Sādhanā (2018) 43:166
https://doi.org/10.1007/s12046-018-0936-5

A novel quantum-inspired evolutionary view selection algorithm

SANTOSH KUMAR[1] and T V VIJAY KUMAR[2,*]

[1] Department of Computer Science and Engineering, ABES Engineering College, Ghaziabad, India
[2] School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, India
e-mail: santoshg25@gmail.com; tvvijaykumar@hotmail.com

MS received 6 April 2016; accepted 20 June 2018; published online 29 August 2018

Abstract. A data warehouse (*DW*) is designed primarily to meet the informational needs of an organization's decision support system. Most queries posed on such systems are analytical in nature. These queries are long and complex, and are posed in an exploratory and ad-hoc manner. The response time of these queries is high when processed directly against a continuously growing *DW*. In order to reduce this time, materialized views are used as an alternative. It is infeasible to materialize all views due to storage space constraints. Further, optimal view selection is an *NP*-Complete problem. Alternately, a subset of views, from amongst all possible views, needs to be selected that improves the response time for analytical queries. In this paper, a quantum-inspired evolutionary view selection algorithm (*QIEVSA*) that selects *Top-K* views from a multidimensional lattice has been proposed. Experimental comparison of *QIEVSA* with other evolutionary view selection algorithms shows that *QIEVSA* is able to select *Top-K* views that are comparatively better in reducing the response times for analytical queries. This in turn aids in efficient decision making.

Keywords. Data warehouse; on-line analytical processing; materialized view selection; quantum-inspired evolutionary algorithm.

1. Introduction

Ever since the invention of computers, and especially since their ever-increasing acceptance with businesses, almost all organizations have nowadays computerized their business operations. Business data are collected and stored using computers. Managing increasing volumes of digital data was a tricky proposition from the very beginning. On-Line Transaction Processing (OLTP) systems were developed to collect transaction data [1]. These systems were designed with complex structures in order to minimize data redundancies on account of very fast write operations. The informational value of this stored transactional data was recognized from the very beginning considering that many applications, based on them, for report generation, trend analysis, etc., were developed for decision support. OLTPs were not designed for fast retrieval operations of huge volumes of data for their analysis; hence it performed very limited analysis and that to inefficiently [1, 2]. With the advent of computer networks having multiple communication protocols, computers with varied hardware configurations, distinct operating systems and disparate database management systems, the stand-alone transactional data sources of an organization became incompatible and obsolete. As a result, the credibility of an organization's data asset for analysis diminished and data analysis using incompatible data sources became an exorbitantly costly process, causing information bottlenecks in the organization [3]. A data warehouse (*DW*) was designed as an alternative to address this crisis with the aim of meeting the informational needs of the organization's decision support system (*DSS*) [4]. *DW* is a centralized repository of historic, integrated, subjected-oriented, time-variant and non-volatile data, created and maintained for supporting complex data analysis, for acquiring information to support strategic and tactical decision making [1–3]. In the *DW*, data from various disparate, remote, heterogeneous and incompatible transactional data sources are extracted, transformed, consolidated and integrated to obtain correct, unambiguous, consistent and complete data. In transactional data sources, data are organized and stored around specific operational applications; however, in a *DW*, data are organized and stored not by their applications, but by business subjects crucial for the organization [2, 3, 5]. Generally, operational systems store the current value of data by overwriting its previous value; they do not maintain a record of all its preceding values. As a result, historical analysis cannot be performed using such operational data. In contrast, a *DW*, in addition to the current value of data, stores all values ever assigned to the data since its creation. The data that enter a *DW* are never deleted, but stored permanently throughout the life of the organization. Such time-variant

*For correspondence

and non-volatile data facilitates historical and predictive analysis. A *DW* stores atomic data, as well as summarized data, to support detailed and fast analytical queries.

The sequence of interactive and multi-perspective data analysis performed on the *DW* using some data analytical tool is called On-Line Analytical Processing (*OLAP*), as this is the primary goal of building a *DW* [2, 6–8]. *OLAP* queries access very large volumes of historical and aggregate data at different levels of granularity using roll-up and drill down *OLAP* operators along certain combinations of dimensions (sets of attributes). They use slice and dice *OLAP* operators to select and project a desired portion of data. A pivot operator is used by them to reorient the data to another view. These complex *OLAP* operations are made possible by the multidimensional data model [6–8]. It provides a multidimensional view of the data comprising measures and dimensions [9]. A measure is the numerical quantification of an event of interest and a dimension is a set of attributes forming the context of a measure [9]. Product, Customer and Time can be dimensions for a sales measure as shown in figure 1. The multidimensional data model is implemented using star schema in relational database management systems. It is implemented using multidimensional arrays in a multidimensional database [9]. The star schema with Sales as a fact table and Customer, Product and Time as the dimensional tables is shown in figure 2.

One of the major challenges during an *OLAP* session is the speed with which the *DW* and *OLAP* components of a *DSS* are able to respond to *OLAP* queries posed by analysts. Although the available *OLAP* operators have been successfully assisting such queries, if *DSS* takes hours or days to answer these, the analysis would be very limited and less productive as response delays would break the chain of ad-hoc analytical thoughts of an analyst and the information obtained at the end of such a long *OLAP* sessions would mostly be obsolete and irrelevant. Not only the depth and width of the information are essential but also the speed at which such information is delivered is all the more important to stay competitive in a dynamic market environment having many other competitors [3]. *OLAP* queries

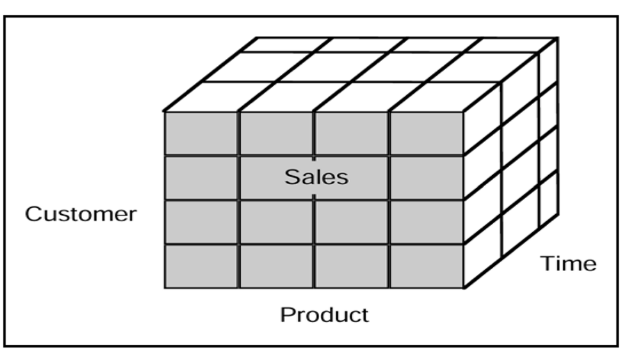

Figure 1. Data cube.

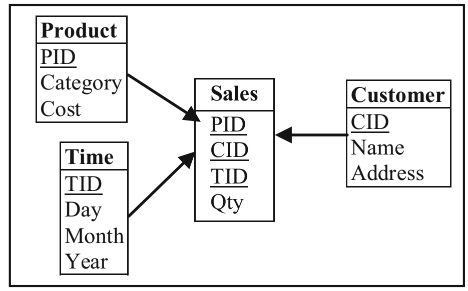

Figure 2. Star schema.

are usually long, complex, exploratory and ad hoc in nature. These queries when posed against a voluminous *DW* consume lot of time for their processing, thereby increasing the query response time. This problem worsens due to the continuously growing data in a *DW*, as these queries may take hours and days to process, where the desired requirement is of only a few seconds and minutes. Although several query optimization techniques and indexing strategies [10–14] exist, they do not scale up with the ever increasing voluminous data in the *DW*. Scalability can be addressed by pre-computing relevant and frequently accessed data in the *DW* and storing it separately as materialized views [15] in the *DW*. This pre-computed information, which is significantly small when compared with the voluminous *DW*, can be used to process *OLAP* queries in an efficient manner.

Materialized views, unlike virtual views, store data along with their query definition. Their primary purpose is to reduce the response time of *OLAP* queries. Amongst the issues associated with materialized views like view maintenance, view synchronization, view adaptation, answering queries using views and view selection [12], view selection is the focus of this paper and is discussed next.

1.1 *View selection*

View selection is concerned with the selection of an appropriate subset of views from amongst all possible views for a given query workload that can improve the query performance, while conforming to the resource constraints in terms of storage space, maintenance cost, etc. [16–20]. It has been studied in the context of query optimization, warehouse design, distributed databases, semantic web databases, etc. [17, 21–23]. According to [20], view selection is concerned with the selection of views, for a given query workload, from a database that is capable of being stored within the available storage space. Since the number of views is exponential with respect to the number of dimensions, all possible views cannot be materialized due to storage space constraint. Thus, there is a need to select a subset of views that conform to the storage space constraints from amongst all possible views. Further, optimal selection of subse t of views is a *NP*-Complete problem [13]. The objective then is to select a subset of good quality views, if not optimal, for materialization. This subset cannot be arbitrarily selected, as it may result in selecting views that are irrelevant and not capable of answering the *OLAP* queries. Several approaches exist that have been attempted to select this subset of views, which are broadly classified as empirically based or heuristically based [24]. The empirically based view selection approach [24–33] considers data accessed by past queries as indicators of data that are likely to be accessed by future queries. These approaches monitor and assess these queries on parameters like the query frequency, data accessed, etc., and use this information to select subsets of views from materialization. On the other hand, heuristically based view selection approaches use heuristics, to prune the search space of all possible views to select views that can reduce the response time of *OLAP* queries. Most of these are based on greedy heuristics [12, 13, 34–50]. Among these, most of them focus on the issues and problems associated with the greedy view selection algorithm given in [13], also referred to as *HRUA* [36, 37, 43–49]. This paper also proposes a view selection algorithm that attempts to address the issues and problems associated with the greedy view selection algorithm *HRUA* given in [13].

HRUA, in each iteration, computes the benefit of each view of a multidimensional lattice, using its size, and selects amongst them the one having maximum benefit. For selecting *Top-K* views, this process continues for *K* number of iterations. *HRUA* aims to select the *Top-K* views that minimize the total cost of evaluating all the views. However, *HRUA* has certain limitations. Since the number of views is exponential with respect to the number of dimensions, increase in the number of dimensions leads to a deterioration in the quality of views selected using *HRUA* with respect to *TVEC*. This is because the possible number of views from which to select *Top-K* views increases with increase in the number of dimensions. Thus *HRUA*, being a greedy algorithm, selects a sub-optimal quality of views. Another limitation with *HRUA* is that it is not scalable for higher dimensions, i.e., it becomes computationally infeasible to select views for higher dimensional data sets. These limitations need to be addressed in order to select good quality views for higher dimensional data sets. One way to address this problem is by selecting views using evolutionary algorithms. Darwin's theory of evolution by means of natural selection has been the heuristic used in evolutionary algorithms [51]. According to it, genetically and behaviourally well adapted progenies survive and replace their parents to reproduce in order to prolong the existence of their species. Many aspects of natural evolution have been viewed as computational processes and have been adapted with stochastic elements to solve complex computational problems [51]. These algorithms have been used for view selection. Evolutionary view selection is discussed next.

1.2 *Evolutionary view selection*

Evolutionary view selection would select views by exploring and exploiting the search space of all possible views using evolutionary operators like selection, crossover and mutation with an aim to select good sets of views for materialization. Several evolution-based view selection algorithms exist [52–63]. Reference [62] used a genetic algorithm (*GA*) to select an optimal set of views, for a given set of multiple query processing plans, to minimize its processing cost. The proposed *GA* performed better than heuristics of [62] and [18]; it was also observed that the hybrid of *GA* and heuristics of [62] and [18] performed better than the *GA* and heuristics of [62] and [18]. Reference [52] used the Genetic Local Search (*GLS*) algorithm using AND–OR view graph to address the view selection problem. The GLS searches in two steps; in the first step it uses a local search to improve the population of chromosomes, whereupon the *GA* is employed to diversify the search to look out for an unexplored solution space. It was applied to a real database to determine its efficiency. *GA*, in conjunction with OR view graph, was used for the maintenance-cost view selection problem [54]. The results obtained were consistently within 10 percent of the optimal solution; further, it exhibited a linear execution time. Reference [61] proposed a two-level structure for materialized view selection and used evolutionary algorithms with Multiple View Processing Plan (*MVPP*) framework to select views. It was observed that evolutionary algorithms had impractically long computation time and the quality of solutions was no better than those of the hybrid of evolutionary algorithms and heuristics of [61] and [18]. Reference [60] proposed a stochastic ranking evolutionary algorithm (*SEA*) for the maintenance cost view selection problem. In finding optimal solutions, it was observed that *SEA* was able to obtain near-optimal results that were very close to those obtained by A*-heuristic; it was also noticed that *SEA* performed much better than the algorithm proposed by [54]. A* and *SEA* performed better than the algorithm proposed by [54] and the inverted-tree greedy for smaller instances of the problem. As A* and inverted-tree greedy were not able to handle lattices with 256 nodes, SEA and the algorithm of [54] were compared for higher dimension data sets; it was observed that the solutions obtained by *SEA* were much better than those obtained through the algorithm of [54], though it took longer time than the algorithm of [54]. Reference [55] proposed a genetic greedy method to select a set of materialized cubes from the data cube; it included a greedy repair procedure to handle infeasible solutions generated by simple GA. Its performance was compared to that of the greedy algorithm of [13] and was observed to be better than the latter. Niched Pareto Genetic Algorithm (*NPGA*) and Multi-Objective Genetic Algorithm (*MOGA*) have been applied to the maintenance cost view selection problem in [53] and the solutions are compared to those produced by *HRUA*. Their performance was studied using real and synthetic data sets. It was observed that the performances of *NPGA* and *MOGA* were better than that of *HRUA* for all problems, but their performance gap decreased for larger instances of the problem. The performances of *NPGA* and *MOGA* were almost similar for most of the problems, but *MOGA* performed better than *NPGA* for skewed problems. Reference [56] proposed *M-OLAP* Genetic and *M-OLAP* co-Genetic algorithm to select distributed *OLAP* cubes for materialization. They were shown to perform better than the greedy algorithm. Reference [57] applied GA to the view selection problem with dual constraints, i.e., space and maintenance cost constraints. The benefit of the solutions obtained by *GA* was at least 64 percent of the benefit obtained of the optimal solution. The results of *GA* were compared with those of exhaustive search and it was observed that the results of *GA* were better than the latter. Reference [58] used the evolutionary algorithm to address the weighted materialized view selection problem, which included both the volume and importance of the retrieved data. Simulation results showed that it performed substantially better than the brute force method and the heuristic approach in terms of the quality of the solutions and execution time. In [64], a *GA*-based view selection algorithm *GVSA* was proposed, which was used to select *Top-K* views from a multidimensional lattice. *GVSA* was shown to perform better than *HRUA*. Further, in [65], memetic-based view selection algorithm *MVSA* was proposed, which incorporated iterative improvement into the evolutionary nature of the algorithm. *MVSA* produced better quality *Top-K* views when compared with those produced using *GVSA* [66]. Later, in [66], algorithm *DEVSA* that selects *Top-K* views from a multidimensional lattice using differential evolution was proposed. *DEVSA*, in comparison with *GVSA* and *MVSA*, was able to select comparatively better quality views.

