Lecture Notes in Electrical Engineering 215

Kuinam J. Kim · Kyung-Yong Chung *Editors*

## IT Convergence and Security 2012

Volume 2

The proceedings approaches the subject matter with problems in technical convergence and convergences of security technology. This approach is new because we look at new issues that arise from techniques converging. The general scope of the proceedings content is convergence security and the latest information technology. The intended readership are societies, enterprises, and research institutes, and intended content level is mid- to highly educated personals.

The most important features and benefits of the proceedings are the introduction of the most recent information technology and its related ideas, applications and problems related to technology convergence, and its case studies and finally an introduction of converging existing security techniques through convergence security. Overall, through the proceedings, authors will be able to understand the most state of the art information strategies and technologies of convergence security.

Electrical Engineering

ISBN 978-94-007-5859-9

▶ springer.com

Kim · Chung *Eds.*

IT Convergence and Security 2012

2

Lecture Notes in Electrical Engineering 215

Kuinam J. Kim
Kyung-Yong Chung
*Editors*

# IT Convergence and Security 2012

Volume 2

Springer

# Improved View Selection Algorithm in Data Warehouse

**Jong-Soo Sohn, Jin-Hyuk Yang and In-Jeong Chung**

**Abstract** In order to minimize the query processing time, a data warehouse maintains materialized views of aggregate data derived from a fact table. However, due to the expensive computing and space costs materializing the whole relations instead of part of the relations results in much worse performance. Consequently, proper selection of appropriate views to be materialized is very important to get a precise and fast response in the data warehouse. However, this view selection problem is NP-hard problem, and there have been many research works on the selection of materialized views. In this paper we propose an improved algorithm to overcome problems of existing view selection algorithms. In the presented algorithm, we first construct the reduced tables in the data warehouse using clustering method among data mining techniques, and then we consider the combination of reduced tables as the materialized views instead of combination of the original base relations. For the justification of the suggested idea, we show the experimental results in which time as well as space costs are about 1.7 times better than the conventional approaches which considered all the tuples in a relation to materialize.

**Keywords** Materialized views · Data warehouse · Clustering

J.-S. Sohn · I.-J. Chung (✉)
Department of Computer and Information Science, Korea University,
208 Seochangri, Sejong City, South Korea
e-mail: chung@korea.ac.kr

J.-S. Sohn
e-mail: mis026@korea.ac.kr

J.-H. Yang
Korea Institute of Planning and Evaluation for Technology in Food Republic of Korea, An-yang, South Korea
e-mail: jhy@ipet.go.kr

K. J. Kim and K.-Y. Chung (eds.), *IT Convergence and Security 2012*,
Lecture Notes in Electrical Engineering 215, DOI: 10.1007/978-94-007-5860-5_111,

# 1 Introduction

The relational database (RDB) is designed mainly for real time transaction processing such as On Line Transaction Processing (OLTP), it is improper for multi-dimensional data analysis such as On Line Analytic Processing (OLAP) or decision support system. In order to overcome this problem and to maximize the use of information from the huge amount of accumulated data effectively, data warehouse was introduced.

A view in a data warehouse is a virtual relation that is derived from a base relation or other view. Since we cannot maintain and materialize all possible views in a data warehouse due to the expensive computing time and space constraint, only a subset of views should be selected to be materialized. However, proper selection of materialized views in a data warehouse is NP-hard problem [1]. So far, there have been many research works on the selection of materialized views in a data warehouse [2–8], however these previous works have problems which we will describe in the following section.

In order to overcome these problems of the related previous works, we propose an efficient algorithm which uses clustering technique to select the materialized views, and thus can accelerate the response time as well as save the storage space. For the justification of the proposed algorithm, we show two independent experimental results: one is the 'pubs' database, used for educational purposes, and the other is much larger Enterprise Resource Planning (ERP) data base, currently being used in one of the leading enterprise in computer form design and manufacturing in South Korea. Both experimental results show that both space and time costs are approximately 1.8 times better than the conventional approaches.

# 2 Related Works

For the proper selection of materialized views, [2] proposed a greedy algorithm to minimize the query processing cost in the special case of the "data cubes". However this paper does not mention the view maintenance cost and space constraint. In [3], an algorithm based on heuristic greedy method was proposed. However, this research has an inefficient evaluation tool. In research [4], the suggested $HA_{mvpp}$ algorithm requires too much time to produce Multiple View Processing Plan (MVPP).

