Influence of Probability of Variation Operator on the Performance of Quantum-Inspired Evolutionary Algorithm for 0/1 Knapsack Problem

Mozammel H.A. Khan*

Department of Computer Science and Engineering, East West University, 43 Mohakhali, Dhaka 1212, Bangladesh

Abstract: Quantum-Inspired Evolutionary Algorithm (QEA) has been shown to be better performing than classical Genetic Algorithm based evolutionary techniques for combinatorial optimization problems like 0/1 knapsack problem. QEA uses quantum computing-inspired representation of solution called Q-bit individual consisting of Q-bits. The probability amplitudes of the Q-bits are changed by application of Q-gate operator, which is classical analogous of quantum rotation operator. The Q-gate operator is the only variation operator used in QEA, which along with some problem specific heuristic provides exploitation of the properties of the best solutions. In this paper, we analyzed the characteristics of the QEA for 0/1 knapsack problem and showed that a probability in the range 0.3 to 0.4 for the application of the Q-gate variation operator has the greatest likelihood of making a good balance between exploration and exploitation. Experimental results agree with the analytical finding.

Keywords: 0/1 knapsack problem, entropy of the probability distribution for the search space, evolutionary algorithm, performance analysis, quantum-inspired evolutionary algorithm.

1. INTRODUCTION

Evolutionary Algorithms (EAs) are nature-inspired, specifically biological evolution based, stochastic search or optimization techniques. In EAs, a population of initial candidate solutions is evolved into new populations of solutions from generation to generation to find out better fit solutions. EAs use two apparently conflicting techniques of exploitation and exploration to generate new solutions. These new solutions compete for survival and if they are better fit than the existing solutions, then they survive and replace some lower fit solutions in the population, otherwise they die out. In the exploitation technique, characteristics of better fit solutions are preserved into the new solutions with the hope to find out much better fit solutions. On the other hand, in the exploration technique, a wide solution space is searched to find out better fit solutions. In practical applications, a good balance between exploitation and exploration is required to find out global solutions within a reasonable time. To have a good balance between exploitation and exploration, the population dynamics such as population size, parent selection, variation operators, reproduction and inheritance, survival competition method, etc. are designed properly.

A nontraditional evolutionary computation technique called Quantum-Inspired Genetic Algorithm was developed by Narayanan [1, 2], where the concept of quantum mechanical interference was included in a modified crossover operator for solving traveling salesman problem. The most notable Quantum-Inspired Evolutionary Algorithm (QEA) was developed by Han [3-9]. In [3], a probabilistic representation and a novel population dynamics inspired by quantum computing were proposed. In [4], the applicability of QEA to a parallel scheme, particularly, PC clustering, was verified successfully. In [5], the basic structure of QEA and its characteristics were formulated and analyzed. It was also experimentally proved that QEA is better than classical GA for solving 0/1 knapsack problem. In [6], a QEA-based disk allocation method (QDM) was proposed, where the average query response times of QDM were equal to or less than those of disk allocation methods using GA (DAGA), and the convergence speed of QDM was 3.2-11.3 times faster than that of DAGA. In [7], QEA was applied to a decision boundary optimization for face verification. Compared with the conventional principal components analysis (PCA) method, improved results were achieved both in terms of the face verification rate and false alarm rate. In [8], some guidelines for setting the parameters of QEA were presented. In [9], extension of the basic QEA of [5] such as termination criterion, a modified version of the variation operator Q-gate called $H_{\in}$ gate, and a two-phase scheme were proposed to improve the performance of the QEA.

In the QEA of [5], Q-gate variation operator is applied on all the Q-bits of a Q-bit individual, that is, the variation operator is applied with a probability of 1. In this paper, we analyzed the characteristics of the QEA for 0/1 knapsack problem and showed that a probability in the range 0.3 to 0.4 for application of the Q-gate variation operator has the greatest likelihood of making a good balance between exploration and exploitation and improves the performance of QEA for 0/1 knapsack problem. We experimented with four sets of data, which agrees with the analytical finding.

The rest of the paper is organized as follows. In Section 2, background on quantum computing system is introduced. The 0/1 knapsack problem and the characteristics of the test data set are discussed in Section 3. In Section 4, we discuss the structure of the QEA for the 0/1 knapsack problem pro-

*Address correspondence to this author at the Department of Computer Science and Engineering, East West University, 43 Mohakhali, Dhaka 1212, Bangladesh; Tel: +8802 9882308; Fax: +8802 8812336;
E-mail: mhakhan@ewubd.edu

posed in [5]. We analyze the characteristics of the QEA in Section 5. In Section 6, we present an analytical model of influence of probability of application of the Q-gate variation operator on the performance of the QEA for 0/1 knapsack problem. We present the experimental results and related discussions in Section 7. Finally, Section 8 concludes the paper.

2. BACKGROUND ON QUANTUM COMPUTING SYSTEM

Understanding of the QEA proposed in [5] requires a sound understanding of the quantum computing system. In this section, we introduce quantum computing system in brief. For more details, readers can see the famous textbook by Nielsen and Chuang [10].

The basic unit of information in a quantum computing system is a quantum bit (qubit). A qubit exists in either $|0\rangle$ state or $|1\rangle$ state, which correspond to the logical values 0 or 1, respectively. These states are called the basis states. A qubit may also exist in linear superposition of the basis states

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

where, α and β are complex numbers representing the probability amplitudes of the basis states. After measurement, the qubit $|\psi\rangle$ becomes $|0\rangle$ with probability $|\alpha|^2$ and $|1\rangle$ with probability $|\beta|^2$, where the normalization condition requires that

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

If $\alpha = \pm 1/\sqrt{2}$ and $\beta = \pm 1/\sqrt{2}$, then the probability of finding the qubit in basis state $|0\rangle$ and $|1\rangle$ are equal, that means, the qubit exists in equal superposition state. A qubit can be geometrically visualized as a unit vector located in any quadrant of the rectangular coordinate system as shown in Fig. (**1**). In every case, the normalization condition satisfies that $|\pm\alpha|^2 + |\pm\beta|^2 = 1$, which implies that the length of the vector is 1.

The state of the qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ is represented by the column vector

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix}.$$

The state of a qubit can be changed by a 1-qubit quantum gate, which is characterized by a unitary operation U acting on the column vector representing the qubit. The unitary operation U is represented by a 2×2 unitary matrix satisfying the condition $U^\dagger U = UU^\dagger = I$, where $U^\dagger$ is the hermitian adjoint of U. There are many nontrivial 2×2 unitary matrices, but only a few of them are used in quantum computation. The frequently used 1-qubit quantum gates are Hadamard, Pauli-X (also known as NOT), Pauli-Y, Pauli-Z, Phase, $\pi/8$, and rotation gates. The readers can see [10] for more details on these 1-qubit gates. In [5], the rotation gate is used as variation operator. The rotation gate is defined below.

