An Angle-expressed Quantum Evolutionary Algorithm for Quadratic Knapsack Problem

Li Hao
School of Electrical Engineering, Southwest Jiaotong university, Chengdu, China
E-mail: hblh198095@163.com

Abstract. The quadratic knapsack problem (QKP) is a typical combinatorial optimization problem. It is arisen in many optimization fields, but there exist no pseudo-polynomial time algorithm to solve it. In this paper, we propose an angle-expressed quantum evolutionary algorithm to solve QKP. In this algorithm, the qubits are expressed in the angle and initialized according to the value densities of their corresponding items, the rotation angle of Q-gate is determined through an analytical formula, and $H\varepsilon$ gate is used to prevent from premature convergence. For infeasible solution, a dynamic value density is introduced to choose item to be selected or dropped. Finally, experiment demonstrates the effectiveness of the algorithm.

1. Introduction
QKP is a typical combinatorial optimization problem. After being introduced by Gallo\cite{1}, it has been widely used in a variety of fields, such as finance, station layout, cryptographic design, telecommunication, VLSI design, and so on\cite{2-5}. The research on QKP is of great significance both in theory and in practice. However, QKP is a NP-complete problem, which makes it difficult for the traditional methods to solve it in polynomial time. In 2005, Julstrom proposed a hybridization of the genetic algorithm with greedy heuristic to solve QKP\cite{6}. In 2009, Pulikanti used an artificial bee colony algorithm to solve QKP\cite{7}. In 2014, Azad proposed a simplified binary artificial fish swarm algorithm for solving QKP\cite{8}

Quantum evolutionary algorithm is an intelligent optimization method inspired by the quantum computing principles. In 2002, Han proposed a quantum-inspired evolutionary algorithm (QEA) based on the concept of qubit and superposition of quantum states\cite{9}. This algorithm introduced the quantum state expression into the genetic code, taken the Q-gate to evolve the chromosome, and used it to solve knapsack problem (KP), the result shows that QEA has better performance than conventional genetic algorithm (CGA), thus, it obtained wide attention and has been applied to many fields\cite{10-13}

In this work, we propose an angle-expressed quantum evolutionary algorithm (AQEA) for QKP. In the algorithm, the qubits are expressed as angle and initialized according to the value densities corresponding to the items, the rotation angle is determined through directly comparing the quantum chromosome with the best solution, $H\varepsilon$ gate is used to prevent from premature convergence. For the infeasible solution, a dynamic value density is taken to choose item to be selected or dropped. Using this algorithm to solve QKP, the simulation results show it has excellent optimization performance.

This paper is organized as follows. QKP and QEA are described in Section 2 and Section 3, respectively. Next, Section 4 presents an AQEA for QKP. And the experimental results are shown in Section 5. Finally, conclusions are given in Section 6.
2. QKP

QKP can be described as: given $n$ items and a knapsack, select a subset of the items to put in the knapsack so as to maximize the profit, that is

$$\max f(X) = \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij} x_{ij}$$

Subject to $\sum_{i=1}^{n} w_i x_i \leq C$

where $w_i$ and $p_{ij}$, $1 \leq i \leq n$ is the weight and the profit of the $i$-th item respectively, $p_{ij} + p_{ji}$ is the additional profit achieved if both items $i$ and $j$ are selected. Without loss of generality, let $p_{ij} = p_{ji} = 0$. $C$ is the capacity of the knapsack. $x_i \in \{0, 1\}$, $x_i = 1$ if the $i$-th item is selected, otherwise $x_i = 0$. So we can see, different from KP, in QKP, not only the value of each item itself, the collaboration value with other items will be considered. When $p_{ij} = 0$ for all $i \neq j$, QKP will become a classic KP.

3. QEA

In QEA, the smallest unit of information is qubit. A qubit can be represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $|\alpha|^2 + |\beta|^2 = 1$, $|\alpha|^2$ and $|\beta|^2$ give the probability that the qubit will be found in the state $|0\rangle$ and $|1\rangle$, respectively. Further, a chromosome can be represented as a string of $n$-qubits, that is

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_n \end{pmatrix}$$

The procedure of QEA is described as follows:

Begin

Initialize $Q(t)$ at $t = 0$
Make $P(t)$ by observing $Q(t)$
Evaluate $P(t)$, Store the best solutions

While (not termination condition) do

$t = t + 1$
Make $P(t)$ by observing $Q(t-1)$
Evaluate $P(t)$
Update $Q(t)$ using Q-gates
Store the best solutions

End

End

where $Q(t) = \{q_1, q_2, \ldots, q_m\}$, $q_j = \begin{pmatrix} \alpha_{j1} \\ \alpha_{j2} \\ \vdots \\ \alpha_{jm} \end{pmatrix}$, $P(t) = \{X'_1, X'_2, \ldots, X'_m\}$, $X'_j = \{x'_{j1}, x'_{j2}, \ldots, x'_{jm}\}$

$B(t) \in X_j$, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, m$, $m$ is the size of the population.

In the step of “Initialize $Q(t)$ at $t = 0$”, $\alpha$ and $\beta$ of all $q_j$ in $Q(0)$ are initialized with $\sqrt{\frac{1}{2}}$, which means that all possible states will be obtained with the same probability.

To obtain the binary string, the step of “Make $P(t)$ by observing $Q(t)$” is implemented. When $Q(t)$ is observed, $x'_j = 0$ or 1 of $P(t)$ will determine according to the probability $|\alpha'|^2$ or $|\beta'|^2$.