Algorithms in [1, 3] aim at minimization of the query processing cost. [1] is a variation of this algorithm and it aims at minimizing the total query processing time under the constraint of total view maintenance cost. Recently some artificial intelligence approaches such as genetic algorithms have been proposed to achieve the improved results in the view maintenance and query processing [5–8]. However these genetic algorithms have some problems due to the impractical solutions [9].

# 3 IVSA: Improved View Selection Algorithm

## *3.1 Improved View Selection Algorithm*

In a different manner of conventional algorithms, we present an improved algorithm for selecting views to materialize using the clustering method among data mining techniques [10–13]. The proposed algorithm Improved View Selection Algorithm (IVSA), firstly finds high density clusters from the dimensions of the given tables, and secondly, produces the reduced tables using the found clusters. Next, the MVPP is produced using the reduced tables, and finally, materialized views are selected from the MVPP in accordance with cost estimation. The technique of materializing views is required to minimize the query response time in a data warehouse, which provides guidelines to enterprise managers through the analysis of market trends by supporting various OLAP capabilities.

The proposed IVSA has following four steps:

Step 1: Find the high-density clusters from k-dimensional relations.
Step 2: Produce the reduced tables using upper and lower bound values of the clusters found.
Step 3: Establish the MVPP using reduced tables.
Step 4: Select the materialized views while considering improvement of query response time and view maintenance cost.

IVSA($\tau$, *n, T, Q, SC, UDT, UET*) {
/* : user's input threshold, n: number of queries or tables, T: set of target tables */
/* Q: set with n queries, SC: user's input space constraint */
/* UDT: user's input clustering dimensions which must be included */
/* UET: user's input clustering dimensions which must be excluded */
$C = \emptyset$;/* set of clusters */RT $= \emptyset$;/* set of reduced tables */
$VP = \emptyset$;/* set of views used in query processing plan */
$MV = \emptyset$;/* set of views to be materialized */
**for** ($i = 0$; $i < n$; $i$ ++) {$C = C \cup$ **find_cluster**($\tau$, *n, Ti, UDT, UET*);}
**for** ($i = 0$; i $< n$; $i$ ++) {$RT = RT \cup$ **generate_reduced_table**(*Ci, Ti, RTi*);}
**make_mvpp**(*n, Q, RT*); **select_view**(*VP*); **return** *MV*;}
/* step 1 */**find_cluster**((*τ, n, Ti, UDT, UET*) {
$T = Ti$; *target* $= 0$;/* variable for attributes' reflection density */
**for** ($i = 0$; $i < n$; $i$ ++) {for ($j = 0$; $j < n$; $j$ ++) {
/* primary key, foreign key, and user's input dimension of tables are excluded */
**if** (*Ti.dj == primary_key* || *Ti.dj == foreign key* || *Ti.dj == UETi.dj*) **continue**;
/* if a dimension is user's specified input dimension, it is included */
**if** (*Ti.dj == UDTi.dj*) {**for** ($k = 0$; *Ti.di.low[k]* ! = NULL; $k$ ++) {
/* select a range of lower bound and upper bound for cluster */
*C.i = Ti.di.low[k], Ti.di.high[k]*;} **break**;}/* move to the next table */