The effectiveness of evolutionary algorithms depends upon their ability to achieve an appropriate balance between exploration and exploitation of the search space, while arriving at solutions to a given problem. Though the traditional evolutionary algorithms are characterized by individual representation, fitness function, selection, crossover and mutation, they are not able to achieve better population diversity and thus are unable to explore and exploit the search space adequately [67]. A quantum-inspired evolutionary algorithm (*QIEA*) has been used as an alternative to achieve such population diversity using the principles of quantum computing such as quantum bit and superposition of states [67, 68]. In *QIEA*, an individual is probabilistically represented using a string of *Q*-bits, where each *Q*-bit is the smallest unit of information. Since the *Q*-bit individual can represent binary solutions probabilistically, it can ensure better population diversity, which in turn would enable better exploration and exploitation of the search space [67]. Further, the *Q*-gate operation in *QIEA* maintains an appropriate balance between the exploration and the exploitation of the search space and thereby drives the individuals towards better solutions [68, 69]. *QIEA* has been shown to perform better than the traditional evolutionary algorithms [67, 70–74]. In this paper, an attempt has been made to use *QIEA* to address the view selection problem in the context of a multidimensional lattice framework. Accordingly, a quantum-inspired-evolution-based view selection algorithm (*QIEVSA*) that selects the *Top-K* views from a multidimensional lattice has been proposed. Further, *QIEVSA* is compared to the existing view selection algorithms *GVSA*, *MVSA* and *DEVSA*. Such comparisons show that *QIEVSA* performs comparatively better than *GVSA*, *MVSA* and *DEVSA*.

1.3 *Organization of the paper*

This paper is organized as follows: view selection using *QIEA* is discussed in section 2 followed by an illustrative example in section 3. Experimental results are given in section 4. Section 5 presents the conclusion.

2. View selection using *QIEA*

Since the proposed view selection algorithm focuses on *HRUA*, it considers a multidimensional lattice for view selection. In a multidimensional lattice framework, the root node represents the fact table while the intermediate nodes represent all other possible combinations of the dimension tables (views). Multidimensional lattice is discussed next.

2.1 *Multidimensional lattice*

An *OLAP* cube, shown in figure 1, can be represented as a multidimensional lattice shown in figure 3. This lattice of figure 3 comprises three dimensions, viz., Customer (*C*), Product (*P*) and Time (*T*), and therefore the total number of possible views is 8 (2^3). The possible views are represented by nodes of the lattice with the view index in parenthesis and the view size alongside. The dependences between the

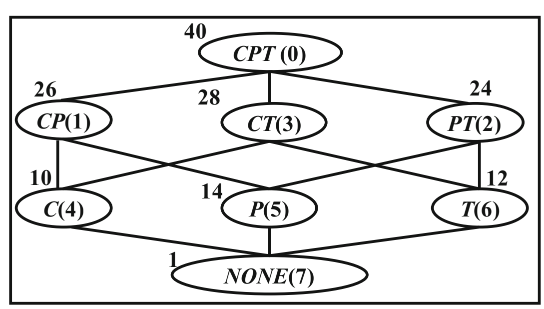

Figure 3. Three-dimensional lattice of views along with the size of each view.

views are represented using edges and each view is either directly or indirectly dependent on the root view, which represents the base fact table *CPT*. View *NONE* has no dependent view. View *CP* is directly dependent on view *CPT*, whereas view *C* is indirectly dependent on view *CPT*.

Next, the *QIEA* that has been discretized and adapted for view selection is discussed.

2.2 *QIEA*

Quantum computation is based on the physical principles of quantum mechanics, like *Q*-bits, superposition, *Q*-gates and quantum measurement, for solving problems in the context of classical computing paradigms [68]. In quantum computer systems, a single bit of information, i.e., a quantum bit (Q-bit), may be in the "0" state or in the "1" state or in any superposition of the "0" state and "1" state [67]. Its state can be given as [67]

$$|\Psi> \; = \alpha|0> \; + \beta|1>$$

where |0> and |1> represent the values of classical bits 0 and 1, respectively, and α and β are complex numbers satisfying:

$$|\alpha|^2 + |\beta|^2 = 1$$

$|\alpha|^2$ and $|\beta|^2$ are the probability of the *Q*-bit being in "0" state and "1" state, respectively.

The state of the Q-bit can be changed using a quantum gate (Q-gate) that is a reversible gate represented by a unary operator "U"; when it acts on the Q-bit it satisfies the condition $U^{+}U=UU^{+}$ [67]. Several Q-gates exist like *NOT* gate, controlled *NOT* gate, rotation gate, etc. [67, 75]. For a system with *m* Q-bits, there are 2^m states. Observing a quantum state collapses it to a single state [67].

Quantum computers were formalized in the late 1980s and active research was carried out in the 1990s, during which research related to merging of quantum computing and evolutionary computing started. Accordingly, several algorithms exist that are broadly classified [68] as evolution-designed quantum algorithms [76–80], quantum evolutionary algorithms [81–86] and *QIEA*s [67, 87–93]. This paper focuses on the use of the *QIEA* to address the view selection problem.

QIEA uses probabilities associated with each state to describe the behaviour of the system. *QIEA* uses Q-bits, Q-gates and the observation process. Q-bits represent individuals, Q-gates operate on Q-bits to generate offsprings and the observation process enables the quantum particle to take state |0> or state |1>. The superposition state represented by Q-bits would collapse to a single state during the observation process. *QIEA* uses Q-bit representation, which is probabilistically a linear superposition of states, to describe individuals of the population. It uses *Q*-gate to generate individuals with better solutions for the next generation. It can also exploit the search space for a global solution having, even, a single element. Several *QIEA* algorithms exist and are broadly classified into three types, namely binary observation *QIEA* (*bQIEA*) [67, 89, 90], real observation *QIEA* (*rQIEA*) [54, 93] and *QIEA*-like algorithms (i*QIEA*) [87, 88, 92]. In this paper, *bQIEA* [67] has been used to address the view selection problem.

An individual in *bQIEA* is represented as a string of Q-bits as follows:

$$\begin{bmatrix} \alpha_1 & | & \alpha_2 & | & \alpha_3 & | & \ldots & | & \alpha_m \\ \beta_1 & | & \beta_2 & | & \beta_3 & | & \ldots & | & \beta_m \end{bmatrix}$$

where $|\alpha_i|^2 + |\beta_i|^2 = 1$, $i = 1, 2, \ldots, m$, where *m* is the number of Q-bits representing an individual.

The algorithm *bQIEA*, as proposed in [67, 68], is given in figure 4.

In Step-1, the generation number *g* is initialized to 1; α and β values in quantum state $Q(g)$ are initialized in a manner such that $|\alpha|^2+|\beta|^2=1$. As a result, each Q-bit individual is a linear superposition of all possible states. Next, in Step-2, a binary solution $P(g)$ is obtained by observing the quantum state $Q(g)$. The resultant binary string of length *m* is formed by selecting either 0 or 1 for each bit based on the probability $|\alpha|^2$ or $|\beta|^2$. In Step-3, each binary solution in $P(g)$ is evaluated based on the problem-specific fitness function. The initial best solutions in $P(g)$ are selected and stored in $B(g)$ and the best solution of $B(g)$ in *b* in Step-4. In Step-5, *g* is incremented by 1. In Step-6, binary solutions in $P(g)$ are obtained by observing $Q(g–1)$, as in Step-2. The solutions in $P(g)$ are evaluated using the problem-specific fitness function in Step-7, as in Step-3. Thereafter, in Step 8, Q-bit individuals in $Q(g)$ are updated by applying rotation Q-gate as follows [67]:

$$G^{g-1}(\theta) = \begin{bmatrix} \cos\theta^{g-1} & -\sin\theta^{g-1} \\ \sin\theta^{g-1} & \cos\theta^{g-1} \end{bmatrix}$$

where θ^g is an adjustable Q-gate rotation angle.

The Q-bit $[\alpha^g, \beta^g]$ is updated as follows [67]:

$$\begin{bmatrix} \alpha^g \\ \beta^g \end{bmatrix} = G^{g-1}(\theta) \begin{bmatrix} \alpha^{g-1} \\ \beta^{g-1} \end{bmatrix}$$

where $\theta^{g-1} = s\left(\alpha^{g-1}, \beta^{g-1}\right) \Delta\theta^{g-1}$; $s\left(\alpha^{g-1}, \beta^{g-1}\right)$ and $\Delta\theta^{g-1}$ are the sign and the magnitude of the rotation gate, respectively. The afore-mentioned particular values are taken from the look-up table [68] given in figure 5; *x* and *y* are bits of $p(g)$ and *b*, respectively.

In Step-9, the best solutions, from among $B(g–1)$ and $P(g)$, are selected and stored in $B(g)$ and *b* is updated with the best solution from amongst solutions in $B(g)$ and *b*. In Step-10, the migration condition is checked. If it is found to

Method
Begin
Step-1: $g \leftarrow 0$
Initialize $Q(g)$
Step-2: Make P(g) by observing the states of $Q(g)$
Step-3: Evaluate $P(g)$
Step-4: Store the best solutions among $P(g)$ into $B(g)$ and the best solution among $B(g)$ into b
Step-5: $g \leftarrow g+1$
Step-6: Make $P(g)$ by observing the states of $Q(g–1)$
Step-7: Evaluate $P(g)$
Step-8: Update $Q(g)$ using Q-gates
Step-9: Store the best solutions among $P(g)$ and $B(g–1)$ into $B(g)$ and store the best solution amongst b and solutions in $B(g)$ into b
Step-10: IF (migration condition)
Migrate b or b_j' to $B(g)$ globally or locally, respectively
END IF
Step-11: IF (Terminating condition NOT satisfied)
GO TO Step-5
End

Figure 4. Algorithm *bQIEA* [67, 68].

x	y	$f(x) \le f(y)$	$\Delta\theta^{g-1}$	$s(\alpha^{g-1},\beta^{g-1})$	
				$\alpha^{g-1}\beta^{g-1} \ge 0$	$\alpha^{g-1}\beta^{g-1} < 0$
0	0	false	0	±1	±1
0	0	true	0	±1	±1
0	1	false	0.01π	1	−1
0	1	true	0	±1	±1
1	0	false	0.01π	−1	1
1	0	true	0	±1	±1
1	1	false	0	±1	±1
1	1	true	0	±1	±1

Figure 5. Look-up table [68].

be satisfied, the best solution b is migrated to $B(g)$ or some of the best solutions in $B(g)$ are migrated to other solutions in $B(g)$. Next, the termination condition is checked in Step-11. If the termination condition is not satisfied, then Step-5 to Step-11 are repeated until the termination condition is satisfied.

The afore-mentioned algorithm *bQIEA* has been adapted to solve the view selection problem. Accordingly, a *QIEVSA* has been proposed that selects the *Top-K* views from a multidimensional lattice using *bQIEA*. *QIEVSA* is discussed next.

2.3 *QIEVSA*

QIEVSA [94] selects *Top-K* views from a multidimensional lattice using *bQIEA*. *QIEVSA* takes the d-dimensional lattice of views, number of views to be selected K and the pre-specified number of generations G for which *QIEVSA* runs as input, and produces the *Top-K* views *TKV* as the output. *QIEVSA* comprises the following steps.

Step-1: Initialization

Initialize the generation $g = 0$. Generate a random population $Q_{TKV}(g)$ of N *Top-K* views with quantum chromosome length Kd, where each quantum gene is represented by d quantum bits (Q-bits) and d is the dimension of the lattice.

$$Q_{TKV}(g) = \{qtkv_1^g, qtkv_2^g, \ldots, qtkv_N^g\}$$

where

$$qtkv_i^g = \left(\begin{array}{cccc|cccc|cc|cccc} \alpha v_{i11}^g & \alpha v_{i12}^g & \cdots & \alpha v_{i1d}^g & \alpha v_{i21}^g & \alpha v_{i22}^g & \cdots & \alpha v_{i2d}^g & \cdots & \cdots & \alpha v_{iK1}^g & \alpha v_{iK2}^g & \cdots & \alpha v_{iKd}^g \\ \beta v_{i11}^g & \beta v_{i12}^g & \cdots & \beta v_{i1d}^g & \beta v_{i21}^g & \beta v_{i22}^g & \cdots & \beta v_{i2d}^g & \cdots & \cdots & \beta v_{iK1}^g & \beta v_{iK2}^g & \cdots & \beta v_{iKd}^g \end{array}\right)$$

$$\left|\alpha v^g_{ijl}\right|^2+\left|\beta v^g_{ijl}\right|^2=1$$
$$\forall i = 1,2,\ldots,N; j=1,2,\ldots,K;\ l = 1,2,\ldots,d.$$

Initially, at $g = 0$, all possible states are superimposed with the same probability, i.e.

$$\alpha v^0_{ijl} = \frac{1}{\sqrt{2}} \text{ and } \beta v^0_{ijl} = \frac{1}{\sqrt{2}}$$
$$\forall i = 1,2,\ldots,N;\ j = 1,\ 2,\ \ldots,K;\ l = 1,2,\ldots,d.$$

In this step of initialization of $Q_{TKV}(g)$, each of the Q-bits (αv^g_{ijl} and βv^g_{ijl}) is initialized to ($1/\sqrt{2}$, $1/\sqrt{2}$). Each Q-bit *Top-K* views $qtkv^g_i$ represents the linear superposition of all the possible solutions with the same probability [67].

