Volume 4, Issue 10, October 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Using Bat Algorithm for Materialized View Selection in Data Warehouse

[1]Nasibe Bagheri*, [2]Mehdi Sadeghzade
Department of Computer Engineering, Science and Research Branch Islamic Azad University, Saveh, Iran
Department of Computer College of Electric and computer Engineering, Mahshahr Branch, Islamic Azad University, Mahshahr, Iran

Abstract— A data warehouse stores historical information, integrated from several large heterogeneous data sources spread across the globe, for the purpose of supporting decision making. The queries for decision making are usually analytical and complex in nature and their response time is high when processed against a large data warehouse. Materialized view selection is one of the most critical techniques to design data warehouse in an optimal manner that can be reduced query response time by materializing views over a data warehouse. Materialized view selection in order to facilitate efficient query processing, the information contained in data warehouses is typically stored as a set of materialized views that minimize view maintenance and query processing costs. To efficiently deal with materialized view selection problem, an efficient materialized view selection algorithm based on Bat Algorithm (BA) is presented in this paper. Experimental results show that the proposed algorithm achieves much better performance than the Genetic Algorithms and the Simulated Annealing. According to the median value of the fitness function for the three algorithms achieved in 20 iterations to the conclusion that Bat Algorithm in minimizing the fitness function than the Genetic algorithms and the Simulated Annealing are more successful and Provides better results.

Keyword s — Data Warehouse, Materialized View, View Selection Problem, Bat Algorithm.

I. INTRODUCTION

In most large organizations different departments maintain local databases for supporting sets of applications. There is a need to provide a common platform over these local databases in order to support global data processing applications. These applications are used to generate integrate global reports, which are useful for developing a decision support system. These systems extract, integrate and store the information spread across several large, heterogeneous local databases and other information sources, into a central repository referred to as a Data Warehouse. This data warehouse can be used for answering complex analytical queries in support of decision making. A data warehouse is also often viewed as architecture, constructed by integrating data from multiple heterogeneous sources to support structured and/or ad hoc queries, analytical reporting, and decision making [1].

Since data in the data warehouse tends to be very large in size, as it grows continuously with time, and queries posed on it may include a large number of complex aggregates, the organization of data in the data warehouse becomes a crucial factor for processing analytical queries efficiently as also in making it compliant with the underlying data sources [2].

The analytical queries, which are long and complex when processed against a large data warehouse, consume a lot of time for processing and thereby, lead to higher response time. This response time can be reduced by using materialized views [3], [4].

It is not feasible to materialize all possible views as the number of possible views is exponential in the number of dimensions, and for higher dimensions it is not possible to store all possible views within the available storage space. Further, optimal view selection is shown to be an NP Complete problem [5]. Thus there is a need to select an appropriate subset of views, from among all possible views, that can reduce the query response time while fitting within the available space for materialization.

In this paper, A Bat Algorithm is proposed that attempts to select reasonably good quality view from a AND-OR Graph. The proposed methods use an algorithm, that materialized views enhance query processing and minimize the limitations of storage space and query processing cost and view maintenance cost, are chosen. The paper is organizes as follow: section II discussed the view selection problem and Bat Algorithm is described in section III. Then proposed approach is given in section IV. Finally in section V and VI we exhibit experimental result and conclusion.

II. VIEW SELECTION PROBLEM

A view takes the output of a query and makes it appear like a virtual table. You can use a view in most places where a table can be used. All operations performed on a view will affect data in the base table and so are subject to the integrity constraints and triggers of the base table. A View can be used to simplify SQL statements for the user or to isolate an application from any future change to the base table definition. A View can also be used to improve security by

restricting access to a predetermined set of rows or columns. In addition to operating on base tables, one View can be based on another; a view can also JOIN a view with a table (GROUP BY or UNION).

There are generally two types of views in data warehouse:

- Virtual view: This view is not stored in the database and only one query to create a relationship. Virtual views are stored in a single query and don't need to change them when the base tables change. And sometimes access to view in this case is expensive.
- Materialized view: Materialized views are schema objects that can be used to summarize, pre compute, replicate, and distribute data. E.g. to construct a data warehouse. A materialized view provides indirect access to table data by storing the results of a query in a separate schema object. Unlike an ordinary view, which does not take up any storage space or contain any data. A materialized view can be stored in the same database as its base table(s) or in a different database. Materialized views stored in the same database as their base tables can improve query performance through query rewrites. Query rewrites are particularly useful in a data warehouse environment.

