A Multi-Level Cache Approach for Realtime Visualization of Massive 3D GIS Data

Xiaoming Li, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China

Weiping Xu, Wuhan University, China

Qing Zhu, Wuhan University, China

Jinxing Hu, Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China

Han Hu, Wuhan University, China

Yeting Zhang, Wuhan University, China

ABSTRACT

The real-time visualization of 3D GIS at a whole city scale always faces the challenge of dynamic data loading with high-efficiency. Based on the multi-tier distributed 3D GIS framework, this paper presents a multi-level cache approach for dynamic data loading. It aims to establish in 3D GIS spatial database engine (3DGIS-SDE) the unified management mechanism of caches on three levels, including: the client memory cache (CMC) oriented to sharing application, the client file cache (CFC) organized by index, as well as the application server memory cache (ASMC) of structural consistency. With the help of the proposed optimized cache replacement policy, multi-level cache consistency maintenance as well as multithread loading model designed in the paper, the engine is able to adaptively make full use of each-level caches according to their own application properties and achieve effective coordination between them. Finally, a practical 3D GIS database based on Oracle 11g is employed for test. The experimental results prove this approach could satisfy multi-user concurrent applications of 3D visual exploration.

Keywords: 3D Geographic Information System (GIS), Application Server Memory Cache, Client File Cache, Client Memory Cache, Dynamic Data Loading, Multi-Level Cache, Multithread Loading Model

INTRODUCTION

With the on-going promotion of 3D digital city construction, the increasing requirements of powerful 3D GIS platform arise (Zlatanova et al., 2002; Butler, 2006). For example, unified data management, fast dynamic data loading, multi-user sharing application and high-security data access are urgently needed in dealing with multi-source, multi-scale and large-scale
3D spatial data (Breunig & Zlatanova, 2011; Zhang et al., 2011). Since 3D data is much more complicated than traditional 2D data in many aspects like the variety in the semantics, geometry and texture of both aboveground and underground entities which also bring their larger data size, its loading efficiency is more inclined to decrease sharply, especially when the real-time visual applications cause frequent database I/Os and even massive data blocks at worse in multi-user access pattern (Zhu et al., 2011). Therefore, how to spare no efforts to improve the data loading capability based on TB-scale 3D spatial database is a key to make a 3D GIS platform more robust.

Caching, which can provide fast access to hot data and high-speed data interchange, has always been playing an important part in the development of computer technology (Bunt et al., 1993; Froese & Bunt, 1996; Huang & Cai, 2006). In the multi-tier distributed framework, multi-level caches can be made full use of in order to effectively release the server’s burden in I/Os and decrease the user’s average waiting time (Muntz & Honeyman, 1992; He et al., 2007). Nowadays, to accelerate the loading of spatial data, more and more GIS commercial softwares begin to pay attention to the technology of caching. For example, the ArcGIS software is able to provide 2D and 3D caching methods, and the caches in ArcGlobe are predefined through tiling scheme (Kennedy, 2009). To meet the practical requirements, it also supports merging caches, on-demand caches and multi-layer caches. The series of SuperMapServer products are able to establish the multi-level cache mechanism for the management of 2D data caches and application caches, where the technology of embedded map grid cache and smart client is applied in the system’s performance optimization on different levels (SuperMap, 2011). Google Earth, Skyline and Virtual Earth 3D all make its data loading more efficient by creating cache files in the clients to cut down on their accesses to the server, which only deal with terrain tiles and less-detailed 3D city models (Aurambout et al., 2008). Some mainstream vendors of database management system (DBMS) also offer customized caching products for their own database. Take Oracle TimesTen for example (Oracle, 2009). It is not only an in-memory relational database of complete durability and restorability, but also used as a kind of memory cache for Oracle database to raise the application software’s response speed.

Since the 3D GIS platform based on the distributed architecture can optimize the whole data loading process by the multi-level cache approach, the paper will mainly focus on: how to effectively organize the cache on each level so as to realize its potential in improving the data loading efficiency, how to make their coordination more complementary, and how to integrate the caching policy with the multithread loading model for the multi-user concurrent access. The rest of the paper is organized as follows. First, we define each-level cache and give a typical data loading flow in 3DGIS-SDE based on the multi-level cache structure, followed by the discussion of some key technologies for the implementation of this approach afterwards. The validity and feasibility of the proposed approach is illustrated in the experiment of real-time applications based on Wuhan 3D spatial database in Oracle 11g in the following section. Finally, conclusions are drawn in the last section.

**MULTI-LEVEL CACHE APPROACH**

**Definition of Multi-Level Caches**

It is recognized as an effective way to manage huge 3D spatial data by large relational database management system (RDBMS). However, the variety of relational databases and their own complexity become an obstacle to high-efficiency data loading in 3D GIS (Zhu et al., 2011). The paper studies the high-performance 3DGIS-SDE with multi-level cache structure to explore its solution. The engine, as a connection channel for the data interaction between the application front-end and the low-level database, is able to support integrated management, hierarchical index establishment, and multithread