AN EFFICIENT DATA MANAGEMENT APPROACH FOR LARGE CYBERCITY GIS

Qing Zhu, Xuefeng Yao, Duo Huang, Yetin Zhang

National Key Lab of Information Engineering for Remote Sensing, Surveying and Mapping,
Wuhan University 430079, P.R. China
E-mail: zhuq@rcgis.wtusm.edu.cn

Commission IV, WG IV/4

KEY WORDS: CyberCity, Spatial Database Engine, R+_trees, Object-relational Database.

ABSTRACT:

CyberCity facilitates the processes of urban planning, communication system design, control and decision making, tourism, etc. However, the high efficient database management has become a bottleneck of CyberCity applications. This paper proposes an efficient approach to manage the integrated databases of large CyberCity. This approach consists of following three schemes: At first, a special R+_tree index was designed to accelerate spatial retrieving. The spatial index of CyberCity includes three different types of indexes, i.e. 3D object index, DEM index and image index. The whole city is divided into rectangular regions, and geometries are then classified into the regions by the center of the rectangular bounding box of each geometry. We call it a R+_tree index because among the bounding boxes of local regions has no intersection. And among all the leaf nodes of the R+_tree (geometry records) there is no repetition. Secondly, different data compression algorithms are adopted to compress the digital elevation models, 3D vector models and images, such as LZ77 lossless compression algorithm for compression of vector data and JPEG compression algorithms for texture images. Thirdly, in order to communicate with the Oracle8i database, the CyberCity GIS spatial database engine (SDE) is designed. At last, based on the SDE prototype a case study is presented. It is hopeful to satisfy the requirement of real time applications of CyberCity GIS. It is proved to establish the efficient spatial index and to adopt proper compression methods as well as to extend the data retrieve strategy of commercial ORDBMS are significant for large CyberCity GIS.

1. INTRODUCTION

The term 'CyberCity' is used to represent the virtual representation of a city that enables a person to explore and interact, in cyberspace, with the vast amounts of environmental and cultural information gathered about the city (Gruber, 1999; Zhu et al., 2000). It facilitates the processes of urban planning, communication system design, control and decision making, tourism, etc. The CyberCity not only shows data in three dimensions, i.e. the 3D city models in most cases, but also presents photorealistic surface description. Therefore, the description of surface character and material parameters, including geometry, photo texture and additional information, are the contents of a CyberCity database, and this would result in the CyberCity database of an entire city facing an amount of some hundred gigabytes of data. Even very complex and large scenes are organized as a collection of files. The access efficiency of the file system is too low to afford the development of the CyberCity, the database management system has been required to manage the huge urban data of CyberCity. The high efficient database management has therefore become a bottleneck of CyberCity applications (Kofler *et al.* , 1996). The relational databases have been established as the most important database technology during the last 30 years, which are widely used in nowadays 2D geographic information systems (GISs). But to the 3D GIS, the disadvantage of relational database has become more and more obvious. Because the CyberCity has various data types, which include many numeric item data, string data, large unstructured data such as texture images of surfaces, structured vector data of 3D objects, and large size of terrain orthoimages. The object-oriented database is still not mature to organize all such kinds of data and their relations, so we choose the object-relational database, such as Oracle 8i, which has the mechanism to organize the individual data into some kind of objects, it reserves the advantage of relational database and adapts the idea of object-oriented into it. It can not only handle the complex relations of the 3D data, but also is good at forming the data into objects which should be treated as a whole logically. On the other hand, the access efficiency of Oracle8i is optimistic and more suitable for CyberCity GIS.

This paper is organized as following: after briefly introducing the architecture of CyberCity geographic information system (CCGIS), the R+_tree spatial index method, data compression algorithms and spatial data engine (SDE) based on object-relational database Oracle8i are proposed respectively. At last, based on the

SDE prototype a case study and conclusion will be presented.

