Complete Algorithm for fragmentation in Data warehouse

ZIYATI ELHOUSSAINE, DRISS ABOUTAJDINE, EL QADI ABDERRAHIM
Mohammed V University GSCM-LRIT
Agdal-Rabat
MAROC

Abstract: - On-line analytical processing (OLAP) queries are strongly affected by the amount data needed to be accessed from the disk. Therefore, there is a need to employ techniques that can facilitate efficient execution of these queries. Data partitioning concept that has been studied in the context of relational databases aims to reduce query execution time and facilitate the parallel execution of queries. In this paper, we develop a new framework based on genetic algorithm for applying the partitioning technique on relational DW schema (star schema) to reduce the total query execution cost. We develop an analytical cost model for executing a set of OLAP queries on a partitioned star schema. We conduct experiments to evaluate the utility of partitioning in efficiently executing OLAP queries. Finally, we show how partitioning can be used to facilitate parallel execution of OLAP queries.

Key-Words: - Partitioning, warehouse, OLAP queries, Genetic algorithm, penalty function, query optimization.

1 Introduction
Data warehousing applications typically involve massive amounts of data that push database management technology to the limit, and also using complex queries due to the presence of join and aggregate operations. Several techniques were proposed and supported by commercial systems. Vertical and horizontal [1, 2, 4] partitioning are two non redundant techniques, several work and commercial systems show their utility and impact in optimizing OLAP queries. But they did not give the same interest to the hybrid or mixed partitioning. In this paper we present a new approach of hybrid partitioning based on genetic algorithm with and without penalty [6]. The experiment results using benchmark APB-1 release II benchmark show enhancement of the process of partitioning in relational data warehouse environment.

2 Vertical and horizontal fragmentation problems
Suppose a relational warehouse modeled by a star schema with d dimension tables and a fact table F. Among these dimension we consider that g tables are fragmented \(g \leq d\). Each dimension table \(D_i\) \((1 \leq i \leq g)\) is partitioned into \(m_i\) fragments: \(\{D_{i1}, D_{i2}, ..., D_{im_i}\}\), where each horizontal fragment \(D_{ij}\) is defined as: \(D_{ij} = \sigma_{c_i^j}^j (D_i)\) with \(c_i^j\) and \(\sigma (1 \leq i \leq g, 1 \leq j \leq m_i)\) represent a conjunction of simple predicates and the selection operator, respectively. Thus the fragmentation schema of the fact table is defined as follows: \(F_i = F \bowtie D_{i1} \bowtie D_{i2} \bowtie ... \bowtie D_{im_i}, \ (1 \leq i \leq m_i)\) with \(\bowtie\) represents the semi jointure operation. Using Relational algebra, vertical algebra could be written as \(F_i = \pi_{attr(F)} (R)\) for all \(j \in \{1, ..., k\}\) verticals fragmentation replaces \(R\) by a set \(\{F_1, ..., F_v\}\) of new relations schema such that:

1) The attributes are distributed, i.e

\[
F_u = \bigcup_{i=1}^{v} F_i,
\]
2) Each relation $r_i$ over $F_i$ is split into relations $r_i = \pi_{r_i}(r_i)$ such that $r = r_i \vDash \ldots \vDash r_i$ holds.

3) In relational algebra, $F_i = F \vDash \ldots \vDash F \pi_{=}$

In both cases horizontal and vertical the problem is: Given a set of dimensions tables $D = \{D_1, D_2, \ldots, D_d\}$ and a set of OLAP queries $Q = \{Q_1, Q_2, \ldots, Q_m\}$ where each query $Q_i$ has an access frequency. The problem of horizontal (vertical) fragmentation consists in determining set dimensions (attributes) to partition and generate a set of fragments in order to minimize total queries cost. We first present the genetic algorithm employed in the horizontal and vertical fragmentation and the impact of introducing the penalty function into the process of fragmentation.

### 2.1 Genetic algorithm for the mixed fragmentation

Starting by the horizontal fragmentation and before presenting the chromosome we should:

1. Extract all simple predicates used by the queries.
2. Generate a complete and minimal set of predicates by using COM_MIN algorithm [4].

For each fragmentation attribute we represented by an array with $n$ cells, where $n$ corresponds to number of its values. Then, each chromosome (fragmentation schema) is represented by multi-dimensional arrays (each array present a fragmentation attribute) [7].

