A Novel Grouping Aggregation Algorithm for Online Analytical Processing

WEI CHEN[1,2]
1) College of Information Science and Engineering, Yanshan University, Qinhuangdao, China
2) The Key Laboratory for Computer Virtual Technology and System Integration of HeBei Province, Qinhuangdao, China
ysu678@hotmail.com

YONGSHAN LIU[1,2]
1) College of Information Science and Engineering, Yanshan University, Qinhuangdao, China
2) The Key Laboratory for Computer Virtual Technology and System Integration of HeBei Province, Qinhuangdao, China

NING WANG[1,2]
1) College of Information Science and Engineering, Yanshan University, Qinhuangdao, China
2) The Key Laboratory for Computer Virtual Technology and System Integration of HeBei Province, Qinhuangdao, China

***Abstract*—As regard to improve the efficiency of grouping aggregation calculation, the data is compressed by using binary encoding, and the dimension hierarchical grouping attribute set encodings of each dimension table are calculated by using the dimension hierarchy tree. Then different encodings are put into temporary table to be sorted and grouped, and the grouping sequence numbers of fact table records which satisfy the query conditions are computed. Finally, the buffer is located by the grouping sequence number and the grouping aggregation calculation is completed. Theoretical analysis and experimental results show that the proposed algorithm can significantly improve the efficiency of grouping aggregation calculation.**

Keywords-OLAP; grouping aggregation; sequence number

I. INTRODUCTION

Online Analytical Processing (OLAP) is a model of multidimensional data analysis [1]. It is a critical application technology in the field of data warehouse, and mainly used to deal with the complex query of huge amounts of data. As a result, the analyst can quickly access information from many aspects [2]. Fast query response speed is usually required in OLAP. Therefore, to improve the performance of OLAP query processing is an important research area in the field of data warehouse.

In recent years, many query algorithms for OLAP are proposed. The characteristics of star schema are taken into consideration in [3], and the join of fact table and many dimension tables is obtained by using the hash table. In [4], firstly, according to the query conditions, all the grouping attribute values of each dimension table are sorted and grouped to obtain their grouping sequence number. Then the key word mapping technique is used to compress the sorting key words. In Group number based Aggregation with Multi-table join (MuGA) [5] algorithm, the approach for dealing with dimension tables is same with [4]. Then the grouping sequence numbers of fact table records are calculated by using these different dimension tables. Finally the grouping aggregation calculation is completed through buffer location. The sorting operations are avoided in MuGA algorithm, so the performance of grouping aggregation algorithm is improved. However, the feature that dimension attributes are hierarchical is ignored in all the above algorithms. Many algorithms based on dimension hierarchies are proposed at present. In [6], the multidimensional hierarchical B+ tree is used to be the index. As a result, the query is performed efficiently. But only the problem of data storage is researched. In [7], the definitions of dimension class, visible dimension hierarchy and invisible dimension hierarchy are proposed. And the condition that invisible dimension hierarchy needs to be converted to visible dimension hierarchy is confirmed. The decline of the system's overall functionality is avoided. However, the storage space of dimension tables is increased. Given the characteristics of the complex multidimensional hierarchies, the hierarchy combined surrogate and pre-grouping sorting on the basis of bitmap join index is adopted in [8]. The efficiency is improved by converting join and aggregation operations of complex multidimensional hierarchies to the queries of fact table. In [9], through the use of the dimension hierarchical encoding and its prefix path, the matching dimension hierarchical encoding is rapidly retrieved and the query range set of each dimension table is calculated. Dimension tables are not needed, so the efficiency of OLAP queries is improved. But when the grouping aggregation calculation is performed, a large number of intermediate results are stored, and these results are deleted when the calculation is finished. Then the fact table records which are not calculated and satisfy the query conditions are inserted. Much time is taken by repeatedly inserting and deleting intermediate results.

In order to improve the efficiency of grouping aggregation calculation, Grouping Aggregation for OLAP Based on the Dimension Hierarchical Encoding (GABDHE) algorithm is proposed. Firstly, according to the query condition, the encodings of each dimension table are calculated. The different encodings are put into temporary table to be sorted and grouped. Then, the fact table records are filtered out by the encodings, and the grouping sequence numbers of the records are calculated. Finally, the grouping aggregation calculation is completed by locating the buffer.

This work is supported in part by the Natural Science Foundation of Hebei Province P.R.China under Grant No.F2009000473 and the 2008 Election Information Industry Development Found of Ministry of Information Industry under Grant No.[2008]97.

II. THE GABDHE ALGORITHM

Definition2.1. When the attribute on the j[th] level of the dimension table D_i is grouped, the attribute is called dimension hierarchical grouping attribute. It is represented as GAh_j^i.

Definition2.2. Nodes on the same level have the same prefix path, so GAh_j^i and its ancestor nodes' attributes are combined into a collection according to the level. The collection is called dimension hierarchical grouping attribute set. The dimension hierarchical grouping attribute set of D_i is represented as GAh_i.