2. THE ARCHITECTURE OF CCGIS

The construction of CyberCity requires 3D city models with realistic material or texture description and related attributes and to provide spatial query and analysis functions. There are usually three basic models involved in the CCGIS, including Digital Elevation Model (DEM), Digital Orthoimage Map (DOM) and 3D Object Models like buildings, roads, pipeline etc. Unlike the true 3D GISs needed in geological or oceanic applications, the main objective of current CyberCity GIS is to deal with the 3D objects as surface models (sometimes they are considered as 2.5D models). It does not matter how the 3D real world is mapped into spatial databases, the main point is the availability of 2.5D and 3D capability in GIS what should be realized in an efficient and robust manner (Fritsch, 1996). As shown in figure 1, the architecture of CCGIS consists of three layers, the top layer is the 3D real time applications such as 3D dynamic interactive visualization (Zhu, 1998). The data access depends on the mid layer, i.e. the CCGIS-SDE, which majors in the communication with the bottom layer, i.e. the database management system (DBMS), which manages the data of CCGIS in database. The database is an object relational database, i.e. Oracle 8.1.6. This SDE extends the data retrieve functionality, and provides an efficient data management approach including special R+_tree index, data compression schemes and fast access strategy as mentioned below.

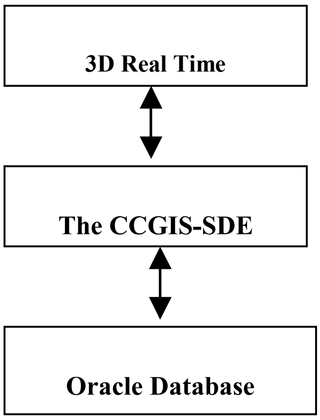


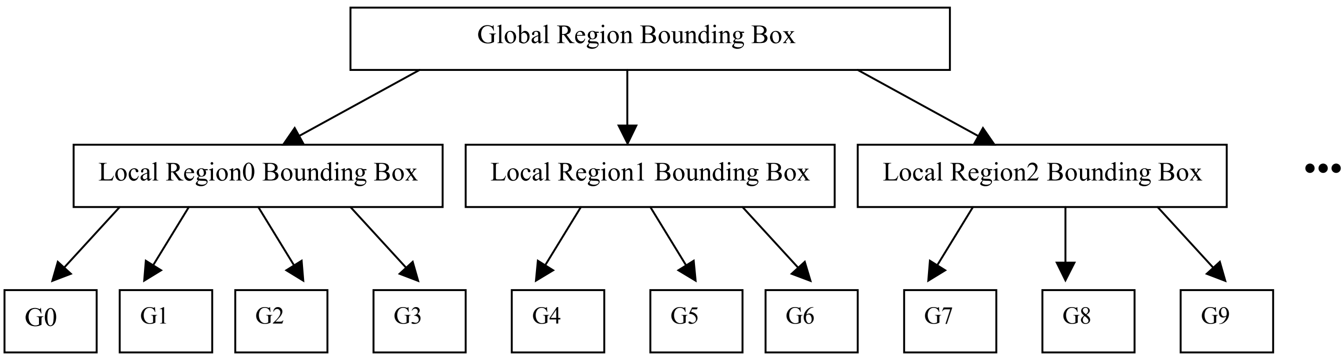
Figure 1 The architecture of CCGIS

3. R+_TREE INDEX FOR FAST DATA RETRIEVE

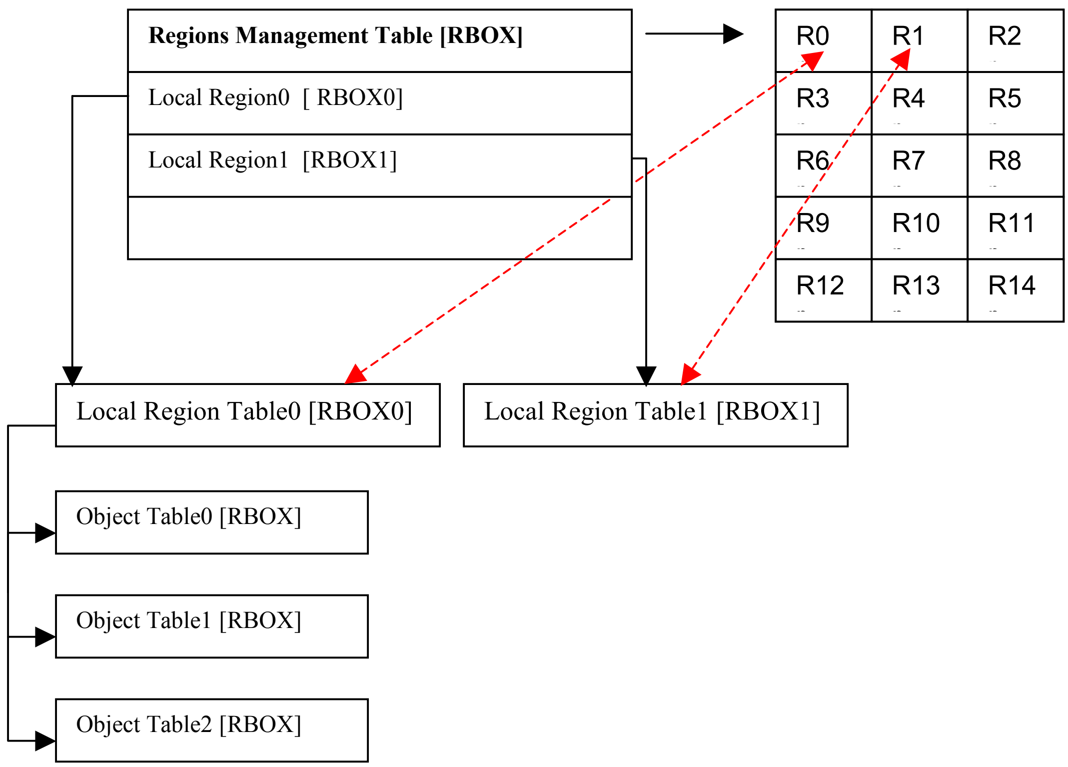
As mentioned above, the data volume of a whole city is huge, if we want to retrieve data from the vast database without spatial index, this process would be very time-consuming, so proper spatial index must be built up to accelerate the data locating. The authors propose to use R+_tree as the basis for a CyberCity GIS to develop a data structure to spatially organize large amounts of 3D data. Herein we call it a R+_tree index because among the bounding boxes of local regions has no intersection. And among all the leaf nodes of the R+_tree (geometry records) there is no repetition, in order to keep the data consistence of the database, one geometry in one city is exclusive. As shown in figure 2, the spatial index of CyberCity includes three different types of indexes, i.e. 3D object index, DEM index and image index. The whole city is divided into rectangular regions, and geometries are then classified into the regions by the center of the rectangular bounding box of each geometry. A global table which manages all the rectangular regions of a city is established