Computers & Geosciences 37 (2011) 814–821

Contents lists available at ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

A fast and robust bulk-loading algorithm for indexing very large digital elevation datasets II. Experimental results

Félix R. Rodríguez *, Manuel Barrena

Media Engineering Group, University of Extremadura, Campus universitario, 10003 Cáceres, Spain

ARTICLE INFO

Article history:
Received 27 February 2009
Received in revised form
17 August 2010
Accepted 8 January 2011
Available online 5 February 2011

Keywords:
Index structures
The Q-tree
Geographic information systems
Digital elevation models
Shuttle Radar Topographic Mission
Performance analysis

ABSTRACT

The spatial indexing of eventually all the available topographic information of Earth is a highly valuable tool for different geoscientific application domains. The Shuttle Radar Topography Mission (SRTM) collected and made available to the public one of the world's largest digital elevation models (DEMs). With the aim of providing on easier and faster access to these data by improving their further analysis and processing, we have indexed the SRTM DEM by means of a spatial index based on the *kd*-tree data structure, called the Q-tree. This paper is the second in a two-part series that includes a thorough performance analysis to validate the bulk-load algorithm efficiency of the Q-tree. We investigate performance measuring elapsed time in different contexts, analyzing disk space usage, testing response time with typical queries, and validating the final index structure balance. In addition, the paper includes performance comparisons with Oracle 11g that helps to understand the real cost of our proposal. Our tests prove that the proposed algorithm outperforms Oracle 11g using around a 9% of the elapsed time, taking six times less storage with more than 96% of page utilization, and getting faster response times to spatial queries issued on 4.5 million points. In addition to this, the behavior of the spatial index has been successfully tested on both an open GIS (VT Builder) and a visualizer tool derived from the previous one.

1. Introduction

Digital elevation models (DEMs) constitute an important data source for geographic information systems (GIS) and geoscience-related applications. There are several DEM sources in the world, most available from the United States Geological Survey (USGS, 2010) agency site[1]. One of the most complete and high-quality DEMs comes from the Shuttle Radar Topography Mission (SRTM), an international project led by the National Geospatial-Intelligence Agency (NGA, 2010)[2] and the National Aeronautics and Space Administration (NASA). This mission elaborated the first ever near-global dataset of land elevations, providing a global high-quality DEM, which was an important step toward globalizing and homogenizing most topographical surface data on Earth. Before the mission, GIS had to use DEM data from different sources, often derived from satellite imagery or stereo photos, with variable availability, which produced heterogeneous representations, in addition to demanding troubleshooting of anyone working with large regions. Worboys and Duckham (2004) highlights the foundations, full extent, and importance of the DEMs, and detailed descriptions about the SRTM can be seen in Kretsch (2000), Werner (2001), Farr et al. (2007), the Consortium for Spatial Information GeoPortal (CGIAR, 2010) Web site[3], and on the Web site of the Jet Propulsion Laboratory (JPL) of (NASA, 2010a)[4].

The Space Shuttle Endeavor collected 12.3 TB of raw data (Farr et al., 2007), covering nearly 80% of the Earth‘s land surface. These data were filtered, gridded, and divided into a collection of data files, each corresponding to one squared degree of latitude and longitude. The SRTM DEM consists of the whole set of flat files, made freely available by the JPL CIT at the official (NASA, 2010b) site[5]. To use the SRTM DEM, applications must access the NASA

* Corresponding author. Current address: Department of Engineering of Informatic and Telematic Systems, Escuela Politécnica, Universidad de Extremadura, Campus universitario, 10003 Cáceres, Spain. Tel.: +34 927 257 000x57553;
fax: +34 927 257 202.

E-mail addresses: felixr@unex.es (F.R. Rodríguez), barrena@unex.es (M. Barrena).

URL: http://gim.unex.es (M. Barrena).

[1] United States Geological Survey. SRTM data, http://srtm.usgs.gov/ (accessed 03 February 2010).

[2] National Geospatial-Intelligence Agency. http://www1.nga.mil/ (accessed 03 February 2010).

[3] Consortium for Spatial Information GeoPortal. http://srtm.csi.cgiar.org/ (accessed 03 February 2010).

[4] National Aeronautics and Space Administration, Jet Propulsion Laboratory of the California Institute of Technology, The Shuttle Radar Topographic Mission. http://www2.jpl.nasa.gov/srtm/ (accessed 03 February 2010).

[5] National Aeronautics and Space Administration, JPL, CIT, SRTM Digital Topographic Data. ftp://e0srp01u.ecs.nasa.gov/ (accessed 03 February 2010).

0098-3004/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cageo.2011.01.004

repository to obtain the flat files that have the data for the region of interest. Every point query about a particular elevation involves selection of the appropriate directory and flat file, reading it and saving its heights in main memory, and translating the height location in agreement with the file name. If the query looks for a geographic region (range queries), the process will likely expand to several files, depending on the required region, which needs to gather, unify, and filter the files in one data set (for instance, to retrieve the whole island of Minorca in a range query, we need to select four SRTM files, while the area covered by the island is much smaller than one SRTM flat file). Batch processing, with automatic and continuous queries (like data mining and data analysis), requires iterative file reading and reinterpretation. All these processes tend to become cumbersome and inefficient, and thus hardly acceptable for many purposes.