Step-2: Make $P_{TKV}(g)$ by observing $Q_{TKV}(g)$

Observe each Q-bit of $Q_{TKV}(g)$ to arrive at $P_{TKV}(g)$ where

$$P_{TKV}(g) = \left\{ptkv^g_1, ptkv^g_2,\ \ldots, ptkv^g_N\right\}$$

$$ptkv^g_i = \left(\underbrace{pv^g_{i11}\,pv^g_{i12}\ \cdots\ pv^g_{i1d}}_{pv^g_{i1}}\ \middle|\ \underbrace{pv^g_{i21}\,pv^g_{i22}\ \cdots\ pv^g_{i2d}}_{pv^g_{i2}}\ \middle|\ \ \ldots\ \ \middle|\ \underbrace{pv^g_{iK1}\,pv^g_{iK2}\ \cdots\ pv^g_{iKd}}_{pv^g_{iK}}\right)$$
$$\forall i = 1,\ 2,\ldots,N;\ j = 1,2,\ldots,K;\ l = 1,2,\ldots,d.$$

$$pv^g_{ijl} = \begin{cases} 0, & \text{random}[0,\ 1) < \left|\alpha v_{ijl}\right|^2 \\ 1, & \text{otherwise} \end{cases}.$$

Each binary solution $ptkv^g_i$ is a binary string of length K, which is formed by selecting either 0 or 1 for each bit with the probability $\left|\alpha v_{ijl}\right|^2$.

Step-3: Evaluate $P_{TKV}(g)$

Each binary solution $P_{TKV}(g)$ is evaluated using the fitness function *TVEC*. Each *Top-K* view $ptkv^g_i$ *in* $P_{TKV}(g)$ is transformed into equivalent decimal representation $dtkv_i$ to arrive at $DP_{TKV}(g)$, where

$$DP_{TKV}(g) = \left\{dptkv^g_1, dptkv^g_2,\ldots, dptkv^g_N\right\} \text{ and}$$
$$dptkv^g_i = \left\{dpv^g_{i1}, dpv^g_{i2},\ldots, dpv^g_{iK}\right\},\ i = 1,2,\ldots N.$$

Compute *TVEC* of each view $dptkv^g_i$ in $DP_{TKV}(g)$ as follows [64–66, 95–97]:

$$TVEC(\ dptkv^g_i) = \sum_{i=1\wedge SM_{V_i}=1}^{n} Size(V_i) + \sum_{i=1\wedge SM_{V_i}=0}^{n} SizeSMA(V_i)$$

where n is the total number of views in the lattice, SM_{Vi} is status materialized of view V_i (SM_{Vi} = 1, if materialized, SM_{Vi} = 0, if not materialized), $Size(V_i)$ is the size of view V_i and $SizeSMA(V_i)$ is the size of smallest materialized ancestor of view V_i

Step-4: Create $B_{TKV}(g)$/$D_{TKV}(g)$ and B_{TKV}/D_{TKV}

In this step, the initial best *Top-K* views are selected, from among the binary *Top-K* views in $P_{TKV}(g)$, and stored into B_{TKV}. Also, the best *Top-K* views among the equivalent decimal representation $DP_{TKV}(g)$ of $P_{TKV}(g)$ are selected and stored into D_{TKV}.

Initially, at $g = 0$, store all N *Top-K* views in $P_{TKV}(g)$ and in corresponding $DP_{TKV}(g)$ into $B_{TKV}(g)$ and $D_{TKV}(g)$, respectively, where

$$B_{TKV}(g) = \left\{btkv^g_1, btkv^g_2,\ldots, btkv^g_N\right\}$$

$$D_{TKV}(g) = \left\{dtkv^g_1, dtkv^g_2,\ldots, dtkv^g_N\right\}$$

$$\begin{aligned} &\mathit{for}\ i = 1,2,\ldots,N \\ &btkv^g_i = \left\{bv^g_{i1}, bv^g_{i2},\ldots, bv^g_{iK}\right\} \\ &dtkv^g_i = \left\{dv^g_{i1}, dv^g_{i2},\ldots, dv^g_{iK}\right\} \\ &\quad \mathit{and}\ \forall i = 1,2,\ldots,N \\ &btkv^g_i = ptkv^g_i, dtkv^g_i = dptkv^g_i. \end{aligned}$$

The best *Top-K* view in $B_{TKV}(g)$ and in the corresponding $D_{TKV}(g)$ are stored into B_{TKV} and D_{TKV}, respectively. Increment g by 1.

Step-5: Obtain $P_{TKV}(g)$ by observing $Q_{TKV}(g–1)$

Binary solutions in $P_{TKV}(g)$ are formed by observing the states of $Q_{TKV}(g\text{-}1)$, as carried out in Step-2.

Step-6: Evaluate $P_{TKV}(g)$

Each binary solution $P_{TKV}(g)$ is evaluated using fitness function *TVEC*, as evaluated in Step-3.

Step-7: Update $Q_{TKV}(g–1)$ using Q-gates

Q-bit individuals in $Q_{TKV}(g)$ are updated by applying rotation Q-gate operator of *QIEA* [67] with the updated Q-bit satisfying the normalization condition

x	b	TVEC(ptkv) ≤ TVEC(btkv)	$\Delta\theta_{ijl}^{g-1}$	$s(\alpha v_{ijl}^{g-1}, \beta v_{ijl}^{g-1})$	
				$\alpha v_{ijl}^{g-1} \beta v_{ijl}^{g-1} \geq 0$	$\alpha v_{ijl}^{g-1} \beta v_{ijl}^{g-1} < 0$
0	0	false	0	±1	±1
0	0	true	0	±1	±1
0	1	false	0.01π	1	−1
0	1	true	0	±1	±1
1	0	false	0.01π	−1	1
1	0	true	0	±1	±1
1	1	false	0	±1	±1
1	1	true	0	±1	±1

Figure 6. Look-up table [68].

$\left|\alpha v_{ijl}^{g}\right|^2 + \left|\beta v_{ijl}^{g}\right|^2 = 1$, where αv_{ijl}^{g} and βv_{ijl}^{g} are the values of the updated Q-bits.

$Q_{TKV}(g)$ is updated using quantum rotation gate as Q-gate. The l^{th} Q-bit of the j^{th} view of the i^{th} *Top-K* view $qtkv_i^g$, $i = 1, 2, \ldots, N$; $j = 1, 2, \ldots, K$; $l = 1, 2, \ldots, d$ is updated by applying the current Q-gate $G_{ijl}^{g}(\theta)$:

$$G_{ijl}^{g-1}(\theta) = \begin{bmatrix} \cos\theta_{ijl}^{g-1} & -\sin\theta_{ijl}^{g-1} \\ \sin\theta_{ijl}^{g-1} & \cos\theta_{ijl}^{g-1} \end{bmatrix}$$

where θ_{ijl}^{g} is an adjustable Q-gate rotation angle.

The Q-bit $[\alpha v_{ijl}^{g}, \beta v_{ijl}^{g}]$ is updated as follows:

$$\begin{bmatrix} \alpha v_{ijl}^{g} \\ \beta v_{ijl}^{g} \end{bmatrix} = G_{ijl}^{g-1}(\theta) \begin{bmatrix} \alpha v_{ijl}^{g-1} \\ \beta v_{ijl}^{g-1} \end{bmatrix}$$

where $\theta_{ijl}^{g-1} = s\left(\alpha v_{ijl}^{g-1}, \beta v_{ijl}^{g-1}\right) \Delta\theta_{ijl}^{g-1}$ and $s(\alpha v_{ijl}^{g-1}, \beta v_{ijl}^{g-1})$ and $\Delta\theta_{ijl}^{g-1}$ are the sign and the magnitude of the rotation gate, respectively. The values of $s(\alpha v_{ijl}^{g-1}, \beta v_{ijl}^{g-1})$ and $\Delta\theta_{ijl}^{g-1}$ are taken from the look-up table [93] given in figure 6; x and b are bits of $ptkv_i^g$ and B_{TKV}, respectively.

Step-8: Update $B_{TKV}(g)$/$D_{TKV}(g)$ and B_{TKV}/D_{TKV}

In this step, $B_{TKV}(g)$ and $D_{TKV}(g)$ are generated and B_{TKV} and D_{TKV} are updated.

The best N *Top-K* views among $P_{TKV}(g)$ and $B_{TKV}(g$–1) are stored into $B_{TKV}(g)$.

The best N *Top-K* views among $DP_{TKV}(g)$ and $D_{TKV}(g$–1) are stored into $D_{TKV}(g)$.

The best *Top-K* views in $B_{TKV}(g)$ and in the corresponding $D_{TKV}(g)$ are stored into B_{TKV} and D_{TKV}, respectively.

Increment g by 1.

Step-9: Migration condition

In this step, the global migration condition is checked. If it is satisfied, the best solution B_{TKV} is migrated to $B_{TKV}(g)$ globally. Otherwise, the best *Top-K* views in a local group in $B_{TKV}(g)$ are migrated to others in the same local group. Variation of probabilities of a Q-bit individual is induced during the migration process. The global migration and the local migration are carried out after a pre-specified number of generations G_{GM} and G_{LM}, respectively.

IF *global migration* THEN

migrate B_{TKV} to $B_{TKV}(g)$ globally

replace all the *Top-K* views in $B_{TKV}(g)$ by B_{TKV}

ELSE

migrate B_{TKV} to $B_{TKV}(g)$ locally

replacing the *Top-K* views in each of the pre-specified number of local groups in $B(g)$ by best *Top-K* views amongst them in the respective groups.

Step-10: Termination condition

QIEVSA runs for a pre-specified number of generations G, whereafter it produces the *Top-K* views D_{TKV} as output.

IF ($g \leq G$) THEN

GO TO Step-5

ELSE

Return D_{TKV} as the *Top-K* views.

Next, an example illustrating the use of *QIEVSA* to select *Top-K* views from a multidimensional lattice is discussed.

3. An example

Consider the selection of *Top*-5 views from a three-dimensional lattice shown in figure 3 using *QIEVSA*.

Step-1: Initialization

Generation, g=0; quantum population size, N=10; number of Q-bits in quantum gene, d=3. Since *Top*-5 views need to be selected, the quantum chromosome length K=5. The quantum population of *Top-K* views at $g = 0$, i.e., $Q_{T5V}(0) = \{qt5v_1^0, qt5v_2^0, \ldots, qt5v_{10}^0\}$ is presented in figure 7. Initially, αv_{ikl}^{0} and βv_{ikl}^{0} are taken as $1/\sqrt{2}$ (0.7071).

Step-2: Make $P_{T5V}(0)$ by observing $Q_{T5V}(0)$

Quantum observed state $P_{T5V}(0) = \{pt5v_1^0, pt5v_2^0, \ldots, pt5v_{10}^0\}$ in binary form, i.e., Q-bit representation of initial population $Q_{T5V}(0)$ is observed. The observation of the first *Top*-5 views $qt5v_1^0$ in $Q_{T5V}(0)$ to generate $pt5v_1^0$ is shown in figure 8.

In a similar manner, quantum observed state for other *Top-K* views in $Q_{T5V}(0)$ are computed to generate $P_{T5V}(0)$, which is given in figure 9.

Step-3: Evaluate $P_{T5V}(0)$

Each *Top*-5 views $pt5v_i^0$ (i = 0, 1, …, 10) in $P_{T5V}(0)$ is transformed into equivalent decimal representation, i.e., $dt5v_i^0$ to generate $DP_{T5V}(0) = \{dpt5v_1^0, dpt5v_1^0, \ldots, dpt5v_{10}^0\}$,

$Q_{T5V}(0)$	$\alpha v_{i11}{}^0$	$\alpha v_{i12}{}^0$	$\alpha v_{i13}{}^0$	$\alpha v_{i21}{}^0$	$\alpha v_{i22}{}^0$	$\alpha v_{i23}{}^0$	$\alpha v_{i31}{}^0$	$\alpha v_{i32}{}^0$	$\alpha v_{i33}{}^0$	$\alpha v_{i41}{}^0$	$\alpha v_{i42}{}^0$	$\alpha v_{i43}{}^0$	$\alpha v_{i51}{}^0$	$\alpha v_{i52}{}^0$	$\alpha v_{i53}{}^0$
	$\beta v_{i11}{}^0$	$\beta v_{i12}{}^0$	$\beta v_{i13}{}^0$	$\beta v_{i21}{}^0$	$\beta v_{i22}{}^0$	$\beta v_{i23}{}^0$	$\beta v_{i31}{}^0$	$\beta v_{i32}{}^0$	$\beta v_{i33}{}^0$	$\beta v_{i41}{}^0$	$\beta v_{i42}{}^0$	$\beta v_{i43}{}^0$	$\beta v_{i51}{}^0$	$\beta v_{i52}{}^0$	$\beta v_{i53}{}^0$
$qt5v_1{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_2{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_3{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_4{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_5{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_6{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_7{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_8{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_9{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_{10}{}^0$	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071

Figure 7. $Q_{T5V}(0)$.

