Materialized View Selection Using Iterative Improvement

T.V. Vijay Kumar and Santosh Kumar

School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi-110067, India

Abstract. A data warehouse is designed for the purpose of answering analytical queries, posed on them for decision making. The complex and exploratory nature of analytical queries which, when processed against large historical information in the data warehouse, consume a lot of time for processing. As a result, the query response time is high. Materialized views provide an alternative platform to address this problem of poor query response time. These views store aggregated and summarized information separately from a data warehouse with the aim of answering analytical queries. All views cannot be materialized, as the number of views is exponential in respect of number of dimensions. Also, optimal view selection is an NP-Complete Problem. Several view selection algorithms exist with most selecting views empirically or based on heuristics like greedy or evolutionary. In this paper, an algorithm based on iterative improvement, a randomized search heuristic technique for selecting top-K views for materialization is proposed. It is shown that the proposed algorithm, in comparison to a well known greedy algorithm, is able to select comparatively better quality views for higher dimensional data sets.

1 Introduction

A large amount of data gets generated, on daily basis, in the various local data sources spread across the globe. This data can be exploited for the purpose of aiding decision making. There are two ways to access this data namely, the on-demand approach or in-advance approach[36]. In the former, the data is integrated from these local data sources based on the content of the user query. The latter approach necessitates the data to be integrated aprior and stored in a central repository. The queries are then posed against this central repository. The latter approach, also referred to as the data warehousing approach, stores data in a central repository called a data warehouse.

A data warehouse contains subject oriented, integrated, time variant and non-volatile data with the aim of supporting decision making [13]. The historical data available in the data warehouse is used to predict future trends in data, which can be useful in laying out business strategies for future. Unlike the operational data sources, where transaction processing is done, data warehouse processing is done with the purpose of helping in timely decision making. Queries for decision making are usually analytical in nature and involve aggregation and summarization of data. This is carried out in an exploratory manner by continuously refining the original query till a desirable result is obtained. The major concern with these analytical queries is that their response times are very high when they are processed against a large data warehouse. This leads

N. Meghanathan et al. (Eds.): Advances in Computing & Inf. Technology, AISC 178, pp. 205–213.
springerlink.com

to delay in decision making. There are several query optimization algorithms proposed in literature[5, 9] but they do not scale up, beneficially, for continuously growing data in a data warehouse[19]. An alternative approach concerned with materialized views[21] has been used to address this problem.

Materialized views, unlike traditional virtual views, store data along with their definitions. These views contain aggregate and summarized information stored separately from a data warehouse. The queries, instead of being posed against the data warehouse, are posed against these materialized views. These being significantly smaller in size, when compared with a data warehouse, can considerably reduce the query response time. This can only be possible if these materialized views contain information that is relevant for answering user queries. The identification of this relevant information from the data warehouse is referred to as the view selection problem[6].

View selection is concerned with selecting appropriate sets of views, that can improve the query response times, while conforming to resource constraints like storage space, memory usage, CPU usage etc[6, 8, 37, 38]. View selection is formally defined in [6] as "Given a database schema R, storage space B, and a workload of queries Q, choose a set of views V over R to materialize, whose combined size is at most B". The views cannot be arbitrarily selected as it may result in selection of views that may not be capable of answering analytical queries and thus becoming an unnecessary storage space overhead. Further, as the number of views grows exponentially with increase in the number of dimensions, all possible views cannot be materialized due to available space constraint. Also optimal view selection, from among all possible views, is shown to be an NP-Complete problem[11]. Alternatively, views can be selected empirically or query based[1,2 3, 4, 7, 16, 17, 18, 23, 24, 29, 30, 33] or based on heuristics like greedy[8, 10, 11, 22, 25, 26, 27, 28, 31, 32, 34, 35], evolutionary[12, 15, 39, 40] etc. This paper focuses on the use of randomized technique to select views for materialization.

In this paper, an algorithm that selects top-K views, from amongst all possible sets of views in a multidimensional lattice, using Iterative Improvement[14, 20] is proposed. This algorithm considers each set of top-K views to be a state in a solution space with an associated total cost of evaluation all the views [11], referred to as Total View Evaluation Cost(TVEC). The algorithm selects top-K views by performing random series of downhill moves in the solution until it attains local minima. A pre-specified number of local minima are computed and thereafter the top-K views with minimum TVEC, are selected. The approach uses TVEC as defined in [11]. Experiments are conducted and the proposed algorithm is compared with the most fundamental greedy algorithm given in [11], hereafter in this paper referred to as HRUA, on the TVEC value of the views selected by them. The proposed algorithm is able to select comparatively better quality views.