Definition 1. The rotation gate is defined using the unitary matrix

$$R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}. \tag{1}$$

If the rotation gate $R(\theta)$ is applied on the qubit $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, then the new state of the qubit will be

$$|\psi'\rangle = R(\theta)|\psi\rangle = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \alpha\cos\theta - \beta\sin\theta \\ \alpha\sin\theta + \beta\cos\theta \end{bmatrix} = \begin{bmatrix} \alpha' \\ \beta' \end{bmatrix} = \alpha'|0\rangle + \beta'|1\rangle,$$

where, $\alpha' = \alpha\cos\theta - \beta\sin\theta$ and $\beta' = \alpha\sin\theta + \beta\cos\theta$.

The application of the rotation gate on a qubit can be geometrically visualized as shown in the polar coordinate system of Fig. (**2**). If θ is positive, then the qubit is rotated θ angle to the counter-clockwise direction. If θ is negative, then the qubit is rotated θ angle to the clockwise direction. The rotation of the qubit changes the relative values of α and β causing the change of probability of finding the qubit in $|0\rangle$ and $|1\rangle$ states.

Theorem 1. (i) If the qubit is in the first or the third quadrant, then a counter-clockwise rotation increases the probability of finding the qubit in the basis state $|1\rangle$ and a clockwise rotation increases the probability of finding the qubit in the basis state $|0\rangle$. (ii) If the qubit is in the second or the fourth quadrant, then a clockwise rotation increases the probability of finding the qubit in the basis state $|1\rangle$ and

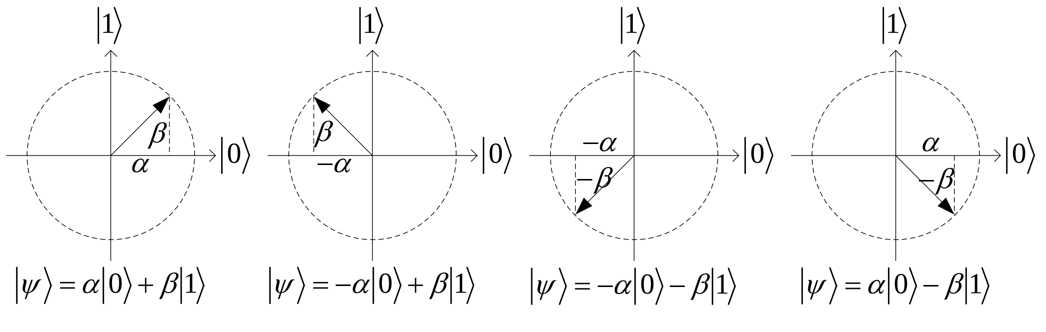
$|1\rangle$
$|0\rangle$
β
α
$-\alpha$
$-\beta$
$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$
$|\psi\rangle = -\alpha|0\rangle + \beta|1\rangle$
$|\psi\rangle = -\alpha|0\rangle - \beta|1\rangle$
$|\psi\rangle = \alpha|0\rangle - \beta|1\rangle$

Fig. (1). Geometric visualization of a Q-bit.

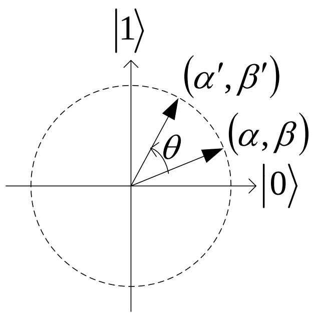

Fig. (2). Geometric visualization of rotation gate.

a counter-clockwise rotation increases the probability of finding the qubit in the basis state $|0\rangle$.

Proof. From Fig. (**3a**), where the qubit is rotated counter-clockwise in the first and the third quadrants, we see that $|\beta'|>|\beta|$ and $|\alpha'|<|\alpha|$. This implies that the probability of finding the qubit in the basis state $|1\rangle$ is increased. From Fig. (**3b**), where the qubit is rotated clockwise in the first and the third quadrants, we see that $|\alpha'|>|\alpha|$ and $|\beta'|<|\beta|$. This implies that the probability of finding the qubit in the basis state $|0\rangle$ is increased. This proves the first part of the theorem. From Fig. (**3c**), where the qubit is rotated clockwise in the second and the fourth quadrants, we see that $|\beta'|>|\beta|$ and $|\alpha'|<|\alpha|$. This implies that the probability of finding the qubit in the basis state $|1\rangle$ is increased. From Fig. **3**(**d**), where the qubit is rotated counter-clockwise in the second and the fourth quadrants, we see that $|\alpha'|>|\alpha|$ and $|\beta'|<|\beta|$. This implies that the probability of finding the qubit in the basis state $|0\rangle$ is increased. This proves the second part of the theorem.

An *m*-qubit quantum computing system exists in linear superposition of 2^m states. For example, if the three individual qubits of a 3-qubit quantum computing system are

$$|\psi_1\rangle = \alpha_1|0\rangle + \beta_1|1\rangle,$$

$$|\psi_2\rangle = \alpha_2|0\rangle + \beta_2|1\rangle,$$

$$|\psi_3\rangle = \alpha_3|0\rangle + \beta_3|1\rangle,$$

then the superposition of the 3-qubit quantum computing system is

$$\begin{aligned}|\psi_{123}\rangle &= \alpha_1\alpha_2\alpha_3|000\rangle + \alpha_1\alpha_2\beta_3|001\rangle + \\ &\alpha_1\beta_2\alpha_3|010\rangle + \alpha_1\beta_2\beta_3|011\rangle + \\ &\beta_1\alpha_2\alpha_3|100\rangle + \beta_1\alpha_2\beta_3|101\rangle + \beta_1\beta_2\alpha_3|110\rangle + \beta_1\beta_2\beta_3|111\rangle\end{aligned} \quad (2)$$

where, the normalization condition satisfies that

$$\begin{aligned}&|\alpha_1\alpha_2\alpha_3|^2 + |\alpha_1\alpha_2\beta_3|^2 + |\alpha_1\beta_2\alpha_3|^2 + \\ &|\alpha_1\beta_2\beta_3|^2 + |\beta_1\alpha_2\alpha_3|^2 + |\beta_1\alpha_2\beta_3|^2 + |\beta_1\beta_2\alpha_3|^2 + |\beta_1\beta_2\beta_3|^2 = 1\end{aligned}.$$

An *m*-qubit quantum computing system represents 2^m states simultaneously in superposition. However, the measurement of the system yields a single state. If $\alpha_1 = \alpha_2 = \cdots = \alpha_m = \beta_1 = \beta_2 = \cdots = \beta_m = \pm 1/\sqrt{2}$, then the *m*-qubit quantum computing system exists in equal superposition state, that means, after measurement, the probability of finding any of the 2^m states are equal. The state of an *m*-qubit quantum computing system can be changed by applying *m*-qubit quantum gates on the system or by applying 1-qubit gates on the individual qubits. An *m*-qubit quantum gate is characterized by a $2^m \times 2^m$ unitary matrix.