Using materialized views is a popular technique for reducing the query response time. Indeed materialize an appropriate set of views to response queries using Views, Can be Speed up Query Processing When access to materialized views much faster than recomputation views.

Therefore, the compute all entry queries can be performed with lower cost of query processing, but, view Support costs increase When keep update materialized views in order to maintain consistency with data sources are.

In addition, query results can be very large compared to the available storage space. Hence, the need to select a set of view is essential so that the three parameters measured included the following:

Cost of processing a query / view maintenance / storage space

There are several issues related to materialized views such as: selection view, maintenance view, evolution view, and response query with using View that In this paper we focus only on the selection view and maintenance view.

In order to identify the advantages and disadvantages of view selection methods, Mami [8] propose two main dimensions along with they can be classified: (i) Frameworks; and (ii) Resource Constraints.

Generally, approaches to the view selection problem consist of two main steps. The first step identifies the candidate views which are promising for materialization. Techniques based on multiquery DAG, syntactical analysis of the workload or query rewriting have been used to obtain the candidate views. Based on the set of candidate views, the second step selects the set of views to materialize under the resource constraints and by applying heuristic algorithms.

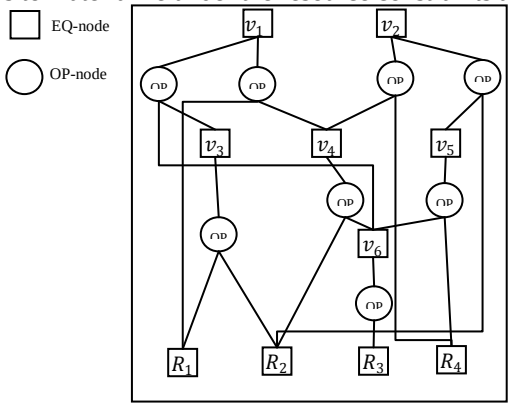

Fig. 1 The AND-OR view graph of the two queries Q1 and Q2.

Generally the dependence relation on queries (or views) has been represented by using DAG. We used of AND_OR View Graph that described by Roy [7] is a Directed Acyclic Graph (DAG) composed of two types of nodes: Operation nodes and Equivalence nodes. Each operation node represents an algebraic expression (Select-Project-Join) with possible aggregate function. An equivalence node represents a set of logical expressions that are equivalent (i.e., that yield the same result). The operation nodes have only equivalence nodes as children and equivalence nodes have only operation nodes as children. The root nodes are the query results and the leaf nodes represent the base relations. A sample AND-OR view graph is shown in Fig. 1. Circles represent operation nodes (Op-Nodes) and boxes represent equivalence nodes (Eq-Nodes). For example, in Fig. 1, view V1 corresponding to a single query Q1, can be computed from V6 and V3 or R1 and V4. If there is only one way to answer or update a given query, the graph becomes an AND view graph. In the data cube which is a specific model of a data warehouse, the AND-OR view graph is an OR view graph, as for each view there are zero or more ways to construct it from other views, but each way involves only one other view [8]. In other words, an OR view graph is an AND-OR view graph in which any node is an equivalence node that can be computed from any one of its children.

Resource constraints considered during the view selection can be taken into account when classifying view selection methods. There are three main models presented in literature: unbounded, space constrained and maintenance cost constrained.

In next level, we need to determine resource constraints that we used of space constrained. Due to the storage space limitation, materializing all views is not always possible. In this setting, a useful notion is that of a view benefit (or query benefit). This is defined as the reduction in the workload evaluation cost that can be achieved by materializing this view.