The paper is organized as follows: The proposed iterative improvement based approach is given in section 2. Section 3 compares the proposed algorithm with HRUA. The conclusion is given in Section 4.

2 Approach

As discussed above, selection of views for higher dimensional data sets using deterministic algorithms becomes growingly infeasible, as the number of views grows

exponentially with the number of dimensions. To overcome this problem, another class of algorithms i.e. Randomized Algorithms, are used to select the views. Randomized algorithms consider each solution to be a state in a solution space with an associated cost, based on a problem specific cost function [14, 20]. These perform random walks on the state space via a series of moves, which are used to construct edges between different solutions in the solution space. Two solutions are connected to each other if, and only if, one can be obtained from the other by applying only one move. Algorithms in this category traverse the graph, so formed, and terminate as soon as a desirable solution is found or when the time limit is exceeded. The approach uses iterative improvement [14], a randomized search heuristic technique, for selecting top-K views for materialization. The iterative improvement algorithm is briefly discussed next.

2.1 Iterative Improvement

The Iterative Improvement algorithm successively improves the initial solution through the series of iterations. It is based on local optimization and uses a well known, hill-climbing, strategy. The algorithm is given in Fig 1. The algorithm starts by choosing a random initial state, S_{init}. It selects some random neighbor, $S_{current}$, of S_{init}, and evaluates it. If it finds the cost of $S_{current}$ lower than the cost of S_{init} then $S_{current}$ is considered as S_{init} i.e. $S_{init} \leftarrow S_{current}$ and the move is performed in the next neighboring state. The algorithm performs a random series of moves, and accepts only downhill moves, until it reaches a local minimum. After a local minimum has been reached a new start state is generated randomly and the same procedure is repeated again with this state. This algorithm is repeated until a stopping condition is fulfilled. The minimum amongst all local minima is returned as the result [14, 20].

Next, the proposed view selection algorithm, based on iterative improvement, is discussed.

```
BEGIN
    SET an initial value of minS as state with very high cost
    WHILE not (stop_condition) DO
        select a random state S_init
        WHILE r-local minimum not reached DO
            S_current = random state in neighbor(S_init)
            IF cost(S_current) < cost(S_init) THEN S_init ← S_current
        END DO
        IF cost(S_init) < cost(minS) then minS = S_init
    END DO
    RETURN (minS)
END
```

Fig. 1. Iterative Improvement Algorithm[14,20]

2.2 Proposed Algorithm

The proposed algorithm based on iterative improvement[14, 20] is given in Fig. 2. The algorithm takes the lattice of views, with size of each view as input, and produces top-K views as output.

```
INPUT: Lattice of Views along with the size of each view
       V_LO =Total number of local optimized TopKViews to be identified
       V_N =Number of consecutively randomly picked neighboring views with higher TVEC
OUTPUT: minV_Top-K
METHOD:
BEGIN
    Set minV_TopK as Top-K views having very high TVEC
    WHILE (I < V_LO)
    DO
        select a random Top-K views V_TopK
        WHILE (J < V_N)
        DO
            Select a random Top-K views, say V_Top-K', in the neighbor of V_Top-K
            IF TVEC(V_Top-K') < TVEC(V_Top-K)
                Assign V_Top-K' to V_Top-K
                J=0
            ELSE
                J=J+1
            END IF
        END WHILE
        IF TVEC(V_Top-K) < TVEC(minV_Top-K)
            minV_Top-K = V_Top-K
            minTVEC = TVEC(V_Top-K)
        END IF
        I = I +1
    END WHILE
    RETURN minV_Top-K
END
```

Fig. 2. Proposed algorithm based on Iterative Improvement

The algorithm first selects a random set of top-K views, $V_{Top\text{-}K}$, from a multidimensional lattice. The top-K views are of the form (V_1, V_2, . . . , V_K), where each V_i represents a view in the multidimensional lattice. The algorithm then identifies the neighbouring view $V_{Top\text{-}K}'$ of the randomly selected view $V_{Top\text{-}K}$, by replacing any one view randomly in $V_{Top\text{-}K}$ by any other view in the multi-dimensional lattice. The total cost of evaluating all the views (TVEC), as defined in [11], is used to compute the cost of randomly selected top-K view $V_{Top\text{-}K}$ and its neighbouring top-K view $V_{Top\text{-}K}'$.