3. THE 0/1 KNAPSACK PROBLEM AND THE CHARACTERISTICS OF THE TEST DATA SET

In [5], the performance of the QEA is tested using 0/1 knapsack problem and we also do the same. For this reason, understandings of the 0/1 knapsack problem and the characteristics of the test data set should be made clear.

The knapsack problem can be described as selection of a subset of items from a given set of items, together with their weights and profits, such that the total weight of the selection does not exceed the given bound, that is, the capacity of the knapsack and the total profit of the selection is maximized. The knapsack problem is NP-hard. The 0/1 knapsack problem is described as follows.

Definition 2. Given a set of m items with weight w_i and profit p_i for $i = 1,2,\cdots,m$ and a knapsack with capacity C, find a binary vector $x = (x_1x_2\cdots x_m)$ such that the total weight $\sum_{i=1}^{m} w_i x_i \le C$ and the total profit $f(\mathrm{x}) = \sum_{i=1}^{m} p_i x_i$ is maximum. If $x_i = 1$, then the ith item is selected for the knapsack.

The performances of the algorithms for the 0/1 knapsack problem are normally analyzed and compared by running the algorithms on several sets of randomly generated test problems. Since the difficulty of such problems is greatly af-

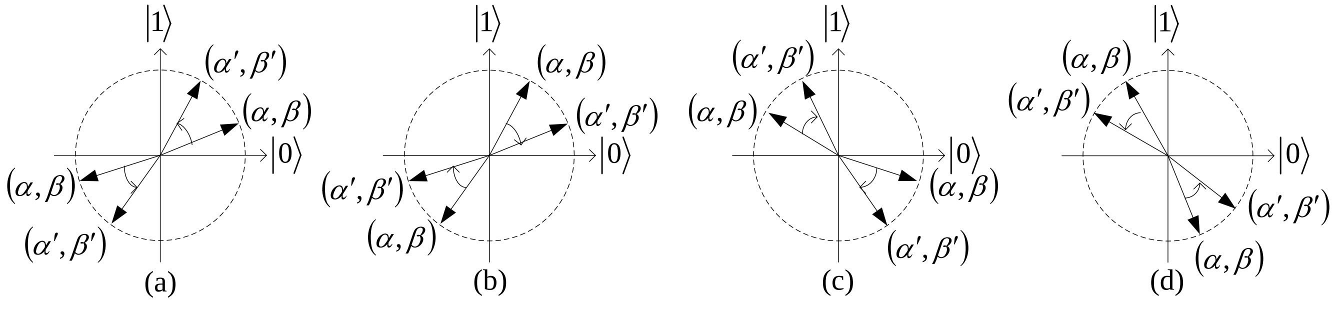

Fig. (3). Rotation of qubit changes the probability of finding the basis states.

fected by the correlation between weights and profits [11, 12], three randomly generated sets of data are usually used:

- uncorrelated:

  w_i = uniformly random $\left[1 \cdots v\right]$

  p_i = uniformly random $\left[1 \cdots v\right]$

- weakly correlated:

  w_i = uniformly random $\left[1 \cdots v\right]$

  $p_i = w_i$ + uniformly random $\left[-r \cdots r\right] > 0$

- strongly correlated:

  w_i = uniformly random $\left[1 \cdots v\right]$

  $p_i = w_i + r$

As reported in [11], higher correlation problems have higher expected difficulty. In [5], strongly correlated data sets are used and the data sets are generated with $v = 10$ and $r = 5$.

In [11], the following two knapsack capacities are suggested:

- restrictive knapsack capacity:

  $C = 2v$

- average knapsack capacity:

  $$C = 0.5\sum_{i=1}^{m} w_i$$

As reported in [11], in the knapsack with restrictive capacity, the optimal solution contains very few items. An area, for which conditions are not fulfilled, occupies almost the whole domain. In the knapsack with average capacity, the optimal solution contains about half of the items. But, this statement is not true for strongly correlated data set and average knapsack capacity as evident from Theorem 2 and Corollaries 1 and 2 below.

In [5], experiments are done with strongly correlated data set with $v = 10$, $r = 5$ and average knapsack capacity. We also do the same. This problem requires some in depth discussion.

Theorem 2. In the case of strongly correlated data set and average knapsack capacity, the global solution contains more than $m/2$ items with smaller weights, where m is the number of items.

Proof. In the case of strongly correlated data set and average knapsack capacity, the profit can be expressed as

$$f(\mathrm{x}) = \sum_{i=1}^{m} p_i x_i = \sum_{i=1}^{m} w_i x_i + r\sum_{i=1}^{m} x_i . \qquad (3)$$

From (3), we see that the profit consists of two parts – the first part is equal to the total weight of the selected items and the second part is equal to r times the number of items selected. Therefore, the profit can be maximized by making the total weight of the selected items equal to the knapsack capacity C that increases the value of the first part of (3) and simultaneously selecting more items that increases the value of the second part of (3). These interdependent conditions intuitively suggest that selecting more items of smaller weights within the capacity constraint will increase the profit. The weights of the m items are uniformly distributed in the range $\left[1 \cdots v\right]$. If we sort the m items in the ascending order of their weights, then the weights will be uniformly distributed from 1 to v in the ascending order and will form an arithmetic progression. In this case, the total weight of the first half of items will be less than the total weight of the second half of items. Obviously, the total weight of the first half of items will be less than the knapsack capacity C, since C is half of the total weight of all the m items. Therefore, if we select more than $m/2$ items from the lower end within the capacity constraints, then the number of selected items will be maximum and the profit will also be maximum.

Corollary 1. In the case of strongly correlated data set with $v = 10$ and average knapsack capacity, the global solution contains about $0.67m$ items with smaller weights, where m is the number of items.

Proof. The weights of the m items are uniformly distributed in the range $\left[1 \cdots 10\right]$. If we sort the m items in the ascending order of their weights, then the weights will be uniformly distributed from 1 to 10 in the ascending order and will form an arithmetic progression with common difference $d = (10-1)/(m-1) = 9/(m-1)$. Then the sum of weights of all m items will be

$$S_m = \frac{m(1+10)}{2} = 5.5m$$

and the capacity of the knapsack will be

$$C = \frac{S_m}{2} = \frac{5.5m}{2} = 2.75m . \qquad (4)$$

Now, let us assume that the sum of the weights of the first t items maximizes the total weight satisfying the capacity constraint $S_t \le C$. The sum of the weights of the first t items is

$$S_t = \frac{t\left[2\cdot 1 + (t-1)d\right]}{2} = \frac{t\left[2 + \frac{9(t-1)}{m-1}\right]}{2} = \frac{t\left[2m + 9t - 11\right]}{2(m-1)} . \qquad (5)$$

Using the equality condition $S_t = C$ and from (4) and (5), we have

$$\frac{t\left[2m + 9t - 11\right]}{2(m-1)} = 2.75m . \qquad (6)$$

After some algebraic manipulation of (6), we have

$$9t^2 + (2m-11)t + (5.5 - 5.5m^2) = 0. \quad (7)$$

From (7), we have that

$$t = \frac{-(2m-11) \pm \sqrt{(2m-11)^2 - 4 \times 9 \times 5.5(1-m^2)}}{2 \times 9} = \frac{(11-2m) \pm \sqrt{202m^2 - 44m - 77}}{18}. \quad (8)$$

If $m > 5$, then $(11-2m)$ will be negative and we can not consider "–" sign for the $\sqrt{202m^2 - 44m - 77}$ part of (8), since we will be working with a large value of m. Then, we can write

$$t = \frac{(11-2m) + \sqrt{202m^2 - 44m - 77}}{18}. \quad (9)$$

For known values of m, from (9) we have the values of t and t/m as shown in Table **1**.

