LUND UNIVERSITY

NOSQL based 3d city model management system

Mao, Bo; Harrie, Lars; Cao, J; Wu, Z; Shenc, J

Link to publication

NOSQL BASED 3D CITY MODEL MANAGEMENT SYSTEM

Bo Mao [a,b, *], Lars Harrie [c], Jie Cao [a,b], Zhiang Wu [a], Jie Shen [d]

[a] Nanjing University of Finance and Economics, 210003 Nanjing, China - (maoboo, caojie6909, zawuster)@gmail.com
[b] Nanjing University of Science and Technology, 210094, Nanjing, China
[c] GIS-Centre, Department of Physical Geography and Ecosystems Science, Lund University,
SE-223 62 Lund- lars.harrie@nateko.lu.se
[d] Nanjing Normal University, 210042, Nanjing, China - shenjie@njnu.edu.cn

Commission VI, WG VI/4

KEY WORDS: 3D city model, NoSQL, data integration, multiple representation structure

ABSTRACT:

To manage increasingly complicated 3D city models, a framework based on NoSQL database is proposed in this paper. The framework supports import and export of 3D city model according to international standards such as CityGML, KML/COLLADA and X3D. We also suggest and implement 3D model analysis and visualization in the framework. For city model analysis, 3D geometry data and semantic information (such as name, height, area, price and so on) are stored and processed separately. We use a Map-Reduce method to deal with the 3D geometry data since it is more complex, while the semantic analysis is mainly based on database query operation. For visualization, a multiple 3D city representation structure CityTree is implemented within the framework to support dynamic LODs based on user viewpoint. Also, the proposed framework is easily extensible and supports geo-indexes to speed up the querying. Our experimental results show that the proposed 3D city management system can efficiently fulfil the analysis and visualization requirements.

1. INTRODUCTION

3D city models are applied in many applications such as urban planning, navigation, disaster management and etc. They have become one of the standard components for city analysis and visualization. Therefore, many municipalities have released their official 3D city models, for example Berlin and Stuttgart. Industry companies like Google, Apple and Nokia are also integrating 3D city models into their map services. Volunteered geographic data communities such as Open Street Map (OSM) are building 3D city models as well. As a result, the volume of 3D city models is increasing dramatically.

This study is devoted to storing, managing, analyzing and visualizing the increasing number of 3D city models. Our working hypothesis is that we have a large amount of 3D city models that we want to visualize in the same service. The service should support visualization in several levels of details. To enable this we are confronted with two challenges. Firstly, we need to create a multiple representation data structure for the 3D city models. In our study the structure is an extension of the CityGML standard. CityGML supports both geometry and semantic information about the city object. It has published as OGC standard and been selected by many cities as their official city model format. Secondly, we need a storage environment that supports fast retrieval of large amount of data.

There have been several studies of creating multiple representation data structures of 3D city models suitable for visualization. Several of those studies are based on the CityGML standard (OGC 2012). This standard has defined five levels of details (LODs) suitable for representation of a city model. To tailor this model for visualization purposes it if often

preferable to complement this standard with new levels of details and also with new geometric data types.

The second challenge in our study is to find a suitable storage environment of the city model. Traditionally, 3D city models are stored in files (e.g. in the formats CityGML or Collada) or object-relational databases (e.g. Oracle spatial or PostGIS). For databases storage specific extensions for 3D city models are often used on top of the object-relational databases. This storage facilitates complex modeling and also analysis of the 3D data. However, the storage has limitations for big data processing. The size of the files or tables in a relational database is limited and the system efficiency drops as the data volumes increases. Therefore, for data intensive applications that do not require advanced modeling and complex analysis it is of interest to seek for alternative storage techniques; one such alternative is the combination of NoSQL databases and cloud computation.

NoSQL databases and cloud computation methods are today used in several data intensive applications such as search engines (e.g. Google), scientific computing, social networks (e.g. twitter) and map services (e.g. Google maps). It has proven to be efficient in big data processing. Different from SQL databases, NoSQL databases are built to service heavy read/write loads and are able scale up and down easily, and therefore they are more natively suited to running on the cloud. Since first proposed in 1998 by Carlo Strozzi, there are more than 150 NoSQL databases developed (NoSQL, 2013). Most of them are open sources and can be used for free. In this study MongoDB is selected to manage the 3D city models. MongoDB is the most popular NoSQL database according to DB-Engines Ranking (2013). It supports Map-Reduce analysis and geospatial indexes, which can speed up the 3D city model

---

*   Corresponding author. Bo Mao, maoboo@gmail.com.

related spatial search. Based on MongoDB, we implement a 3D city management system that supports model input/output and search/visualization.