If TVEC of $V_{Top\text{-}K}'$ is less than TVEC of $V_{Top\text{-}K}$ then $V_{Top\text{-}K}'$ becomes the new $V_{Top\text{-}K}$ else there is no change in the $V_{Top\text{-}K}$. This identification of neighboring view of $V_{Top\text{-}K}$ continues till a predefined number of consecutively randomly picked neighboring views, i.e. V_N, have TVEC values greater than the TVEC of $V_{Top\text{-}K}$. $V_{Top\text{-}K}$ is then stored in $minV_{Top\text{-}K}$ and its TVEC is stored in minTVEC. The algorithm again selects, randomly a set of views and repeats the above steps, continually updating the $minV_{Top\text{-}K}$ and minTVEC. The algorithm terminates when a pre-defined number of locally optimized Top-K views, i.e. V_{LO}, are identified. The $minV_{Top\text{-}K}$ representing the Top-K views is then produced as output.

As an example consider selection of top-5 views, in a 3-dimensional lattice shown in Fig. 3, using the proposed algorithm. The index of the view is shown in parenthesis alongside the name of the view. The size of the views is shown alongside the node in the lattice.

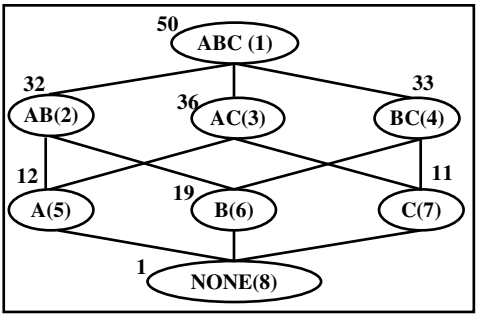

Fig. 3. 3-dimensional lattice along with size and index of each view

Since queries on root view cannot be answered by any other view in the lattice, the root view is assumed to be materialized i.e. view with index 1 is materialized. The selection of remaining top-4 views, given root view is materialized, is shown in Fig. 4. The pre-defined number of locally optimized Top-K views, i.e. V_{LO}, is taken as 2 and the number of consecutively randomly picked neighbors with higher TVEC, i.e. V_N, is taken as 4.

The first locally optimized Top-K views selected by the proposed algorithm are 12765 i.e. views ABC, AB, A, B and C. The next locally optimized Top-K views found are17254 i.e. ABC, C, AB, A and BC. The TVEC value 231 due to views corresponding to 17254 is less than the TVEC value 235 due to views corresponding to 12765. Thus the views selected by the proposed algorithm are ABC, C, AB, A and BC.

	V_{Top-K}	TVEC(V_{Top-K})	$V_{Top-K'}$	TVEC($V_{Top-K'}$)	MinV_{Top-K}	TopKViews
LocalOptimization-I	12764	238	12564	241	235	**ABC, AB, A, B, C**
	12764	238	12765	235		
	12765	235	14765	236		
	12765	235	12465	241		
	12765	235	12763	241		
	12765	235	12764	238		
LocalOptimization-II	13254	240	16254	241	231	**ABC, AB, BC, A, C**
	13254	240	17254	231		
	17254	231	17654	236		
	17254	231	17253	234		
	17254	231	17264	238		
	17254	**231**	**17256**	**235**		

Fig. 4. Selection of top-5 views using Proposed Algorithm

Next, experimental results showing comparison of the proposed algorithm (PA) with the most fundamental greedy algorithm HRUA is discussed.

3 Experimental Results

Algorithms PA and HRUA were implemented using JDK 1.6 in Windows-7 environment. The two algorithms were compared by conducting experiments on an Intel based 2.13 GHz PC having 3 GB RAM. The comparisons were carried out on parameter TVEC.

Graphs were plotted to compare PA and HRUA algorithms on TVEC against the number of dimensions for selecting top-10 views for materialization. These graphs for the number of neighbors V_N = 30, V_N =40, V_N =50 and V_N = 60 are shown in Fig. 5. V_N is denoted by *n* in the graphs.