Table 1. Values of t and t/m from (9)

m	t	t/m
100	68.3717	0.684
250	170.1449	0.681
500	339.7656	0.680
1000	679.0066	0.679

As we have considered the equality condition $S_t = C$ to determine the value of t rather than considering the capacity constraint condition $S_t \leq C$, then from Table **1**, we can conservatively take the value of t/m to be 0.67. Thus, about $0.67m$ items with smaller weights will maximize the selected weight and will also maximize the number of items selected. Form (3) and subsequent discussion, we have that this selection will produce the global optimal solution.

For experimentation, we have generated four sets of strongly correlated data set with $v = 10$ and $r = 5$ for m = 100, 250, 500, 1000. After sorting the generated data sets in the ascending order of weights, we have computed the value of t such that the sum of the first t items maximizes the total weight within the capacity constraint $S_t \leq C$ as shown in Table **2**. The experimental data of Table **2** is reasonably consistent with the data from Table **1**.

Table 2. Values of t and t/m from Generated Data Sets

m	t	t/m
100	67	0.670
250	168	0.672
500	338	0.676
1000	675	0.675

Corollary 2. In the case of strongly correlated data set with $v = 10$ and average knapsack capacity, if less than $0.67m$ items, where m is the number of items, maximize the total selected weight within the capacity constraint, then the selection will produce a local solution.

Proof. From Corollary 1, we see that if less than $0.67m$ items maximize the total weight within the capacity constraint, then the selected items will be of higher weights and the total number of items selected will be very small. From (3), we see that this selection can not produce a global solution and therefore, will produce a local solution.

Corollaries 1 and 2 can be used to develop new heuristics for exploiting the search mechanism.

4. THE QUANTUM-INSPIRED EVOLUTIONARY ALGORITHM FOR 0/1 KNAPSACK PROBLEM

In the QEA proposed in [5], two simultaneous representations of the solutions onto the individuals are used – one is qubit based representation and the other is binary solution corresponding to the vector x as defined in Definition 2. The representation of the binary vector x is already clear. Here, we describe the qubit based representation from [5].

Definition 3. A Q-bit is defined as the smallest unit of information in the QEA, which is defined as a pair of numbers (α, β) as

$$\begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

such that $|\alpha|^2 + |\beta|^2 = 1$. $|\alpha|^2$ gives the probability the Q-bit will produce a 0 and $|\beta|^2$ gives the probability that the Q-bit will produce a 1.

The Q-bit is an adaptation of the concept of qubit for the QEA. Like qubit, a Q-bit may be in the "1" state, in the "0" state, or in a linear superposition of the two.

Definition 4. A Q-bit individual is a string of m Q-bits defined as

$$\left[\begin{array}{c|c|c|c} \alpha_1 & \alpha_2 & \ldots & \alpha_m \\ \beta_1 & \beta_2 & & \beta_m \end{array}\right]$$

such that $|\alpha_i|^2 + |\beta_i|^2 = 1$ for $i = 1, 2, \cdots, m$.

The advantage of representing solution using Q-bit individual is that, from (2), we see that a single Q-bit individual is capable of representing 2^m binary solutions probabilistically.

The QEA maintains four data structures as described below.

Population of Q-bit individuals: The population of Q-bit individuals denoted $Q(t) = \{q_1^t, q_2^t, \cdots, q_n^t\}$ is a population of n (size of the population) Q-bit individuals of length m (length of each individual), t is the generation number, and q_j^t for $j = 1, 2, \cdots, n$ is defined as in Definition 4.

Population of observed binary solutions: The population of observed binary solutions denoted $P(t) = \left\{x_1^t, x_2^t, \cdots, x_n^t\right\}$ is a population of n binary strings of length m each observed from $Q(t-1)$, where t is the generation number and x_j^t is observed from q_j^{t-1} for $j = 1,2,\cdots,n$. The observation is made by the QEA operation **make** and will be discussed latter on.

Population of stored binary solutions: The population of stored binary solutions denoted $B(t) = \left\{b_1^t, b_2^t, \cdots, b_n^t\right\}$ is a population of best n binary strings of length m each selected from the populations $P(t)$ and $B(t-1)$. The population $B(t)$ stores the best n solutions so far generated.

The best binary solution: The best binary solution denoted b stores the best binary solution so far generated.

The structure of the QEA for the 0/1 knapsack problem is given below.

Procedure QEA for the 0/1 Knapsack Problem

```
begin
          $t \leftarrow 0$
(i)       initialize $Q(t)$
(ii)      make $P(t)$ by observing $Q(t)$
(iii)     repair $P(t)$
(iv)      evaluate $P(t)$
(v)       store $P(t)$ in $B(t)$
          while (t < MAX_GEN) do
          begin
            $t \leftarrow t+1$
(vi)        make $P(t)$ by observing $Q(t-1)$
(vii)       repair $P(t)$
(viii)      evaluate $P(t)$
(ix)        update $Q(t)$
(x)         store the best $n$ solutions among $B(t-1)$ and
            $P(t)$ into $B(t)$
(xi)        store the best solution among $B(t)$ to $b$
(xii)       if (global migration condition)
            then migrate $b$ to $B(t)$ globally
(xiii)      else if (local migration condition)
            then migrate $b_j^t$ in $B(t)$ to $B(t)$ locally
          end
end
```

The steps of the QEA are discussed below step wise

Step i: In the step of "initialize $Q(t)$", values of α_i and β_i for $i = 1,2,\cdots,m$ of all q_j^0 for $j = 1,2,\cdots,n$ are initialized to $1/\sqrt{2}$. It means that one Q-bit individual, q_j^0 represents the linear superposition of all 2^m possible binary solutions with equal probability.

Step ii: This step makes binary solutions in $P(0)$ by observing the states of $Q(0)$, where $P(0) = \left\{x_1^0, x_2^0, \cdots, x_n^0\right\}$ at generation $t = 0$. One binary solution, x_j^0 for $j = 1,2,\cdots,n$ is a binary string of length m, which is formed using the following **make (x)** procedure, which is a QEA operation. The make operation probabilistically determines the state of a Q-bit to be either 0 or 1 depending on the value of the probabilities $|\alpha_i|^2$ and $|\beta_i|^2$. Thus, a binary string of length m, x_j^0 is formed from the Q-bit individual q_j^0, which represents a solution observed from the Q-bit individual q_j^0. For notational simplicity, x_j^t is written as x in the procedure **make (x)**.