The rest of the paper is structured as follows. Related work is introduced in section 2. Section 3 describes the proposed 3D city model management system and explains the features of the framework. A case study is given in Section 4 to demonstrate the proposed framework. Finally, Section 5 concludes the whole paper and suggests our future research direction.

2. RELATED WORK

2.1 Multiple representation structure

In 3D city visualization, multiple representation structures of different LODs are essential to reduce the loading time and to improve the visual efficiency. Coors (2003) as well as Kolbe and Gröger (2003) proposed models for obtaining consistency between the LODs in 3D city models where geometric elements are shared; i.e., a geometric element becomes a part of a geometry in an adjacent LOD. Parry et al. (2002) described a scalable building organisation structure appropriate for dense 3D urban areas based on quadtrees. Mao et al. (2011) suggested a multiple representation structure called CityTree based on the block division and 3D buildings aggregation. In this paper, we will implement the CityTree structure with the proposed framework based on MongoDB system to support the dynamic visualization of 3D city models.

There has also been extensive research in generalization methods to populate the multiple representation databases with geographic data (see e.g. Meng and Forberg, 2007 for an overview). Mayer (2005) and Forberg (2007) create a scale-space technique partly based on the morphological operators opening and closing to simplify 3D building model. Half space model is used by Kada (2006) to detect the main outline of a building. Lu et al. (2011) suggest using transition polygon to represent 3D buildings, but this method can only support flat roof structures. Fan et al. (2012) generated 3D building exterior representation by combing roof and ground plan.

2.2 Storage method of 3D city models

Today 3D city models are generally stored in object-relational databases such as Oracle and Postgis (Stadler et al. 2009) and in most cases these databases are extended with dedicated city model tools such as 3DCityDB (2013) and DB4Geo (Breunig et al. 2010). Strauss and Carey (1992) introduced object oriented databases for 3D model management; these databases have also been used for 3D building applications (Long et al., 2011). Felix (2013) implemented the CityGML schema in an object-relational database. However, the CityGML schema is complex and leads to hundreds of tables in the database but most of these tables may not be used in most applications. These current frameworks are difficult to implement and cannot be easily extended and parallelized as can be done with cloud computation and NoSQL databases.

The cloud computing and NoSQL technology are developed to high performance, big data storage, high scalability and high availability. These properties make the technology suitable for 3D model management. Luan et al. (2014) proposed a 3D model management strategy based on Hadoop distributed file system to tackle the big data problem that 3D model management is facing. Sugumaran et al. (2014) developed and implemented a web-based 3D data processing system using Amazon's EC2 cloud computing environment. Their tests demonstrated the advantages of cloud computing over traditional approaches in time, cost and performance. Dobos et al. (2012) employ NoSQL database (Mongodb) to store 3D models in aid of public consultation. Han and Stroulia (2012) build their 3D data management system based on another NoSQL database called Hbase. Cloud computation method such as Map-Reduce (Park et al. 2011) can be deployed on NoSQL database to increase the analysis speed. It is suitable for big data applications such as 3D city model generalization and visualization.

3. METHODLOGY

The multiple representation data structure CityTree is this study (which is an improvement and extension of the tree in Mao et al., 2011) is based on the CityGML standard using the following LODs:

LOD1 – box models of buildings using flat roofs (box representation)
LOD2 – building models using roof structures and building facades (constructive solid geometry representation, CSG)
LOD3 – detailed outdoor building models where doors, windows, balconies etc. are explicitly modelled (boundary representation, B-rep).

In this paper, LOD3 models are the original or input models. LOD2 representations are generated from the LOD3 models, and it contains building centroid, ground plan, height, roof style and etc. Then from the CSG, box models in LOD1 are created, which contain centroid, height, minimum containing rectangle (MCR) of the ground plan and etc. Furthermore, LOD1 box models are aggregated according to their distance and stored in the CityTree. The CityTree will be saved in database for visualization applications.

