Information Sciences 328 (2016) 288–301

Contents lists available at ScienceDirect

Information Sciences

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

Integrating frequent pattern clustering and branch-and-bound approaches for data partitioning

Yin-Fu Huang*, Chen-Ju Lai

Department of Computer Science and Information Engineering, National Yunlin University of Science and Technology, 123 University Road, Section 3, Touliu, Yunlin, Taiwan 640, ROC

ARTICLE INFO

Article history:
Received 28 October 2014
Revised 8 August 2015
Accepted 23 August 2015
Available online 5 September 2015

Keywords:
Data partitioning
Vertical partitioning
Branch-and-bound
Apriori algorithm
Cosine similarity

ABSTRACT

In this paper, we propose an approach integrating frequent pattern clustering and branch-and-bound algorithms for finding an optimal database partition. First, the Apriori algorithm and cosine similarity are used to determine weighted frequent patterns according to a transaction profile. On the basis of the weighted frequent patterns, we developed two methods for partitioning a database: the candidate method and the optimal method. The optimal method involves using a branch-and-bound algorithm and considering costs in each step of combining attributes until an optimal solution is reached. Furthermore, we refined the optimal method for expediting the execution by reducing the search space. Finally, the experimental results show that the proposed optimal method performs the highest among all examined methods, and the refined method is considerably more efficient than the original method.

1. Introduction

In a relational database system, efficiency is paramount when a transaction is processed. For accessing databases, a large amount of data can be extracted in a transaction, but some of them can be useless in subsequent processing. Therefore, several approaches such as indexing and data partitioning have been proposed for accessing databases more efficiently and reducing disk input/output (I/O).

In general, database design can be divided into logical and physical database design. For logical database design, a database administrator (DBA) selects the content of the database at an abstract level or creates the conceptual schema. For physical database design, the DBA also decides how the data must be represented in the database. Once database design is completed, the database schema (or structure) would work for a long time until the schema must be modified. Therefore, determining an optimal database design is critical to the performance of the database. Applications that involve accessing data in a structure-variant manner are not suitable for using a database.

In this paper, we propose an approach consisting of two processes for finding an optimal partition. First, the Apriori algorithm and cosine similarity [13] are combined to determine weighted frequent patterns. The Apriori algorithm is used to ascertain the combination of attributes (or patterns) frequently accessed in transactions. Cosine similarity is used to measure the similarity of patterns, which are then merged into a weighted frequent pattern. Next, a branch-and-bound algorithm is used to produce an optimal database partition according to a cost model.

The remainder of this paper is organized as follows. In Section 2, previous studies related to data partitioning are reviewed. Section 3 presents the database partitioning problem and the system architecture for finding an optimal partition. In Section 4,

* Corresponding author. Tel.: +886 5 5342601; fax: +886 5 5312170.
E-mail address: huangyf@yuntech.edu.tw (Y.-F. Huang).

http://dx.doi.org/10.1016/j.ins.2015.08.047
0020-0255/

we mine weighted frequent patterns by using the Apriori algorithm and cosine similarity. In Section 5, the candidate method and the optimal method are proposed for deriving the partition with the minimum cost. Therein, we also refine the optimal method by reducing the search space. In Section 6, we present the experimental results of comparing all examined methods (particularly the original optimal method and refined optimal method), the influences of different thresholds used to filter the high-weighted frequent patterns, and feasible strategies for meeting designer requests. Finally, we draw conclusions in Section 7.

2. Previous work

Data partitioning, which has been applied to physical database design in previous studies, can be classified based on various considerations: (1) systems: centralized, distributed/parallel, and cloud databases; (2) approaches: vertical, horizontal, and mixed partitions that entail using different methods; and (3) goals: optimal, near-optimal, and feasible solutions.