Procedure make (x)

```
begin
  $i \leftarrow 0$
  while ($i < m$) do
  begin
      $i \leftarrow i+1$
      if random [$0 \cdots 1$] < $|\beta_i|^2$
      then $x_i \leftarrow 1$
      else $x_i \leftarrow 0$
  end
end
```

Step iii: This step repairs the binary solution x_j^0 for $j = 1,2,\cdots,n$ in $P(0)$ for overfilled and under-filled corrections using the following **repair (x)** procedure, which is a QEA operation. If the knapsack is overfilled, then the first **while** loop of the **repair (x)** procedure converts some randomly selected 1s to 0s to reduce the total weight within the capacity constraint. If the knapsack is under-filled, the **repair (x)** procedure converts some randomly selected 0s to 1s to maximize the total weight within the capacity constraint.

Procedure repair (x)

```
begin
  knapsack-overfilled $\leftarrow$ false
```

```
    if ( $\sum_{i=1}^{m} w_i x_i > C$ )
    then knapsack-overfilled ← true
    while (knapsack-overfilled) do
    begin
        randomly select an $i$ such that $x_i = 1$
        $x_i \leftarrow 0$
        if ( $\sum_{i=1}^{m} w_i x_i \le C$ )
        then knapsack-overfilled ← false
    end
    while (not knapsack-overfilled) do
    begin
        randomly select an $i$ such that $x_i = 0$
        $x_i \leftarrow 1$
        if ( $\sum_{i=1}^{m} w_i x_i > C$ )
        then knapsack-overfilled ← true
    end
    $x_i \leftarrow 0$
end
```

Step iv: In this step of "evaluate $P(t)$", the fitness of each of the initial binary solution x_j^0 for $j = 1,2,\cdots,n$ in $P(0)$ is computed using the profit equation

$$f(x) = \sum_{i=1}^{m} p_i x_i \ .$$

Step v: In this step, the initial binary solutions in $P(0)$ is stored in $B(0)$ as best solutions so far generated.

Step vi: This step, within the **while** loop, makes the binary individuals in $P(t)$ by observing the Q-bit individuals in $Q(t-1)$ as in step ii.

Step vii: This step repairs the binary individuals in $P(t)$ as in step iii.

Step viii: This step evaluates the binary solutions in $P(t)$ as in step iv.

Step ix: In the "update $Q(t)$" step, Q-bit individuals in $Q(t)$ are updated by using the **update (** q **)** procedure, which uses Q-gates as variation operator as defined below.

Definition 5. A Q-gate is defined as a variation operator of the QEA, by which the values of α and β of a Q-bit are updated to α' and β' such that the normalization condition $|\alpha'|^2 + |\beta'|^2 = 1$ is satisfied.

In [5], the rotation gate $R(\theta)$ as defined in Definition 1 is used as a Q-gate. From Theorem 1, we see that the rotation gate $R(\theta)$ as a Q-gate rotates the Q-bit θ angle towards either 0 or 1 depending on the sign of the angle θ and the quadrant of the Q-bit. The value of the angle θ is determined as a function of the *i*th bit of the best solution b_j^t and the *i*th bit of the observed binary solution x_j^t. In [5], the experimentally found best value of θ as a function of the *i*th bit of the best solution b_j^t and the *i*th bit of the observed binary solution x_j^t is reported as in Table **3**.

Table 3. Experimentally Found Best Value of θ Used in the Rotation Gate $R(\theta)$ as Variation Operator in [5]

x_i	b_i	$f(x_j^t) \ge f(b_j^t)$	θ
0	0	false	0
0	0	true	0
0	1	false	0.01π
0	1	true	0
1	0	false	-0.01π
1	0	true	0
1	1	false	0
1	1	true	0

The **update (** q **)** procedure is given below.

```
Procedure update ( $q$ )
begin
    $i \leftarrow 0$
    while ( $i < m$ ) do
    begin
        $i \leftarrow i + 1$
        determine $\theta_i$ from Table 3.
        if ( $q$ is located in the first or third quadrant)
        then $\begin{bmatrix} \alpha_i' \\ \beta_i' \end{bmatrix} = R(\theta) \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix}$
        else $\begin{bmatrix} \alpha_i' \\ \beta_i' \end{bmatrix} = R(-\theta) \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix}$
    end
    $q \leftarrow q'$
end
```

The rotation gate used as a Q-gate in the **update (** q **)** procedure induces the convergence of each Q-bit to either 0 or 1. However, a Q-bit converged to either 0 or 1 cannot escape the state by itself, the **make (** x **)** procedure will always observe the converged value restricting any further exploration of the solution space. To prevent the premature convergence of Q-bit, a modified form of the rotation gate is proposed in [9] and called $H_{\in}$ gate as defined in Definition 6.

Definition 6. A $H_{\in}$ gate is defined as a Q-gate extended from the rotation gate as

$$\begin{bmatrix}\alpha' \\ \beta'\end{bmatrix} = H_{\in}(\alpha, \beta, \theta)$$

where for

$$\begin{bmatrix}\alpha'' \\ \beta''\end{bmatrix} = R(\theta)\begin{bmatrix}\alpha \\ \beta\end{bmatrix}:$$

(i) if $|\alpha''|^2 \le \in$ and $|\beta''|^2 \ge 1-\in$, then

$$\begin{bmatrix}\alpha' \\ \beta'\end{bmatrix} = \begin{bmatrix}\sqrt{\in} \\ \sqrt{1-\in}\end{bmatrix};$$

(ii) if $|\alpha''|^2 \ge 1-\in$ and $|\beta''|^2 \le \in$, then

$$\begin{bmatrix}\alpha' \\ \beta'\end{bmatrix} = \begin{bmatrix}\sqrt{1-\in} \\ \sqrt{\in}\end{bmatrix};$$

(iii) otherwise

$$\begin{bmatrix}\alpha' \\ \beta'\end{bmatrix} = \begin{bmatrix}\alpha'' \\ \beta''\end{bmatrix}$$

where $0 < \in << 1$, $R(\theta)$ is the rotation gate, and θ is the rotation angle.

The $H_{\in}$ gate can be visualized as shown in Fig. (**4**). [9], where $\lim_{\in \to 0} H_{\in}(\cdot)$ is the same as the rotation gate. While the rotation gate makes the probability of $|\alpha|^2$ or $|\beta|^2$ converge to either 0 or 1, $H_{\in}$ gate makes it converge to $\in$ or $(1-\in)$. It should be noted that if $\in$ is too big, the convergence tendency of a Q-bit individual may disappear. In [9], $\in = 0.01$ is suggested.

