Its Application
• Test Results and Analysis
• Conclusion
What is a Spatial Database?

- **Spatial Database** - is a database that is optimized to store and query data that is related to objects in space, including points, lines and polygons.
- While typical databases can understand various numeric and character types of data, additional functionality needs to be added for databases to process spatial data types.
- **Spatial Queries** - where predicates for selection deal with spatial parameters.
- **Spatial Analysis Operations** - used to uncover spatial relationships within and among mapped data layers.
Introduction

• Utilizes ArcSDE (Spatial Data Engine)
• In their system, they incorporate the SDO_Geometry, which is an abstract data type provided by Oracle Spatial.
• They incorporate Oracle C++ Calling Interface (OCCI) instead of ODBC or JDBC to access and update the spatial data.
• This paper introduces the multilevel spatial data updating method to schedule and update massive 3D spatial data, and improve the efficiency, interaction and integrity of spatial databases.
The Analysis of Features of 3D Spatial Data Update
Features of 3D Spatial Data

• SDO_Geometry is composed of several parameters indicating geometry type (point, plane, line, etc.), dimension, and an array with the x,y,z coordinates.

• Traditional 2D data poses its own challenges:
  o Spatial features
  o Unstructured features
  o Spatial relationship features
  o Classification and coding features
  o Massive data
  o Non-standard algebraic operations

• 3D data poses those challenges as well as:
  o Semantics of aboveground and underground
  o Multiplicity of geometry and texture properties
  o Huge quantity of data
  o Complicated relationships between models.
Difficulties During 3D Data Updating

• Traditionally, 3D spatial data is modeled using the general model.
  o Integrates aboveground and underground.
  o Geological bodies, pipelines, roads, discrete buildings, and other object differ significantly is aspects of:
    • Geometry
    • Appearance
    • Semantics
    • Topology
  o Massive 3D spatial data has multilevel and multi-category features
    • These features cause users to ignore the integrity of the spatial entity
    • This leads to decreasing efficiency.

• This project utilizes an object oriented model to improve updating efficiency.
Multilevel Data Updating Method and Its Application
Multilevel Data Updating Method

• The object-oriented model treats the object entity as a unit to store and manage spatial geometric elements and emphasizes the integrity of the spatial entity.

• Utilizes existing topological relationships between entities to reduce the input quantity of data while holding the basic features of spatial entities.

• Local updating of OO data models is more convenient.

• Inserting, deleting, or updating one entity will not affect the other entities, so efficiency is higher.
Multilevel Structure of Spatial Data

• In common GIS systems spatial data is usually expressed by using a layered structure, so it’s convenient to manipulate and store features, but the elements have obvious hierarchy on the space used.

• This means combining features of structured, unstructured, and OO features.

• This project adopts the following logic model:
Table 1: Logic Model of Feature Objects

<table>
<thead>
<tr>
<th>Column Name</th>
<th>Data Type</th>
<th>Null-able</th>
</tr>
</thead>
<tbody>
<tr>
<td>FID</td>
<td>NUMBER(20,0)</td>
<td>NO</td>
</tr>
<tr>
<td>FNAME</td>
<td>VARCHAR2(50 BYTE)</td>
<td>NO</td>
</tr>
<tr>
<td>FSTATE</td>
<td>NUMBER(10,0)</td>
<td>YES</td>
</tr>
<tr>
<td>FLODS</td>
<td>NUMBER(1,0)</td>
<td>NO</td>
</tr>
<tr>
<td>FBOUND</td>
<td>G3D_TYPE_BOUND</td>
<td>NO</td>
</tr>
<tr>
<td>FORCLBOUND</td>
<td>SDO_GEOMETRY</td>
<td>YES</td>
</tr>
<tr>
<td>FCREATETIME</td>
<td>NUMBER(20,0)</td>
<td>NO</td>
</tr>
<tr>
<td>FUPDATETIME</td>
<td>NUMBER(20,0)</td>
<td>NO</td>
</tr>
<tr>
<td>FLODINFO</td>
<td>BLOB</td>
<td>YES</td>
</tr>
<tr>
<td>FGEOM</td>
<td>BLOB</td>
<td>NO</td>
</tr>
<tr>
<td>FTOPO</td>
<td>BLOB</td>
<td>YES</td>
</tr>
<tr>
<td>R_LEN1</td>
<td>NUMBER(10,3)</td>
<td>NO</td>
</tr>
<tr>
<td>......</td>
<td>......</td>
<td>......</td>
</tr>
</tbody>
</table>
Updating Process of BLOB field

1. Clear the content of BLOB field in database
2. Create buffer area
3. Get BLOB content from updating feature object and write it into buffer
4. Read the BLOB field from database again
5. Compute the buffer and that to be inputted BLOB each time
6. Write buffer into BLOB by section
7. Is it buffer’s tail part?
   - Yes: Execute the updating statement
   - No: Repeat from step 2
The updating of SDO_Geometry field

- In the object-relational model of Spatial the description of the geometry of spatial data entities is stored in the object type SDO_Geometry, which is defined in tables by users.
- SDO_Geometry is used to increase the speed of dispatching and querying data by storing the bounding box information of the spatial data
  - Integrating aboveground and underground information to establish a spatial index.
- SDO_GTYPE – type of spatial entity (fixed)
- SDO_ELEM_INFO – (fixed) interprets the coordinate values defined by:
  - SDO_ORDINATES – coordinate value
    - Defined by variable length array
    - Type NUMBER to store coordinate value and form the boundary
- The process of updating SDO_GTYPE and SDO_ELEM_INFO is mainly to update SDO_ORDINATES.
Updating **SDO_GEOMETRY**