**else if** ($\prod(Ti.di, Ti.dj) > \tau$ && $[C.i] > target$) {$target = [C.i]$;
**for** ($k = 0$; $Ti.di.low[k]$ ! = NULL; $k$ ++) {$C.i = Ti.di.low[k]$, $Ti.di.high[k]$;}}} **return** $C$;}
/* step 2 */**generate_reduced_table**($Ci$, $Ti$) {/* operator ← returns index */
$tmp \leftarrow Ti.Ci.low[0]$;
**for** ($k = 0$; $Ti.Ci.low[k]$ ! = NULL; $k$ ++) {
**while** ([$tmp$] ≥ $Ti.Ci.low[k]$ && [$tmp$] ≤ $Ti.Ci.high[k]$) {Copy tuple from $Ti$ to $RTi$; $tmp$ ++;}}
**return** $RTi$;}
/* step 3 */**make_mvpp**($n$, $Q$, $RT$) {
**for** ($i = 0$; $i < n$; $i$ ++) {
Make vpi using Q and RT as base relation instead of T;
Count the number of nodes in vpi and save into NNi;}
**for** ($i = 0$; $i < n$; $i$ ++) {**for** ($j = 0$; $j < NNj$; $j$ ++) {**for** ($k = 0$; $k < NNk$; $k$ ++) {
$VP = VP \cup vpi$; **if** ($vpi.nodej$ == $VPi.nodek$) $VPi.nodek.fq$ ++;}} **return** $VP$;}
/* step 4 */**select_view**($VP$) {
**for** ($i = 0$; $i < n$; $i$ ++) {**for** ($j = 0$; $j < n$; $j$ ++) {
$VPi.Ca = VPi.Ca + VPi.nodej.Ca$; $VPi.Cm = VPi.Cm + VPi.nodej.Cm$;
$VPi.Cv = VPi.Cv + VPi.Ca + VPi.Cm$;}
$VP.Ca = VP.Ca + VPi.Ca$; $VP.Cm = VP.Cm + VPi.Cm$; $VP.Cv = VP.Cv + VP.Ca + VP.Cm$;}
/* sort the elements of VP in ascending order according to the value of Cv */
**Sort**($VP$);/* select views within the bound of specified SC */
**for** ($i = 0$; $i < n$; $i$ ++) {/* operator Σ returns storage space */
**if** ($\Sigma TMV < SC$) {$MV = MV \cup VPi$; $MV.Cv = MV.Cv + VPi.Cv$;} **else break**;}
**return** $MV$;}

## 3.2 Properties of Improved View Selection Algorithm

In the first step of the algorithm, the high-density cluster for target base relations is found using the clustering method of among data mining techniques. For each dimension of the table, the dimension with the maximum density value is selected, which exceeds the user's input threshold $\tau$. As a novel approach which is not considered in conventional algorithms, this technique with clustering is crucial from the standpoint of providing an opportunity to implicitly utilize important information overlooked.

In the second step of the algorithm, reduced tables containing the only corresponding tuples are produced by using the lower and upper bound values of the selected dimension for each table. While traditional algorithms consider all the tuples of a base relation for materializing, the targets of materializing are restricted to the tuples of the reduced tables in the proposed algorithm IVSA. Therefore, it

can achieve the goals of improvement in query response time and saving of storage for views.

In the third step of the algorithm, we produce an MVPP using the reduced tables generated in the previous step. The existing algorithm [4] proposed the 0–1 integer programming method and $HA_{mvpp}$ for establishing MVPP. While this 0–1 integer programming technique produces optimal MVPP, it takes too much time to implement. In our algorithm, we propose the off-line procedure for establishing MVPP using query frequency.

In the fourth step of the algorithm, the views which can derive benefits in the case of materialized ones were selected within the bounds of the user's input space constraint, while considering view processing time cost and view maintenance cost in the produced MVPP. The conventional algorithms consider only the cost for join operation and restrict query frequency to the query itself. We argue that these cost estimation methods leave out some important factors in cost. In the IVSA, cost for the select operation is supplemented to cost estimation formulation.

# 4 Implementation Results and Analysis

## *4.1 Experimental Result of Materials Handling in the ERP*

In this chapter, we present an experimental result on the large Enterprise Resource Planning (ERP) data base, currently being used in one of the leading enterprise in computer form design and manufacturing in South Korea. We use the keyword-based search method to accelerate the query response time. The ERP database of that company has altogether 981 tables where each table has rows from thousands to tens of thousands. In this paper we show some examples of making materialized view from the materials handling tables in the ERP database. Followings are the tables used in purchasing materials for production in the company.