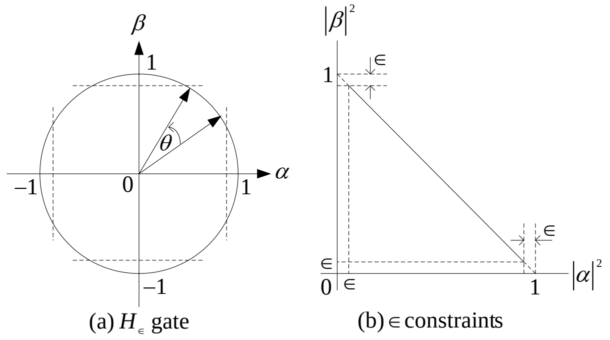

Fig. (4). $H_{\in}$ gate based on rotation gate [9].

Step x: In this step, the best n solutions among $B(t-1)$ and $P(t)$ are selected and stored into $B(t)$.

Step xi: In this step, if the best solution stored in $B(t)$ is fitter than the stored best solution b, then b is replaced by the best solution stored in $B(t)$.

Step xii: In this step, after a specified period known as global migration period, the best solution b is copied to all binary individuals in $B(t)$.

Step xiii: In this step, after a specified period known as local migration period, the best solution b_j^t among some of the solutions in $B(t)$ is copied to them.

5. ANALYSIS OF THE QEA FOR 0/1 KNAPSACK PROBLEM

In this section, we analyze the characteristics of the QEA operations.

Theorem 3. The **make (** x **)** procedure explores the solution space.

Proof. From Definition 3, we see that a Q-bit is represented by two probability amplitudes α and β, where $|\alpha|^2$ is the probability that the Q-bit will produce a 0 and $|\beta|^2$ is the probability that the Q-bit will produce a 1. Thus, from Definition 4, we see that a Q-bit individual of length m represents all 2^m solutions probabilistically. The **make (** x **)** procedure produces either 0 or 1 depending on the value of $|\beta|^2$ of the Q-bit and the random number generated within the procedure. Though, there is a bias of producing either 0 or 1 depending on the value of $|\beta|^2$, but probabilistically any of 0 or 1 may be generated, which practically explores the search space.

Theorem 4. The **repair (** x **)** procedure leads the solution towards a local optima.

Proof. If the knapsack is overfilled, then the **repair (** x **)** procedure reduces the total weight within the capacity constraint by reducing the number of 1s in the solution, but the possible maximum weight is produced. This maximizes the first part of (3). The number of 1s may be small by selecting higher-weight items or large by selecting lower-weight items. As there is no idea about the number of 1s in the repaired solution, we have no idea whether the second part of (3) is maximized or not. As the first part of (3) is maximized, we can say that the solution goes towards local optima. If the knapsack is under-filled, then the **repair (** x **)** procedure increases the total weight by increasing the number of 1s in the solution. In this case, both the first part and the second part of (3) are increased and the solution goes towards local optima, if not the global optima.

Theorem 5. (a) If $f(x_j^t) \ge f(b_j^t)$ is false, then the **update (** q **)** procedure converges the Q-bit individual q_j^t to binary solution b_j^t and (b) if $f(x_j^t) \geq f(b_j^t)$ is true, then the **update (q)** procedure keeps the Q-bit individual q_j^t unchanged.

Proof. From the **update (q)** procedure and Table **3**, we see that if $f(x_j^t) \geq f(b_j^t)$ is false, that is, the fitness of the observed binary individual x_j^t from the Q-bit individual q_j^t is less than the fitness of the corresponding stored binary solution b_j^t:

(i) If $x_i = 0$, $b_i = 1$, and the Q-bit is located in the first or the third quadrant, then the rotation angle is $\theta = 0.01\pi$, which increases the probability of 1 as evident from Theorem 1. That means, the Q-bit converges to $b_i = 1$.

(ii) If $x_i = 0$, $b_i = 1$, and the Q-bit is located in the second or the fourth quadrant, then the rotation angle is $\theta = -0.01\pi$, which increases the probability of 1 as evident from Theorem 1. That means, the Q-bit converges to $b_i = 1$.

(iii) If $x_i = 1$, $b_i = 0$, and the Q-bit is located in the first or the third quadrant, then the rotation angle is $\theta = -0.01\pi$, which increases the probability of 0 as evident from Theorem 1. That means, the Q-bit converges to $b_i = 0$.

(iv) If $x_i = 1$, $b_i = 0$, and the Q-bit is located in the second or the fourth quadrant, then the rotation angle is $\theta = -(-0.01\pi) = 0.01\pi$, which increases the probability of 0 as evident from Theorem 1. That means, the Q-bit converges to $b_i = 0$.

(v) If $x_i = b_i = 0$ or $x_i = b_i = 1$, and the Q-bit is located in any of the four quadrants, then the rotation angle is $\theta = 0$, which does not change the probabilities of 0 and 1. In this case, the Q-bit is already converged to b_i.

The above discussions reveal that the **update (q)** procedure converges the Q-bit individual q_j^t to stored binary solution b_j^t, which proves the part (a) of the theorem.

If $f(x_j^t) \geq f(b_j^t)$ is true, that is, the fitness of the observed binary individual x_j^t from the Q-bit individual q_j^t is greater than or equal to the fitness of the corresponding stored binary solution b_j^t, then for any combination of x_i and b_i, and for any location of the Q-bit in any quadrant, the rotation angle is $\theta = 0$, which does not change the probabilities of 0 and 1. The condition $f(x_j^t) \geq f(b_j^t)$ happens due to probabilistic exploration of the search space by the **make (x)** procedure. This new better fit x_j^t will be then stored in $B(t)$ in step x. This proves part (b) of the theorem.

Theorem 6. The **update (q)** procedure exploits the property of the already generated best solutions.

Proof. From the proof of Theorem 5, we see that if $f(x_j^t) \geq f(b_j^t)$ is true, then the Q-bit individual is not changed. But, if $f(x_j^t) \geq f(b_j^t)$ is false, then the **update (q)** procedure converges the Q-bit individual q_j^t to the stored binary solution b_j^t, which exploits the already generated best solution.

Theorem 7. The migration operation allows the QEA to escape local optima.

Proof. If the probabilities of all m Q-bits of the Q-bit individual q_j^t for $j = 1,2,\cdots,n$ converged to the corresponding bits (either 0 or 1) of the stored binary solution b_j^t for $j = 1,2,\cdots,n$, then the **make (x)** procedure will produce the same observed binary solution x_j^t repeatedly in the subsequent generations and the produced x_j^ts will be exactly equal to the corresponding stored binary solution b_j^ts. In this situation, no b_j^t will be replaced in step x and the QEA will be stuck in local optima. In this situation, if the best solution b is copied to all b_j^ts as the global migration process, then the **update (q)** procedure will have scope to change the probabilities of some Q-bits of some Q-bit individuals and the **make (x)** procedure will have scope to explore new solutions to escape the local optima. Similarly, if the probabilities of all m Q-bits of a subset of Q-bit individuals converged to the corresponding bits of their corresponding stored binary solutions, then that subset will suffer from similar problem. In this situation, copying the best solution b_j^t of the set to all solutions of the set as the local migration process will help to escape the problem.

Theorem 8. If the stored best solution b and the stored best solutions b_j^t for $j = 1,2,\cdots,n$ in $B(t)$ are equal, then the migration process fails to escape the local optima.

Proof. If all of the stored best solutions b_j^t for $j = 1,2,\cdots,n$ in $B(t)$ are equal to the stored best solution b, then both local and global migration will fail to change any of the stored binary solution in $B(t)$. Therefore, the QEA will be stuck at local optima.

The migration conditions are design parameters and should be carefully determined to make balance between exploration and exploitation.

6. INFLUENCE OF PROBABILITY OF VARIATION OPERATOR ON THE PERFORMANCE OF THE QEA FOR 0/1 KNAPSACK PROBLEM

From Theorem 4, we see that the **repair (x)** procedure generates local solutions. On the other hand, from Theorems 5 and 6, we see that the **update (q)** procedure exploits the stored local solutions and converge the Q-bit individuals to those local solutions. From Theorem 7, we see that by using migration we can escape from being stuck into these local solutions. But, from Theorem 8, we see that the QEA may fail to escape from local optima. The primary reason is that the **update (** q **)** procedure converges the Q-bit individuals to already generated local solutions. Therefore, in the subsequent generations, the **make (** x **)** procedure fails to explore new solutions. In general, we can say that the proposed QEA exploits more than explores. To improve the performance of the QEA, we should investigate some other means of making balance between the exploitation and exploration, such that the QEA can reach the global solution. In this section, we investigate the influence of the probability of application of the variation operator Q-gate in the **update (** q **)** procedure on the performance of the QEA.

Definition 7. The Hamming distance H_d of two binary strings, x_1 and x_2, is the number of positions where the bits of the two strings are not equal and is computed as

$$H_d(x_1, x_2) = \sum_{i=1}^{m}\left(x_{1i} \oplus x_{2i})\right)$$

where, m is the length of the binary strings and $\oplus$ is modulo 2 addition.

Theorem 9. The entropy of the probability distribution for the search space explored by the QEA with probability p_v of application of the Q-gate variation operator in the **update (** q **)** procedure is

$$H = -\sum_{h=0}^{m}\left(\frac{m!}{h!(m-h)!}\left(1-p_v\right)^{(m-h)} p_v^h \left(\log_2\left(\left(1-p_v\right)^{(m-h)} p_v^h\right)\right)\right) \quad (10)$$

where m is the length of the Q-bit individual.

Proof. Let x_j^t be the observed binary solution from the Q-bit individual q_j^t and b_j^t be the corresponding stored binary solution. Let h be the hamming distance between x_j^t and b_j^t. In the **update (** q **)** procedure, h Q-bits of the Q-bit individual q_j^t will converge to their corresponding bits in the b_j^t with a probability of p_v. Then the probability of converging to b_j^t is

$$p(b_j^t) = (1-p_v)^{(m-h)} p_v^h,$$

and the number of all possible b_j^t is

$$n(b_j^t) = \frac{m!}{h!(m-h)!}.$$

Therefore, the entropy [13] of the probability distribution for the search space explored by the QEA with probability p_v of application of the Q-gate variation operator in the **update (** q **)** procedure is

$$H = -\sum_{h=0}^{m}\left(n(b_j^t)p(b_j^t)\log_2 p(b_j^t)\right)$$

$$= -\sum_{h=0}^{m}\left(\frac{m!}{h!(m-h)!}\left(1-p_v\right)^{(m-h)} p_v^h \left(\log_2\left(\left(1-p_v\right)^{(m-h)} p_v^h\right)\right)\right)$$

The expression of entropy of the probability distribution for the search space of (10) is valid for $0 < p_v < 1$.

Theorem 10. The probability of application of the Q-gate variation operator in the range 0.3 to 0.4 will have the greatest likelihood of making a good balance between the exploration and the exploitation.

Proof. We have computed the value of the entropy H of the search space from (10) for $m = 10,20,30$ and different values of p_v and tabulated in Table **4**. From Table **4**, we see that $p_v = 0.5$ provides the highest exploration of the search space and is not desirable. If we choose $p_v > 0.5$, then exploration will be reduced but more Q-bits of the Q-bit individual q_j^t will be converged to the stored binary solution b_j^t and will provide more exploitation leading to local optima. If we choose $p_v << 0.5$, then exploration will be reduced but very few Q-bits of the Q-bit individual q_j^t will be converged to the stored binary solution b_j^t and will provide very small exploitation. Therefore, there is the greatest likelihood of making a good balance between the exploitation and exploration for the value of p_v in the range 0.3 to 0.4.

Table 4. Entropy *H* of the Search Space for Different Values of p_v and *m*

		m	
p_v	10	20	30
0.01	0.81	1.62	2.42
0.1	4.69	9.38	14.07
0.2	7.22	14.44	21.66
0.3	8.81	17.63	26.44
0.4	9.71	19.42	29.13
0.5	10.00	20.00	30.00
0.6	9.71	19.42	29.13
0.7	8.81	17.63	26.44
0.8	7.22	14.44	21.66
0.9	4.69	9.38	14.07
0.99	0.81	1.62	2.42

7. EXPERIMENTAL RESULTS

To verify the claim of Theorem 10, we experimented with four knapsack problems with 100, 250, 500, and 1000 items. We considered strongly correlated data set with $v = 10$ and $r = 5$, and average knapsack capacity. The data were unsorted. We experimented with two QEAs. The population sizes of QEA1 and QEA2 were 1 and 10, respectively. The global migration period for QEA2 was 1 generation and local migration was not used. For each QEA, both rotation gate and $H_{\in}$ gate were used as variation operators. The probability of the application of the variation operator was taken to be 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, and 1.0. As the performance measure, we collected the best solution

Table 5. Experimentally Found Average Best Profit Generated within 1000 Generations over 30 Runs

							p_v					
Item	QEA	Q-gate	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
	QEA1	R	582.4	587.6	589.2	**590.4**	589.5	588.4	590.1	588.0	586.8	587.3
100		$H_{\in}$	582.3	587.7	**591.0**	589.7	589.8	589.5	588.7	586.4	587.0	586.8
	QEA2	R	593.7	599.6	600.5	599.5	**600.8**	599.2	599.5	599.2	597.3	597.9
		$H_{\in}$	593.8	598.7	**601.4**	599.9	600.1	599.2	599.9	598.8	598.7	598.3
	QEA1	R	1414.1	1430.7	1436.7	**1440.8**	1439.5	1440.2	1438.0	1436.1	1433.3	1433.1
250		$H_{\in}$	1414.5	1434.5	1441.0	1440.1	1437.6	**1442.0**	1437.1	1434.1	1434.0	1432.7
	QEA2	R	1443.2	1469.0	**1473.9**	**1473.9**	1473.5	1470.3	1467.4	1465.1	1465.1	1463.5
		$H_{\in}$	1443.4	1468.2	1473.5	**1474.0**	1470.9	1469.1	1470.3	1466.4	1464.2	1464.5
	QEA1	R	2780.4	2810.1	2820.1	**2827.4**	2823.6	2820.1	2820.2	2812.8	2812.4	2810.7
500		$H_{\in}$	2779.8	2808.4	2820.4	**2821.9**	2821.7	2817.6	2814.9	2816.0	2814.2	2811.9
	QEA2	R	2824.8	2876.2	**2886.3**	2885.7	2877.6	2878.8	2873.6	2868.0	2870.0	2868.2
		$H_{\in}$	2826.4	2872.2	2885.0	**2885.6**	2883.4	2876.8	2875.1	2871.2	2868.0	2863.1
	QEA1	R	5479.4	5525.3	5548.5	**5560.5**	5550.5	5542.2	5542.8	5544.3	5540.2	5532.9
1000		$H_{\in}$	5481.4	5530.8	5552.1	**5558.7**	5546.8	5545.8	5542.3	5545.4	5535.1	5533.4
	QEA2	R	5557.0	5629.5	5651.7	**5658.1**	5645.2	5640.8	5631.0	5633.0	5627.4	5630.8
		$H_{\in}$	5560.6	5633.1	5651.3	**5652.8**	5646.6	5638.6	5635.5	5626.0	5624.2	5620.8