Random var.	${\alpha v^0_{1jl}}^2$	pv^0_{1jl}	$pt5v^0_i$
0.40175587	0.49999997	0	011
0.51350707	0.49999997	1	
0.6879447	0.49999997	1	
0.4345283	0.49999997	0	001
0.34838796	0.49999997	0	
0.7124095	0.49999997	1	
0.7756701	0.49999997	1	101
0.30042487	0.49999997	0	
0.9100001	0.49999997	1	
0.78266287	0.49999997	1	110
0.9099875	0.49999997	1	
0.000876069	0.49999997	0	
0.58374248	0.49999997	1	111
0.56351848	0.49999997	1	
0.69518647	0.49999997	1	

Figure 8. $pt5v^0_1$ of $qt5v^0_1$ in $Q_{T5V}(0)$.

where $dpt5v^0_i = \{dpv^0_{i1}, dpv^0_1, \ldots, dpv^0_5\}$. $DP_{T5V}(0)$ is shown in figure 10. *TVEC* of each view $dpt5v^0_i$ in $DP_{T5V}(0)$ is computed. *TVEC* computation of $dpt5v^0_1$ comprising views *CT*, *CP*, *P*, *T* and *None* is shown in figure 11. In a similar manner, *TVEC* of other $dpt5v^0_i$ in $DP_{T5V}(0)$ are computed and are shown in figure 12.

Step-4: Create $B_{T5V}(0)$ and B_{T5V}

$P_{T5V}(0)$ and $DP_{T5V}(0)$ are stored into $B_{T5V}(0)$ and $D_{T5V}(0)$, respectively. $B_{T5V}(0)$ and $D_{T5V}(0)$ are shown in figures 13 and 14, respectively.

The best *Top*-5 views in $B_{T5V}(0)$ and $D_{T5V}(0)$ are stored into B_{T5V} and D_{T5V}, respectively. They are shown in figure 15. Next, g is incremented by 1, i.e., $g = 1+1 = 2$.

Step-5: Make $P_{T5V}(1)$ by observing $Q_{T5V}(0)$

Quantum observed state $P_{T5V}(1) = \{pt5v^1_1, pt5v^1_2, \ldots, pt5v^1_{10}\}$ in binary form is generated. The observation of the first *Top*-5 views $qt5v^0_1$ in $Q_{T5V}(0)$ to generate $pt5v^1_1$ is shown in figure 16.

In a similar manner, quantum observed states for other *Top-K* views in $Q_{T5V}(0)$ are computed to generate $P_{T5V}(1)$, which is given in figure 17.

Step-6: Evaluate $P_{T5V}(1)$

Each *Top*-5 views $pt5v^0_i$ ($i = 0, 1, \ldots, 10$) in $P_{T5V}(1)$ is transformed into equivalent decimal representation, i.e., $dt5v^0_i$, to generate $DP_{T5V}(1) = \{dpt5v^1_1, dpt5v^1_1, \ldots, dpt5v^1_{10}\}$, where $dpt5v^1_i = \{dpv^1_{i1}, dpv^1_1, \ldots, dpv^1_5\}$. $DP_{T5V}(1)$ is shown in figure 18. *TVEC* of each view $dpt5v^1_i$ in $DP_{TKV}(1)$ is computed and are shown in figure 19.

Step-7: Update $Q_{T5V}(0)$ using Q-gates

Update Q-bits (αv^0_{ikl} and βv^0_{ikl}) of each *Top*-5 views in $Q_{T5V}(0)$ using corresponding *Top*-5 views in $P_{T5V}(1)$ and best *Top*-5 views B_{T5V}. Updation of Q-bit (αv^0_{1kl} and βv^0_{1kl}) of the first individual $qt5v^0_1$ of $Q_{TKV}(0)$ using $pt5v^1_1$, in $P_{T5V}(1)$, and B_{T5V}, shown in figure 20, is given below:

Computation of θ^0_{111} for the first bit

Bit $x = 1$, bit $b = 1$, $TVEC(pt5v^1_1) = 187$ and $TVEC(B_{T5V}) = 181$. Since $TVEC(pt5v^1_1) \le TVEC(B_{T5V})$ is false, using the look-up table, $s(\alpha v^0_{111}, \beta v^0_{111}) = 1$ and $\Delta\theta^0_{111} = 0$.

$P_{T5V}(0)$	pv_{i1}^0	pv_{i2}^0	pv_{i3}^0	pv_{i4}^0	pv_{i5}^0
$pt5v_1{}^0$	011	001	101	110	111
$pt5v_2{}^0$	001	011	110	111	101
$pt5v_3{}^0$	001	110	100	101	000
$pt5v_4{}^0$	011	110	001	101	111
$pt5v_5{}^0$	110	111	011	101	010
$pt5v_6{}^0$	100	111	101	110	011
$pt5v_7{}^0$	101	100	001	011	111
$pt5v_8{}^0$	001	011	010	101	111
$pt5v_9{}^0$	111	010	100	101	011
$pt5v_{10}{}^0$	101	110	100	001	111

Figure 9. $P_{T5V}(0)$.

$P_{T5V}(0)$	$D_{T5V}(0)$	dpv_{i1}^0	dpv_{i2}^0	dpv_{i3}^0	dpv_{i4}^0	dpv_{i5}^0
$pt5v_1{}^0$	$dpt5v_1{}^0$	3	1	5	6	7
$pt5v_2{}^0$	$dpt5v_2{}^0$	1	3	6	7	5
$pt5v_3{}^0$	$dpt5v_3{}^0$	1	6	4	5	7
$pt5v_4{}^0$	$dpt5v_4{}^0$	3	6	1	5	7
$pt5v_5{}^0$	$dpt5v_5{}^0$	6	7	3	5	2
$pt5v_6{}^0$	$dpt5v_6{}^0$	4	7	5	6	3
$pt5v_7{}^0$	$dpt5v_7{}^0$	5	4	1	3	7
$pt5v_8{}^0$	$dpt5v_8{}^0$	1	3	2	5	7
$pt5v_9{}^0$	$dpt5v_9{}^0$	7	2	4	5	3
$pt5v_{10}{}^0$	$dpt5v_{10}{}^0$	5	6	4	1	7

Figure 10. $DP_{T5V}(0)$.

$$\theta_{111}^0 = s(\alpha v_{111}^0, \beta v_{111}^0)\Delta\theta_{111}^0 = 1 \times 0 = 0.$$

Computation of θ_{112}^0 for the second bit

Bit $x = 1$, bit $b = 1$, $TVEC(pt5v_1^1) = 187$ and $TVEC(B_{T5V}) = 181$. Since $TVEC(pt5v_1^1) \leq TVEC(B_{T5V})$ is false, using the look-up table, $s(\alpha v_{112}^0, \beta v_{112}^0) = 1$ and $\Delta\theta_{112}^0 = 0$.

$$\theta_{112}^0 = s(\alpha v_{112}^0, \beta v_{112}^0)\Delta\theta_{112}^0 = 1 \times 0 = 0.$$

Computation of θ_{113}^0 for the third bit

Bit $x = 0$, bit $b = 1$, $TVEC(pt5v_1^1) = 187$ and $TVEC(B_{T5V}) = 181$. Since $TVEC(pt5v_1^1) \leq TVEC(B_{T5V})$ is false, using the look-up table, $s(\alpha v_{113}^0, \beta v_{113}^0) = 1$ and $\Delta\theta_{113}^0 = 0.0314$.

$$\theta_{113}^0 = s(\alpha v_{113}^0, \beta v_{113}^0)\Delta\theta_{113}^0 = 1 \times 0.0314 = 0.0314.$$

In a similar manner, θ_{1kl}^0 for the other quantum bits of the first individual $qt5v_1^0$ of $Q_{T5V}(0)$ is computed and is shown in figure 21.

$P_{T5V}(0)$	$D_{T5V}(0)$	$TVEC$
$pt5v_1{}^0$	$dpt5v_1{}^0$	187
$pt5v_2{}^0$	$dpt5v_2{}^0$	187
$pt5v_3{}^0$	$dpt5v_3{}^0$	183
$pt5v_4{}^0$	$dpt5v_4{}^0$	187
$pt5v_5{}^0$	$dpt5v_5{}^0$	187
$pt5v_6{}^0$	$dpt5v_6{}^0$	185
$pt5v_7{}^0$	$dpt5v_7{}^0$	187
$pt5v_8{}^0$	$dpt5v_8{}^0$	183
$pt5v_9{}^0$	$dpt5v_9{}^0$	181
$pt5v_{10}{}^0$	$dpt5v_{10}{}^0$	183

Figure 12. $TVEC$ of $dpt5v_i^0$ in $DP_{T5V}(0)$.

$B_{T5V}(0)$	bv_{i1}^0	bv_{i2}^0	bv_{i3}^0	bv_{i4}^0	bv_{i5}^0
$bt5v_1{}^0$	011	001	101	110	111
$bt5v_2{}^0$	001	011	110	111	101
$bt5v_3{}^0$	001	110	100	101	000
$bt5v_4{}^0$	011	110	001	101	111
$bt5v_5{}^0$	110	111	011	101	010
$bt5v_6{}^0$	100	111	101	110	011
$bt5v_7{}^0$	101	100	001	011	111
$bt5v_8{}^0$	001	011	010	101	111
$bt5v_9{}^0$	111	010	100	101	011
$bt5v_{10}{}^0$	101	110	100	001	111

Figure 13. $B_{T5V}(0)$.

$D_{T5V}(0)$	dv_{i1}^0	dv_{i2}^0	dv_{i3}^0	dv_{i4}^0	dv_{i5}^0
$dt5v_1{}^0$	3	1	5	6	7
$dt5v_2{}^0$	1	3	6	7	5
$dt5v_3{}^0$	1	6	4	5	7
$dt5v_4{}^0$	3	6	1	5	7
$dt5v_5{}^0$	6	7	3	5	2
$dt5v_6{}^0$	4	7	5	6	3
$dt5v_7{}^0$	5	4	1	3	7
$dt5v_8{}^0$	1	3	2	5	7
$dt5v_9{}^0$	7	2	4	5	3
$dt5v_{10}{}^0$	5	6	4	1	7

Figure 14. $D_{T5V}(0)$.

$$\sum_{i=1 \wedge SM_{V_i}=1}^{8} Size(dpt5v_1{}^0) = (Size(CPT) + Size(CT) + Size(CP) + Size(P) + Size(T) + Size(None))$$

$$= (40 + 28 + 26 + 14 + 12 + 1) = 121,$$

$$\sum_{i=1 \wedge SM_{V_i}=0}^{8} SizeSMA(dpt5v_1{}^0) = (SizeSMA(PT) + SizeSMA(C)) = (40 + 26) = 66,$$

$$TVEC = \sum_{i=1 \wedge SM_V=1}^{8} Size(dpt5v_1{}^0) + \sum_{i=1 \wedge SM_V=0}^{8} SizeSMA(dpt5v_1{}^0) = 121 + 66 = 187.$$

Figure 11. *TVEC* computation if views *CT*, *CP*, *P*, *T* and *None* are selected for materialization.

Top-5 views	View-1	View-2	View-3	View-4	View-5	*TVEC*
B_{T5V}	111	010	100	101	011	181
D_{T5V}	7	2	4	5	3	181

Figure 15. B_{T5V} and D_{T5V}.

Random [0, 1)	α^1_{1jl}	pv^1_{1jl}	$pt5v^1_i$
0.5011488	0.49999997	1	110
0.70176893	0.49999997	1	
0.043875396	0.49999997	0	
0.73296607	0.49999997	1	101
0.1709851	0.49999997	0	
0.93461823	0.49999997	1	
0.9480489	0.49999997	1	111
0.6981055	0.49999997	1	
0.7665917	0.49999997	1	
0.02548498	0.49999997	0	011
0.51505903	0.49999997	1	
0.64097357	0.49999997	1	
0.4138329	0.49999997	0	010
0.9327696	0.49999997	1	
0.0682677	0.49999997	0	

Figure 16. $pt5v^1_1$ of $qt5v^0_1$ in $Q_{T5V}(0)$.

Updation of αv^0_{111} and βv^0_{111} of the first Q-bit $\begin{bmatrix}\alpha v^1_{111}\\ \beta v^1_{111}\end{bmatrix} = \begin{bmatrix}\cos 0 & -\sin 0\\ \sin 0 & \cos 0\end{bmatrix} \times \begin{bmatrix}0.70710677\\ 0.70710677\end{bmatrix} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \times \begin{bmatrix}0.70710677\\ 0.70710677\end{bmatrix} = \begin{bmatrix}0.70710677\\ 0.70710677\end{bmatrix}$.

Updation of αv^0_{112} and βv^0_{112} of the second Q-bit

$$\begin{bmatrix}\alpha v^1_{112}\\ \beta v^1_{112}\end{bmatrix} = \begin{bmatrix}\cos 0 & -\sin 0\\ \sin 0 & \cos 0\end{bmatrix} \times \begin{bmatrix}0.70710677\\ 0.70710677\end{bmatrix} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \times \begin{bmatrix}0.70710677\\ 0.70710677\end{bmatrix} = \begin{bmatrix}0.70710677\\ 0.70710677\end{bmatrix}.$$

Updation of αv^0_{113} and βv^0_{113} of the third Q-bit

$$\begin{bmatrix}\alpha v^1_{113}\\ \beta v^1_{113}\end{bmatrix} = \begin{bmatrix}\cos 0.0314 & -\sin 0.0314\\ \sin 0.0314 & \cos 0.0314\end{bmatrix} \times \begin{bmatrix}0.70710677\\ 0.70710677\end{bmatrix} = \begin{bmatrix}0.9995 & -0.0314\\ 0.0314 & 0.9995\end{bmatrix} \times \begin{bmatrix}0.70710677\\ 0.70710677\end{bmatrix} = \begin{bmatrix}0.6846\\ 0.7290\end{bmatrix}.$$

In a similar manner, Q-bits of each $qt5v^0_i$ of $Q_{T5V}(0)$ are updated to generate $Q_{T5V}(1)$, which is shown in figure 22.

Step-8: Update $B_{T5V}(1)/D_{T5V}(1)$ and B_{T5V}/D_{T5V}

The 10 *Top*-5 views, from amongst $P_{T5V}(1)$ and $B_{T5V}(0)$, are stored into $B_{T5V}(1)$. The 10 *Top*-5 views, from amongst

$P_{TKV}(1)$	pv^1_{i1}	pv^1_{i2}	pv^1_{i3}	pv^1_{i4}	pv^1_{i5}
$pt5v_1{}^1$	110	101	111	011	010
$pt5v_2{}^1$	001	011	110	101	111
$pt5v_3{}^1$	111	011	110	001	101
$pt5v_4{}^1$	101	110	010	111	001
$pt5v_5{}^1$	001	111	100	110	101
$pt5v_6{}^1$	110	010	011	111	101
$pt5v_7{}^1$	001	011	110	010	100
$pt5v_8{}^1$	010	101	001	111	110
$pt5v_9{}^1$	001	100	111	101	110
$pt5v_{10}{}^1$	111	001	011	101	010

Figure 17. $P_{T5V}(1)$.

$P_{T5V}(1)$	$D_{T5V}(1)$	dpv^1_{i1}	dpv^1_{i2}	dpv^1_{i3}	dpv^1_{i4}	dpv^1_{i5}
$pt5v_1{}^1$	$dpt5v_1{}^1$	6	5	7	3	2
$pt5v_2{}^1$	$dpt5v_2{}^1$	1	3	6	5	7
$pt5v_3{}^1$	$dpt5v_3{}^1$	7	3	6	1	5
$pt5v_4{}^1$	$dpt5v_4{}^1$	5	6	2	7	1
$pt5v_5{}^1$	$dpt5v_5{}^1$	1	7	4	6	5
$pt5v_6{}^1$	$dpt5v_6{}^1$	6	2	3	7	5
$pt5v_7{}^1$	$dpt5v_7{}^1$	1	3	6	2	4
$pt5v_8{}^1$	$dpt5v_8{}^1$	2	5	1	7	6
$pt5v_9{}^1$	$dpt5v_9{}^1$	1	4	7	5	6
$pt5v_{10}{}^1$	$dpt5v_{10}{}^1$	7	1	3	5	2

Figure 18. $DP_{T5V}(1)$.

$P_{T5V}(1)$	$D_{T5V}(1)$	*TVEC*
$pt5v_1{}^1$	$dpt5v_1{}^1$	187
$pt5v_2{}^1$	$dpt5v_2{}^1$	187
$pt5v_3{}^1$	$dpt5v_3{}^1$	187
$pt5v_4{}^1$	$dpt5v_4{}^1$	183
$pt5v_5{}^1$	$dpt5v_5{}^1$	183
$pt5v_6{}^1$	$dpt5v_6{}^1$	187
$pt5v_7{}^1$	$dpt5v_7{}^1$	174
$pt5v_8{}^1$	$dpt5v_8{}^1$	183
$pt5v_9{}^1$	$dpt5v_9{}^1$	183
$pt5v_{10}{}^1$	$dpt5v_{10}{}^1$	183

Figure 19. *TVEC* of $dpt5v^1_i$ in $DP_{T5V}(1)$.

$pt5v_1{}^1$	110	101	111	000	010
B_{T5V}	111	010	100	101	011

Figure 20. $pt5v^1_1$ and B_{T5V}.

$DP_{T5V}(1)$ and $D_{T5V}(0)$, are stored into $D_{T5V}(1)$. $B_{T5V}(1)$ and $D_{T5V}(1)$ are shown in figures 23 and 24, respectively. The best *Top-K* views in $B_{T5V}(1)$, and in the corresponding $D_{T5V}(1)$, are stored into B_{T5V} and D_{T5V}, respectively. The updated B_{T5V} and D_{T5V} are shown in figure 25. This is followed by incrementing g by 1.

Step-9: Perform local/global migration

Considering that local migration is performed in each step, the resultant $D_{T5V}(1)$, after performing local migration for local group size 2, is given in figure 26.

Bit	x	b	$TVEC(pt5v_1^1)$	$TVEC(B_{T5V})$	$TVEC(pt5v_1^1) \leq TVEC(B_{T5V})$	$\Delta\theta^0_{1jl}$	Sign	θ^0_{1jl}
1	1	1	187	181	False	0	1	0
2	1	1	187	181	False	0	1	0
3	0	1	187	181	False	0.0314	1	0.0314
4	1	0	187	181	False	0.0314	–1	–0.0314
5	0	1	187	181	False	0.0314	1	0.0314
6	1	0	187	181	False	0.0314	–1	–0.0314
7	1	1	187	181	False	0	1	0
8	1	0	187	181	False	0.0314	–1	–0.0314
9	1	0	187	181	False	0.0314	–1	–0.0314
10	0	1	187	181	False	0.0314	1	0.0314
11	0	0	187	181	False	0	1	0
12	0	1	187	181	False	0.0314	1	0.0314
13	0	0	187	181	False	0	1	0
14	1	1	187	181	False	0	1	0
15	0	1	187	181	False	0.0314	1	0.0314

Figure 21. θ^0_{1kl} of quantum bits of $qt5v_1^0$ in $Q_{T5V}(0)$.

$Q_{T5V}(1)$	αv_{i11}^1	αv_{i12}^1	αv_{i13}^1	αv_{i21}^1	αv_{i22}^1	αv_{i23}^1	αv_{i31}^1	αv_{i32}^1	αv_{i33}^1	αv_{i41}^1	αv_{i42}^1	αv_{i43}^1	αv_{i51}^1	αv_{i52}^1	αv_{i53}^1
	βv_{i11}^1	βv_{i12}^1	βv_{i13}^1	βv_{i21}^1	βv_{i22}^1	βv_{i23}^1	βv_{i31}^1	βv_{i32}^1	βv_{i33}^1	βv_{i41}^1	βv_{i42}^1	βv_{i43}^1	βv_{i51}^1	βv_{i52}^1	βv_{i53}^1
$qt5v_1^1$	0.7071	0.7071	0.6846	0.7290	0.6846	0.7290	0.7071	0.7290	0.7290	0.6846	0.7071	0.6846	0.7071	0.7071	0.6846
	0.7071	0.7071	0.7282	0.6839	0.7282	0.6839	0.7071	0.6839	0.6839	0.7282	0.7071	0.7282	0.7071	0.7071	0.7282
$qt5v_2^1$	0.6846	0.6846	0.7282	0.6839	0.7282	0.7057	0.7071	0.7057	0.6839	0.7282	0.7071	0.7282	0.7057	0.7071	0.7057
	0.7282	0.7282	0.7071	0.7071	0.7071	0.6846	0.7071	0.6846	0.7071	0.7071	0.7071	0.7071	0.7289	0.7071	0.7289
$qt5v_3^1$	0.7282	0.7282	0.7071	0.7071	0.7071	0.7065	0.7071	0.7065	0.7071	0.7274	0.6600	0.7500	0.7290	0.7057	0.7289
	0.7071	0.7071	0.7071	0.7071	0.7071	0.6846	0.7071	0.6846	0.7071	0.7296	0.7071	0.7071	0.6839	0.7289	0.7071
$qt5v_4^1$	0.7071	0.6846	0.7071	0.7290	0.7071	0.6846	0.6846	0.7064	0.7071	0.7513	0.7290	0.7071	0.7288	0.7290	0.7071
	0.7071	0.7282	0.7071	0.6839	0.7071	0.7071	0.7282	0.6846	0.7071	0.7071	0.6839	0.7071	0.7296	0.6839	0.7071
$qt5v_5^1$	0.6846	0.7282	0.6846	0.6839	0.6846	0.7071	0.7282	0.6846	0.7071	0.7296	0.7057	0.6846	0.7065	0.7478	0.7487
	0.7282	0.7071	0.7282	0.7071	0.7282	0.7071	0.7071	0.7071	0.7071	0.7071	0.6846	0.7282	0.6846	0.7302	0.7071
$qt5v_6^1$	0.7282	0.7071	0.7057	0.7071	0.7282	0.7071	0.6846	0.7290	0.7290	0.7290	0.6833	0.7487	0.7064	0.6833	0.7261
	0.7071	0.7071	0.7289	0.7071	0.7071	0.7071	0.7282	0.6839	0.6839	0.7071	0.6853	0.7071	0.7289	0.6853	0.7296
$qt5v_7^1$	0.7071	0.7071	0.7289	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7289	0.6853	0.7296	0.6846	0.7705	0.7487
	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071	0.7071
$qt5v_8^1$	0.6846	0.7071	0.6846	0.7290	0.6846	0.7290	0.6846	0.7071	0.7290	0.7071	0.7290	0.7071	0.7290	0.7071	0.6846
	0.7282	0.7071	0.7282	0.6839	0.7282	0.6839	0.7282	0.7071	0.6839	0.7071	0.6839	0.7071	0.6839	0.7071	0.7282
$qt5v_9^1$	0.7057	0.6846	0.7282	0.7057	0.7057	0.6839	0.7487	0.7057	0.7057	0.7071	0.6839	0.7071	0.6600	0.6846	0.7261
	0.7289	0.7282	0.7071	0.6846	0.7289	0.7071	0.7071	0.6846	0.6846	0.7071	0.7071	0.7071	0.7071	0.7282	0.7296
$qt5v_{10}^1$	0.7064	0.7057	0.6846	0.6846	0.7064	0.7290	0.6846	0.7065	0.7065	0.7071	0.7071	0.6846	0.6846	0.7290	0.6846
	0.7289	0.7289	0.7282	0.7071	0.7289	0.6839	0.7282	0.6846	0.6846	0.7071	0.7071	0.7282	0.7282	0.6839	0.7282

Figure 22. $Q_{T5V}(1)$.

Step-5 to Step-9 are repeated for a pre-specified number of generations G, whereafter the *Top*-5 views are produced as output.

Next, experiment-based comparison of *QIEVSA* to the evolutionary view selection algorithms *DEVSA*, *MVSA*, *GVSA* and *HRUA* is discussed.

4. Experimental results

Algorithms *QIEVSA*, *DEVSA*, *GVSA*, *MVSA* and *HRUA* were implemented using *JDK* 1.7 in a Windows-7 environment on an Intel-based 2.13 GHz PC having 4 GB RAM. First, the appropriate value of number of generations

$B_{T5V}(1)$	bv_{i1}^1	bv_{i2}^1	bv_{i3}^1	bv_{i4}^1	bv_{i5}^1
$bt5v_1^1$	001	011	110	010	100
$bt5v_2^1$	111	010	100	101	011
$bt5v_3^1$	101	110	010	111	001
$bt5v_4^1$	001	111	100	110	101
$bt5v_5^1$	010	101	001	111	110
$bt5v_6^1$	001	100	111	101	110
$bt5v_7^1$	111	001	011	101	010
$bt5v_8^1$	001	110	100	101	000
$bt5v_9^1$	001	011	010	101	111
$bt5v_{10}^1$	101	110	100	001	111

Figure 23. $B_{T5V}(1)$.

$D_{TKV}(1)$	dv_{i1}^1	dv_{i2}^1	dv_{i3}^1	dv_{i4}^1	dv_{i5}^1	*TVEC*
$dt5v_1^1$	1	3	6	2	4	174
$dt5v_2^1$	5	6	2	7	1	183
$dt5v_3^1$	7	2	4	5	3	181
$dt5v_4^1$	1	7	4	6	5	183
$dt5v_5^1$	2	5	1	7	6	183
$dt5v_6^1$	1	4	7	5	6	183
$dt5v_7^1$	7	1	3	5	2	183
$dt5v_8^1$	1	6	4	5	7	183
$dt5v_9^1$	1	3	2	5	7	183
$dt5v_{10}^1$	5	6	4	1	7	183

Figure 24. $D_{T5V}(1)$.

Top-5 views	View-1	View-2	View-3	View-4	View-5	*TVEC*
B_{T5V}	001	011	110	010	100	174
D_{T5V}	1	3	6	2	4	174

Figure 25. B_{T5V} and D_{T5V}.

$D_{TKV}(1)$	dv_{i1}^1	dv_{i2}^1	dv_{i3}^1	dv_{i4}^1	dv_{i5}^1
$dt5v_1^1$	1	3	6	2	4
$dt5v_1^1$	1	3	6	2	4
$dt5v_3^1$	7	2	4	5	3
$dt5v_3^1$	7	2	4	5	3
$dt5v_5^1$	2	5	1	7	6
$dt5v_5^1$	2	5	1	7	6
$dt5v_7^1$	7	1	3	5	2
$dt5v_7^1$	7	1	3	5	2
$dt5v_{10}^1$	5	6	4	1	7
$dt5v_{10}^1$	5	6	4	1	7

Figure 26. $D_{T5V}(1)$ after local migration.

after which local migration G_{LM} and global migration G_{GM} are to be performed, for which *QIEVSA* is able to select views having lower *TVEC*, is determined. The mean value of *TVEC* of *Top*-10 views over four simulation runs after each generation up to 500 generations for various combinations of G_{LM} and G_{GM} was used for plotting graphs [98]. These graphs showing the *TVEC* of the *Top*-10 views for (G_{LM}, G_{GM}) = {(1, 50), (1, 100), (2, 50), (2, 100), (4, 50),

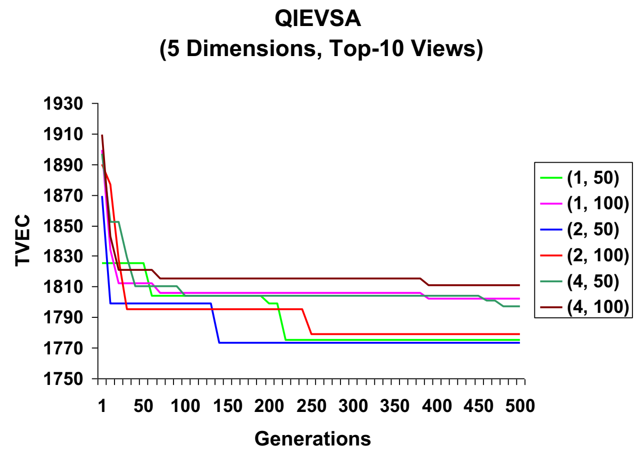

Figure 27. *QIEVSA* - *TVEC* Vs. Generations - Top-10 Views - 5 Dimensions.

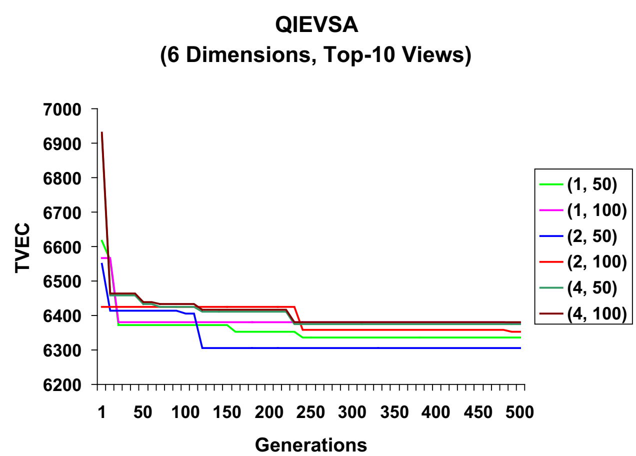

Figure 28. *QIEVSA-TVEC* vs. generations – Top-10 views – 6 dimensions.

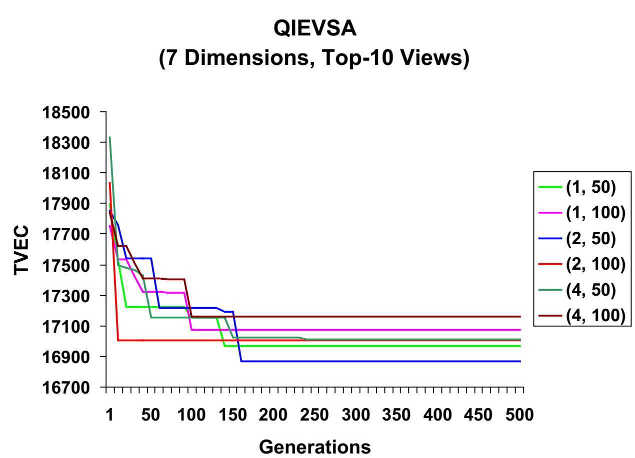

Figure 29. *QIEVSA-TVEC* vs. generations – Top-10 views – 7 dimensions.

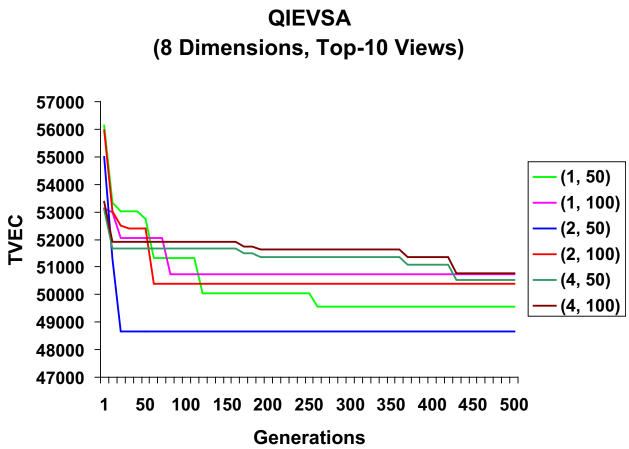

Figure 30. *QIEVSA-TVEC* vs. generations – Top-10 views – 8 dimensions.

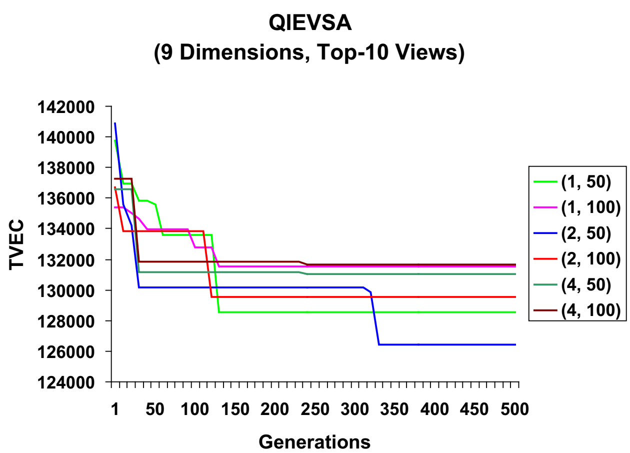

Figure 31. *QIEVSA-TVEC* vs. generations – Top-10 views – 9 dimensions.

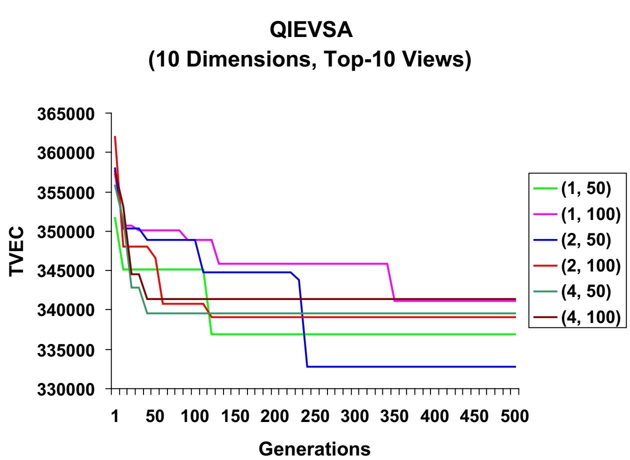

Figure 32. *QIEVSA-TVEC* vs. generations – Top-10 views – 10 dimensions.

Dimension = 5				
(G_{LM}, G_{GM})	*minTVEC*	*maxTVEC*	*meanTVEC*	*stdTVEC*
(1, 50)	1755	1802	1777.25	20.9687
(1, 100)	1780	1824	1801.50	17.0953
(2, 50)	1752	1794	1773.25	15.8646
(2, 100)	1759	1801	1779.25	18.8994
(4, 50)	1768	1823	1796.75	24.0767
(4, 100)	1784	1835	1810.50	20.7665
Dimension = 6				
(G_{LM}, G_{GM})	*minTVEC*	*maxTVEC*	*meanTVEC*	*stdTVEC*
(1, 50)	6303	6369	6336.25	30.3757
(1, 100)	6345	6408	6376.50	27.3724
(2, 50)	6281	6334	6306.25	21.2294
(2, 100)	6312	6390	6352.25	33.7667
(4, 50)	6341	6404	6374.50	26.6505
(4, 100)	6349	6424	6381.25	30.1859
Dimension = 7				
(G_{LM}, G_{GM})	*minTVEC*	*maxTVEC*	*meanTVEC*	*stdTVEC*
(1, 50)	16875	17087	16965.25	83.6313
(1, 100)	16979	17165	17071.50	85.8298
(2, 50)	16813	16956	16868.25	57.1856
(2, 100)	16924	17089	17004.25	75.1844
(4, 50)	16937	17081	17013.75	60.4126
(4, 100)	17059	17227	17157.50	69.5899
Dimension = 8				
(G_{LM}, G_{GM})	*minTVEC*	*maxTVEC*	*meanTVEC*	*stdTVEC*
(1, 50)	49415	49665	49551.50	106.3332
(1, 100)	50601	50899	50746.75	127.5233
(2, 50)	48561	48763	48669.75	85.0922
(2, 100)	50285	50517	50398.75	108.5504
(4, 50)	50455	50644	50531.25	106.2835
(4, 100)	50651	50915	50785.25	112.3975
Dimension = 9				
(G_{LM}, G_{GM})	*minTVEC*	*maxTVEC*	*meanTVEC*	*stdTVEC*
(1, 50)	128311	128787	128559.50	201.8831
(1, 100)	131230	131787	131539.50	228.3533
(2, 50)	126241	126613	126417.25	171.2547
(2, 100)	129311	129749	129512.75	211.1378
(4, 50)	130811	131273	131024.25	204.6306
(4, 100)	131379	131941	131682.25	225.5076
Dimension = 10				
(G_{LM}, G_{GM})	*minTVEC*	*maxTVEC*	*meanTVEC*	*stdTVEC*
(1, 50)	336255	337557	336860.75	574.3424
(1, 100)	340445	341787	341094.50	593.7935
(2, 50)	332181	333141	332745.75	422.6851
(2, 100)	338715	339654	339088.75	524.4666
(4, 50)	339141	340165	339621.25	522.8061
(4, 100)	340611	341981	341328.25	556.3926

Figure 33. *QIEVSA-minTVEC*, *maxTVEC*, *meanTVEC*, *stdTVEC* – *Top*-10 views.

(4, 100)} for dimensions 5, 6, 7, 8, 9 and 10 are shown in figures 27, 28, 29, 30, 31 and 32, respectively. From each of these graphs, it can be inferred that the *Top*-10 views selected by *QIEVSA* have a lower *TVEC* for G_{LM} = 2 and G_{GM} = 50. Further, *minTVEC* (minimum value of *TVEC*), *maxTVEC* (maximum value of *TVEC*), *meanTVEC* (mean value of *TVEC*) and the *stdTVEC* (standard deviation of

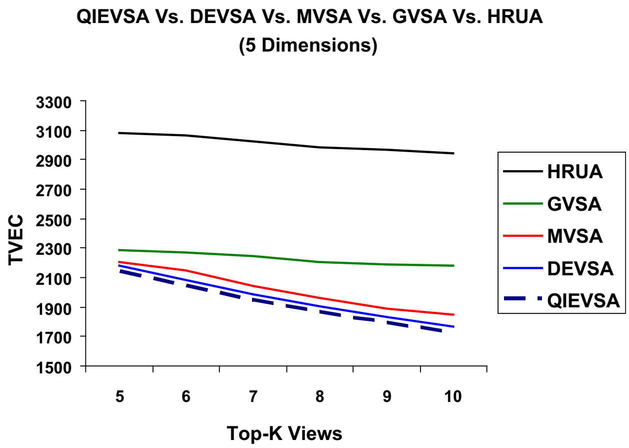

Figure 34. *QIEVSA* vs. *DEVSA* vs. *MVSA* vs. *GVSA* vs. *HRUA*-*TVEC* vs. *Top-K* views – 5 dimensions.

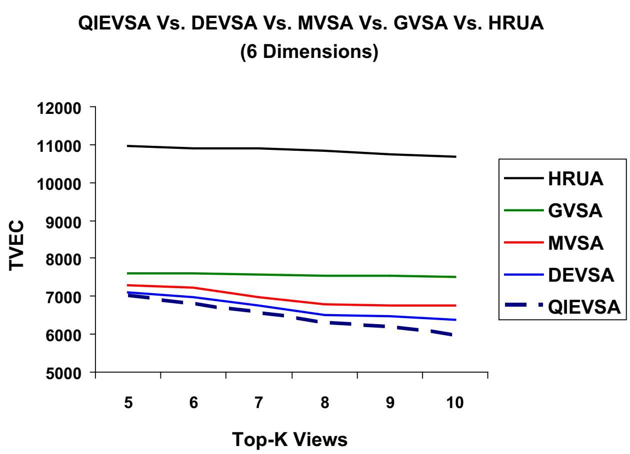

Figure 35. *QIEVSA* vs. *DEVSA* vs. *MVSA* vs. *GVSA* vs. *HRUA*-*TVEC* vs. *Top-K* views – 6 dimensions.

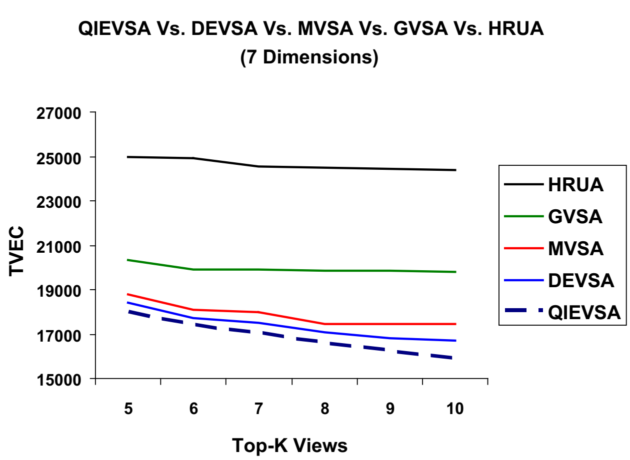

Figure 36. *QIEVSA* vs. *DEVSA* vs. *MVSA* vs. *GVSA* vs. *HRUA*-*TVEC* vs. *Top-K* views – 7 dimensions.

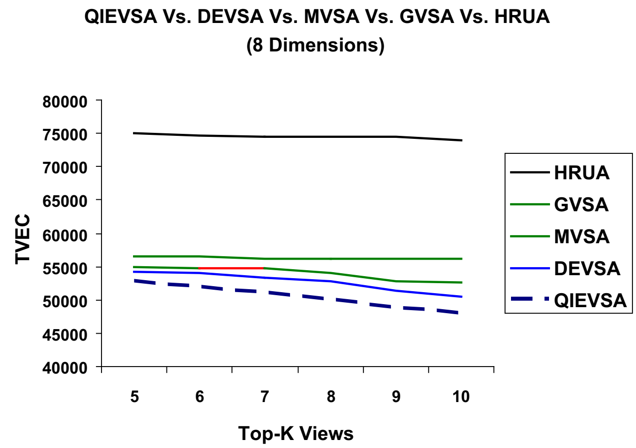

Figure 37. *QIEVSA* vs. *DEVSA* vs. *MVSA* vs. *GVSA* vs. *HRUA*-*TVEC* vs. *Top-K* views – 8 dimensions.

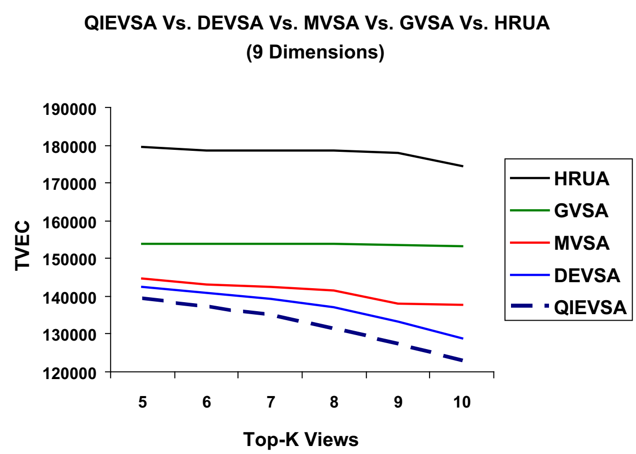

Figure 38. *QIEVSA* vs. *DEVSA* vs. *MVSA* vs. *GVSA* vs. *HRUA*-*TVEC* vs. *Top-K* views – 9 dimensions.

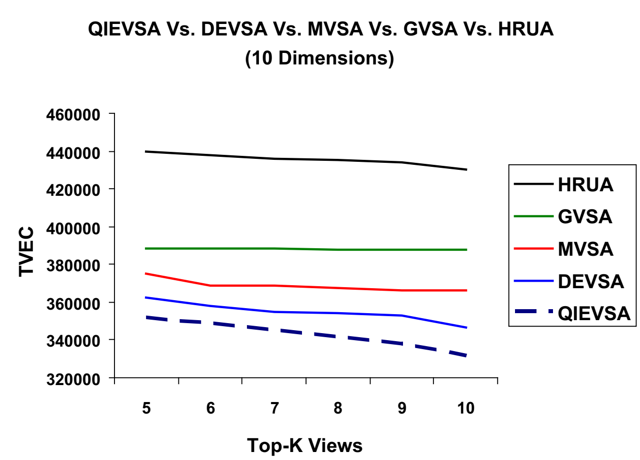

Figure 39. *QIEVSA* vs. *DEVSA* vs. *MVSA* vs. *GVSA* vs. *HRUA*-*TVEC* vs. *Top-K* views – 10 dimensions.

Dimension = 5					
Views	*HRUA*	*GVSA*	*MVSA*	*DEVSA*	*QIEVSA*
Top-5	3082	2287	2203	2182	2140
Top-6	3064	2270	2147	2081	2045
Top-7	3024	2249	2044	1985	1948
Top-8	2984	2208	1960	1905	1864
Top-9	2964	2190	1890	1831	1795
Top-10	2944	2182	1850	1766	1726
Dimension = 6					
Views	*HRUA*	*GVSA*	*MVSA*	*DEVSA*	*QIEVSA*
Top-5	10972	7603	7293	7098	6998
Top-6	10908	7597	7244	6991	6791
Top-7	10905	7574	6985	6768	6568
Top-8	10832	7540	6775	6519	6278
Top-9	10737	7543	6765	6485	6206
Top-10	10688	7526	6751	6394	5984
Dimension = 7					
Views	*HRUA*	*GVSA*	*MVSA*	*DEVSA*	*QIEVSA*
Top-5	24964	20344	18787	18406	17996
Top-6	24904	19907	18104	17694	17389
Top-7	24552	19898	17978	17508	17046
Top-8	24476	19879	17475	17105	16587
Top-9	24462	19868	17464	16835	16229
Top-10	24384	19786	17453	16710	15898
Dimension = 8					
Views	*HRUA*	*GVSA*	*MVSA*	*DEVSA*	*QIEVSA*
Top-5	74993	56620	55016	54206	52798
Top-6	74742	56468	54814	53981	51871
Top-7	74525	56240	54682	53268	50969
Top-8	74491	56228	53980	52851	49882
Top-9	74489	56168	52837	51405	48887
Top-10	73984	56159	52589	50573	47926
Dimension = 9					
Views	*HRUA*	*GVSA*	*MVSA*	*DEVSA*	*QIEVSA*
Top-5	179577	153975	144570	142507	139432
Top-6	178662	153765	143259	140767	137143
Top-7	178652	153764	142481	139413	134856
Top-8	178478	153738	141554	137030	131455
Top-9	177950	153682	138212	133219	127219
Top-10	174592	153369	137612	128985	122985
Dimension = 10					
Views	*HRUA*	*GVSA*	*MVSA*	*DEVSA*	*QIEVSA*
Top-5	439504	388733	375200	362483	351456
Top-6	438066	388359	368866	358228	348696
Top-7	436078	388250	368736	354751	345574
Top-8	435098	387970	367671	354120	341691
Top-9	433758	387798	366483	353032	337457
Top-10	430336	387737	365948	346765	331289

Figure 40. *QIEVSA* vs. *DEVSA* vs. *MVSA* vs. *GVSA* vs. *HRUA* (*TVEC* of *Top-K* views for 5–10 dimensions).

TVEC) over four simulation runs for selecting *Top*-10 views after 500 generations are computed for dimensions 5–10.They are given in a table in figure 33. From the table also, it can be observed that *QIEVSA* performs best for $G_{LM} = 2$ and $G_{GM} = 50$. These values of G_{LM} and G_{GM} for *QIEVSA* were used for further comparisons to *DEVSA*, *GVSA*, *MVSA* and *HRUA*.

Next, *TVEC*s of the *Top-K* (K = 5, 6, 7, 8, 9, 10) views selected using *HRUA*, *GVSA*, *MVSA*, *DEVSA* and *QIEVSA* for dimensions 5, 6, 7, 8, 9 and 10 are plotted and are shown in figures 34, 35, 36, 37, 38 and 39, respectively. The comparisons were based on the values *GVSA* (crossover probability $P_c = 0.6$, mutation probability $P_m = 0.05$) observed in [64], *MVSA* (crossover probability $P_c = 0.8$, mutation probability $P_m = 0.05$) observed in [65], *DEVSA* (crossover rate $CR = 0.6$, scaling factor $F = 0.1$) observed in [66] and *QIEVSA* ($G_{LM} = 2$, $G_{GM} = 50$) observed from figure 33. The mean of the *TVEC* values, for each of the five algorithms over four simulation runs, was taken for plotting graphs. It can be inferred from each of these graphs that the *Top-K* views selected using *QIEVSA*, in comparison with those selected using *DEVSA*, *MVSA*, *GVSA* and *HRUA*, have a lower *TVEC*. Further, it can be observed from figure 40 that the difference in the TVEC value increases with increase in the dimensions and the value of K. Furthermore, the performance of *DEVSA* is the next best followed by *MVSA* and *GVSA*. Views selected using *HRUA*, in comparison with others, have a higher *TVEC*.

5. Conclusions

In this paper, a *QIEA* has been suitably adapted and discretized to address the view selection problem in a multidimensional lattice framework. Accordingly, view selection algorithm *QIEVSA* that selects the *Top-K* views from a multidimensional lattice has been proposed. The Q-bits, Q-gates and the observation process in *QIEA* have been suitably adapted and discretized in *QIEVSA* to generate a population of *Top-K* views for the subsequent generation. *QIEVSA*, at first, randomly selects a population of Q-bit *Top-K* views. Binary sets of *Top-K* views are generated by observing the quantum state of the *Top-K* views in the population. *TVEC* of these views is then computed, whereafter the best set of *Top-K* views are updated. Thereafter, the Q-bit *Top-K* views are updated by applying the rotation Q-gate operator. *QIEVSA* terminates after running for a pre-specified number of generations, whereupon the *Top-K* views having minimum *TVEC* are produced as output. Further, experimental comparison of *QIEVSA*, with other evolutionary view selection algorithms based on multidimensional lattice framework like *DEVSA*, *MVSA*, *GVSA* and *HRUA*, shows that *QIEVSA* is able to select *Top-K* views at a comparatively lesser *TVEC* for the observed values of G_{LM} and G_{GM}. This performance improves for higher dimensions. Further, *QIEVSA* is able select views for higher dimensional data sets, as, in *QIEVSA*, population of the *Top-K* views is randomly generated and their *TVEC* is computed from the lattice. This is unlike the case in *HRUA*, where *HRUA* needs to compute the *Top-K* views from an exponentially large search space of possible views and for higher dimensional data sets, it becomes almost infeasible to select views for materialization using *HRUA*. Thus, it can be reasonably inferred that *QIEVSA* is able to select reasonably good quality views, for higher dimensional data sets, that are capable of reducing the response time of analytical queries, which thereby would lead to efficient decision making. As future work, *QIEVSA* would be compared to existing swarm-based view selection algorithms [99–107].

References

[1] Inmon W H 2003 *Building the data warehouse*, 3rd ed. India: Wiley Dreamtech India Pvt. Ltd

[2] Rainardi V 2008 *Building a data warehouse: with examples in SQL server*. New York: Apress

[3] Kimball R and Ross M 2002 *The data warehouse toolkit: the complete guide to dimensional modelling*. Hoboken: Wiley

[4] Widom J 1995 Research problems in data warehousing. In: *Proceedings on the 4th international conference on information and knowledge management*, Baltimore, Maryland, pp. 25–30

[5] Ponniah P 2004 *Data warehousing fundamentals: a comprehensive guide for IT professionals*. Hoboken: Wiley

[6] Cabibbo L and Torlone R 1998 A logical approach to multidimensional databases. In: *Advances in database technology-EDBT'98*, *Lecture notes in computer science*, vol. 1377, pp. 183–197

[7] Choong Y W, Laurent A and Laurent D 2007 Pixelizing data cubes: a block-based approach. In: *Pixelization paradigm*, *Lecture notes in computer science*, vol. 4370, pp. 63–76

[8] Codd E F, Codd S B and Salley C T 1993 *Providing OLAP (on line analytical processing) to user-analysts: an IT mandate*. Dorset: E. F. Codd and Associates

[9] Thomsen E 2002 *OLAP solutions: building multidimensional information systems*. Hoboken: Wiley

[10] Golfarellia M, Maniezzo V and Rizzi S 2004 Materialization of fragmented views in multidimensional databases. *Data Knowl. Eng.* 49: 325–351

[11] Gray J, Bosworth A, Lyaman A and Pirahesh H 1996 Data cube: a relational aggregation operator generalizing GROUP-BY, CROSS-TAB, and SUB-TOTALS. In: *Proceedings of the 12th IEEE international conference on data engineering*, pp. 152–159

[12] Gupta H 1997 Selection of views to materialize in a data warehouse. In: *Proceedings of the 6th ICDT*, pp. 98–112

[13] Harinarayan V, Rajaraman A and Ullman J D 1996 Implementing data cubes efficiently. In: *Proceedings of ACM SIGMOD*, Montreal, Canada, pp. 205–216

[14] Niemi T, Nummenmaa J and Thanisch P 2001 Constructing OLAP cubes based on queries. In: *Proceedings of the 4th ACM international workshop on data warehousing and OLAP*, pp. 9–15

[15] Roussopoulos N 1997 Materialized views and data warehouse. In: *Proceedings of the 4th KRDB workshop*, Athens, Greece, August

[16] Chirkova R and Yang J 2011 Materialized views. *Found. Trends Databases* 4(4): 295–405

[17] Mami I and Bellahsene Z 2012 A survey of view selection methods. *ACM SIGMOD Record* 41(1): 20–29

[18] Yang J, Karlapalem K and Li Q 1997 Algorithms for materialized view design in data warehousing environment. *Very Large databases (VLDB) J* 136–145

[19] Yousri N A R, Ahmed K M and El-Makky N M 2005 Algorithms for selecting materialized views in a data warehouse. In: *Proceedings of the ACS/IEEE 2005 international conference on computer systems and applications*, AICCSA'05, IEEE Computer Society, pp. 27–1

[20] Chirkova R, Halevy A Y and Suciu D 2001 A formal perspective on the view selection problem. In: *Proceedings of the 27 VLDB conference*, Italy, pp. 59–68

[21] Chaves L W F, Buchmann E, Hueske F and Bohm K 2009 Towards materialized view selection for distributed databases. In: *Proceedings of the 12th international conference on extending database technology: advances in database technology*, pp. 1088–1099

[22] Goasdoue F, Karanasos K, Leblay J and Manolescu I 2011 View selection in semantic web databases. In: *Proceedings of the VLDB endowment*, vol. 5(2), pp. 97–108

[23] Katsifodimos A, Manolescu I and Vassalos V 2012 Materialized view selection for XQuery workloads. In: *Proceedings of the international conference on management of data (SIGMOD'12)*, pp. 565–576

[24] Teschke M and Ulbrich A 1997 *Using materialized views to speed up data warehousing*. Technical Report, IMMD 6, Universität Erlangen-Nümberg

[25] Agrawal S, Chaudhuri S and Narasayya V R 2000 Automated selection of materialized views and indexes in SQL databases. In: *Proceedings of the 26th international conference on very large databases*, Cairo, Egypt, pp. 496–505

[26] Aouiche K and Darmont J 2009 Data mining-based materialized view and index selection in data warehouse. *J. Intell. Inf. Syst.* 33(1): 65–93

[27] Aouiche K, Jouve P E and Darmont J 2006 Clustering-based materialized view selection in data warehouses. In: *Proceedings of the 10th East-European conference on advances in databases and information systems (ADBIS06)*, Thessaloniki, Greece, LNCS, vol. 4152, pp. 81–95

[28] Baralis E, Paraboschi S and Teniente E 1997 Materialized view selection in a multidimensional database. In: *Proceedings of the 23rd VLDB conference*, Greece, pp. 156–165

[29] Lehner W, Ruf T and Teschke M 1996 Improving query response time in scientific databases using data aggregation. In: *Proceedings of the 7th international conference and workshop on database and expert systems applications*, DEXA 96, Zurich, pp. 201–206

[30] Theodoratos D and Sellis T 1997 Data warehouse configuration. In: *Proceedings of VLDB*, Athens, Greece, pp. 126–135

[31] Vijay Kumar T V and Devi K 2013 An architectural framework for constructing materialized views in a data warehouse. *Int. J. Innov. Manag. Technol.* 4(2): 192–197

[32] Vijay Kumar T V and Devi K 2012 Materialized view construction in data warehouse for decision making. *Int J Bus Inf Syst* 11(4): 379–396

[33] Vijay Kumar T V, Goel A and Jain N 2010 Mining information for constructing materialized views. *Int. J. Inf. Commun. Technol.* 2(4): 386–405

[34] Gupta H and Mumick I S 2005 Selection of views to materialize in a data warehouse. *IEEE Trans. Knowl. Data Eng.* 17(1): 24–43

[35] Gupta H, Harinarayan V, Rajaraman V and Ullman J 1997 Index selection for OLAP. In: *Proceedings of the 13th international conference on data engineering*, ICDE 97, Birmingham, UK, pp. 208–219

[36] Haider M and Vijay Kumar T V 2011 Materialised view selection using size and query frequency. *Int. J. Value Chain Manag.* 5(2): 95–105

[37] Haider M and Vijay Kumar T V 2017 Query frequency based view selection. *Int. J. Bus. Anal.* 4(1): 36–55

[38] Serna-Encinas M T and Hoyo-Montano J A 2007 Algorithm for selection of materialized views: based on a costs model. In: *Proceedings of the 8th international conference on current trends in computer science*, pp. 18–24

[39] Shah B Ramachandran K and Raghavan V 2006 A hybrid approach for data warehouse view selection. *Int. J. Data Wareh. Min.* 2(2): 1–37

[40] Shukla A, Deshpande P M and Naughton J F 1998 Materialized view selection for multidimensional datasets. In: *Proceedings of VLDB*, pp. 488–500

[41] Valluri S, Vadapalli S and Karlapalem K 2002 View relevance driven materialized view selection in data warehousing environment. *Aust. Comput. Sci. Commun.* 24(2): 187–196

[42] Vijay Kumar T V and Ghoshal A 2009 A reduced lattice greedy algorithm for selecting materialized views. *Commun. Comput. Inf. Sci.* 31: 6–18

[43] Vijay Kumar T V, Haider M and Kumar S 2011 A view recommendation greedy algorithm for materialized views selection. *Commun. Comput. Inf. Sci*. 141: 61–70

[44] Vijay Kumar T V and Haider M 2010 A query answering greedy algorithm for selecting materialized views. In: *Lecture notes in artificial intelligence* (LNAI). Berlin: Springer, vol. 6422, pp. 153–162

[45] Vijay Kumar T V and Haider M 2011 Greedy views selection using size and query frequency. *Commun. Comput. Inf. Sci.* 125: 11–17

[46] Vijay Kumar T V and Haider M 2012 Materialized views selection for answering queries. In: *Lecture notes in computer science* (LNCS). Berlin: Springer, vol. 6411, pp. 44–51

[47] Vijay Kumar T V and Haider M 2015 Query answering based view selection. *Int. J. Bus. Inf. Syst.* 18(3): 338–353

[48] Vijay Kumar T V and Haider M 2011 Selection of views for materialization using size and query frequency. *Commun. Comput. Inf. Sci.* 147: 150–155

[49] Vijay Kumar T V, Haider M and Kumar S 2010 Proposing candidate views for materialization. *Commun. Comput. Inf. Sci.* 54: 89–98

[50] Vijay Kumar T V 2013 Answering query-based selection of materialised views. *Int. J. Inf. Decision Sci.* 5(1): 103–116

[51] De Jong K A 2006 *Evolutionary computation: a unified approach*. Cambridge: MIT Press

[52] Horng J T, Chang Y J, Liu B J and Kao C Y 1999 Materialized view selection using genetic algorithms in a data warehouse system. In: *Proceedings of the world congress on evolutionary computation*, IEEE CEC, Washington, DC, USA, vol. 3, pp. 2221–2227

[53] Lawrence M 2006 Multiobjective genetic algorithms for materialized view selection in OLAP data warehouses. In: *Proceedings of the 8th annual conference on genetic and evolutionary computation*, GECCO'06, July 8–12, Seattle, Washington, USA, pp. 699–706

[54] Lee M and Hammer J 2001 Speeding up materialized view selection in data warehouse using a random algorithm. *Int. J. Coop. Inf. Syst.* 10: 327–353

[55] Lin W and Kuo I C 2004 A genetic algorithm for OLAP data cubes. *Int. J. Knowl. Inf. Syst.* 6(1): 83–102

[56] Loureiro J and Belo O 2006 An evolutionary approach to the selection and allocation of distributed cubes. In: *Proceedings of the 10th international database engineering and applications symposium*, IDEAS'06, IEEE, pp. 243–248

[57] Talebian S H and Abdul Kareem S 2009 Using genetic algorithm to select materialized views subject to dual constraints. In: *Proceedings of the international conference on signal processing systems*, IEEE, pp. 633–638

[58] Wagner N and Agrawal V 2013 Using an evolutionary algorithm to solve the weighted view materialisation problem for data warehouses. *Int. J. Intell. Inf. Database Syst.* 7(2): 163–179

[59] Wang Z and Zhang D 2005 Optimal genetic view selection algorithm under space constraint. *Int. J. Inf. Technol.* 11(5): 44–51

[60] Yu J X, Yao Y X, Choi C and Gou G 2003 Materialized view selection as constrained evolutionary optimization. *IEEE Trans. Syst. Man Cybern.* 33(4): 458–467

[61] Zhang C, Yao X and Yang J 2001 An evolutionary approach to materialized views selection in a data warehouse environment. *IEEE Trans. Syst. Man Cybern.* 31(3): 282–294

[62] Zhang C, Yao X and Yang J 1999 Evolving materialized views in a data warehouse. In: *Proceedings of the IEEE congress on evolutionary computations*, vol. 2, pp. 823–829

[63] Zhou L, He X and Li K 2012 An improved approach for materialized view selection based on genetic algorithm. *J. Comput.* 7(7): 1591–1598

[64] Vijay Kumar T V and Kumar S 2012 Materialized view selection using genetic algorithm. *Commun. Comput. Inf. Sci.* 306: 225–237

[65] Vijay Kumar T V and Kumar S 2013 Materialized view selection using memetic algorithm. In: *Lecture notes in artificial intelligence* (LNAI). Berlin: Springer, vol. 8284, pp. 316–327

[66] Vijay Kumar T V and Kumar S 2014 Materialized view selection using differential evolution. *Int. J. Innov. Comput. Appl.* 6(2):102–113

[67] Han K H and Kim J W 2002 Quantum-inspired evolutionary algorithm for a class of combinatorial optimization. *IEEE Trans. Evolut. Comput.* 6(6): 580–593

[68] Zhang G 2011 Quantum-inspired evolutionary algorithms: a survey and empirical study. *J. Heuristics* 17: 303–351

[69] Manju A and Nigam M J 2014 Applications of quantum inspired computational intelligence: a survey. *Artif. Intell. Rev.* 42: 79–156

[70] Han K H, Park K H, Lee C H and Kim J H 2001 Parallel quantum inspired genetic algorithm for combinatorial optimization problem. In: *Proceedings of the 2001 congress on evolutionary computation*, Seoul, Korea, vol. 2, pp. 1422–1429

[71] Li B B and Wang L 2006 A hybrid quantum-inspired genetic algorithm for multi-objective scheduling. In: *Proceedings of the 2006 international conference on intelligent computing*, Kunming, China

[72] Narayan A and Moore M 1995 Quantum-inspired genetic algorithms. In: *Proceedings of the 1996 IEEE international conference on evolutionary computation (ICEC96)*, pp. 61–66

[73] Zhou S and Sun Z 2005 A new approach belonging to EDAs: quantum-inspired genetic algorithm with only one chromosome. In: *Lecture notes in computer science* (LNCS). Berlin: Springer, vol. 3612, pp. 141–150

[74] Zhou R and Cao J 2014 Quantum novel genetic algorithm based on parallel subpopulation computing and its application. *Artif Intell Rev* 41(3): 359–371

[75] Hey T 1999 Quantum computing: an introduction. *Comput. Control Eng. J.* 10(3): 105–112

[76] Grigorenko I and Garcia M 2002 Calculation of the partition function using quantum genetic algorithms. *Physica A Stat. Mech. App.* 313(3–4): 463–470

[77] Grigorenko I and Garcia M 2001 Ground-state wave functions of two-particle systems determined using quantum genetic algorithms. *Physica A Stat. Mech. Appl.* 291(1–4): 439–448

[78] Koza J R, Al-Sakran S H and Jones L W 2005 Cross-domain features of runs of genetic programming used to evolve designs for analog circuits, optical lens systems, controllers, antennas, mechanical systems, and quantum computing circuits. In: *Proceedings of NASA/DoD EH*, pp. 205–212

[79] Sahin M and Tomak M 2005 The self-consistent calculation of a spherical quantum dot: a quantum genetic algorithm study. *Physica E Low Dimens. Syst. Nanostruct.* 28(3): 247–256

[80] Spector L, Barnum H, Bernstein H and Swamy J N 1999 Finding a better-than-classical quantum AND/OR algorithm using genetic programming. In: *Proceedings of the CEC*, pp. 2239–2246

[81] Malossini A, Blanzieri E and Calarco T 2004 *QGA: a quantum genetic algorithm.* Technical Report No. DIT-04-105, Informatica e Telecommunicazioni, University of Trento

[82] Rylander B, Soule T, Foster J and Alves-Foss J 2000 Quantum genetic algorithms. In: *Proceedings of the GECCO*, 373–377

[83] Sahin M, Atav U and Tomak M 2005 Quantum genetic algorithm method in self-consistent electronic structure calculations of a quantum dot with many electrons. *Int. J. Mod. Phys.* C16(9): 1379–1393

[84] Sofge D A 2006 Toward a framework for quantum evolutionary computation. In: *Proceedings of the CIS conference*, pp. 789–794

[85] Spector L, Barnum H and Bernstein H 1998 Genetic programming for quantum computers. In: *Proceedings of the 3rd annual conference on genetic programming*, pp. 365–373

[86] Udrescu M, Prodan L and Vladutiu M 2006 Implementing quantum genetic algorithms: a solution based on Grover's algorithm. In: *Proceedings of CF*, pp. 14–16

[87] Abs da Cruz A, Hall Barbosa C, Pacheco M and Vellasco M 2004 Quantum-inspired evolutionary algorithms and its application to numerical optimization problems. In: *Proceedings of ICONIP 2004*, LNCS 3316. Berlin: Springer, pp. 212–217

[88] Abs da Cruz A, Vellasco M and Pacheco M 2006 Quantum-inspired evolutionary algorithm for numerical optimization. In: *Proceedings of the CEC*, pp. 2630–2637

[89] Han K and Kim J 2000 Genetic quantum algorithm and its application to combinatorial optimization problem. In: *Proceedings of the CEC*, vol. 2, pp. 1354–1360

[90] Han K and Kim J 2004 Quantum-inspired evolutionary algorithms with a new termination criterion, h-epsilon gate, and two-phase scheme. *IEEE Trans. Evolut. Comput.* 8(2): 156–169

[91] Liu H, Zhang G, Liu C and Fang C 2008 A novel memetic algorithm based on real-observation quantum inspired evolutionary algorithms. In: *Proceedings of ISKE*, pp. 486–490

[92] Sailesh Babu G S, Bhagwan Das D and Patvardhan C 2008 Real-parameter quantum evolutionary algorithm for economic load dispatch. *IET Gener. Transm. Distrib.* 2(1): 22–31

[93] Zhang G X and Rong H N 2007 Real-observation quantum-inspired evolutionary algorithm for a class of numerical optimization problems. In: *Lecture notes in computer science*, vol. 4490, p. 996

[94] Kumar S 2015 *Materialized view selection using randomized and evolutionary algorithms*. Ph.D. Thesis, Jawaharlal Nehru University, New Delhi

[95] Vijay Kumar T V and Kumar S 2012 Materialized view selection using simulated annealing. In: *Lecture notes in computer science (LNCS)*. Berlin: Springer, vol. 7678, pp. 168–179

[96] Vijay Kumar T V and Kumar S 2012 Materialized view selection using iterative improvement. *Adv. Intell. Syst. Comput.* 178: 205–214

[97] Vijay Kumar T V and Kumar S 2015 Materialized view selection using randomized algorithms. *Int. J. Bus. Inf. Syst.* 19(2): 224–240

[98] Derrac J, García S, Molina D and Herrera F 2011 A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evolut. Comput.* 1(1): 3–18

[99] Arun B and Vijay Kumar T V 2015 Materialized view selection using marriage in honey bees optimization. *Int. J. Nat. Comput. Res.* 5(3): 1–25

[100] Arun B and Vijay Kumar T V 2015 Materialized view selection using improvement based bee colony optimization. *Int. J. Softw. Sci. Comput. Intell.* 7(4): 35–61

[101] Vijay Kumar T V and Arun B 2016 Materialized view selection using BCO. *Int. J. Bus. Inf. Syst.* 22(3): 280–301

[102] Arun B and Vijay Kumar T V 2017 Materialized view selection using artificial bee colony optimization. *Int. J. Intell. Inf. Technol.* 13(1): 26–49

[103] Arun B and Vijay Kumar T V 2017 Materialized view selection using bumble bee mating optimization. *Int. J. Decision Support Syst. Technol.* 9(3): 1–27

[104] Vijay Kumar T V and Arun B 2017 Materialized view selection using HBMO. *Int. J. Syst. Assur. Eng. Manag.* 8(1): 379–392

[105] Kumar A and Vijay Kumar T V 2017 Improved quality view selection for analytical query performance enhancement using particle swarm optimization. *Int. J. Reliab. Quality Saf. Eng.* 24(6): https://doi.org/10.1142/S0218539317400010

[106] Vijay Kumar T V, Kumar A and Arun B 2017 Cuckoo search based view selection. In: *Emerging research in computing, information, communication and applications*. Berlin: Springer, pp. 327–337

[107] Kumar A and Vijay Kumar T V 2018 Materialized view selection using set based particle swarm optimization. *Int. J. Cogn. Inf. Nat. Intell.* 12(3): 18–39