A New Parallel Data Cube Construction Scheme

Dong Jin, Qingdao University, China
Tatsuo Tsuji, University of Fukui, Japan

ABSTRACT

The pre-computation of data cubes is critical for improving the response time of OLAP (On-Line Analytical Processing) systems. To meet the need for improved performance created by growing data sizes, parallel solutions for data cube construction are becoming increasingly important. This paper presents a new parallel data cube construction scheme based on an extendible multidimensional array, which is dynamically extendible along any dimension without relocating any existing data. The authors implemented and evaluated their parallel data cube construction methods on shared-memory multiprocessors. Given the performance limit, the methods achieve close to linear speedup with load balance. The authors’ experiments also indicate that their parallel methods can be more scalable on higher dimensional data cube construction.

Keywords: Data Cube, Extendible Array, MOLAP, Parallel Database, Shared-Memory Multiprocessors

1. INTRODUCTION

The pre-computation of the various views (group-bys) of a data cube, i.e., the forming of aggregates for every combination of GROUP-BY attributes, is critical for improving the response time of OLAP queries in decision support systems (Gray, Bosworth, Layman, & Pirahesh, 1996). When the number of dimension attributes is \( n \), the data cube computes \( 2^n \) group-bys, each of which is called a cuboid. A lattice can be used to express dependencies among cuboids (Harinarayan, Rajaraman, & Ullman, 1996). Figure 1 shows a lattice for a 4-dimensional data cube with dimension \( a, b, c \) and \( d \). An edge between two cuboids indicates that the target cuboid can be computed from the source cuboid by aggregation along one dimension. We call the cuboid \( abcd \) at the bottom of the lattice base cuboid. The others are called dependent cuboids because they can be computed directly or indirectly from the base cuboid.

As the number of dimensions increases, data cube computation cost grows exponentially. Parallel solutions on multiprocessor systems are becoming very popular for fast data cube computation (Chen, Dehne, Eavis, & Rau-Chaplin, 2004; Dehne, Eavis, Hambrusch, & Rau-Chaplin, 2002; Dehne, Eavis, & Rau-Chaplin, 2006; Goil & Choudhary, 1997, 2001; Jin, Yang, Vaidyanathan, & Agrawal, 2005; Lu, Huang, & Li, 1997; Lu, Yu, Feng, & Li, 2003; Muto & Kitsuregawa, 1999; Ng, Wagner, &
Yin, 2001; Yang, Jin, & Agrawal, 2003). They are all based on two kinds of cluster architecture: shared-storage or shared-nothing architecture depending on the nature of disks access. In this paper, we choose to implement on shared-memory multiprocessors. The processors can use shared memory to exchange data between each other to avoid the large data communication cost which may be caused by parallel data cube computation on shared-nothing multiprocessors systems.

In previous works on parallel systems (Chen et al., 2004; Dehne et al., 2002, 2006; Goil et al., 1997, 2001; Jin et al., 2005; Lu et al., 1997, 2003; Muto et al., 1999; Ng et al., 2001; Yang et al., 2003), data management is one challenge because they all organize a data cube on cuboid level as far as we know. Data management becomes very complex for high dimensional data cubes because a full data cube is usually stored into $O(2^n)$ files corresponding to the $2^n$ cuboids. All of the parallel methods are also challenged by load imbalance which is caused by different cuboid sizes or data skew (Chen et al., 2004; Muto et al., 1999). Due to dependency among cuboids, there must be parallel performance limit in the previous works (Chen et al., 2004; Dehne et al., 2002, 2006; Goil et al., 1997, 2001; Jin et al., 2005; Lu et al., 1997, 2003; Muto et al., 1999; Ng et al., 2001; Yang et al., 2003). However, the performance limit was not explicitly discussed in those papers.

In this paper we overcome the challenge of complex data management by implementing a single array based data cube. The parallel data cube construction algorithms proposed in this paper achieves load balancing by simple solutions on shared-memory multiprocessors. We will also discuss the parallel performance limit quantitatively together with the related construct, parallel scalability.

There are two basic data cube representations: ROLAP representations where cuboids are represented as relational tables and MOLAP representations where cuboids are represented as multi-dimensional arrays. Multi-dimensional arrays are natural to express the multi-dimensionality of OLAP data, which makes MOLAP more suitable for data analysis. Among the parallel data cube construction papers, papers (Chen et al., 2004; Dehne et al., 2002, 2006; Lu et al., 1997, 2003; Muto et al., 1999; Ng et al., 2001) are for ROLAP; papers (Goil et al., 1997, 2001; Jin et al., 2005; Yang et al., 2003) are for MOLAP. Fixed-size multidimensional arrays are used in MOLAP papers (Goil et al., 1997, 2001; Jin et al., 2005; Yang et al., 2003). In this paper, we use the extendible multidimensional array model proposed by Hasan, Kuroda, Azuma, and Tsuji (2005) as a basis for data cube construction in MOLAP. Unlike a fixed

![Figure 1. A 4-dimensional data cube lattice](image-url)
Related Content

On Construction of a Multi-Grid Resource Selection Strategy on Grids

Model-Based Application Deployment on Cloud Computing
[www.igi-global.com/article/model-based-application-deployment-on-cloud-computing/226964?camid=4v1a](www.igi-global.com/article/model-based-application-deployment-on-cloud-computing/226964?camid=4v1a)
Context Related Software Under Ubiquitous Computing
[www.igi-global.com/chapter/context-related-software-under-ubiquitous/64505?camid=4v1a](www.igi-global.com/chapter/context-related-software-under-ubiquitous/64505?camid=4v1a)

Fog Computing: Applications, Concepts, and Issues
[www.igi-global.com/article/fog-computing/188795?camid=4v1a](www.igi-global.com/article/fog-computing/188795?camid=4v1a)