Also relevant in this context is the per-unit benefit, obtained by dividing the view benefit by its space occupancy. It has been shown [6] that the per-space unit benefit of a view can only decrease as more views are selected (monotonic property). The space constrained model minimizes the query processing cost plus the view maintenance cost under a space constraint.

$$minimize\left(\sum_{Q_i \in Q} f_{Q_i} * Q_c(Q_i, M) + \sum_{v_i \in M} f_u(v_i) * M_c(V_i, M)\right) \quad (1)$$

$$under \sum_{v_i \in M} size(v_i) \leq S$$

Where S is the storage space capacity.

Mami [6] Classify the view selection methods according to several dimensions characterizing their algorithms (i) resource constraints they consider during the view selection process and (ii) frameworks they use to obtain the candidate views. Based on this classification, we review most of the view selection methods. The best-known heuristic algorithms proposed in literature to solve the view selection problem, namely: deterministic algorithms, randomized algorithms, hybrid algorithms or constraint programming.

III. BRIEF REVIEW OF BAT ALGORITHM

Bat Algorithm is a type of metaheuristic methods. The vast majority of heuristic & metaheuristic algorithm have been derives from the behaviour of biological systems and/or physical systems in nature. Bat Algorithm is based on the echolocation behaviour of bats and proposes by Yang in 2010.

Micro bats use a type of sonar, called, echolocation, to detect prey, avoid obstacles, and locate their roosting crevices in the dark. These bats emit a very loud sound pulse and listen for the echo that bounces back from the sur-rounding objects.

Though each pulse only lasts a few thousandths of a second (up to about 8 to 10 ms), however, it has a constant frequency which is usually in the region of 25 kHz to 150 kHz. When hunting for prey, the rate of pulse emission can be sped up to about 200 pulses per second when they fly near their prey.

For simplicity, we now use the following approximate or idealized rules:

1. All bats use echolocation to sense distance, and they also 'know' the difference between food/prey and background barriers in some magical way;
2. bats fly randomly with velocity v_i at position xi with a fixed frequencyf_{min}, varying wavelength λ and loudness A_0 to search for prey. They can automatically adjust the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r \in [0, 1]$, depending on the proximity of their target;
3. Although the loudness can vary in many ways, we assume that the loudness varies from a large (positive) A_0 to a minimum constant valueA_{min}.

Based on these approximations and idealization, the basic steps of the Bat Algorithm (BA) can be summarized as the pseudo code shown in Fig. 2.

```
Bat Algorithm
    Objective function f(x),   x = (x_1, ..., x_d)^T
    Initialize the bat population x_i (i = 1,2,...,n) and v_i
    Define pulse frequency f_i at x_i
    Initialize pulse rates r_i and the loudness A_i
    while (t <Max number of iterations)
    Generate new solutions by adjusting frequency,
    and updating velocities and locations/solutions [equations (2) to (4)]
        if (rand > r_i)
        Select a solution among the best solutions
        Generate a local solution around the selected best solution
        end if
        Generate a new solution by flying randomly
        if (rand < A_i & f(x_i) < f(x_*))
        Accept the new solutions
        Increase r_i and reduce A_i
        end if
    Rank the bats and find the current best x_*
    end while
    Postprocess results and visualization
```

Fig. 2 pseudo code of Bat Algorithm (BA) [9]

In simulations, we use virtual bats naturally. We have to define the rules how their positions x_i and velocities v_i in a d-dimensional search space are updated. The new solutions x_i^t and velocities v_i^t at time step t are given by

$$f_i = f_{min} + (f_{max} - f_{min})\beta \quad (2)$$

$$v_i^t = v_i^{t-1} + (x_i^t - x_*)f_i \quad (3)$$

$$x_i^t = x_i^{t-1} + v_i^t \quad (4)$$

Where $\beta \in [0, 1]$ is a random vector drawn from a uniform distribution. Here x_* is the current global best location (solution) which is located after comparing all the solutions among all the n bats.

The update of the velocities and positions of bats have some similarity to the procedure in the standard particle swarm optimization [10] as f_i essentially controls the pace and range of the movement of the swarming particles. To a degree, BA can be considered as a balanced combination of the standard particle swarm optimization and the intensive local search controlled by the loudness and pulse rate. [9]

IV. PROPOSED APPROACH

In this paper, subject is ability of BA for optimization materialized view selection.