found within 1000 generations and averaged them over 30 runs. In every run, the random generator was seeded with random seed. The results are summarized in Table **5**. In every case, the best profit is shown in bold face. From Table 5, we see that best results are found for three cases at $P_v = 0.3$, for one case at $P_v = 0.3, 0.4$, for ten cases at $P_v = 0.4$, for one case at $P_v = 0.5$, and for one case at $P_v = 0.6$. From these observations, we see that P_v in the range 0.3 to 0.4 will give the greatest likelihood of getting the best result, which agrees with the Theorem 10.

We also have collected the best solution generated in every generation and averaged them over 30 runs. The convergence trends for QEA2 with 1000 items and rotation gate as variation operator for p_v = 0.1, 0.4, 0.7, 1.0 are shown in Fig. (**5**). From Fig. (**5**), we see that p_v = 0.1 converges very slowly and generates the smallest profit, which indicates that it provides very small exploitation and randomly explores the search space. On the other hand, p_v = 0.7 and 1.0 prematurely converged due to more exploitation than exploration and produced local solution. p_v = 0.4 reasonably explored and exploited and generated the best solution.

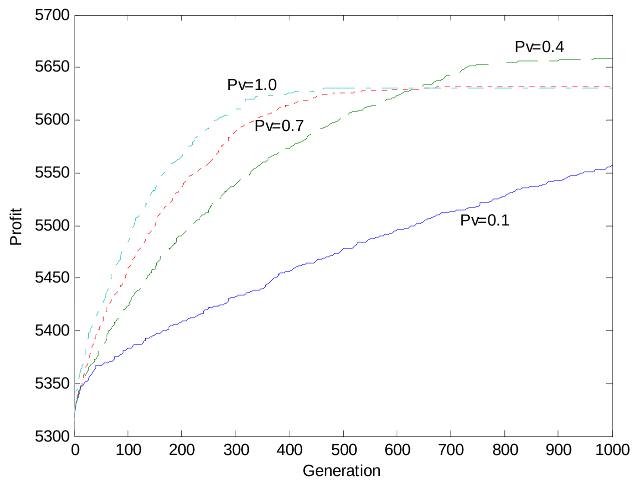

Fig. (5). Convergence trends for QEA2 with 1000 items and rotation gate variation operator for different values of probability of application of the variation operator.

CONCLUSION

In [5], a non-traditional evolutionary algorithm called Quantum-Inspired Evolutionary Algorithm (QEA) was proposed and showed to be better performing than classical Genetic Algorithms for 0/1 knapsack problem. In [9], some improvements were proposed. In this paper, we analyzed the characteristics of the QEA operators and showed that the QEA may fail to produce global solution due to over exploitation than exploration. In the QEA of [5], the variation operator was applied with probability 1. We analytically show that a lower probability of application of the variation operator will provide a good balance between the exploration and exploitation. A very low probability of application of the variation operator exploits less and explores the search space randomly and takes larger number of generation to produce even the local solution. A large probability of the application of the variation operator exploits more and prematurely converged to local solution. We show that the probability of application of the variation operator in the range 0.3 to 0.4 will have the greatest likelihood of providing good balance between exploration and exploitation. The experimental results agree with the claim.

We analyzed the behavior of the knapsack problem with strongly correlated data set and average knapsack capacity and showed that the global solution contains more than 50% items with lower weights. This knowledge can be used to device new heuristic to improve the performance of the QEA for 0/1 knapsack problem.

REFERENCES

[1] A. Narayan, and M. Moore, "Quantum-inspired genetic algorithms", in *IEEE International Conference on Evolutionary Computation*, 1996, pp. 61-66.

[2] A. Narayan, "Quantum computing for beginners", in *IEEE Congress on Evolutionary Computation*, 1999, pp. 2231-2238.

[3] K.-H. Han and J.-H. Kim, "Genetic quantum algorithm and its application to combinatorial optimization problem", in *IEEE Congress on Evolutionary Computation*, 2000, pp. 1354-1360.

[4] K.-H. Han, K.-H. Park, C.-H. Lee, and J.-H. Kim, "Parallel quantum-inspired genetic algorithm for combinatorial optimization problem", in *IEEE Congress on Evolutionary Computation*, 2001, pp. 1422-1429.

[5] K.-H. Han and J.-H. Kim, "Quantum-inspired evolutionary algorithm for a class of combinatorial optimization", *IEEE Trans. Evol. Comput.*, vol. 6, pp. 580-593, December 2002.

[6] K.-H. Kim, J.-Y. Hwang, K.-H. Han, J.-H. Kim, and K.-H. Park, "A quantum-inspired evolutionary computing algorithm for disk allocation method", *IEICE Trans. Inform. Syst.*, vol. E86-D, pp. 645-649, March 2003.

[7] J.-S. Jang, K.-H. Han, and J.-H. Kim, "Quantum-inspired evolutionary algorithm-based face verification", in *Genetic Evolutionary Computation Conference*, 2003, pp. 2147-2156.

[8] K.-H. Han and J.-H. Kim, "On setting the parameters of quantum-inspired evolutionary algorithm for practical applications", in *IEEE Congress on Evolutionary Computation*, 2003, pp. 178-184.

[9] K.-H. Han and J.-H. Kim, "Quantum-inspired evolutionary algorithms with a new termination criterion, $H_{\in}$ gate, and two-phase scheme", *IEEE Trans. Evol. Comput.*, vol. 8, pp. 156-169, April 2004.

[10] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge University Press, 2000.

[11] S. Martello and P. Toth, *Knapsack problems*, Chichester, UK: John Wiley, 1990.

[12] Z. Michalewicz, *Genetic algorithms + data Structures = evolution programs*, New York: Springer-Verlag, 1999.

[13] S. J. Russel and P. Norvig, *Artificial intelligence: a modern approach*. Englewood Cliffs, NJ: Prentice-Hall, 1995.

Received: December 17, 2009 Revised: March 06, 2010 Accepted: March 08, 2010