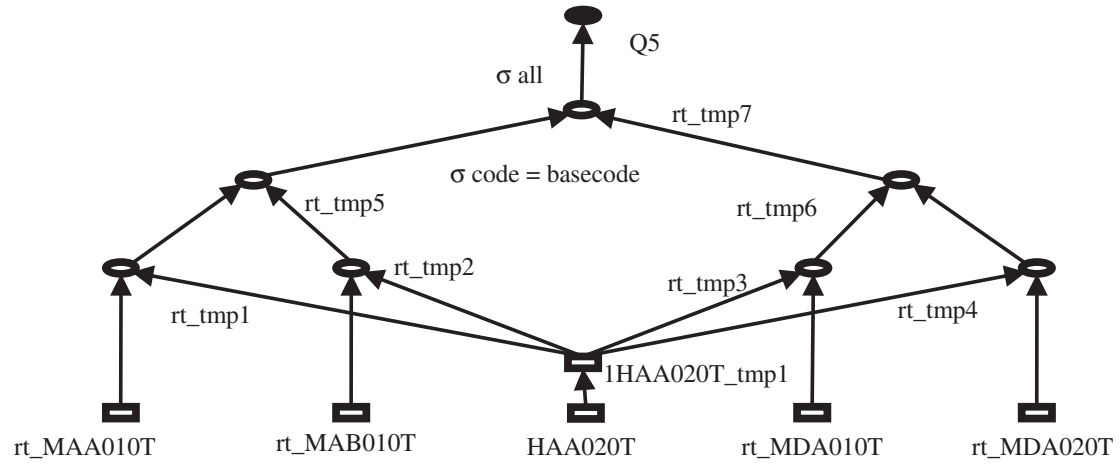


**Fig. 1** MVPP for the query Q5

**Table 1** Cost computation for the query Q1 with reduced tables

| Table | *fq* | *t#* | *Ca* Q5 | *Cm* Q5 | *Cv* Q5 |
|---|---|---|---|---|---|
| HAA020T | 1 | 483 | 483 | 0 | 46,851 |
| rt_MAA010t | 1 | 1,372 | 133,267 | 0 | 133,267 |
| rt_MAB010T | 1 | 2,148 | 202,119 | 0 | 202,119 |
| rt_MDA010T | 1 | 8,567 | 803,262 | 0 | 803,262 |
| rt_MDA020T | 1 | 1,652 | 157,093 | 0 | 157,093 |
| HAA020_tmp | 1 | 93 | 1,428,179 | 0 | 1,428,179 |
| rt_tmp1 | 1 | 194,333 | 204,513 | 390,869 | 595,382 |
| rt_tmp2 | 1 | 15,891 | 16,437 | 31,681 | 48,118 |
| rt_tmp3 | 1 | 912,767 | 91,466 | 984,148 | 1,075,614 |
| rt_tmp4 | 1 | 192,778 | 196,452 | 381,445 | 577,897 |
| rt_tmp5 | 1 | 137,445 | 142,483 | 274,329 | 416,812 |
| rt_tmp6 | 1 | 1,449,420 | 1,445,282 | 2,836,808 | 4,282,090 |
| rt_tmp7 | 1 | 27,843 | 27,843 | 4,801,296 | 4,829,139 |

We show the detailed steps to solve several query processing for purchasing items and placing order of materials in the company and then evaluate their performance. In order to improve response time of keyword based search, we consider the articles of the law containing the related notices as clustering targets. After generating the reduced table for the materials handling of ERP: ERP handling materials, we subsequently generated MVPP on the following query. Consider the following query Q1.

(Q1) Make a list for unit prices and enterprises on the orders

Query Q1 searches the table concerning clients and materials. User enters employee number and query Q1 displays the result to the user. Then query Q1 makes temporary table 'HAA020T_tmp1' of input employee number by searching HAA020T table. HAA202T_tmp1 is used to search specific information for clients and materials Fig. 1 displays MVPP of query Q1.

Table 1 is a result of time cost estimation in executing query Q1 when we insert the material purchase data into this algorithm—IVSA. Table 2 is a computation result of executing query Q1 with conventional method without using reduced table.

In this example, we set up user variables SC as 150,000. And then tuples rt_temp1, rt_temp3, rt_temp4 and rt_temp6 are selected one by one as materialized view. As displayed in Table 3, we can check that algorithm presented in this paper shows 1.62 times better storage space and 1.9 times better performance in terms of total cost.

Table 3 is a summarization of Tables 1 and 2. Table 3 shows that the query processing time is 1.77 times faster and storage space is 1.79 times smaller using the suggested method in this paper when user space constraint variables SC were not given.

**Table 2** Cost computation for the query Q1 without reduced tables

| Table | *fq* | *t#* | *Ca* Q5 | *Cm* Q5 | *Cv* Q5 |
|---|---|---|---|---|---|
| HAA020T | 1 | 483 | 483 | 0 | 483 |
| MAA010t | 1 | 2,332 | 227,759 | 0 | 227,759 |
| MAB010T | 1 | 3,652 | 356,579 | 0 | 356,579 |
| MDA010T | 1 | 14,564 | 1,422,165 | 0 | 1,422,165 |
| MDA020T | 1 | 2,808 | 274,240 | 0 | 274,240 |
| HAA020_tmp | 1 | 93 | 2,011,820 | 0 | 2,011,820 |
| tmp1 | 1 | 330,366 | 350,188 | 666,942 | 1,017,130 |
| tmp2 | 1 | 27,015 | 28,635 | 54,537 | 83,172 |
| tmp3 | 1 | 1,369,151 | 1,451,300 | 2,764,041 | 4,215,341 |
| tmp4 | 1 | 308,445 | 326,951 | 622,688 | 949,639 |
| tmp5 | 1 | 206,168 | 218,538 | 416,211 | 634,748 |
| tmp6 | 1 | 2,029,188 | 2,150,939 | 4,096,525 | 6,247,464 |
| tmp7 | 1 | 28,170 | 28,170 | 8,448,525 | 8,476,695 |