According to last section, first problem is select one model for represent views in data warehouse, second problem is select a state for resource constraint and finally implementation BA for find best set of view to materialization.

As regard last explanation, we used AND_OR view graph and space constraint. We present an example to motivate the discussion of materialized view selection in data warehouse.

Our example is taken from a sample data warehouse application that analyses trend in sales and which was used in [11]. The relation and the attributes of the running example's schema are:

Item (I_id , I_name , I_price)
Part (P_id , P_name , I_id)
Supplier (S_id , S-name , P-id , city , cost)
Sales (I_id , month , year , amount)

Furthermore, we assume that all of these relations are stored at the same site and we do not need to consider data communication costs in our cost calculation. Suppose that we have the four following frequently used queries:

Q_1= select I_id, sum (amount* I_price)
From Item, Sales
Where I_name like {MAZDA, NISSAN, TOYOTA}(O1)
And year=1996 (O2)
And Item.I_id= Sales.I_id (O5)
Group by I_id

Q_2= select P_id, month, sum (amount* no)
From Item, Sales, Part
Where I_name like {MAZDA, NISSAN, TOYOTA}(O1)
And year=1996(O2)
And Item.I_id= Sales.I_id (O5)
And Part.I_id= Item.I_id (O6)
Group by P_id, month

Q_3= select P_id, min (cost), max (cost)
From Part, Supplier
Where Part. P_id= Supplier. P_id O (7)
And P_name like {spark_plug , gas_kit} O(3)
Group by P_id

Q_4= select I_id, sum (amount*number*min_cost)
From Item, Sales, Part
Where I_name like {MAZDA, NISSAN, TOYOTA} O (1)
And year=1996 O (2)
And Item.I_id= Sales.I_id (O5)
And Part.I_id= Item.I_id (O6)
And Part.I_id=
(Select P_id, min (cost) as min_cost
From Supplier
Group by P_id) O (9)
Group by I_id

In Fig. 3 we show an AND_OR view graph for the above four queries.

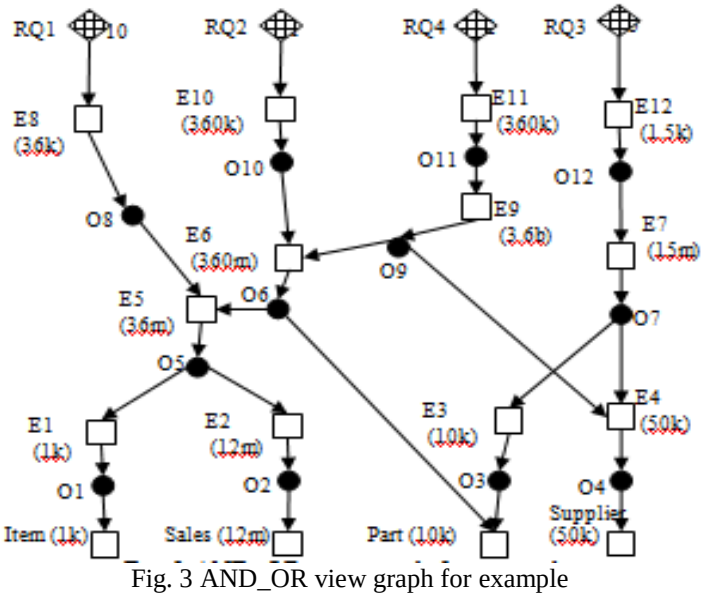

Fig. 3 AND_OR view graph for example

The query access frequencies are indicates above each query node. For simplicity, we assume that the base relations Item, Sales, Part, and Supplier are updated twice during the process of materializes view selection.

Suppose there are some materialized intermediate nodes in the AND_OR view graph. For each query, the cost of query processing is its query frequencies multiplied by the cost of the query accesses to the materialized nodes. The maintenance cost for materialized view is the cost used for construction of the view (here we assume that rebuilding is used whenever an update of an involved base relation occurs) [11].

Now, discuses about steps of BA based materialized view selection algorithm that are described as follow:

Step 1: Initialization and Structure of bats. In the initialization process, a set of bats is created under the constraint that total space occupied by M (set of materialized view) is less than S (a constant number for space limitations). Each view in AND_OR view graph is assign one number and each bat randomly is consist of 5 views under search space.