It is observed from the above graphs that, with increase in the number of dimensions, the TVEC value of views selected using PA is lower than those selected using HRUA for different values of V_N used to compute the local minima. The performance of PA is best for V_N=60 i.e. when 60 neighbors are used to compute the locally optimized top-K views. Further, it can also be noted from these graphs that the difference in the TVEC values becomes significant for higher dimensions.

Thus it can be stated from the above graphs that the proposed algorithm is able to select reasonably good quality Top-K views in comparison to those selected using HRUA. Further, experimentations showed that the proposed algorithm is able to select Top-K views for dimensions higher than 10, where it becomes computationally infeasible for HRUA to select Top-K views for materialization.

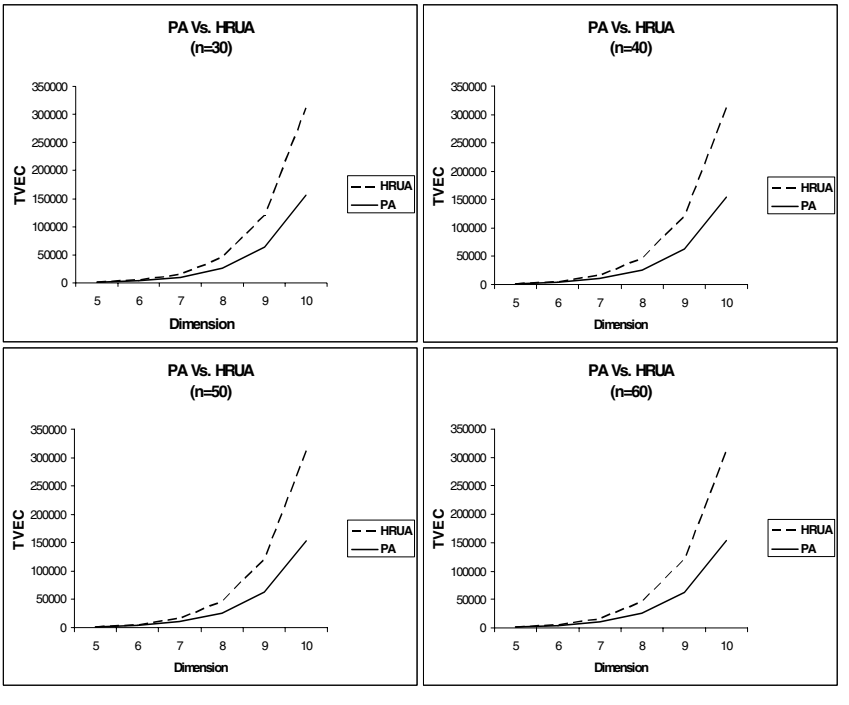

Fig. 5. Comparison of PA with HRUA – TVEC Vs. Dimension (n=30, 40, 50, 60)

4 Conclusion

In this paper, an iterative improvement based view selection algorithm is proposed. This algorithm considers each set of top-K views to be a state in a solution space, comprising of possible sets of K views in a multidimensional lattice, with an associated cost TVEC. The algorithm performs random walks via a series of downhill moves, involving identification of neighboring sets of K views having lower TVEC, until it attains a locally optimized Top-K views i.e Top-K views with minimum TVEC is achieved. In this way it determines a pre-defined number of locally optimized Top-K views and then selects, from amongst them, the one having minimum TVEC. The proposed algorithm selects reasonably good quality top-K views. Further, experimental results show that the views selected using the proposed algorithm have comparatively lower value of TVEC, when compared with those selected using the most fundamental greedy algorithm HRUA. Furthermore, the proposed algorithm is able to select Top-K views for higher dimensional data sets, where it becomes computationally infeasible for HRUA to select Top-K views for materialization. Thus, it can be said that it is feasible to select fairly good quality views for higher dimensional data sets using the proposed iterative improvement based view selection algorithm.