For a centralized database, McCormick et al. [18] applied an attribute clustering technique called the bond energy algorithm (BEA) to identify groups (or cluster elements) by permuting the rows and columns of an input data array so that the cluster elements of the data array could be combined. Hoffer [14] used the BEA to estimate affinity between attributes that are based on the required attributes of transactions. Navathe et al. [19] extended the work of Hoffer by generating an attribute affinity matrix containing all attribute pairs from transaction profiles, and then used the BEA to rearrange the rows and columns of the attribute affinity matrix. For determining an optimal partition, Navathe and Ra [20] and Cornell and Yu [7] have applied an optimal binary partitioning algorithm, Huang and Van [15] used the A* algorithm, and Song and Gorla [29] and Ng et al. [21] have employed a genetic algorithm. Agrawal et al. [2] integrated vertical and horizontal partitioning to optimize the performance of a database for a given representative workload. Subsequently, Agrawal et al. [3] proposed the Database Tuning Advisor (DTA) for Microsoft SQL Server 2005, using horizontal partitioning and index structures to achieve optimized physical database design.

In distributed/parallel database research, Cheng et al. [6] explored a genetic algorithm for vertical partitioning to achieve high (or optimal) database retrieval performance. Rao et al. [23] proposed a horizontal partitioning method for a DB2 parallel database system for achieving overall optimal performance. Du [9] proposed a measurement procedure called attraction for evaluating the affinity between attributes, and used two algorithms: (1) the preliminary connection-based partitioning algorithm, which derives fragments; and (2) the connection-based partitioning algorithm, which removes irrelevant fragments. Son and Kim [28] presented an adaptable vertical partitioning (AVP) method supporting two objectives: (1) one is to derive an optimal partition with the minimum cost, and (2) the other is to generate a specific number of fragments required by users. Abuelyaman [1] proposed an optimized scheme for vertical partitioning of a distributed database, independent of query frequencies. Without depending on the schema layout, Curino et al. [8] presented Schism, a workload-aware approach for database partitioning and replication designed to improve the scalability of a shared-nothing distributed database. Because distributed transactions are expensive in online transaction processing settings, Schism attempts to minimize the number of distributed transactions while producing balanced partitions. Rodríguez and Li [25] proposed a multimedia adaptable vertical partitioning algorithm for accessing distributed multimedia databases (MAVP), which is extended from AVP. MAVP also accounts for the size of multimedia data to derive an optimal partition. Similar to Curino et al. [8], Pavlo et al. [22] presented an approach for automatically partitioning a database in a shared-nothing parallel database management system. Their algorithm uses a large-neighborhood search technique with an analytical cost model to minimize the number of distributed transactions while controlling the amount of skew. Wu et al. [30] not only proposed several software implementations of data partitioning but also deployed specialized hardware to further improve the efficiency of a parallel database. Fresno et al. [10] presented an approach for integrating dense and sparse data management in parallel programming. Bellatreche and Kerkad [5] proposed an approach for selecting a horizontal partitioning schema of a data warehouse in a divide-and-conquer manner to achieve an improved trade-off between the optimization algorithm's speed and the quality of the solution.

In cloud database research, Kumar et al. [16] presented a comprehensive, workload-aware framework for solving the problems of data placement and replication in cloud data management systems, to minimize the average number of machines involved in executing a query. Aridhi et al. [4] proposed a novel approach for large-scale subgraph mining in large graph databases by using a density-based partitioning technique with the MapReduce framework. Romero et al. [26] studied the feasibility of solving online analytical processing queries with Hadoop (the Apache project implementing MapReduce) while benefiting from secondary indices and partitioning in HBase.

Most relevant research has used an attribute affinity matrix to measure the affinity between attributes [12,17,19,20]; however, the matrix cannot reflect the affinity when more than two attributes are involved. Therefore, data mining techniques have been used in data partitioning to discover crucial combinations of attributes. Gorla and Pang [11] used the Apriori algorithm to filter frequent itemsets and generate candidate partitions, and then produced the partition with the minimum cost as a feasible solution. Rodríguez and Li [24] combined an attribute affinity matrix according to the concept of supports to define an affinity-based support factor used to find an optimal partition.

3. System overview

3.1. Problem definitions

To reduce unnecessary access in a transaction and improve system efficiency, a database is always partitioned into several smaller tables according to the access patterns and frequencies of transactions. Database partitioning can be classified into two