For example we define two fragmentations attributes with there domains figure 1:

![Figure 1](image_url)

An example of a possible solution is:

<table>
<thead>
<tr>
<th>Age</th>
<th>1</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gender</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

Table 1

In table 1, attribute1 and attribute3 will be in site $A$ and attribute2 in the site $B$ and so on for other dimensions (Client, Store). The cost model is given [4] as:

$$\text{Cost}(Q_i) = \sum_{j=1}^{N_{\text{Sel}}} \text{valid}(Q_i, S_j) \prod_{i=1}^{M} \left[ \frac{\text{Sel}^i \times |F| \times L}{PS} \right]$$

(1)

where, $M_j, F, L$ and $PS$ represent respectively the numbers of selection predicates defining the fact fragment of the sub star schema $S_j$, the cardinality of the fact table (number of tuples) $F$, the width, in bytes, of a tuple of a table $F$ and the page size of the file system, respectively. And also

$$\text{valid}(Q_i, S_j) = \begin{cases} 1 & \text{if the sub star schema } S_j \text{ is needed for } Q_i \\ 0 & \text{otherwise} \end{cases}$$

(2)

The total cost of executing a set of queries $Q$ is given by:

$$\text{TC}(Q) = \sum_{j=1}^{m} \text{Cost}(Q_j)$$

(3)

The selectivity factors are chosen using an uniform distribution $UD$.

#### 2.1.1 Selection

The roulette wheel method is used in both algorithms: each chromosome is associated with its fitness value calculated using the cost model defined above, the chromosome with high fitness values have chances to be selected.

#### 2.1.2 Crossover Operation

In both situations we selected a two-point crossover mechanism to gives the same chances to the attributes with high and low number of sub domains.
2.1.3 Mutation
Mutation is needed to create new genes that may not be present in any number of a population and enable the algorithm to reach all possible solutions in the search space. Example Horizontal fragmentation:

![Horizontal Fragmentation Diagram](image)

Fig.2

Initial chromosome has 3 fragments concerning attr2. After mutation process the resulting chromosome has only 2 fragments, in the same way for the vertical partitioning for example:

![Vertical Fragmentation Diagram](image)

Fig.3

We have in the first case 3 vertical fragments, but, in the second chromosome we have two (attr1, attr3 and attr4 are located in site A and attr2 in site B). In the same way, mutations could occur on several attributes of the individual.

3 Problem Solution
The proposed method for solving this problem of fragmentation in a relational data warehouse to minimize the execution time of OLAP queries is a mixed or combined fragmentation based on adaptative Genetic Algorithm (in both cases vertical and horizontal fragmentation):

The algorithm is presented as:

**Input:**
Set of Queries: \( Q \)
Set of a relational data warehouse dimensions: \( D \)

**Output:** vertical and fragmentation schemes.

**Begin**
Extract (access frequencies, simple predicates)
Start \( \text{horizontal\_genetic\_fragmentation}() \)
Start \( \text{vertical\_genetic\_fragmentation}() \)
  // fragment the warehouse horizontally
  // using the generated schemes
  // in both cases

**end**

4 Experimental performance analysis
In order to analyze and test the performance and convergence of the genetic algorithms, we use the dataset from the APB-1 benchmark [3], the vertical fragmentation, programs have been developed in java, all our experiments run on Pentium IV 1,8 GHz (256 Mo). We have considered a set of OLAP queries. Each query has selection predicates, where each one has its selectivity. In the first we compare using only the horizontal fragmentation and no the case where no fragmentation figure 4.

![Horizontal vs No Fragmentation Graph](image)

Fig.4

And then the mixed fragmentation with only horizontal fragmentation figure 5.

![Mixed Fragmentation Graph](image)

Fig.5

Also we want to see the impact of the numbers of vertical fragments to the performance figure 6.

![Vertical Fragmentation Performance Graph](image)

Fig.6

And utility of introducing a penalty function [5, 6] to some infeasible solutions during the process figure 7.
5 Conclusion

In this paper we present a complete algorithm for vertical and horizontal fragmentation using genetic algorithm and a defined cost model evaluating the cost of a set of frequently queries performed in a top of the partitioned relational data warehouse scheme in order to enhance physical design in data warehouse. Our experiments results show that our method can provide a significantly better solution than previous algorithms in terms of minimization of query processing cost [4,7]. There are many other future works especially query nature (use aggregate function SUM, AVG, COUNT, STDDEV, VAR.....).

References:


