A Multi-Objective Approach for Materialized View Selection

Jay Prakash, School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, India

T.V. Vijay Kumar, School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, India

ABSTRACT

In today's world, business transactional data has become the critical part of all business-related decisions. For this purpose, complex analytical queries have been run on transactional data to get the relevant information, from therein, for decision making. These complex queries consume a lot of time to execute as data is spread across multiple disparate locations. Materializing views in the data warehouse can be used to speed up processing of these complex analytical queries. Materializing all possible views is infeasible due to storage space constraint and view maintenance cost. Hence, a subset of relevant views needs to be selected for materialization that reduces the response time of analytical queries. Optimal selection of subset of views is shown to be an NP-Complete problem. In this article, a non-Pareto based genetic algorithm, is proposed, that selects Top-K views for materialization from a multidimensional lattice. An experiments-based comparison of the proposed algorithm with the most fundamental view selection algorithm, *HRUA*, shows that the former performs comparatively better than the latter. Thus, materializing views selected by using the proposed algorithm would improve the query response time of analytical queries and thereby facilitate in decision making.

KEYWORDS

Data Warehouse, Decision Making, Evolutionary Algorithm, Materialized View Selection, Multi-Objective Optimization, Online Analytical Processing

1. INTRODUCTION

The penetration of smart technologies has made it increasingly convenient to capture and store the data of day to day business transactions. It has become standard practice in the business world to make every transaction in digital form. The transactional data has a hidden value, which can provide useful insight about business performance. These trends and insights aid smarter business decisions. If this transactional data is analyzed and used properly, it can empower the business world to make smarter decisions about their future business operations. In today's competitive business environment, smarter decisions are necessary in order to sustain in the global market. The multi-national companies capture business transactional data and store them in multiple disparate databases spread across the globe.

There are two approaches to access this information namely the lazy (on-demand) approach or eager (in-advance) approach (Widom, 1995). In the former approach, the data is gathered based on the

user query and is used when data at local data sources changes frequently. In the latter approach, the data is accumulated and stored apriori in a central repository and queries are processed against this already stored information. A data warehouse is based on the latter approach. In a data warehouse, relevant data accumulated from multiple disparate databases, spread across multiple locations, is integrated and stored for analytical query processing. A data warehouse stores subject-specific data, which is non-volatile and time-variant, integrated from multiple sources for supporting strategic decision making (Inmon, 2003; Kimball & Ross, 2003). The complex analytical queries are posed against the data in the data warehouse in order to get insights and trends for business operations. These complex and analytical queries take a lot of time for processing considering that a data warehouse grows continuously with time as data in it is non-volatile. This processing time can be reduced by materializing views in the data warehouse and use these for querying purposes (Mohania et al., 1999). Since all possible views cannot be materialized due to storage space constraints, an appropriate subset amongst them needs to be selected that conform to the storage space constraint and can result in efficient decision making. The selection of such a subset of relevant views is referred to as view selection (Chirkova et al., 2002). View selection is discussed next.

1.1. View Selection

View selection, as defined in (Chirkova et al., 2002), is "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". This problem is an NP-complete problem (Gupta, 1997; Dondi et al., 1999). Several view selection algorithms exist, most of which are empirical-based or heuristic-based. The empirical algorithms consider data accessed by past queries as indicators of data likely to be accessed by future queries (Lehner et.al., 1996, Vijay Kumar & Devi, 2012, 2013; Vijay Kumar et al., 2010a). On the other hand, heuristic based view selection algorithms (Harinarayan et.al., 1996; Gupta et.al., 1997; Choi et. al., 2002, Theodoratos & Sellis, 1997; Yang et.al., 1997; Baralis et al., 1997; Gupta & Mumick, 2005; Horng et.al., 2003; Lin & Kernighan, 1973; Zhang et.al., 2001; Valluri et al., 2002) find solutions, from among all possible solutions, without guaranteeing the globally optimal solution. Three frameworks have been considered for the view selection problem, i.e. the data cube lattice (Harinarayan et.al., 1996; Gray et al., 1997), *AND/OR* view graph (Roussopoulos et al., 1982) and Multiview processing plan (*MVVP*) (Yang et al., 1997). The data cube lattice framework, which has been considered in this paper for view selection, is based on star schema, which is widely used for data warehouse design. The multi-dimensional lattice is a simple representation of all possible views. The node of the lattice represents a view and the top most, or root view, represents the fact table. A multi-dimensional lattice is a dependency lattice framework, where the query on any node can be answered by its ancestor node. For example, consider a 3-dimensional lattice shown in Figure 1.

Figure 1. Lattice 3-dimensional with view size
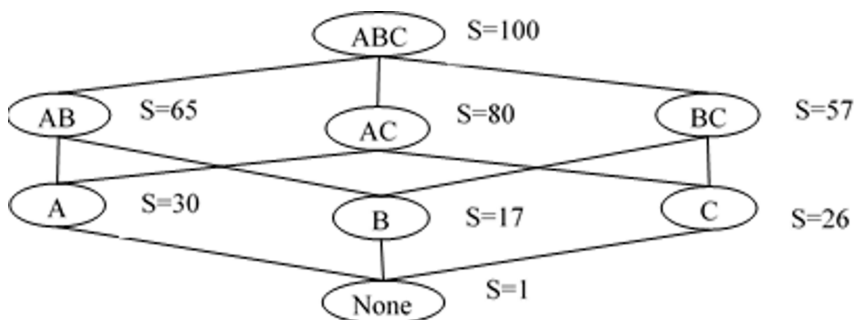
The node *ABC* is a fact table and all other nodes are possible views i.e. *AB*, *AC*, *BC*, *A*, *B*, *C* and *None*.

Size S of each views is shown alongside the view. The direct dependencies are captured by edges between the adjacent hierarchies whereas, the indirect dependencies are captured transitively between views in the same hierarchy, but not in adjacent hierarchy. The edge between view *A* and *AB* shows direct dependency whereas *A* is indirectly dependent on *ABC*. All views directly or indirectly depend on the root view *ABC* and the view *NONE* has no dependent view.

The greedy view selection algorithm proposed in (Harinarayan et al.,1996) has been employed on the multidimensional lattice to select *Top-K* views from it. This algorithm, hereinafter referred to as *HRUA*, is the most fundamental view selection algorithm in literature. However, for higher dimension data sets, it becomes almost infeasible for *HRUA* to select the *Top-K* views for materialization. Heuristics like randomized (Vijay Kumar & Kumar, 2012a, 2012c, 2015), greedy (Harinarayan et al., 1996; Shukla et al., 1998; Haider & Vijay Kumar, 2011, 2017; Vijay Kumar, 2013; Vijay Kumar & Ghoshal, 2009; Vijay Kumar & Haider, 2010, 2011a, 2011b, 2012, 2015, Vijay Kumar et al., 2010b, 2011), evolutionary (Vijay Kumar & Kumar, 2012b, 2013, 2014, 2018b) and swarm (Arun & Vijay Kumar, 2015a, 2015b, 2017a, 2017b; Vijay Kumar & Arun, 2016, 2017, Vijay Kumar et al., 2017, Kumar & Vijay Kumar, 2017, 2018a) have addressed this limitation with the *HRUA*. Their objective was to select *Top-K* views from the multidimensional lattice that minimized the total cost of evaluating all the views, this cost is referred to as the Total View Evaluation Cost (TVEC) and is defined in (Vijay Kumar & Kumar, 2012a, 2012b, 2012c, 2013, 2014, 2015) as:

$$TVEC\left(V_{Top-K}\right) = \sum_{i=1 \wedge SM_{v_i}=1}^{N} Size\left(V_i\right) + \sum_{i=1 \wedge SM_{v_i}=0}^{N} SizeSMA\left(V_i\right)$$

where, *N* is total number of views in the lattice, SM_{v_i} is the Status Materialized of view V_i (SM_{v_i} =1 materialized, and $SM_{v_i} = 0$ Not materialized), $Size\left(V_i\right)$ is the size of view V_i, $SizeSMA\left(V_i\right)$ is size of smallest materialized ancestor of view V_i.

The existing view selection algorithms addressed the view selection problem as a single objective optimization problem with the objective to minimize the *TVEC*. In this paper, an attempt has been made to formulate this view selection problem as a bi-objective optimization problem so as to simultaneous optimize and achieve a better trade-off between the cost due to materialized views and the cost due to non-materialized views. Further, an attempt has been made to address this bi-objective Materialized View Selection (*MVS*) problem using a non-Pareto based vector evaluated genetic algorithm (*VEGA*) (Schaffer, 1985). Accordingly, a *VEGA* based *MVS* algorithm has been proposed that selects the *Top-K* views from a multi-dimensional lattice. Further, experiment-based comparison of the proposed view selection algorithm with *HRUA* is carried out in terms of the quality of the *Top-K* views selected by them.

1.2. Organization of the Paper

The paper is organized as follows: The formulation of the view selection problem as a bi-objective optimization problem along with the *VEGA* based MVS algorithm to address it is presented in section 2. An example illustrating the use of the proposed multi-objective view selection algorithm is given in section 3. Section 4 discusses the experimental based comparison of the proposed view selection algorithm with *HRUA*. Conclusion is given in section 5.

2. *MVS* USING *VEGA*

As mentioned before, the objective of a view selection algorithms is to select the *Top-K* views from amongst all possible views that entail the least *TVEC*. The existing view selection algorithms addresses the *MVS* problem as a single objective optimization problem with the objective to minimize *TVEC*. In this paper, the *MVS* problem is formulated as a bi-objective optimization problem as discussed next.

2.1. The Bi-Objective MVS Optimization Problem

TVEC, as define above, comprises the summation of the two costs namely the cost due to materialized views (C_{MV}) and the cost due to non-materialized views (C_{NMV}) as given below:

$$C_{MV} = \sum_{i=1 \wedge SM_{v_i} =1}^{N} Size\left(V_i\right)$$

C_{MV} is a summation of sizes of all views that are materialized, as queries on these views are answered by the view itself. On the other hand, C_{NMV} is the summation of the sizes of the smallest materialized ancestor of the non-materialized views as queries on non-materialized views would be cost effectively answered by that ancestor view that is materialized and has the smallest size. Minimization of the costs C_{MV} and C_{NMV} would lead to minimization of the *TVEC*. However, reducing the cost C_{MV}, i.e. materializing fewer number views, would lead to increase in the number of non-materialized views, which in turn would increase the cost C_{NMV}. On the other hand, reducing the cost C_{NMV}, i.e. more views need to be materialized, would in turn lead to an increase in C_{MV}. That is, costs C_{MV} and C_{NMV} conflict with each other as minimization or improvement in C_{MV} lead to a deterioration in C_{NMV} and vice-versa. Therefore, costs C_{MV} and C_{NMV} needs to be simultaneously optimized or minimized in order to achieve an optimum trade-off between the two costs. Thus, the single objective *MVS* optimization problem is formulated as a bi-objective *MVS* optimization problem that would require optimization of the two objectives given below:

$$Minimize\, C_{MV} = \sum_{i=1 \wedge SM_{v_i} =1}^{N} Size\left(V_i\right)$$

and:

$$Minimize\, C_{NMV} = \sum_{i=1 \wedge SM_{v_i} =0}^{N} SizeSMA\left(V_i\right)$$

Several algorithms exist to solve such multi-objective optimization problems. One way to address such a problem is by using the scalar based approach, which employs weighted aggregation of multiple objectives to form a single objective optimization problem. The original problem of minimization of *TVEC* is a weighted aggregation of the two costs C_{MV} and C_{NMV}, each having the same weight. The major limitation of this approach is the requirement of aprior knowledge of the weights. Alternatively, approaches exist that does not require aprior knowledge of weights. These are majorly classified as *Pareto* based and non-*Pareto* based approaches (Schaffer, 1985; Deb, 2001). The *Pareto*-based approaches use the concept of *Pareto* optimality to simultaneously optimize the multiple objectives whereas, the non-*Pareto* approaches address the multiple objectives individually to simultaneously

optimize them. In this paper, a non-*Pareto* based multi-objective genetic algorithm *VEGA* (Schaffer, 1985) has been used to address the bi-objective *MVS* optimization problem. *VEGA* is discussed next.
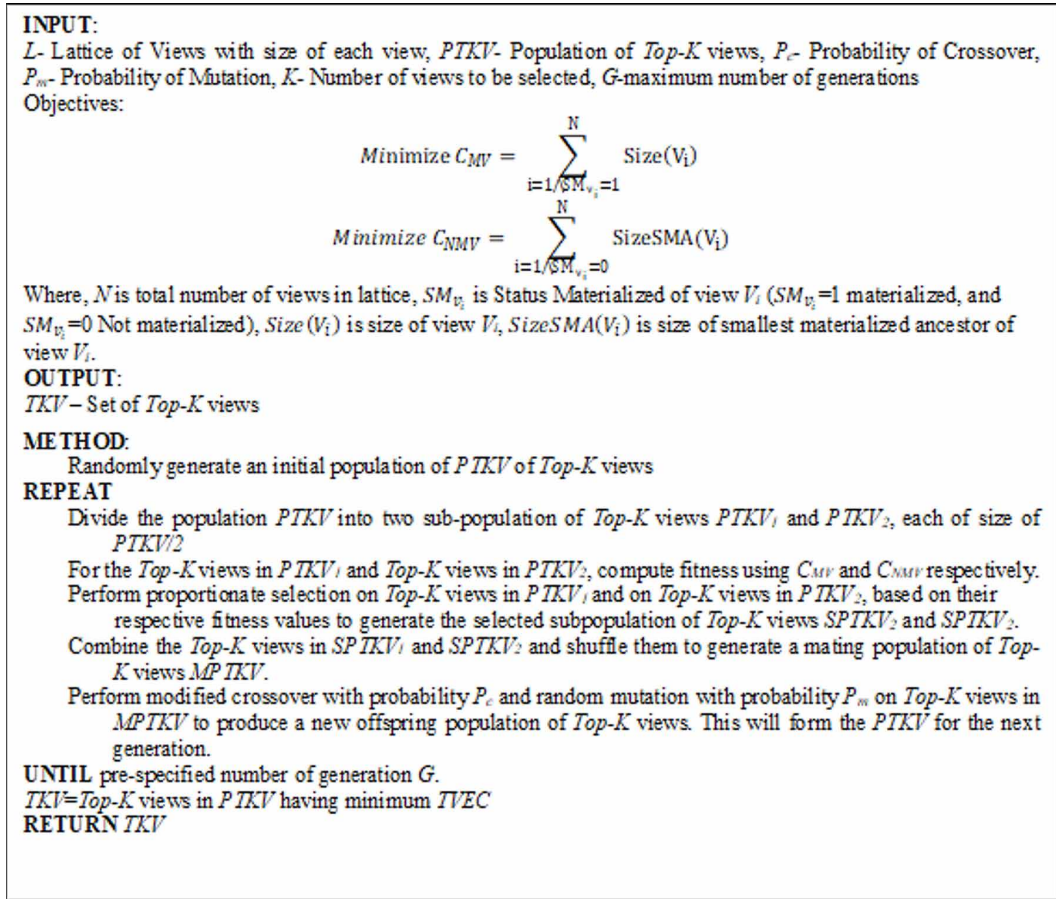
2.2. *VEGA*

VEGA (Schaffer, 1985; Deb, 2001) is the first multi-objective variant of the GA, which is a minor modification of the single objective GA to solve multi-objective optimization problems. In this algorithm, an objective vector is evaluated in which every element of the vector represents an objective function. In *VEGA*, for an *M* objective optimization problem, the population of individuals is divided into *M* equal subpopulations in every generation. The fitness of individuals in each subpopulation is evaluated using only one amongst the M objectives. Further, no two subpopulation individuals are evaluated on the same objective. In this way, individuals in each of the subpopulations are evaluated on a distinct objective and their respective fitness is computed. Next, from each subpopulation, individuals are selected using proportionate selection (Goldberg, 1989). The selected individuals from each of the subpopulations are combined together and shuffled to generate a mating pool of individuals. This is followed by applying crossover and mutation operators to generate a new offspring population for the next generation. The algorithms run for a pre-specified number of generations after which the best set of individuals is produced as the output. *VEGA* has been used to address the bi-objective *MVS* problem. Accordingly, a *VEGA* based *MVS* algorithm (MVS_{VEGA}) has been proposed and is discussed next.

2.3. MVS_{VEGA}

In the *MVS* problem, there is a need to select an appropriate subset of the *Top-K* views that minimizes the *TVEC*. Since the *MVS* problem is an *NP*-Complete problem, there is no deterministic view selection algorithm that can select the globally optimal *Top-K* views. One way to address such problems is by using the Genetic Algorithm. Further, the existing view selection algorithms solve the *MVS* problem as a single-objective optimization problem with the objective to minimize the *TVEC*. As mentioned above, the *TVEC* comprises of two costs C_{MV} and C_{NMV} and these two costs need to be simultaneously optimized, in order to reduce the *TVEC*. The proposed algorithm MVS_{VEGA} attempts to select the *Top-K* views by simultaneously minimizing these two costs. The algorithm MVS_{VEGA} is given in Figure 2 and the process in MVS_{VEGA} is depicted in Figure 3. The algorithm MVS_{VEGA} takes a lattice of views with size of each view *L*, probability of crossover P_c, probability of mutation P_m, number of views to be selected *K* and the maximum number of generations for which the algorithms should run *G* as the input and produces the *Top-K* views *TKV* as the output. In MVS_{VEGA}, initially a population of the *Top-K* views, *PTKV*, is randomly generated. Since *MVS* is formulated as a bi-objective optimization problem, the population *PTKV* is divided into two sub-populations $PTKV_1$ and $PTKV_2$, each of size *PTKV*/2. The fitness of the *Top-K* views in $PTKV_1$ is computed using the first objective function, i.e. C_{MV}, and fitness of *Top-K* views in $PTKV_2$ is computed using the second objective function, i.e. C_{NMV}. Next, the *Top-K* views from each of the subpopulations $PTKV_1$ and $PTKV_2$ are selected using the proportionate selection based on the fitness of the *Top-K* views in it and are computed using the corresponding objective function to generate the selected subpopulation of *Top-K* views $SPTKV_1$ and $SPTKV_2$. The *Top-K* views in $SPTKV_1$ and $SPTKV_2$ are then combined and shuffled to generate a mating population of *Top-K* views *MPTKV*. Thereafter, the modified crossover (Davis, 1985) and the random mutation (Goldberg, 1989) operators are applied to generate the offspring population of the *Top-K* views. This forms the new *PTKV* for the next generation. MVS_{VEGA} runs for a pre-specified number of generations *G* whereafter, the *Top-K* views in *PTKV* having the least *TVEC*, i.e. *TKV*, is produced as the output. An example illustrating use of MVS_{VEGA} to compute the *Top-K* views is given next.

Figure 2. Algorithm MVS_{VEGA}

INPUT:
L- Lattice of Views with size of each view, *PTKV*- Population of *Top-K* views, P_c- Probability of Crossover, P_m- Probability of Mutation, *K*- Number of views to be selected, *G*-maximum number of generations
Objectives:

$$Minimize\ C_{MV} = \sum_{i=1/SM_{v_i}=1}^{N} Size(V_i)$$

$$Minimize\ C_{NMV} = \sum_{i=1/SM_{v_i}=0}^{N} SizeSMA(V_i)$$

Where, *N* is total number of views in lattice, SM_{v_i} is Status Materialized of view V_i (SM_{v_i} =1 materialized, and SM_{v_i} =0 Not materialized), $Size(V_i)$ is size of view V_i, $SizeSMA(V_i)$ is size of smallest materialized ancestor of view V_i.
OUTPUT:
TKV – Set of *Top-K* views

METHOD:
    Randomly generate an initial population of *PTKV* of *Top-K* views
REPEAT
    Divide the population *PTKV* into two sub-population of *Top-K* views *PTKV₁* and *PTKV₂*, each of size of *PTKV/2*
    For the *Top-K* views in *PTKV₁* and *Top-K* views in *PTKV₂*, compute fitness using C_{MV} and C_{NMV} respectively.
    Perform proportionate selection on *Top-K* views in *PTKV₁* and on *Top-K* views in *PTKV₂*, based on their respective fitness values to generate the selected subpopulation of *Top-K* views *SPTKV₂* and *SPTKV₂*.
    Combine the *Top-K* views in *SPTKV₁* and *SPTKV₂* and shuffle them to generate a mating population of *Top-K* views *MPTKV*.
    Perform modified crossover with probability P_c and random mutation with probability P_m on *Top-K* views in *MPTKV* to produce a new offspring population of *Top-K* views. This will form the *PTKV* for the next generation.
UNTIL pre-specified number of generation *G*.
TKV=*Top-K* views in *PTKV* having minimum *TVEC*
RETURN *TKV*

3. AN EXAMPLE

Consider the selection of the *Top-4* views from a three-dimensional lattice shown in Figure 1. Let the initial parameters for MVS_{VEGA} be *PTKV* = 8, P_c =0.8 and P_m =0.1. The initial random population of the *Top-4* views *PT4V* is generated as shown in Figure 4.

Since there are two objectives, i.e. Minimize C_{MV} and Minimize C_{NMV}, the population is divided in two equal sub-populations *PT4V₁* and *PT4V₂*. These are given in Figure 5.

The fitness of the *Top-4* views in *PT4V₁* and *PT4V₂* are computed using C_{MV} and C_{NMV} respectively. The fitness so computed is given in Figure 6.

Next, the *Top-4* views in *PT4V₁* and the *Top-4* views in *PT4V₂* are selected using the proportionate selection (Goldberg, 1989) considering their respective fitness values to generate the selected subpopulation of the *Top-4* views *SPT4V₁* and *SPT4V₂*. These are given in Figure 7.

The *Top-4* views in *SPT4V₁* and *SPT4V₂* are then combined and shuffled to generate a mating population of *Top-4* views *MPT4V*, as given in Figure 8.

This is followed by applying the modified crossover on the *Top-4* views in *MPT4V* with probability 0.8 as shown in Figure 9.

After crossover, the random mutation with probability 0.1 is performed on the *Top-4* views produced after crossover, as shown in Figure 10. The *Top-4* views after mutation form the new
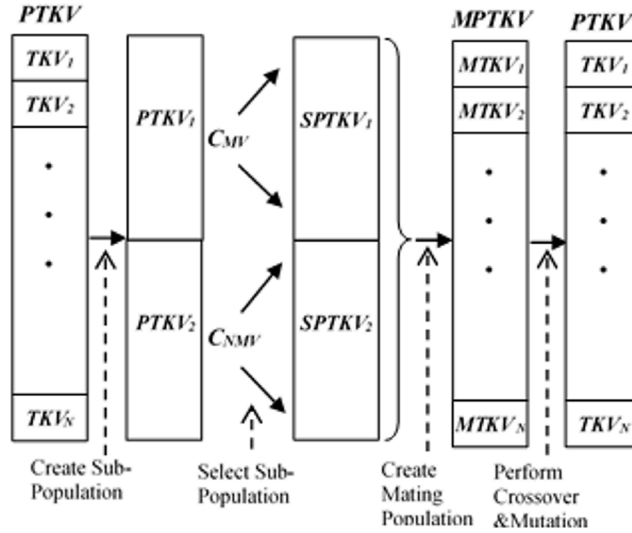
Figure 3. View selection in MVS_{VEGA}

Figure 4. PT4V

population of *Top-4* views *PT4V* for the next generation. The algorithms run for a pre-defined number of generations *G*. Thereafter, the *Top-K* views with having the least *TVEC* is produced as the output.

Next, experimental results comparing the outputs generated by MVS_{VEGA} and the most fundamental greedy view selection algorithm *HRUA*, is discussed.

4. EXPERIMENTAL RESULTS

The proposed view selection algorithm MVS_{VEGA} and the fundamental greedy view selection algorithm MVS_{HRUA}, were implemented using MATLAB-2012 in a Windows-7 environment on an Intel based

Figure 5. PT4V1 and PT4V2

PT4V₁					PT4V₂			
4	2	5	6		7	3	6	2
4	6	7	5		3	5	4	2
3	4	6	2		4	2	3	7
3	5	6	4		5	7	2	3

Figure 6. C_{MV} and C_{NMV} of *Top-4* views in $PT4V_1$ and $PT4V_2$

PT4V₁	Fitness Function C_{MV}	C_{MV}
4 2 5 6		269
4 6 7 5	$$C_{MV} = \sum_{i=1 \wedge SM_{v_i}=1}^{N} Size(V_i)$$	230
3 4 6 2		319
3 5 6 4		284
PT4V₂	Fitness Function C_{NMV}	C_{NMV}
7 3 6 2		182
3 5 4 2	$$C_{NMV} = \sum_{i=1 \wedge SM_{v_i}=0}^{N} SizeSMA(V_i)$$	144
4 2 3 7		148
5 7 2 3		191

Figure 7. $SPT4V_1$ and $SPT4V_2$

PT4V₁					Proportionate Selection	*SPT4V₁*			
4	2	5	6			4	2	5	6
4	6	7	5			4	6	7	5
3	4	6	2			4	6	7	5
3	5	6	4			3	5	6	4
PT4V₂					Proportionate Selection	*SPT4V₂*			
7	3	6	2			7	3	6	2
3	5	4	2			4	2	3	7
4	2	3	7			3	5	4	2
5	7	2	3			4	2	3	7

Figure 8. MPT4V

MPT4V			
4	2	5	6
4	6	7	5
7	3	6	2
4	6	7	5
4	2	3	7
3	5	4	2
3	5	6	4
4	2	3	7

Figure 9. Modified crossover on *Top-4* **views in** *MPT4V*

Top-4 views for Crossover	Crossover Point	*Top-4* views after Crossover
4 2 5 6 4 2 3 7	3	4 2 5 7 4 2 3 6
4 6 7 5 7 3 6 2	2	4 6 2 7 7 3 5 4
4 2 3 7 3 5 4 2	1	4 5 2 3 3 2 7 4
4 2 3 7 4 6 7 5	2	4 2 7 5 4 6 3 7

2.27 Ghz PC having 8 GB RAM. The experiments have been conducted to ascertain the comparative performance of MVS_{VEGA} and MVS_{HRUA} in terms of the quality of the *Top-K* views selected by them. First, experiments were carried out to determine the appropriate value of the probability of crossover and the probability of mutation for which MVS_{VEGA} is able to select the *Top-10* views with least *TVEC* for dimensions 5, 6, 7, 8, 9 and 10. The corresponding graphs for the dimensions 5, 6, 7, 8, 9 and 10 are shown in Figure 11, Figure 12, Figure 13, Figure 14, Figure 15 and Figure 16 respectively. From these graphs, it can be inferred that MVS_{VEGA} is able to select *Top-K* views that have the minimum

Figure10. Mutation on Top-4 views produced after crossover

Top-4 views for Mutation	Mutation Point	*Top-4* views after Mutation
4 2 5 7	-	4 2 5 7
4 2 3 6	4	4 2 3 7
4 6 2 7	-	4 6 2 7
7 3 5 4	3	7 3 6 4
4 5 2 3	-	4 5 2 3
3 2 7 4	1	5 2 7 4
4 2 7 5	2	4 6 7 5
4 6 3 7	-	4 6 3 7

Figure 11. MVS_{VEGA} – *TVEC* of *Top-10* views, Dimensions=5
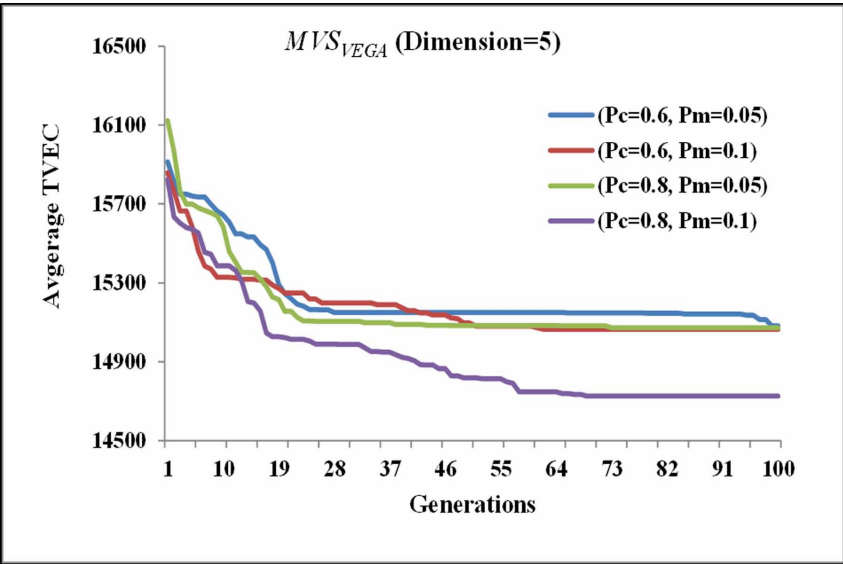
TVEC for P_c=0.8 and P_m=0.1. Using these values of P_c and P_m, MVS_{VEGA} is compared with the most fundamental view selection algorithm *HRUA* (MVS_{HRUA}) in terms of the *TVEC* of the *Top-K* views selected by the two algorithms for dimensions 5, 6, 7, 8, 9 and 10. For these, the graphs are plotted and are shown in Figure 17, Figure 18, Figure 19, Figure 20, Figure 21 and Figure 22 respectively.

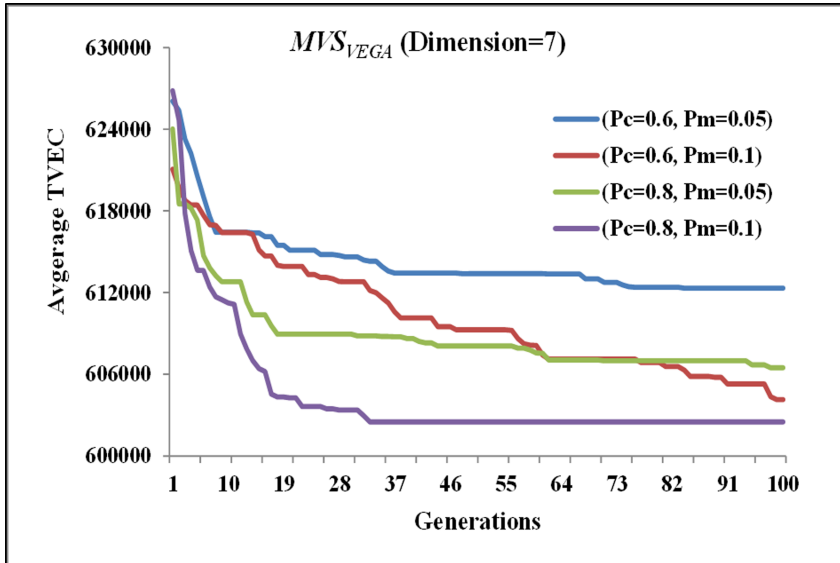**Figure 12.** MVS_{VEGA} – *TVEC* **of** *Top-10* **views, Dimensions=6**
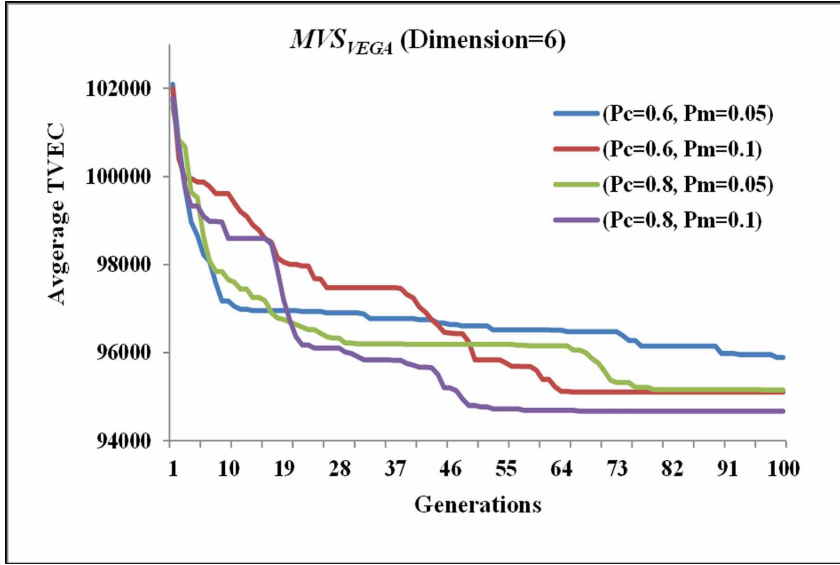
Figure 13. MVS_{VEGA} – *TVEC* **of** *Top-10* **views, Dimensions=7**

From these graphs, it can be observed that *TVEC* values of the *Top-K* views selected using MVS_{VEGA} is comparatively lower than those selected using MVS_{HRUA} for all dimensions and for all values of *K*. The difference in *TVEC* values of MVS_{VEGA} and MVS_{HRUA} increases with increase in the value of *K* and with an increase in the number of dimensions. The better performance of MVS_{VEGA} may be attributed to its ability to optimize the two objectives simultaneously, and thereby achieving a trade-off between the two objectives leading to a comparatively lower value of *TVEC*.

Figure 14. *MVS$_{VEGA}$* – *TVEC* **of** *Top-10* **views, Dimensions=8**
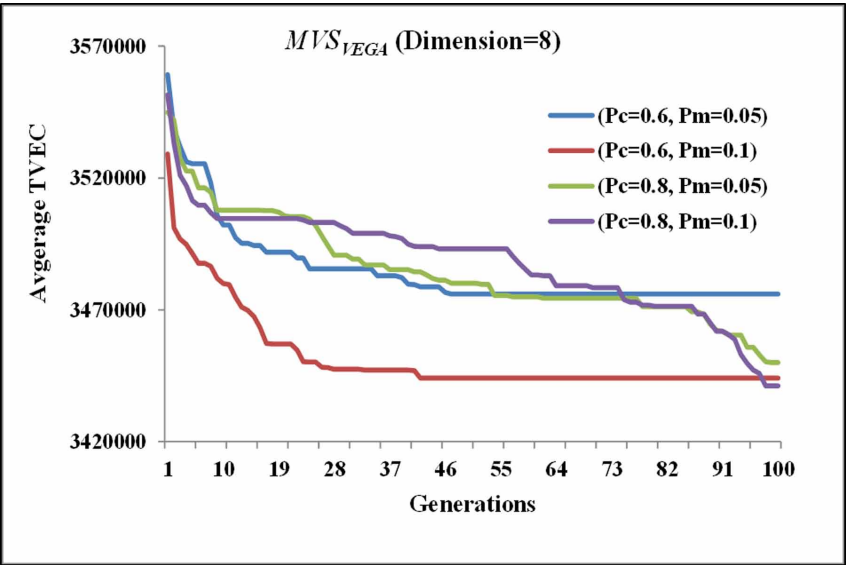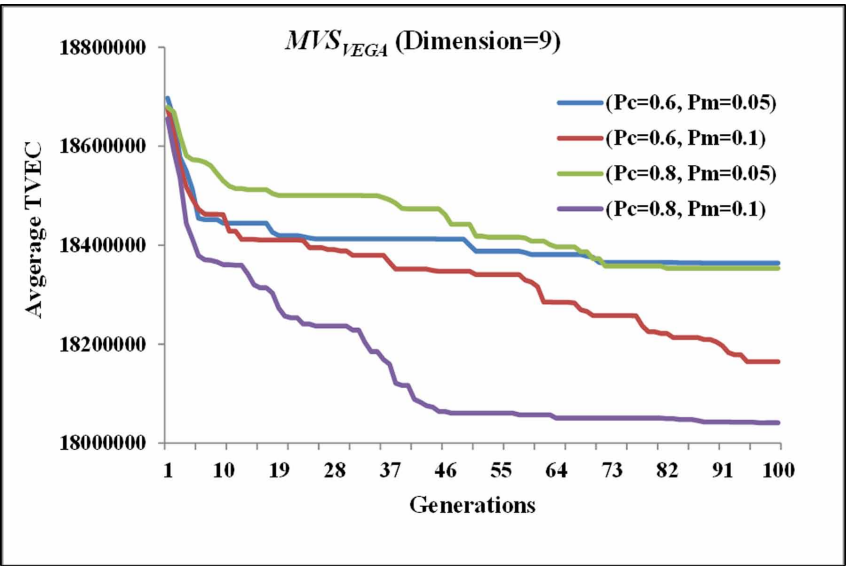

Figure 15. *MVS$_{VEGA}$* – *TVEC* **of** *Top-10* **views, Dimensions=9**

5. CONCLUSION

In this paper, the materialized views selection problem is formulated as a bi-objective optimization problem with the two objectives being the minimization of the cost due to materialized views and the minimization of the cost due to non-materialized views. These objectives, being conflicting in

Figure 16. MVS_{VEGA} – *TVEC* **of** *Top-10* **views, Dimensions=10**
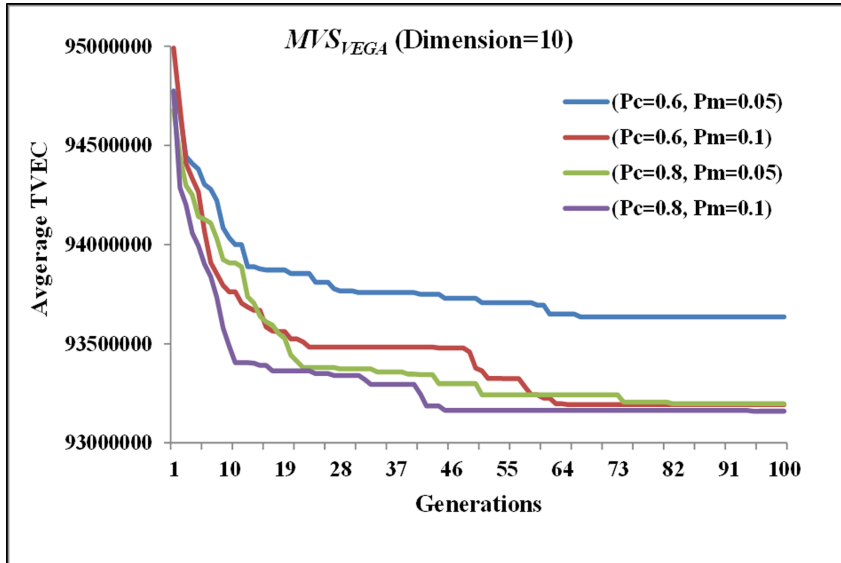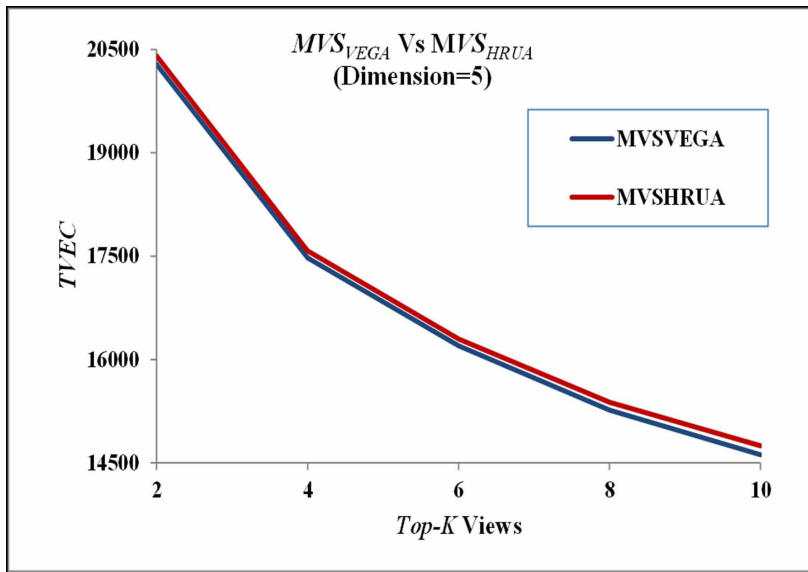

Figure 17. MVS_{VEGA} **Vs.** MVS_{HRUA} **- Dimensions=5**

nature, are simultaneously optimized using *VEGA*. Accordingly, *VEGA* based materialized view selection algorithm MVS_{VEGA} has been proposed that selects *Top-K* views from a multidimensional lattice. Experiments based comparison of MVS_{VEGA} with the most fundamental view selection algorithm MVS_{HRUA} shows that the former is able to select comparatively better quality of *Top-K*

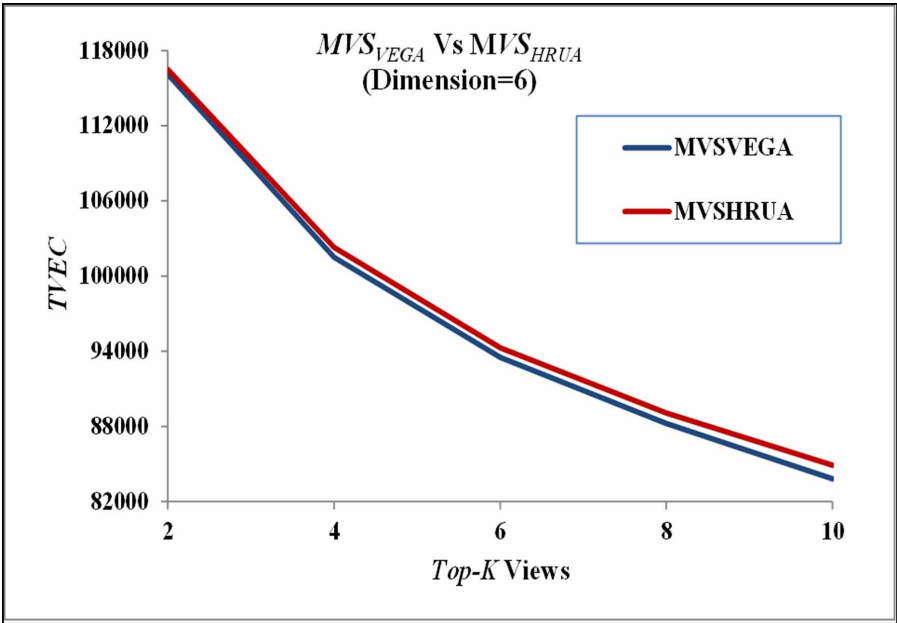**Figure 18.** MVS_{VEGA} **Vs.** MVS_{HRUA} **- Dimensions=6**

Figure 19. MVS_{VEGA} **Vs.** MVS_{HRUA} **- Dimensions=7**
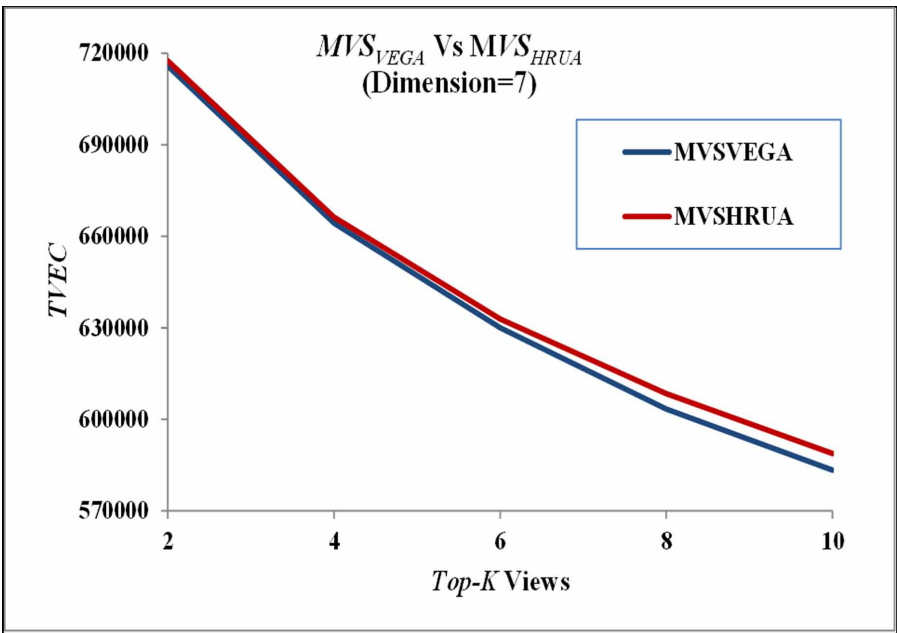
Figure 20. MVS_{VEGA} **Vs.** MVS_{HRUA} **- Dimensions=8**
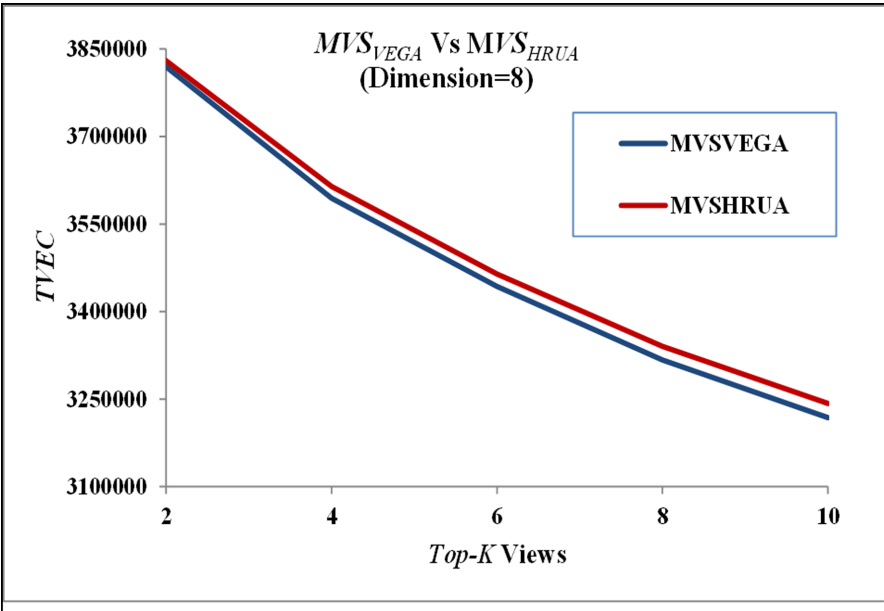
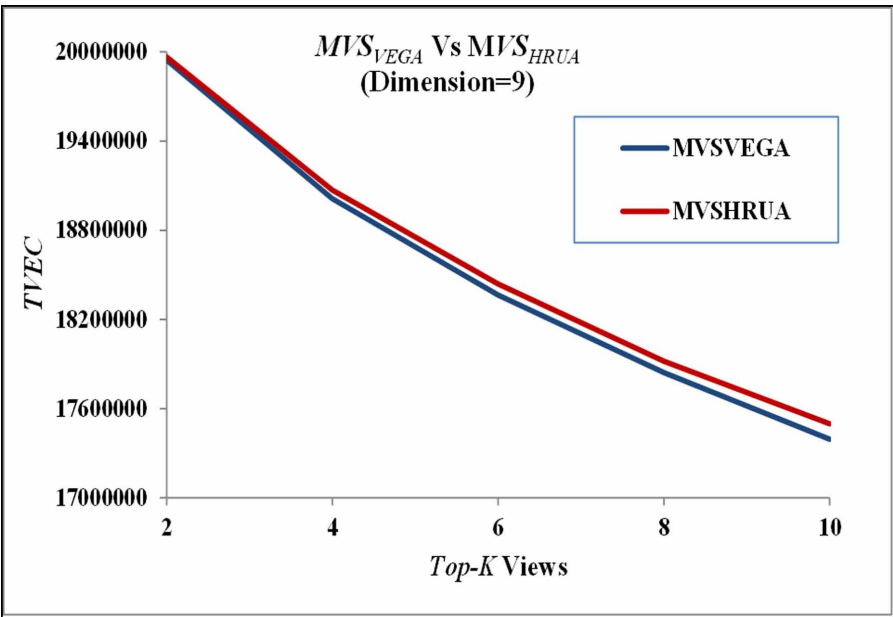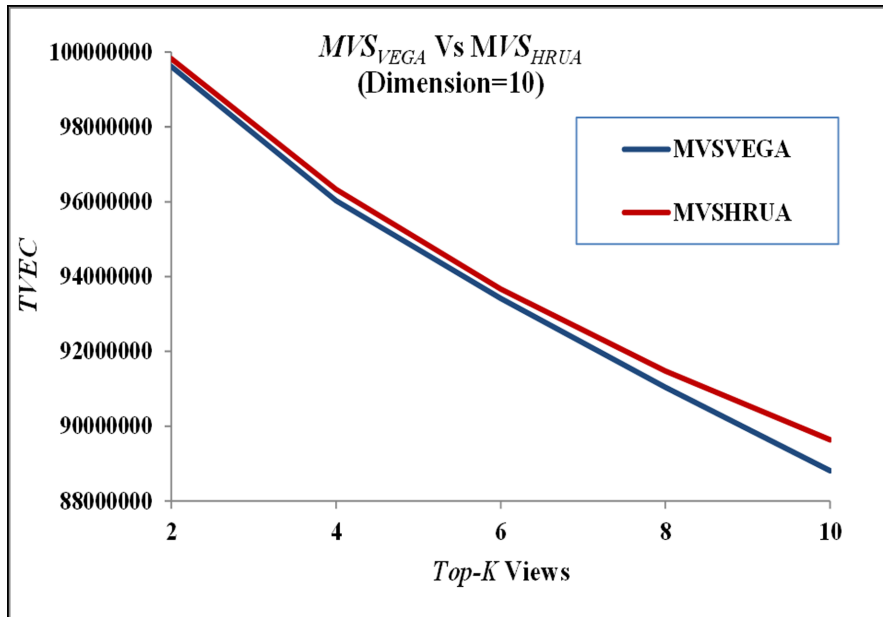**Figure 21.** MVS_{VEGA} **Vs.** MVS_{HRUA} **- Dimensions=9**

Figure 22. MVS_{VEGA} **Vs.** MVS_{HRUA} **- Dimensions=10**

views, in terms of *TVEC*, than those selected by the latter. Resultantly, views selected using MVS_{VEGA}, if materialized, would improve the query response time of analytical queries and thereby greatly facilitate decision making.

ACKNOWLEDGMENT

REFERENCES

Arun, B., & Vijay Kumar, T. V. (2015a). Materialized View Selection using Marriage in Honey Bees Optimization. *International Journal of Natural Computing Research*, *5*(3), 1–25. doi:10.4018/IJNCR.2015070101

Arun, B., & Vijay Kumar, T. V. (2015b). Materialized View Selection using Improvement Based Bee Colony Optimization. *International Journal of Software Science and Computational Intelligence*, *7*(4), 35–61. doi:10.4018/IJSSCI.2015100103

Arun, B., & Vijay Kumar, T. V. (2017a). Materialized View Selection using Artificial Bee Colony Optimization. *International Journal of Intelligent Information Technologies*, *13*(1), 26–49. doi:10.4018/IJIIT.2017010102

Arun, B., & Vijay Kumar, T. V. (2017b). Materialized View Selection using Bumble Bee Mating Optimization. *International Journal of Decision Support System Technology*, *9*(3), 1–27. doi:10.4018/IJDSST.2017070101

Baralis, E., Paraboschi, S., & Teniente, E. (1997). Materialized view selection in a multidimensional database. In *VLDB '97 Proceedings of the 23rd International Conference on Very Large Data Bases* (pp. 156–165).

Chirkova, R., Halevy, A. Y., & Suciu, D. (2002). A formal perspective on the view selection problem. *The VLDB Journal*, *11*(3), 216–237. doi:10.1007/s00778-002-0070-0

Choi, C. H., Yu, J. X., & Gou, G. (2002). What difference heuristics make: maintenance-cost view selection revisited. In *Proceedings of the 3rd International Conference on Advances in Web-Age Information Management* (pp. 313-350). Springer-Verlag. doi:10.1007/3-540-45703-8_23

Davis, L. (1985). Applying adaptive algorithms to epistatic domains. In *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 162–164).

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*. New York, NY: John Wiley & Sons, Inc.

Dondi, R., Mauri, G., & Zoppis, I. (1999). On the Complexity of the View-Selection Problem. In PODS'99 (pp. 167–173).

Goldberg, D. E. (1989). *Genetic Algorithms in Search* (Vol. 1). Boston, MA: Optimization, and Machine Learning.

Gray, J., Chaudhuri, S., Bosworth, A., Layman, A., Reichart, D., Venkatrao, M., & Pirahesh, H. et al. (1997). Data Cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-totals. *Data Mining and Knowledge Discovery*, *1*(1), 29–53. doi:10.1023/A:1009726021843

Gupta, H. (1997). Selection of views to materialize in a data warehouse. In Database Theory—ICDT'97 (pp. 98-112).

Gupta, H., Harinarayan, V., Rajaraman, A., & Ullman, J. D. (1997, April). Index selection for OLAP. In Proceedings 13th International Conference on Data Engineering (pp. 208-219). doi:10.1109/ICDE.1997.581755

Gupta, H., & Mumick, I. S. (2005). Selection of views to materialize in a data warehouse. *IEEE Transactions on Knowledge and Data Engineering*, *17*(1), 24–43. doi:10.1109/TKDE.2005.16

Haider, M., & Vijay Kumar, T. V. (2011). Materialized Views Selection using Size and Query Frequency. *International Journal of Value Chain Management*, *5*(2), 95–105. doi:10.1504/IJVCM.2011.042071

Haider, M., & Vijay Kumar, T. V. (2017). Query Frequency based View Selection. *International Journal of Business Analytics*, *4*(1), 36–55. doi:10.4018/IJBAN.2017010103

Harinarayan, V., Rajaraman, A., & Ullman, J. D. (1996). Implementing data cubes efficiently. *SIGMOD Record*, *25*(2), 205–216. doi:10.1145/235968.233333

Horng, J. T., Chang, Y. J., & Liu, B. J. (2003). Applying evolutionary algorithms to materialized view selection in a data warehouse. *Soft Computing*, *7*(8), 574–581. doi:10.1007/s00500-002-0243-1

Inmon, W. H. (2003). *Building the Data Warehouse* (3rd ed.). Wiley Dreamtech.

Kimball, R., & Ross, M. (2002). *The Data Warehouse Toolkit* (2nd ed.). Wiley Computer Publishing.

Kumar, A., & Vijay Kumar, T. V. (2017). Improved Quality View Selection for Analytical Query Performance Enhancement using Particle Swarm Optimization. *International Journal of Reliability Quality and Safety Engineering*, *24*(6), 1740001. doi:10.1142/S0218539317400010

Kumar, A., & Vijay Kumar, T. V. (2018a). Materialized View Selection using Set based Particle Swarm Optimization. *International Journal of Cognitive Informatics and Natural Intelligence*, *12*(3), 18–39. doi:10.4018/IJCINI.2018070102

Kumar, S., & Kumar, T. V. (2018). A novel quantum-inspired evolutionary view selection algorithm. *Sadhana*, *43*(10), 166.

Kumar, T. V., & Haider, M. (2011). Selection of views for materialization using size and query frequency. In Information Technology and Mobile Communication (pp. 150-155). Springer. doi:10.1007/978-3-642-20573-6_24

Lehner, W., Ruf, T. & Teschke, M. (1996). 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 96*, Zurich (pp. 201-206).

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations Research*, *21*(2), 498–516. doi:10.1287/opre.21.2.498

Mohania, M., Samtani, S., Roddick, J. F., & Kambayashi, Y. (1999). Advances and research directions in data-warehousing technology. *AJIS. Australasian Journal of Information Systems*, *7*(1), 41–59. doi:10.3127/ajis.v7i1.287

Roussopoulos, N. (1982). The Logical Access Path Schema of a Database. *IEEE Transactions on Software Engineering*, *SE-8*(6), 563–573. doi:10.1109/TSE.1982.235886

Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms* (pp. 93–100).

Shukla, A., Deshpande, P. M., & Naughton, J. F. (1998). Materialized View Selection for Multidimensional Datasets. In *Proceedings of the 24rd International Conference on Very Large Data Bases (VLDB-1998)*, August 24-27 (pp. 488-499).

Theodoratos, D., & Sellis, T. K. (1997). Data warehouse configuration. In *Proceedings of 23rd International Conference on Very Large Data Bases (VLDB-1997)*, August 25-29 (pp. 126-135).

Valluri, S. R. S., Vadapalli, S., & Karlapalem, K. (2002). View relevance driven materialized view selection in data warehousing environment. *Australian Computer Science Communications*, *24*(2), 187–196.

Vijay Kumar, T. V. (2013). Answering query-based selection of materialized views. *International Journal of Information and Decision Sciences*, *5*(1), 103–116. doi:10.1504/IJIDS.2013.052015

Vijay Kumar, T. V., & Arun, B. (2016). Materialized View Selection using BCO. *International Journal of Business Information Systems*, *22*(3), 280–301.

Vijay Kumar, T. V., & Arun, B. (2017). Materialized View Selection using HBMO. *International Journal of Systems Assurance Engineering and Management*, *8*(1), 379–392. doi:10.1007/s13198-015-0356-4

Vijay Kumar, T. V., & Devi, K. (2012). Materialized View Construction in Data Warehouse for Decision Making. *International Journal of Business Information Systems*, *11*(4), 379–396. doi:10.1504/IJBIS.2012.050172

Vijay Kumar, T. V., & Devi, K. (2013). An Architectural Framework for Constructing Materialized Views in a Data Warehouse. system. *International Journal of Innovation, Management and Technology*, *4*(2), 192–197.

Vijay Kumar, T. V., & Ghoshal, A. (2009). A Reduced Lattice Greedy Algorithm for Selecting Materialized Views. In International Conference on Information Systems, Technology and Management (pp. 6-18). Springer. doi:10.1007/978-3-642-00405-6_5

Vijay Kumar, T. V., Goel, A., & Jain, N. (2010a). Mining information for constructing materialized views. *International Journal of Information and Communication Technology*, *2*(4), 386–405. doi:10.1504/IJICT.2010.034979

Vijay Kumar, T. V., & Haider, M. (2010). A Query Answering Greedy Algorithm for Selecting Materialized Views. In International Conference on Computational Collective Intelligence (pp. 153-162). Springer.

Vijay Kumar, T. V., & Haider, M. (2011a). Greedy Views Selection using Size and Query Frequency. In Advances in Computing, Communication and Control (pp. 11-17). Springer. doi:10.1007/978-3-642-18440-6_2

Vijay Kumar, T. V., & Haider, M. (2012). Materialized Views Selection for Answering Queries. In Data Engineering and Management (pp. 44-51). Springer. doi:10.1007/978-3-642-27872-3_7

Vijay Kumar, T. V., & Haider, M. (2015). Query Answering Based View Selection. *International Journal of Business Information Systems*, *18*(3), 338–353. doi:10.1504/IJBIS.2015.068168

Vijay Kumar, T. V., Haider, M., & Kumar, S. (2010b). Proposing candidate views for materialization. *Communications in Computer and Information Science*, *54*, 89–98. doi:10.1007/978-3-642-12035-0_10

Vijay Kumar, T. V., Haider, M., & Kumar, S. (2011). A View Recommendation Greedy Algorithm for Materialized Views Selection. *Communications in Computer and Information Science*, *141*, 61–70. doi:10.1007/978-3-642-19423-8_7

Vijay Kumar, T. V., Kumar, A., & Arun, B. (2017). *Cuckoo Search Based View Selection, Emerging Research in Computing, Information, Communication and Applications (ERCICA-2016)* (pp. 326–338). Springer Verlag.

Vijay Kumar, T. V., & Kumar, S. (2012a). Materialized View Selection using Iterative Improvement. *Advances in Intelligent Systems and Computing*, *178*, 205–214. doi:10.1007/978-3-642-31600-5_21

Vijay Kumar, T. V., & Kumar, S. (2012b). Materialized View Selection using Genetic Algorithm. *Communications in Computer and Information Science*, *306*, 225–237. doi:10.1007/978-3-642-32129-0_26

Vijay Kumar, T. V., & Kumar, S. (2012c). Materialized View Selection using Simulated Annealing. *Lecture Notes in Computer Science*, *7678*, 168–179. doi:10.1007/978-3-642-35542-4_15

Vijay Kumar, T. V., & Kumar, S. (2013). Materialized View Selection using Memetic Algorithm. *Lecture Notes in Artificial Intelligence*, *8284*, 316–327.

Vijay Kumar, T. V., & Kumar, S. (2014). Materialized View Selection using Differential Evolution. *International Journal of Innovative Computing and Applications*, *6*(2), 102–113. doi:10.1504/IJICA.2014.066499

Vijay Kumar, T. V., & Kumar, S. (2015). Materialized View Selection using Randomized Algorithms. *International Journal of Business Information Systems*, *19*(2), 224–240. doi:10.1504/IJBIS.2015.069432

Widom, J. (1995), Research problems in data warehousing, in *Proceedings of the fourth international conference on Information and knowledge management - CIKM '95*, pp. 25–30. doi:10.1145/221270.221319

Yang, J., Karlapalem, K., & Li, Q. (1997). Algorithms for materialized view design in data warehousing environment. In *Proceedings of 23rd International Conference on Very Large Data Bases (VLDB-1997)*, August 25 - 29 (pp. 136-145).

Zhang, C., Yao, X., & Yang, J. (2001). An evolutionary approach to materialized views selection in a data warehouse environment. *Systems, IEEE Transactions on Man, and Cybernetics, Part C: Applications and Reviews*, *31*(3), 282–294. doi:10.1109/5326.971656

Jay Prakash is currently pursuing PhD in computer science from Jawaharlal Nehru University, New Delhi, India. He has completed his M.Tech in Computer Science and Technology from Jawaharlal Nehru University, New Delhi, India after completing his MCA and BCA from Indira Gandhi National Open University, New Delhi, India. His research interests are databases, evolutionary algorithms for multi-objective optimization and Nature Inspired Computing.

T.V. Vijay Kumar has completed his PhD in the area of Databases from Jawaharlal Nehru University, New Delhi, India after completing his MPhil and MSc in Operational Research, and BSc (Hons) in Mathematics, from the University of Delhi, Delhi, India. His research interests are databases, data warehousing, data mining, machine learning, nature inspired algorithms, big data and analytics.