to locate the local regions with the bounding box information. Simultaneously several management tables are set up according to the number of local region, each local region has a management table to manage all the geometry tables in this rectangular region. To insert geometries into the geometry table, the proper local region and proper geometry table based on the center of the rectangular bounding box of each geometry is selected.

3.1 Management of large DEM

A so-called gridded manner algorithm was designed to handle large DEM as raster data. In gridded manner method, tile is the logical subdivision of the whole region and server as the foundation of data index. Block is the basic data storage and access unit. For 3D real time visualization applications, block is also the basic rendering and LOD (levels of detail) process unit. When blocks are stored every block is aligned with each other in tile and neighbour block and an overlay line data in four directions. Cell is the minimum subdivision of DEM data. It consists four original elevation points. All block within a specific grid interval constitute a layer. In another word layer is the DEM dataset of the same grid interval at a certain scale or resolution. LOD is an important concept. As far as DEM data is concerned, LOD relates to various grid interval DEM data (DEM pyramid). A large grid interval layer is created from the neighbouring minor grid interval layer. All layers have the same structure that is composed of blocks.
In object-relational DBMS, one layer data is stored in one table and the friend type of DEM data stream is large object binary (Lob). In modern DBMS, Lob is a popular data type. Though Lob is an unstructured data type, it can be trimed or appended arbitrarily and can store up to 4GB in some DBMS. So Lob is convenient to store large unstructured data, for instance, the image and video. In DEM layer table an elevation value in Lob field can consist of 2,3 or 4 bytes. When source data is submitted to DEM database, it is decomposed to blocks and each of them is indexed according to database original point and correspond (x, y) block width and height. This process



(a) The concept of R+_tree



(b) The table structure

Figure 2 The R+_tree index

creates a series of float-formatted elevation data stream. Each block stream is the minimum data storage unit in Lob field. After all source data is stored in DEM database the initial DEM layer table is created.

The most advantage of this kind of DEM database is that the DEM data can be accessed arbitrarily based on block unit. Because of the limited data size of a block, the gridded manner means that we can roam in an unlimited DEM data freely, irrespective of region area or data size.

