Available online at www.sciencedirect.com
SCIENCE DIRECT®
Information Sciences 168 (2004) 25–50
INFORMATION SCIENCES
AN INTERNATIONAL JOURNAL
www.elsevier.com/locate/ins

Performance analysis of "Groupby-After-Join" query processing in parallel database systems

David Taniar [a,*], Rebecca Boon-Noi Tan [a], C.H.C. Leung [b], K.H. Liu [c]

[a] *School of Business Systems, Monash University, Clayton, Victoria 3800, Australia*
[b] *School of Computer Science and Mathematics, Victoria University, P.O. Box 14428 MCMC, Melbourne 8001, Australia*
[c] *Blueridge Systems, 2115 Aldrin Road, #12B, Ocean, NJ 07712, USA*

Received 1 January 2003; received in revised form 1 September 2003; accepted 1 September 2003

Abstract

Queries containing aggregate functions often combine multiple tables through join operations. This query is subsequently called "*Groupby-Join*". There is a special category of this query whereby the group-by operation can only be performed after the join operation. This is known as "*Groupby-After-Join*" queries—the focus of this paper. In parallel processing of such queries, it must be decided which attribute is used as a partitioning attribute, particularly join attribute or group-by attribute. Based on the partitioning attribute, two parallel processing methods, namely join partition method (JPM) and aggregate partition method (APM) are discussed. The behaviours of these parallelization methods are described in terms of cost models. Experiments are performed based on simulations. The simulation results show that the aggregate partition method performs better than the join partition method.

Keywords: Groupby queries; Groupby-join queries; Parallel query processing; Parallel query optimization; Parallel databases; Performance analysis

* Corresponding author. Tel.: +61-3-99059693; fax: +61-3-99055159.
E-mail addresses: david.taniar@infotech.monash.edu.au (D. Taniar), rebecca.tan@infotech.monash.edu.au (R.B.-N. Tan), clement@matilda.vu.edu.au (C.H.C. Leung), kliu@ieee.org (K.H. Liu).

0020-0255/$ - see front matter
doi:10.1016/j.ins.2003.09.029

1. Introduction

Queries involving aggregates are very common in database processing, especially in on-line analytical processing (OLAP), and data warehouse [1,3]. These queries are often used as a tool for strategic decision making. Queries containing aggregate functions summarize a large set of records based on the designated grouping. The input set of records may be derived from multiple tables using a join operation. This kind of queries is called "*Groupby-Join*" queries, in which the queries contain aggregate functions and join operations.

As the data repository for integrated decision making grows, aggregate queries need to be executed efficiently. Large historical tables need to be joined and aggregated each other; consequently, effective optimization of aggregate functions has the potential to result in huge performance gains. This paper will focus on the use of parallel query processing techniques in Groupby-Join queries, whereby the group-by operations can only be performed after the join operation—therefore we call this "*Groupby-After-Join*" queries.

The work presented in this paper is part of a larger project on parallel aggregate query processing consisting of three parts: *parallel group-by* [14], *parallel groupby-before-join* [16–18] and *parallel groupby-after-join* [15]. The first part of this project involved with parallelization of Group-By queries on a single table and there is no involvement of join operation. The results have been reported in the Computer Systems: Science and Engineering International Journal [14]. The second part focused on parallelization Groupby-Join queries where the Join attribute is the same as the Group-by attribute resulting that the group-by operation can be performed first before the join for optimization purposes. The outcome of the second part was published at Springer LNCS [17]. In this paper, the focus is mainly on the third part, parallel groupby-after-join, also known as aggregate-join. It concentrates on the parallelization of GroupBy-Join queries where the Group-By attributes are different from the Join attributes; consequently the join operation must be carried out first and then followed by group-by operation.

Previous work [15] identified two parallel processing methods for groupby-after-join queries, namely *join partition method (JPM)*, *aggregate partition method (APM)*. The *JPM* and *APM* methods mainly differ in the selection of partitioning attribute for distributing workloads over the processors.

The objective of this paper is not to propose new parallelization methods for Groupby-After-Join queries, but rather to perform an evaluation of the join partition method and aggregate partition method. The main reason is that most existing work concentrates on identifying parallelization models for this type of query. A complete analysis has yet to be made. In this paper, a through analysis of the two parallelization techniques proposed in our previous work [15] is presented. A comparison between these two parallelization methods is also made.