Also, in this step, position & velocities of each bats in 5-dimentional search space random is determinate.

Step 2: Fitness Function Definition. Since objective in view selection mathematical model is to minimize the sum of query and maintenances views. We define the fitness function from the cost function in equation (1).

$$Fitness(G\ ,M) = \tau\ (G\ ,M)$$

In this step, according to available fitness function, function value for each bat calculation and store.

Step 3: Global Best Position Computation. Each bat I memorizes its own fitness value and choose the minimized one, which had been better so far as personal best position. The global best position is announced to all bats.

Step 4: modify the position (x) and velocity (v) of each bat at time step t. In this step, according to global best position in before step, update position and velocity by equations (2) to (4).

Step 5: stop condition. If the best evaluation value (global best position) is not obviously improves or the iteration number t reaches the given maximum, then go to step 6. Otherwise, go to step 2.

Step 6: the bat that generates the best Fitness function values is the output materialized view.

V. EXPERIMENTAL RESULTS

Bat Algorithm for AND_OR view graph in figure 2 to select an optimal set views for materialization has been implemented in MATLAB environment.

The main parameters of Bats Algorithm are set as follow:

Population size= 500, loudness (A) =0.25, Pulse Rate (r) = 0.3 ,$\beta = 0.3$,$f_{min} = 0$,$f_{max} = 2$, iteration number = 200.

After implementation, our proposed algorithm to display the effectiveness is compare whit Genetic Algorithm and Simulated Annealing.

For comparison algorithms is considered 20 iterations for each algorithm and the fitness function value and CPU time for each case is investigated. The results of the fitness function are presented in Table I for 20 iterations.

Table I: Fitness function values of three algorithms

	fitness function value		
	BAT	GA	SA
1	105208000	105208000	105208000
2	150185000	146564000	150185000
3	105238000	135208000	171511500
4	171511500	115208000	105208000
5	126564500	125238000	105208000
6	105208000	155208000	201642000
7	126564500	150185000	150185000
8	150435000	146564000	150435000
9	105208000	171511500	150435000
10	171511500	125208000	150435000
11	105208000	115238000	105208000
12	171511500	125208000	105208000
13	246869000	115208000	105208000
14	105208000	125208000	150185000
15	150435000	115208000	150435000
16	105208000	246869000	171511500
17	201642000	105208000	105208000
18	126564500	155208000	246869000
19	201642000	246869000	150435000
20	105238000	115208000	201642000
MEDIN	141858000	142076725	146618100
Standard deviation	41341377.03	4025659.18	39376356.42

Linear curve of Fitness function are shown in Fig. 4.

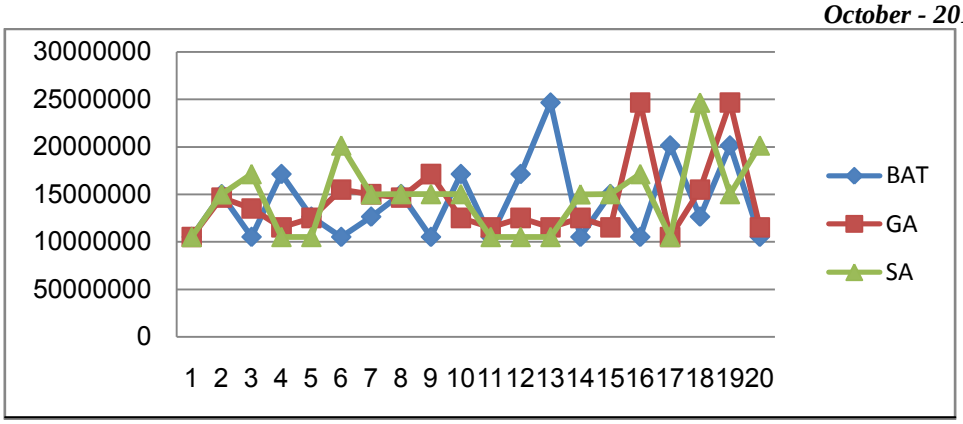

Fig. 4 Linear curve of Fitness function for 3 algorithms.