**Table 3** Performance comparison on the database of ERP system

| | | Conventional algorithms | IVSA |
|---|---|---|---|
| Partial materialization case | Materialized views | tmp1, tmp2, tmp5, tmp6 | rt_tmp1, rt_tmp2, rt_tmp5, rt_tmp6 |
| | Total cost | 127,417 | 72,632 |
| | Storage space | 41,769 | 23,612 |
| Full materialization case | Materialized views | ALL | ALL |
| | Total cost | 593,591 | 332,180 |
| | Storage space | 84,713 | 47,201 |

## 5 Conclusions

As a technique of materialized views, this paper proposes IVSA algorithm which adopts the data mining clustering technique. In the proposed algorithm, the user can specify a dimension for mandatory clustering. This function excludes the possibility of leaving out the important information. The user can also specify the threshold value that indicates the compression strength of clusters. Finally, the user is able to input a space constraint value within which materialized views are selected. These kinds of user interfaces are not found in conventional algorithms.

The proposed algorithm IVSA, firstly, finds high density clusters from the dimensions of the given tables, and secondly, produces the reduced tables using the found clusters. Next, the MVPP is produced using the reduced tables, and finally, materialized views are selected from the MVPP in accordance with cost estimation.

As shown in the experimental results, the proposed algorithm achieves 1.76 times better on average performance in terms of both query response time and storage space of materialized views. Even in the case where the value of the space

constraint variable is not specified (i.e., when we assume there is no space constraint), our algorithm shows 1.78 times better on average performance database for ERP database, respectively.

Broadly, there lie two issues with the data warehouse. The first is selection of materialized views, and the other is maintenance of the views for consistency of a data warehouse. IVSA in this paper is in regards to the first issue. As future works, we will focus on how to update and maintain the reduced tables when there occurs any update in the source data.

## References

1. Gupta H, Mumick I (1999) In: Selection of views to materialize under a maintenance cost constraint. Database theory—ICDT'99, pp 453–470
2. Harinarayan V, Rajaraman A, Ullman JD (1996) Implementing data cubes efficiently. SIGMOD Rec 25(2):205–216
3. Gupta H (1997) In: Selection of views to materialize in a data warehouse. Database theory—ICDT'97, pp 98–112
4. Yang J, Karlapalem K, Li Q (1997) Algorithms for materialized view design in data warehousing environment. 136–145
5. Zhang C, Yao X, Yang J (2001) An evolutionary approach to materialized views selection in a data warehouse environment. IEEE Trans Syst Man Cybern Part C: Appl Rev 31(3):282–294
6. Lee M, Hammer J (2001) Speeding up materialized view selection in data warehouses using a randomized algorithm. Int J Coop Inf Syst 10(3):327–354
7. Choi CH, Yu J, Gou G (2002) What difference heuristics make: Maintenance-cost view-selection revisited. In: Advances in web-age information management, 313–350
8. Yu JX, Yao X, Choi CH et al. (2003) Materialized view selection as constrained evolutionary optimization. IEEE Trans Syst Man Cybern Part C-Appl Reviews 33(4):458–467
9. Ashadevi B, Subramanian R (2009) Optimized cost effective approach for selection of materialized views in data warehousing. Int J Computer Sci Technol 9(1)
10. Chen MS, Han J, Yu PS (1996) Data mining: An overview from a database perspective. IEEE Trans Knowl Data Eng 8(6):866–883
11. Agrawal R, Imielinski T, Swami A (1993) Database mining: a performance perspective. IEEE Trans Knowl Data Eng 5(6):914–925
12. Berson A, Smith SJ (1997) Data warehousing, data mining, and OLAP. Computing McGraw-Hill, New York
13. Kalnis P, Mamoulis N, Papadias D (2002) View selection using randomized search. Data Knowl Eng 42(1):89–111