3.1 Input

Figure 1 shows the system architecture. Since city models can be generated from different sources the system architecture should support multiple format for data input. In this study CityGML is selected as the standard for 3D city dataset integration; Collada and 3D Max files are also supported. 3D models in different formats are first read into the system with specific parsers, for example we use citygml4j for CityGML files, xj3d for X3D files and j3d for KML and Collada files. Then, the geometry content and semantic features (if there is) of the input 3D models are extracted and stored into NoSQL database in JSON format which can support different schemas for different models (see more details in section 4). For example, all buildings have the features such as Id, type, centroid, height and so on, but one building may have the feature "owner" while others do not. All these buildings can be stored in a collection of NoSQL, which is not supported in a table of relational database since its columns have to be predefined.
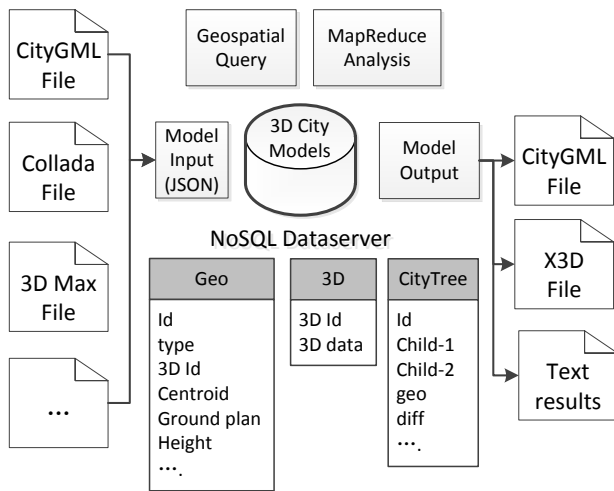
Figure 1. 3D city model management framework

3.2 Storage

The multiple representation data are stored in three collections in the NoSQL database (similar with tables in the relational database):

CityTree
The CityTree collection contains information about the multiple representation data (cf. Figure 1).

Geo
Stores attribute data of the buildings such as type of building, centroid, etc. It also stores an id-number for the 3D representation (stored in the 3D collection).

3D
This collection stores the 3D representation of a building or a generalized building group.

The reason for separating the two latter collections in this study is performance. The querying of the data is mainly implemented on the Geo collection which supports a 2D geospatial index. It should be noted here that the NoSQL database does not support join operations between collections. It is therefore vital that the Geo table contains all the relevant information used in data search. In other words, semantic information about the building is stored in Geo table; 3D geometry data is stored in 3D table and its dynamic visualization structure is stored in the CityTree.

The querying on 3D information is implemented with a map-reduce method. Since the map-reduce methods makes full use of the parallel computing, it is suitable for the large volume of 3D data.

3.3 Output

The output of the model is in three formats:

CityGML
To support the city model exchanging with other applications, we use CityGML as the query output format. The CityGML file contains both semantic and geometry information of 3D city models and it can be integrated with the existing GIS frameworks such as ERSI system or LandXplore from AutoCAD.

X3D
If the requirement is focused on visualization, the output can be in X3D format. X3D files can be rendered in WebGL supported browsers without additional plugins. Therefore, the X3D output can be viewed in multiple platforms including Android, Windows, iMac and Linux. Currently, the iOS platform does not support WebGL, so plugins are required to show 3D city models in iphone/ipad.

Text file
If the query output is just the simple analysis results, we can use the text file as output. These text file can be in simple text, XML or JSON format as we currently used.

The case study shows the proposed framework can deal with the large number of building models with a cluster of common PCs, which can save the implementation cost quite much. Our framework can be extended as the data volume grows by just plug in the new PCs into the system with a simple configuration.