References

1. Agrawal, S., Chaudhari, S., Narasayya, V.: Automated Selection of Materialized Views and Indexes in SQL databases. In: 26th International Conference on Very Large Data Bases (VLDB 2000), Cairo, Egypt, pp. 495–505 (2000)
2. Aouiche, K., Jouve, P.-E., Darmont, J.: Clustering-Based Materialized View Selection in Data Warehouses. In: Manolopoulos, Y., Pokorný, J., Sellis, T.K. (eds.) ADBIS 2006. LNCS, vol. 4152, pp. 81–95. Springer, Heidelberg (2006)
3. Aouiche, K., Darmont, J.: Data mining-based materialized view and index selection in data warehouse. Journal of Intelligent Information Systems, 65–93 (2009)
4. Baralis, E., Paraboschi, S., Teniente, E.: Materialized View Selection in a Multidimansional Database. In: 23rd International Conference on Very Large Data Bases (VLDB 1997), Athens, Greece, pp. 156–165 (1997)
5. Chaudhuri, S., Shim, K.: Including Groupby in Query Optimization. In: Proceedings of the International Conference on Very Large Database Systems (1994)
6. Chirkova, R., Halevy, A.Y., Suciu, D.: A Formal Perspective on the View Selection Problem. In: Proceedings of VLDB, pp. 59–68 (2001)
7. Golfarelli, M., Rizzi, S.: View Materialization for Nested GPSJ Queries. In: Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW 2000), Stockholm, Sweden (2000)
8. Gupta, H., Mumick, I.S.: Selection of Views to Materialize in a Data warehouse. IEEE Transactions on Knowledge & Data Engineering 17(1), 24–43 (2005)
9. Gupta, A., Harinarayan, V., Quass, D.: Generalized Projections: A Powerful Approach to Aggregation. In: Proceedings of the International Conference of Very Large Database Systems (1995)

10. Gupta, H., Harinarayan, V., Rajaraman, V., Ullman, J.: Index Selection for OLAP. In: Proceedings of the 13th International Conference on Data Engineering, ICDE 1997, Birmingham, UK (1997)
11. Harinarayan, V., Rajaraman, A., Ullman, J.D.: Implementing Data Cubes Efficiently. In: ACM SIGMOD, Montreal, Canada, pp. 205–216 (1996)
12. Horng J. T., Chang Y. J., Liu B. J., Kao, C.Y.: Materialized View Selection Using Genetic Algorithms in a Data warehouse System. In: Proceedings of the 1999 Congress on Evolutionary Computation, Washington, D.C., USA, vol. 3 (1999)
13. Inmon, W.H.: Building the Data Warehouse, 3rd edn. Wiley Dreamtech India Pvt. Ltd. (2003)
14. Ioannidis, Y.E., Kang, Y.C.: Randomized Algorithms for Optimizing Large Join Queries. In: Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data, ACM SIGMOD Record, vol. 19(2), pp. 312–321 (1990)
15. Lawrence, M.: Multiobjective Genetic Algorithms for Materialized View Selection in OLAP Data Warehouses. In: GECCO 2006, Seattle, Washington, USA, July 8-12 (2006)
16. Lehner, W., Ruf, T., Teschke, M.: Improving Query Response Time in Scientific Databases Using Data Aggregation. In: Proceedings of 7th International Conference and Workshop on Database and Expert Systems Applications, DEXA 1996, Zurich(1996)
17. Lin, Z., Yang, D., Song, G., Wang, T.: User-oriented Materialized View Selection. In: The 7th IEEE International Conference on Computer and Information Technology (2007)
18. Luo, G.: Partial Materialized Views. In: International Conference on Data Engineering (ICDE 2007), Istanbul, Turkey (April 2007)
19. Mohania, M., Samtani, S., Roddick, J., Kambayashi, Y.: Advances and Research Directions in Data Warehousing Technology. Australian Journal of Information Systems (1998)
20. Nahar, S., Sahni, S., Shragowitz, E.: Simulated Annealing and Combinatorial Optimization. In: Proceedings of the 23rd Design Automation Conference, pp. 293–299 (1986)
21. Roussopoulos, N.: Materialized Views and Data Warehouse. In: 4th Workshop KRDB 1997, Athens, Greece (August 1997)
22. Shah, B., Ramachandran, K., Raghavan, V.: A Hybrid Approach for Data Warehouse View Selection. International Journal of Data Warehousing and Mining 2(2), 1–37 (2006)
23. Teschke, M., Ulbrich, A.: Using Materialized Views to Speed Up Data Warehousing, Technical Report, IMMD 6, Universität Erlangen-Nümberg (1997)
24. Theodoratos, D., Sellis, T.: Data Warehouse Configuration. In: Proceeding of VLDB, Athens, Greece, pp. 126–135 (1997)
25. Valluri, S., Vadapalli, S., Karlapalem, K.: View Relevance Driven Materrialized View Selection in Data Warehousing Environment. Australian Computer Science Communications 24(2), 187–196 (2002)
26. Vijay Kumar, T.V., Ghoshal, A.: A Reduced Lattice Greedy Algorithm for Selecting Materialized Views. In: Prasad, S.K., Routray, S., Khurana, R., Sahni, S. (eds.) ICISTM 2009. CCIS, vol. 31, pp. 6–18. Springer, Heidelberg (2009)
27. Vijay Kumar, T.V., Haider, M., Kumar, S.: Proposing Candidate Views for Materialization. In: Prasad, S.K., Vin, H.M., Sahni, S., Jaiswal, M.P., Thipakorn, B. (eds.) ICISTM 2010. CCIS, vol. 54, pp. 89–98. Springer, Heidelberg (2010)
28. Vijay Kumar, T.V., Haider, M.: A Query Answering Greedy Algorithm for Selecting Materialized Views. In: Pan, J.-S., Chen, S.-M., Nguyen, N.T. (eds.) ICCCI 2010. LNCS(LNAI), vol. 6422, pp. 153–162. Springer, Heidelberg (2010)