Update feature(ID, Feature) where ID is the feature class’ ID and Feature is the updating object:

Begin

...... //update the attribute data
Bound = Feature.Get bound(); //obtain updated information of bounding box
SDO_GEOMETRY sdo_geometry; //establish new geometry object
NUMBER gtype = 3008;
Vector<NUMBER> elem_info;
elem_info.pushback (1);
elem_info.pushback (1007);
elem_info.pushback (3); //establish sdo_gtype and sdo_elem_info
sdo_geometry.Set std_gtype (gtype);
sdo_geometry.Set std_elem_info (elem_info);
sdo_geometry.Set std_ordinates (bound);
Set object (sdo_geometry); //set information of SDO_GEOMETRY object
...... //update un-structured data
End
The Design of the Adaptive Import Algorithm of Attribute Data

- Three types of file formats:
  - Txt text
  - Excel sheet
  - Access database

- Users select input data to judge if the attribute structure matches the existing one currently in database.

- If it’s a match, user selects input model
  - Item-by-item – Attribute item has to be chosen
  - Entire model – quantity consistency must be calculated

- If consistent, attribute data will be imported to database successfully.
The Adaptive Import Algorithm of Attribute Data

1. Attribute Data
2. If the attribute structure matched
   - Yes
   - No (To select again)
3. Selecting the import model
   - Item by item
   - Entire model
4. Is the number of feature objects consistent?
   - No
   - Yes
5. To continue to select
   - No
   - Yes (Complete?)
6. Importing attribute data
The Design of the Update Module in the Application Layer

The main functions of the update module of the 3D Spatial Data Management Tool in the application layer are:

- Insert
- Modify
- Batch replace the feature objects
- Adaptive import attribute data in multi-mode method
Design of the Update Module Interface

3D Spatial Data Management Tool (CGV3dCatalog Interface)

- CGV3dUpdateDataManager
  - GV3dImportProp(int type):void
  - GV3dInsertFeature(FeatureSharedPtr pFeature):void
  - GV3dModifyFeature(FeatureSharedPtr pFeature):void
  - GV3dDeleteFeature(FeatureSharedPtr pFeature):void

CGV3dManagerLib

- ImportTxtDlg Class
  - ReadFile
  - ImportSingle
  - ImportAll
  - UpdateDialog
  - ImportTxtSave

- ImportExcelDlg Class
  - ReadFile
  - ImportSingle
  - ImportAll
  - UpdateDialog
  - ImportExcelSave

- ImportAccessDlg Class
  - ReadFile
  - ImportSingle
  - ImportAll
  - UpdateDialog
  - ImportAccessSave

- UpdateDataDlg Class
  - DeleteData
  - ModifyData
  - ReplaceData
  - SaveData
  - SaveData
  - UpdateDialog

CGV3dDataLib

- GV3dreadFacade(Proxy* proxy):BOOL
- GV3dreadFeatures(FCLSID id, Proxy* proxy):BOOL
- GV3dmodifyFeatures(FCLSID id, vector<FID> fid, vector<std::string> dataset):BOOL
- GV3dimportPropData(FCLSID id, vector<FID> fid, vector<std::string> propdata):BOOL

3D Spatial Data Engine (SDE) (Uniform Access Interface)
Steps For Updating Data

• The data management tools in client-side send request to uniform access interface of SDE, including the information of project, feature class and feature objects;

• To see if the awaiting updating data exist in cache. If it exists, dispatch them to application layer, otherwise, connect the database and dispatch data from database.

• The data is updated in client side and transmits updated data to be warehoused requires by users, then, the updated data waits to be loaded into database while...

• The consistency and integrity of data are checked.

• If it mismatches the condition, the error message is returned to client side, then update is restarted

• If matches the condition, updated data will be loaded into database, the updated feature objects and the operations are recorded by database log, which can be consulted by users.
Flow Chart of Updating Data Procedure

1. Start
   - transmision requires

2. Is awaiting updating data existing in cache?
   - Yes
     - Update awaiting updating data (integration processing)
   - No
     - Dispatching data from database

3. Checking consistency and integrity of data
   - mismatch condition
   - match condition

4. Recording by database log

5. Data imported to database
Importing Attribute Data with DTA Management Tool
Test Results and Analysis
The test

- The update module of 3D spatial data management tool "GV3dCatalog" is developed.
- It is used for aboveground and underground data.
- The Oracle_11g is adopted as the background database.
- Visual C++ is used as the developing language.
- The tool supplies the functions of warehousing, browsing, updating, and exporting aboveground and underground data.
Importing Original Data into Database

The original data is imported into the database. Element type of commercial building is selected.
Selecting Feature Object

Select one building to delete, and the others to modify in batch model.
Deleting the Appointed Feature Object

During update the element object to be modified is checked to see if it matches consistency and integration of data.
Conclusion

“In order to solve the issue of complexity, integrity and efficiency of spatial data, the OCCI technology and object relation model supported by Oracle Spatial is adopted for three dimensional spatial data multilevel updating and its realization based on research and analysis on spatial data engine theory and technology. With application of this method in three dimensional spatial data management tool for scheduling and updating the massive spatial data, the updating efficiency is improved significantly. Meanwhile, how to coordinate the relationship between updating spatial information and visualization environment, spatial analysis operation, is still to be considered in future.”
Questions?
Sources

- Multilevel Updating Method of Three-dimensional Spatial Database: Yanting Liu, Gang Liu*, Zhenwen He, Zhengping Weng; Faculty of Computer Science and Technology, China University of Geosciences(Wuhan); Wuhan, China