4. CASE STUDY

We built a cluster with NoSQL database MongoDB to store and management the proposed multiple representation structures of 3D city models. Mongodb system is naturally designed for distribution clusters. It supports auto-sharding that allows us to add or remove data servers without affect the applications running on the database. In our laboratory, we use five PCs to build our database as shown in Figure 2. The main database process is running on an App Server (main server for database related applications such as configuration, querying or updating) that also preserve the cluster configure information. The Test Server is a backup of App Server and can also be used for loading balance. The data are mainly stored in the DB nodes (DB01 and etc.), which can be dynamically added (as long as it is less than a threshold) or removed from the cluster without interfere applications running on the cluster. All data can have multiple copies for redundancy according to the configuration.
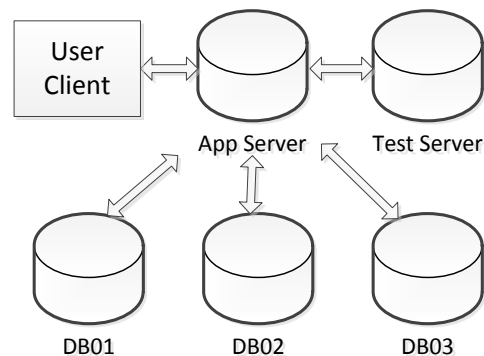
Figure 2. Mongodb cluster

The user client is standard web browser such as Chrome, IE or Firefox. We develop a Nodejs based web server to access the 3D city models. Nodejs is an event-driven I/O server-side JavaScript environment based on Chrome V8 Engine. It can be used to implement the web server easily. In Nodejs, a web application framework called Express provides robust tooling for HTTP server. Express employs Model -View-Control (MVC) structure to handle the user requires and can be easily extended for different application. Combined with MongoDB, the Express framework can support user management, which is

important for 3D city model related applications. The 3D visualization is also integrated into the web browser. Currently, we have tested with Threejs and X3DOM, both of which are WebGL based browser 3D rendering tools. The experiments indicate that Threejs can supply better 3D rendering effect such as e.g. shading, UV and lighting. While X3DOM supports better interactive/animation and is easier to implement. Both of these visualization techniques can be applied into the online 3D city model management system.

In our experiments, the hardware configuration is as follows: App server, Intel i7 3770 CPU with 4G memory and 1T hard disk; Test server, Xeon E7330 duo core CPU with 4G memory and 300G hard disk; three data servers, E8300 CPUs with 2G memories and 250G hard disks. The total capacity of the system is around 2.5T, we use a triple redundancy so the effective storage is around 0.8T. All servers run Ubuntu 13.04 X64 server operation system with Mongodb 2.4.5. The Nodejs server is in version 0.8.18 and is running on the app server. The whole system has run for four months without crush down for once, which shows the stability of Mongodb system. Figure 3 shows 3D visualization results of the original model and the CityTree generalized one.
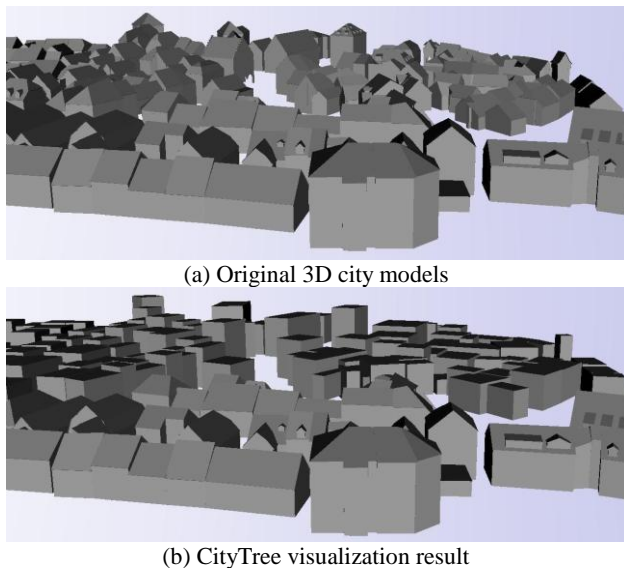

(a) Original 3D city models

(b) CityTree visualization result

Figure 3. City Model visualization results

5. CONCLUSIONS

In this paper, we have built a 3D city model management system with NoSQL database—Mongodb. In this framework, 3D city models in different format such as Collada, X3D, CityGML and 3DS are stored and managed with a multiple representation structure—CityTree. A MongoDB cluster is built to store the models and a Nodejs based web server is developed to access the 3D city models. The experimental results indicate that the proposed framework can fulfil the requirements of 3D city management. In the future, a larger scale system will be built to in more depth evaluate the possibilities and performance of parallel computations.