Box plot of the fitness function is shown in Fig. 5.

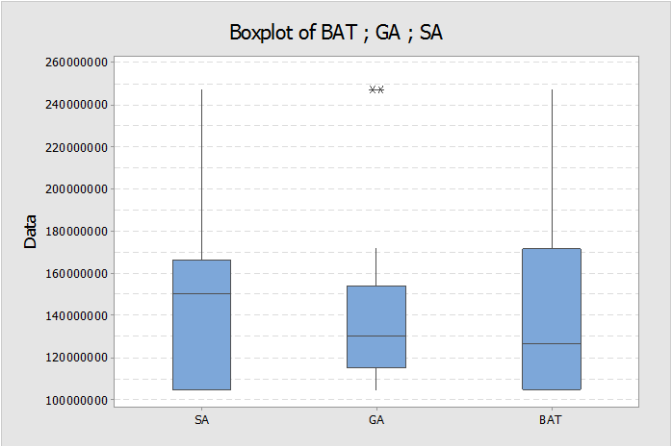

Fig. 5 Box plot of Fitness function for 3 algorithms.

The amount of CPU time for 20 iterations is presented in Table II.

Table II: CPU Time values of three algorithms

	CPU Time		
	BAT	GA	SA
1	3.378	4.5732	1.8388
2	3.3486	4.4583	1.8158
3	3.3509	4.4497	1.8188
4	3.3416	4.4855	1.8165
5	3.3569	4.5265	1.8111
6	3.3769	4.469	1.8243
7	3.3788	4.4755	1.8179
8	3.3757	4.4571	1.8221
9	3.3311	4.4618	1.8187
10	3.373	4.4837	1.8162
11	3.3556	4.4221	1.8264
12	3.3485	4.4664	1.8165
13	3.377	4.4691	1.811
14	3.3488	4.465	1.8123
15	3.3555	4.4681	1.8189
16	3.364	4.4836	1.8172
17	3.3887	4.4826	1.8096
18	3.3637	4.4737	1.8147
19	3.3923	4.4627	1.808
20	3.376	4.4695	1.818
MEDIN	3.36408	4.475155	1.81764
Standard deviation	0.016513	0.030175425	0.006816

Linear curve of CPU Time are shown in Fig. 6.

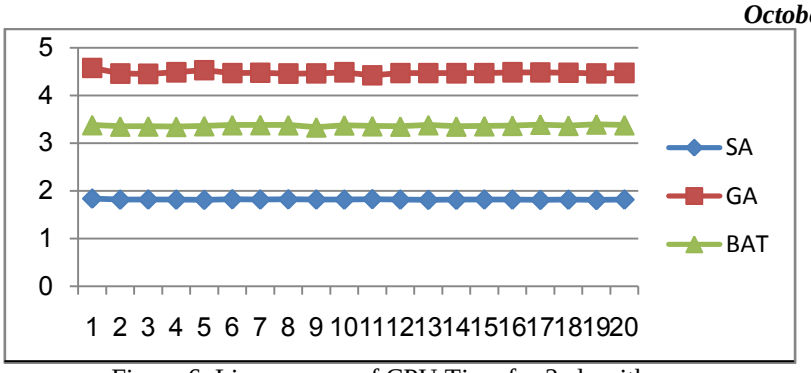

Figure 6: Linear curve of CPU Time for 3 algorithms.

Box plot of CPU Time is shown in Fig. 7.