Aiming at providing easier and faster access to the SRTM data by improving their further analysis and processing, we have indexed the huge number of DEM data coming from SRTM through a multidimensional structure, the Q-tree (Jurado and Barrena, 2002), an original *kd*-tree (Bentley, 1975)-based index structure developed by our research group, and presented in the first paper of this two-part series. In this second part, we present the experimental results of our research on SRTM DEM indexing to facilitate data processing tasks. The proposed method constitutes the first successful experience in achieving accessibility for such a huge number of data.

The remainder of this paper is organized as follows: Section 2 shows experimental results about our bulk-load algorithm, including four sections on the measures: bulk-load performance in Section 2.1, space utilization in Section 2.2, query performance in Section 2.3, and index balance in Section 2.4. Section 3 describes the conclusions.

2. Experimental results

To measure the efficiency of our bulk-load algorithm, we have conducted a series of tests on three different Intel platform computers: (i) double Xeon, 2.4 GHz, 8 GB of main memory, and eight 250 GB-disks on a RAID; (ii) P-IV, 3 GHz, 2 GB main memory, and 250 GB hard disk; and (iii) P-M, 1.4 GHz, 512 MB main memory, 30 GB hard disk, and 1 TB external hard disk. These three different platform computers show that high performance computers are not required; personal computers or small servers are now sufficient. All the tests were performed on each machine, and the results shown for each test are the average of the results obtained on the three computer platforms. Although the number of *I*/*O* accesses is as usual, we have decided to show exact time measurements to present the performance results of access methods. The aim is to provide a clear idea of the cost of this time-consuming process. In addition to measuring elapsed time in different contexts, we investigate performance via three other significant parameters: disk space usage, response time with typical queries, and the balancing of the index structure.

The tests consist of partial and complete bulk loads with a varying number of elevation points. The numbers range from 1.44 million points in one SRTM file to nearly 21 billion points for all the coarse-granularity SRTM DEM. Experimental results are displayed for uniformly increasing volumes of data. The graphs in this section show (on the abscissa) this number of input data. Partial input usually corresponds to well-delimited regions on Earth (e.g., the Iberian Peninsula, the Canary Islands, or the South American continent).

To evaluate the performance of our bulk-load algorithm, we have carried out comparisons with commercial databases incorporating spatial extensions (commonly through the use of R-tree indices). Therefore, we use the Oracle Spatial extension (Oracle (2009), see also Kothuri et al. (2002, 2004)), mainly because it is a good reference for the database community and it uses an R-tree to manage geographical data, which let us to evaluate the performance of the algorithm.

2.1. Bulk-load performance

The first approach to bulk-load performance is shown in Fig. 1. This figure presents the time it takes to load a total of 122 SRTM files (corresponding to the Iberian Peninsula) in one run. This cost is contrasted with all the possible alternatives dealt with. This figure shows that any bulk-load strategy is better than the insertion of points tuple-by-tuple. The load times for Oracle vary depending on whether we consider the time to create the R-tree index structure when inserting the complete data. An interesting point is that our Q-tree bulk-load algorithm, by using the simplest file and block selection strategies, only takes 9% of the time required by the speed loading process in Oracle 11g. In fact, with these differences, we can perceive that the two processes, Oracle and the Q-tree bulk loader, cannot be placed at the same comparison level. While Oracle has to run dozens of internal processes to accomplish the complete data load, the Q-tree loader runs in isolation. But even with this premise, the achievement of a 91% time reduction with respect to Oracle 11g gives us definite reasons to consider our approach an actual contribution.

Due to the bulk load in Oracle spatial, the process takes place in two phases, consisting of the load of rough data with the SQL*Ldr tool and the index creation (Kothuri et al., 2004, Chapter 8), We decided to measure both processes separately. Fig. 2 points to the good performance of the Q-tree bulk-load algorithm as compared to the Oracle 11g bulk-load operation (at several moments of the complete load) taken by the different processes to complete the load of South America. The load times always improve in the Q-tree algorithm, as shown in Fig. 2a. The Oracle bulk load via the SQL*Ldr is quite close, in terms of *I*/*O* operations, to the Q-tree bulk load; however, it becomes more distant when the index creation process of the Oracle spatial comes into play, shown in Fig. 2b.

To check whether the page size significantly influences Q-tree bulk-load performance, we have measured both processing time and numbers of *I*/*O* operations with the load of one SRTM file having different page sizes every time, while the data and index pages

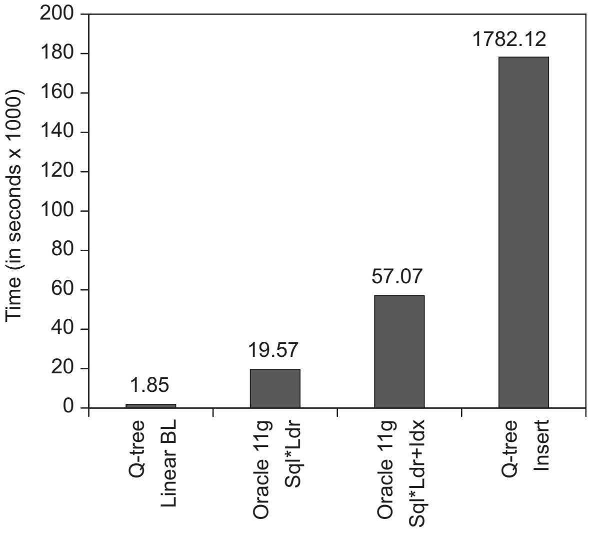

Fig. 1. A contrastive view of load times for the Iberian Peninsula (122 SRTM files) with 8 kB of page size, based on the bulk-loading strategy.