6. REFERENCES

3DCityDB, 2013, http://opportunity.bv.tu-berlin.de/software/projects/3dcitydb (10 March. 2014)

Breunig, M., Schilberg, B., Thomsen, A., Kuper, P., Jahn, M., Butwilowski, E., 2010. DB4GeO, a 3D/4D Geodatabase and Its Application for the Analysis of Landslides, *Lecture Notes in Geoinformation and Cartography* 2010, pp 83-101

DB-Engines Ranking, 2013. http://db-engines.com/en/ranking (10 March. 2014)

Dobos, J., Steed, A., 2012. 3D Revision Control Framework, *Web3D*, 2012

Fan H. and Meng L., 2012. A three-step approach of simplifying 3D buildings. *International Journal of Geoinformation Science* 26(6), pp. 1091-1107

Forberg, A., 2007. Generalisation of 3D building data based on scale-space approach. *ISPRS Journal of Photogrammetry and Remote Sensing* 62 (2), pp. 104-111.

Han, D., Stroulia, E., 2012. A three-dimensional data model in hbase for large time-series dataset analysis, IEEE 6th International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems(MESOCA), pp. 47–56.

Hua Luan, Yachun Fan, Mingquan Zhou, Xuesong Wang, 2014, Towards Effective 3D Model Management on Hadoop, *Lecture Notes in Electrical Engineering* Volume 279, 2014, pp. 131-139

Kada, M., 2006. 3D Building Generalisation based on Half-Space Modeling. Proc. ISPRS Workshop on Multiple Representation and Interoperability of Spatial Data, Hannover, Germany. 7 p. (on CD-ROM).

Kunde, F., 2013. CityGML in PostGIS : portability, usage and performance analysis using the example of the 3D City Database of Berlin, *Master Thesis conducted at the Department of Geography*, University of Potsdam, 2013.

Long N., Fleming, K. Brackney, L., 2011. An Object-Oriented Database for Managing Building Modelling Components and Metadata, Presented at Building Simulation 2011, Sydney, Australia, November 14-16, 2011, http://www.nrel.gov/docs/fy12osti/51835.pdf (10 March. 2014)

Lu Y., Behar E., Donnelly S., Lien J.-M., Camelli F., Wong D., 2011. Fast and robust generation of city-scale seamless 3D urban models, *Computer Aided Design*, 43(11), pp. 1380-1390.

Mao B., Ban Y, Harrie L., 2011 A Multiple Representation Data Structure for Dynamic Visualisation of Generalised 3D City Models, *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(2): pp. 198-208.

Mayer, H., 2005. Scale-spaces for generalization of 3D buildings. In: *International Journal of Geographical Information Science*. Vol. 19, No. 8-9, September-October 2005, pp. 975-997.

Meng, L. and Forberg A., 2007. 3D Building generalisation, In: W. Mackaness, A. Ruas, and L. T. Sarjakoski (Eds.), *Generalisation of Geographic Information: Cartographic Modelling and Applications*, Elsevier, pp. 211-232.

NoSQL, 2013. http://nosql-database.org/ (10 March. 2014)

OGC, 2012. OGC City Geography Markup Language (CityGML) Encoding Standard, Version 2.0. Available at: http://www.opengeospatial.org/standards/citygml (10 March. 2014).

Park, J., Yun, C., Jung, H., Lee, Y., 2011. Visualization of Urban Air Pollution with Cloud Computing, I*EEE World Congress on Services*, pp. 578-583

R Sugumaran, J Burnett, M Armstrong ,2014, Using a Cloud Computing Environment to Process Large 3D Spatial Datasets, *Big Data: Techniques and Technologies in Geoinformatics*, edited by Hassan A. Karimi, CRC press

Stadler, A., Nagel, C., König, G, Kolbe, T., 2008. Making interoperability persistent: A 3D geo database based on CityGML in: Lee & Zlatanova (eds.), *3D Geo-Information Sciences*, Selected papers from the 3rd International Workshop on 3D Geo-Information, Seoul, Korea. LNG&C series, Springer Verlag, pp. 175-192. 2008

Strauss, P., Carey, R., 1992. An Object-Oriented 3D Graphics Toolkit, *Computer Graphics*, 26(2) pp. 341-349, July 1992

Vainio, T., Kotala, O., 2002. Developing 3D information systems for mobile users: some usability issues. In Proceedings of the second Nordic conference on Human-computer interaction (NordiCHI '02). ACM, New York, NY, USA, pp. 231-234.

7. ACKNOWLEDGEMENTS