3.2 Management of large DOM

The process of DOM data is similar with that of DEM. Both the original points, the block structures and the layer structures of DOM database are identical with DEM's. It is apparent that DOM database is bound up with DEM database. These features result in the same query scope in DEM database can get the DOM data in the same region within DOM database. Because the resolution of the orthoimage is usually higher than that of DEM at the same layer, the data volumes of each block at the same layer are then quite different. For example, a DEM block may consists of 1K nodes as 4K bytes elevation values, but a DOM block may involve 64K pixels as 196K bytes RGB color values.

3.3 Management of large 3D object models

The 3D vector objects are classified into five basic types, i.e., the point objects, line objects, surface objects, body objects and group objects. Usually we treat the trees, streetlamps etc as point objects, they always have an anchor point. The pipelines and the wires are usually treated as line objects, which are looked as polylines abstractly, and always organized as point string. The surface objects include the roads and the waterways. The last is the main body of urban data, the group objects includes all kinds of buildings as the body objects generally are treated as the element of group objects, one group object represent one building, it can be composed of one to several body objects. We use the five types to organize the whole 3D objects. In a city scale, the data amounts to several hundred Gigabytes, two kinds of tables are therefore setup to organize the whole data of the five types. One is the geometry table, and another one is the manager table to manage the geometry tables. The large unstructured data and huge structured vector data of 3D objects are compressed and stored in Blob, each Blob item can hold 4-Gigabyte data.

4. DATA COMPRESSION SCHEMES

Because we have to manage large amount of data in CyberCity applications, even the vector data of 3D objects (buildings, roads, pipelines, etc.) still accounts for large percents. An obvious way to reduce the size of data is compression, compression saves a lot of disk space at the server and reduces network load tremendously. On the other hand, the data must be decompressed at the client with high efficiency. Especially, for vector data the compression would not result in any loss, but for image just without visible loss of quality is ok. Therefore, we choose LZ77 compression algorithm for vector data, the LZ77 is a dictionary compression algorithm without loss. The zip proportion is about 1 : 10 when it is used to compress text data, but when we use this algorithm on the vector data, the zip proportion is about 1 : 8, and the speed of unzip is very fast as shown in table 1. We call the coded bits instant-code that means we can decode the bits by the flag stored in it after we get it. At the same time, some loss compression algorithms are used for texture images, for example the JPEG and wavelet methods are employed.

Table 1 the statistics of vector data compression

Buildings	Bodys	Data size(Bytes)	Coded data size(Bytes)	Decode time(s)
1	2	28768	3940	0.003496
10	24	112744	15349	0.017157
100	238	1138569	142321	0.155967