29. Vijay Kumar, T.V., Jain, N.: Selection of Frequent Queries for Constructing Materialized Views in Data Warehouse. The IUP Journal of Systems Management 8(2), 46–64 (2010)
30. Vijay Kumar, T.V., Goel, A., Jain, N.: Mining Information for Constructing Materialised Views. International Journal of Information and Communication Technology 2(4), 386–405 (2010)
31. Vijay Kumar, T.V., Haider, M.: Greedy Views Selection Using Size and Query Frequency. In: Unnikrishnan, S., Surve, S., Bhoir, D. (eds.) ICAC3 2011. CCIS, vol. 125, pp. 11–17. Springer, Heidelberg (2011)
32. Vijay Kumar, T.V., Haider, M., Kumar, S.: A View Recommendation Greedy Algorithm for Materialized Views Selection. In: Dua, S., Sahni, S., Goyal, D.P. (eds.) ICISTM 2011. CCIS, vol. 141, pp. 61–70. Springer, Heidelberg (2011)
33. Vijay Kumar, T.V., Devi, K.: Frequent Queries Identification for Constructing Materialized Views. In: The Proceedings of the International Conference on Electronics Computer Technology (ICECT-2011), April 8-10, vol. 6, pp. 177–181. IEEE, Kanyakumari (2011)
34. Vijay Kumar, T.V., Haider, M.: Selection of Views for Materialization Using Size and Query Frequency. In: Das, V.V., Thomas, G., Lumban Gaol, F. (eds.) AIM 2011. CCIS, vol. 147, pp. 150–155. Springer, Heidelberg (2011)
35. Vijay Kumar, T.V., Haider, M.: Materialized Views Selection for Answering Queries. In: Kannan, R., Andres, F. (eds.) ICDEM 2010. LNCS, vol. 6411, pp. 44–51. Springer, Heidelberg (2012)
36. Widom, J.: Research Problems in Data Warehousing. In: 4th International Conference on Information and Knowledge Management, Baltimore, Maryland, pp. 25–30 (1995)
37. Yang, J., Karlapalem, K., Li, Q.: Algorithms for Materialized View Design in Data Warehousing Environment. The Very Large databases (VLDB) Journal, 136–145 (1997)
38. Yousri, N.A.R., Ahmed, K.M., El-Makky, N.M.: Algorithms for Selecting Materialized Views in a Data Warehouse. In: The Proceedings of the ACS/IEEE 2005 International Conference on Computer Systems and Applications, AICCSA 2005, pp. 27–1. IEEE Computer Society (2005)
39. Zhang, C., Yao, X., Yang, J.: Evolving Materialized Views in a Data Warehouse. IEEE CEC, 823–829 (1999)
40. Zhang, C., Yao, X., Yang, J.: An Evolutionary Approach to Materialized Views Selection in a Data Warehouse Environment. IEEE Transactions on Systems, Man and Cybernatics, 282–294 (2001)