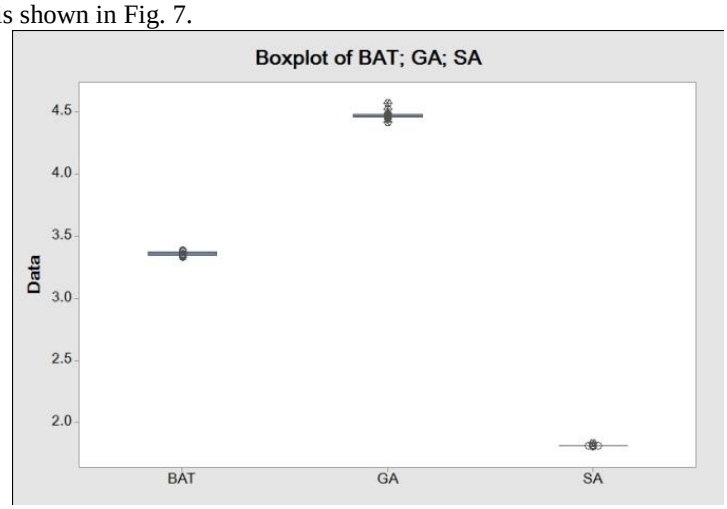

Figure 7: Box plot of CPU Time for 3 algorithms.

According to the Box plot and in terms of Median value of the Fitness function, it is known that Bats Algorithm has better performance in minimizing the Fitness function value, and then Genetic Algorithm and Simulated Annealing are placed. However, the Simulated Annealing in CPU time has shown better performance. And Bat Algorithm is in second place. And finally, genetic algorithm, which has a very high time resolution, is located.

As I mentioned, the Bat Algorithm in each iteration do 3 updates and is associated with three parameters, but the simulated annealing algorithm involved with a single temperature, that this reduces the time of simulated annealing Towards Bat Algorithm. For this reason, time in simulated annealing showed better performance than the Bat Algorithm.

VI. CONCLUSIONS

One of the important issues in data warehouse design is select views for materialization. Here we propose an algorithm for materialized view selection that chooses a set of views somehow that minimize the total cost of query processing and maintenance view.

This algorithm has been implemented in MATLAB environment, Experimental results of the algorithm implementation; Show that the proposed algorithm has better performance than the algorithms GA and SA. In the framework of this algorithm have been considered all cost parameters related to materialized view such as the query frequency and base relation maintenance frequency and limitations of storage space.

According to CPU time in our algorithm and achieved less time in simulated annealing, recommended to reduce time a combination of bat and simulated annealing algorithm is implemented on this issue, until achieved a more favourable result. And also one of the challenging tasks of future work can address the problem of view selection in a distributed environment. So for future research in materialized view selection, it is suggested that do more research in view selection problem on Semantic Web databases and studies on this problem in a distributed environment.

REFERENCES

[1] Inmon, W.H. , *Building the Data Warehouse*, Third Edition, Wiley Dreamtech, 2003.

[2] Mohania, M., Samtani, S., Roddick, J. and Kambayshi, Y.: *Advances and Research Directions in Data Warehousing Technology*, Australian Journal of Information Systems, Vol 7, No 1; 1999.

[3] Gupta, H., Mumick, I.: *Selection of Views to Materialize in a Data Warehouse*, IEEE Transactions Knowledge and Data Engineering, 17(1), pp. 24–43, 2005.

[4] Baralis E., Paraboschi S., Teniente E.: *Materialized view selection in a multidimensional database*, In Proceeding of 23rd VLDB Conference Greece, pp. 156–165, 1997.

[5] Harinarayan, V., Rajaraman, A., Ullman, J. D.: *Implementing data cubes efficiently*, ACMSIGMOD International Conference on Management of Data, pages 205-227, 1996.

[6] Imene Mami and Zohra Bellahsene, *A Survey of View Selection Methods*, SIGMOD Record, March 2012 (Vol. 41, No. 1)

[7] P. Roy, S. Seshadri, S. Sudarshan, and S. Bhobe. *Efficient and extensible algorithms for multi query optimization.* In SIGMOD Conference, pages 249–260, 2000.

[8] H. Gupta. *Selection of views to materialize in a data warehouse*. In ICDT, pages 98–112, 1997.

[9] Xin-She Yang, *A New Metaheuristic Bat-Inspired Algorithm*, Studies in Computational Intelligence, Springer Berlin, 284, Springer, 65-74, 2010.

[10] Kennedy, J. and Eberhart, R.: *Particle swarm optimization*, Proc. IEEE Int. Conf. Neural Networks. Perth, Australia, 1942-1945 (1995).

[11] J. Yang, K. Karlapalem, and Q. Li. *Algorithms for materialized view design in data warehousing environment*. In VLDB, pages 136–145, 1997.