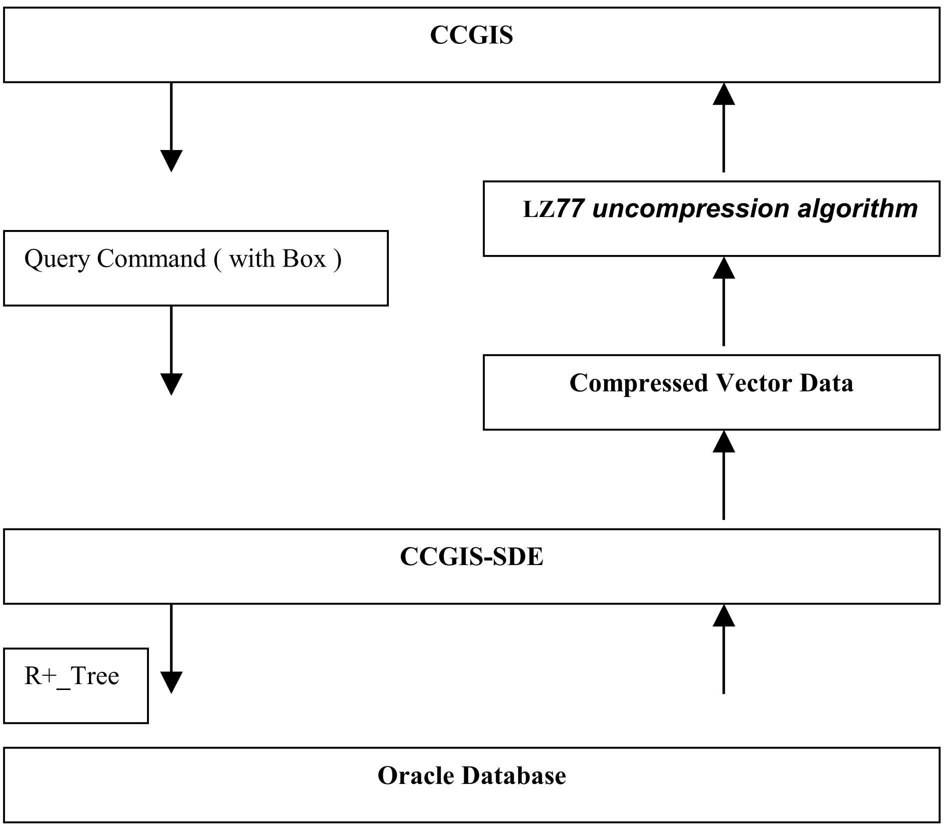


Figure3 The fetching process of 3D objects

5. SPATIAL DATA ENGINE

In order to communicate with the Oracle8i database, the CyberCity GIS needs spatial database engine (SDE) to improve the performance of data access. As shown in figure 3, to fetch the vector data, a message is sent to SDE with the bounding box of query region, SDE firstly finds all the local regions the query refers, then locates the specific geometries in the geometry table through the special R+_tree index, but only the compressed data is obtained from database hereto. Of course, the real time decoding algorithms are provided for further process.

6. CASE STUDY

Based on the Oracle 8.16, a SDE prototype is designed and used for an experiment. At the Oracle sever (Pentium III, 600Mhzs, 100Mbps network), the whole number of buildings is more than 100,000, about 2M bytes of 200 buildings' 3D vector geometric data and texture images can be accessed in 0.125s from server to client. It is hopeful to satisfy the requirement of real time 3D visualization applications of CyberCity GIS. It is proved to establish the efficient spatial index and to adopt proper compression methods as well as to extend the data retrieve strategy of commercial ORDBMS are significant for large CyberCity GIS.

7. CONCLUDING REMARKS

This paper introduced the architecture and the mechanism of the data management system of the CyberCity GIS. More efficient spatial index must be created to support more complex data access. On the other hand, the fast compression algorithm should be found to compress the object texture images and the terrain orthoimages, and the fast decoding algorithm with multi-resolution for real time applications is quite significant.

8. ACKNOWLEDGMENT

This project is supported by the NSFC (No.40001017), the Fok Ying Tung Education Foundation (No.71017) and the major state basic research program (No.G1999043801).

9. REFERENCES

Fritsch D., 1996, Three-Dimensional Geographic Information Systems—Status and Prospects, In: *International Archives of Photogrammetry and Remote Sensing*. Vol. XXXI, Part B3 , pp. 215-221.

Gruber M and Wilmersdof E., 1997, Urban Data Management - A Modern Approach, *Computers , Environment and Urban System*, 21(2):147-158.

Gruber M. 1999, Managing Large 3D Urban Databases, *47th Photogrametric Week,* (Dieter Fritsch and Rudi Spiller, editors), Wichmann Verlag, Germany, pp.341-349.

Kofler M., Rehatschek H. and Gruber M., 1996, A Database for a 3D GIS for Urban Environments Supporting Photorealistic Visualization, Int. Archives of Photogrammetry and Remote Sensing, Vol. XXXI, B2, Vienna 1996.

Molenaar, M., 1992. "A topology for 3D vector maps", *ITC Journal,* No. 1, pp. 25-33.

Rikkers, R., Molenaar, M., 1994. "A query oriented implementation of a topologic data structure for 3-dimensional vector maps", *International Journal of Geographical Information Systems*, 8(3): 243-260.

Zhu Qing, 1998, Three Dimensional Dynamic Visualization Model, *Journal of Wuhan Technical University of Surveying and Mapping (in Chinese)*, 23(2):83-87.

Zhu Qing, Li Deren, Gong Jianya, Xiong Hanjiang, 2000, The Integrated Spatial Databases of GeoStar, In: *International Archives of Photogrammetry and Remote Sensing ,* Vol. XXXIII, Part B4, pp. 1243 – 1246.