Definition2.3. B^{GAh_i} is the dimension hierarchical grouping attribute set encoding of D_i, and it can be calculated in (1).

$$B^{GAh_i} = (...((B^{GAh_1^i} << BitGAh_2^i \mid B^{GAh_2^i}) << BitGAh_3^i \mid B^{GAh_3^i})...) << BitGAh_j^i \mid B^{GAh_j^i} \quad (1)$$

$B^{GAh_j^i}$ is the dimension hierarchical grouping attribute encoding, $BitGAh_j^i$ is the bit number of the binary encoding of GAh_j^i.

The central idea of GABDHE algorithm: firstly, the complex query is converted to sub-query for each dimension table, and the dimension hierarchical grouping attribute sets of each dimension table are obtained. Then all dimension hierarchical grouping attribute encodings of each dimension table are obtained by finding corresponding dimension hierarchical tree. Finally, the fact table records are screened out by using the dimension hierarchical grouping attribute set encodings of each dimension table. When all aggregation results can be stored in the buffer, the buffer is located by the grouping sequence number, and the measures of records which have the same grouping sequence numbers are calculated.

GABDHE Algorithm.

Input: the fact table *FT*, the dimension tables DT_1, DT_2,..., DT_m, , query condition *Q*

Output: the table of the aggregation measures $Agg\_Mes\_result(GAh_1, GAh_2, ..., GAh_m, A)$

(1) Analyze the query condition Q, convert it to sub-query conditions for each dimension, that is $Q_1, Q_2, ..., Q_i, ..., Q_m$. Meanwhile, get the dimension hierarchical grouping attribute set of each dimension table: $GAh_1, GAh_2, ..., GAh_m$ and the aggregation attribute $sum(Agg)$;

(2) for $i = 1$ to m do

(3) Submit Q_i, and get all the dimension hierarchical grouping attribute encodings by finding the corresponding dimension hierarchical tree;

(4) Calculate the dimension hierarchical grouping attribute set encodings;

(5) Put the different dimension hierarchical grouping attribute set encodings into temporary table to be sorted and grouped, compute the corresponding grouping sequence number $GroupNo^{GAh_i}$ and the corresponding groups number $Groups^{GAh_i}$;

(6) End for

(7) Initialize the *GroupBuf*, that is $GroupBuf = \prod_{i=1}^{m} Groups^{GAh_i}$;

(8) While (scan the fact table *FT* in sequence, and filter out the records by using the dimension hierarchical grouping attribute set encodings of each dimension table.) do

(9) $GroupNo = 0$;

(10) for $i = 1$ to m do

(11) Get the $GroupNo^{GAh_i}$ by using the corresponding B^{GAh_i};

(12) if i >1 then

(13) $GroupNo = GroupNo * Groups^{GAh_i}$;

(14) $GroupNo = GroupNo + GroupNo^{GAh_i}$;

(15) End if

(16) End for

(17) Generate new record $R^1(GroupNo, Agg)$;

(18) Locate the buffer *GroupBuf* by using *GroupNo*, and put *Agg* into the field *sumA* of the cell of *GroupBuf* which has the same *GroupNo* with R^1;

(19) $sumA = sumA + Agg$;

(20) End while

(21) for $i = 0$ to $Groups - 1$ do

(22) Get *sumA* of the i+1 record of *GroupBuf*;

(23) if *sumA* !=0 then

(24) $GroupNo = i$;

(25) for $k = m$ to 1 do

(26) $GroupNo^{GAh_k} = GroupNo \bmod Groups^{GAh_k}$;

(27) $GroupNo = GroupNo / Groups^{GAh_k}$;

(28) Get the corresponding B^{GAh_k} by using the $GroupNo^{GAh_k}$ to access the temporary table;

(29) End for

(30) Aggregation results $(B^{GAh_1}, B^{GAh_2}, ..., B^{GAh_m}, sumA)$ are written to the external memory, and connected to the non-dimension hierarchical attributes;

(31) End if

(32) End for

(33) return $Agg\_Mes\_result(GAh_1, GAh_2, ..., GAh_m, A)$

Given n is the number of fact table's records, that is |FT|=n, r is the number of the aggregation results. The number of fact table's records is much bigger than that of aggregation results, that is n>>r. So the computational complexity of the GABDHE algorithm is *O(n).*

III. EXPERIMENTAL RESULTS

The experiments are conducted under the circumstance of a 1.8GHZ Pentium Personal Computer with 2GB memory. The dataset used in the experiments is the ERP data of a small enterprise.

In the first set of experiments, we compare the data compressions of the two algorithms. The numbers of different attributes of the three levels are: I: 10*10*100, II: 10*100*99, III: 10*100*100, IV: 99*100*100, V: 100*100*100. The experimental result is shown in figure 1. From the experiment result, we can see that, on the average, the bits of data compressed by MuGA is 3.0 times longer than GABDHE algorithm.

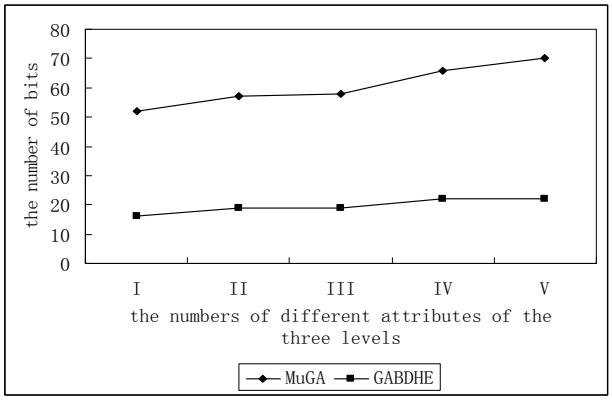

Figure 1. Performance comparison for data compressions.

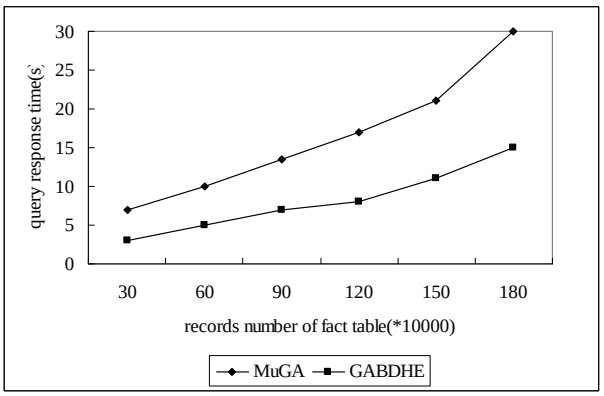

Figure 2. Performance comparison for query response times.

In the second set of experiments, we compare the query response times of the two algorithms. The experimental result is shown in figure 2. From the experiment result, we can see that, on the average, the response time of MuGA is 2.0 times longer than GABDHE algorithm. In particular, when the record number of fact table is 180, the response time of MuGA is 2.3 times longer than GABDHE algorithm. This is because the fact table needs to connect the dimension tables in the MuGA algorithm, and a lot of time is consumed by the connection operation.

IV. CONCLUSION

In this paper, based on the characteristic that dimension attributes are hierarchical, GABDHE algorithm is proposed. The definition and formula of the dimension hierarchical grouping attribute set encoding are given, and the binary encoding is used to replace the foreign keys of fact table. As a result, the target data is compressed. In addition, according to the query condition, all dimension hierarchical grouping attribute encodings of each dimension table are obtained. Then the dimension hierarchical grouping attribute set encodings are calculated, and the operation of pre-grouping dimension tables is completed. Finally, the fact table records are screened out by using the dimension hierarchical grouping attribute set encodings, and the grouping sequence numbers which are used to locate the buffer to finish the grouping aggregation calculation are given. Intermediate results are not stored in GABDHE algorithm, so it avoids the operation of repeatedly inserting and deleting records. The experimental results show that the query performance is significantly improved in the GABDHE algorithm.

REFERENCES

[1] Y. S. Zhang, M. Jiao, Z. W. Wang, S. Wang, and X. Zhou, "One-size-fits-all OLAP Technique for Big Data Analysis," Chinese Journal of Computers, vol. 10, pp. 1936-1946, 2011.

[2] L. Y. Wang, and H. N. Lu, "Aggregation Implementation on MOLAP Dimensions with Hierarchies," Computer Engineering and Design, vol. 19, pp. 4595-4597, 2007.

[3] X. D. Jiang, and L. Z. Zhou, "A Multi-table Join Algorithm for Data Warehouse Query Processing," Journal of Software, vol. 2, pp. 190-195, 2001.

[4] X. D. Jiang, J. H. Feng, and L. Z. Zhou, "A Novel Aggregation Algorithm for Online Analytical Processing Query Evaluation," Journal of Software, vol. 1, pp. 65-70, 2002.

[5] J. H. Feng, X. D. Jiang, and X. H. Meng, "An Aggregation Algorithm Based on Group Numbers," Journal of Software, vol. 2, pp. 222-229, 2003.

[6] Q. J. Zhao, S. F. Chen, and K. F. Hu, "Extensible Storage Structure Based on Multidimensional Hierarchical Cube," Journal of Applied Sciences, vol. 2, pp. 166-170, 2007.

[7] Y. Zhang, X. F. Xia, and G. Yu, "Method for Conversion from Dimension Classes to Dimension Hierarchies," Journal of Chinese Computer Systems, vol. 8, pp. 1498-1501, 2008.

[8] Z. H. Huang, Y. S. Xue, J. J. Duan, and J. B. Wang, "A Join and Aggregate Algorithm for Complex Multi-Dimensional Hierarchies," Journal of Computer Research and Development, vol. 8, pp. 1345-1351, 2004.

[9] K. F. Hu, Y. S. Dong, and L. Z. Xu, "A Novel Aggregation Algorithm for Online Analytical Processing Queries Evaluation based on Dimension Hierarchical Encoding," Journal of Computer Research and Development, vol. 4, pp. 608-614, 2004.