In the “update” operation, Q-gate is used to update the chromosome, it is as follows

$$\begin{pmatrix} \alpha_j \\ \beta_j \end{pmatrix} = \begin{pmatrix} \cos \Delta \theta_j & -\sin \Delta \theta_j \\ \sin \Delta \theta_j & \cos \Delta \theta_j \end{pmatrix} \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix}$$

where $\Delta \theta_j$ is the rotation angle of the Q-gate. It is determined from a lookup table\textsuperscript{9}.
4. An AQEA for QKP

4.1 Qubit with angle expression
In AQEA, qubit is represented in an angle \( \theta \). Specifically, \( \alpha \) and \( \beta \) are expressed as \( \alpha = \cos \theta \), \( \beta = \sin \theta \), respectively, so \( |\psi\rangle = \cos \theta |0\rangle + \sin \theta |1\rangle \). It also meet \( \cos^2 \theta + \sin^2 \theta = 1 \). When a qubit is measured, 0 or 1 will be gotten with the probability of \( \cos \theta \) or \( \sin \theta \), respectively. Further, a chromosome can be expressed as \( q = (\theta_1 |0\rangle \cdots |0\rangle) \). The Q-gate operation can be converted as \( \theta' = \Delta \theta + \theta \). Obviously, the expression and evolution of the chromosome can be simplified sharply if the qubits are expressed with angle.

4.2 Determination of rotation angle of Q-gate
The determination of the rotation angle is the key to affect the performance of QEA. Considering a bit is "0" or "1" can correspond to a special case that the \( \theta \) is 0 or \( \pi/2 \), so we can directly compare the qubit and the best solution to determine the rotation angle. For a binary number \( b \), we define its angle \( \theta_b = 0 \) if \( b = 0 \), and \( \theta_b = \pi/2 \) if \( b = 1 \). Further, we define the rotation angle:

\[
\Delta \theta_i = (\theta_{b_i} - \theta_i) \cdot r
\]

where \( \theta_i \) is the \( i \)th qubit of a chromosome, \( \theta_{b_i} \) is the angle of \( i \)th bit of individual best solution of this chromosome, and \( r \in [0,1] \) is an adjusting factor.

4.3 Population initialization
In QEA, all qubits are initialized with \( 1/\sqrt{2} \), and all solutions will be found with the same probably. Actually, we can set the initial values according to the actual situation so as to achieve better results. Aim at QKP, we firstly define an initial value density for each item, that is, \( d_i = p_i/w_i, i = 1, \ldots, n \), where \( p_i = \sum_{j=1}^n p_j \). Further, in initialization of AQEA, we can increase the probability that the items with high value density be selected and decrease the probability that the items with low value density be selected to improve the optimization performance.

4.4 He gate
To avoid premature convergence, the concept of He gate is introduced, it can be define

\[
\theta'_i = \begin{cases} 
\varepsilon & \theta_i \leq \varepsilon \\
\theta_i & \varepsilon \leq \theta_i \leq \pi/2 - \varepsilon \\
\pi/2 - \varepsilon & \theta_i \geq \pi/2 - \varepsilon 
\end{cases}
\]

where \( 0 < \varepsilon < \pi/2 \), \( \theta_i \) is the qubit angle after rotating. Can be seen, He gate makes the angle always in the area \( [\varepsilon, \pi/2 - \varepsilon] \) to maintain the randomness.

The procedure of AQEA for QKP is described as follows:

Begin
Sort items by value density, initialize \( Q(t) \) at \( t = 0 \)
Make \( P(t) \) by observing the states of \( Q(t) \)
Evaluate \( P(t) \), store \( P(t) \) into \( B(t) \), store the best solutions among \( B(t) \)
While (not termination condition) do
\( t = t + 1 \)
Make \( P(t) \) by observing the states of \( Q(t-1) \)
Evaluate \( P(t) \), store the best solutions
Update \( Q(t) \) using Q-gates
End
End
5. Experimental results

5.1 Performance testing
To verify the performance of AQEA, 30 benchmark test instances are taken\(^\left[14\right]\). In which the number of variables (NV) is 100, 200, 300, respectively. And the non-zeros in the profit coefficients is 50\%. In each of the three cases, 10 samples are taken. In AQEA, the population size is 20, the maximum number of iteration is 10NV, \(\varepsilon = 0.03\pi\), \(r = 0.003\). For the infeasible solutions, a repair method is introduced. Firstly, we define the dynamic value density of the \(i\)th items

\[ d_i' = \frac{p_i'}{w_i}, \quad i = 1, \ldots, n \]  \(\quad (6)\)

where \(p_i' = 2 \sum_{j \in \Omega} p_y + p_z\) is the increased or decreased value if the \(i\)th item is selected or dropped, \(\Omega\) is the set in which the items are selected. According to \(d_i'\), we can determine the dynamic value of an item if it will be added into a feasible solution or dropped from an infeasible solution. Then in the algorithm, every binary string \(X\) is repaired with the strategy below

Procedure repair \(X\)

Begin

All items in the knapsack are sorted according to \(d_i'\)
Knapsack-overfilled = false
If \(\sum_{i=1}^{n} w_i x_i > C\) Knapsack-overfilled = true
While (Knapsack-overfilled = true) do
Select the \(j\)-th item which is the last item being chosen, let \(x_j = 0\)
End
While (Knapsack-overfilled = false) do
Select the \(j\)-th item which is the first item being not chosen, let \(x_j = 1\)
End
\(x_j = 0\)
End

All experiments run 50 times, the results are shown in Table 1. In which BS, BR, AR, WR, NB means the best solution, the best result, the average result, the worst result, the number of getting best solution, respectively, and \(r_{m\_n\_k}\) means the \(k\)th instance with NV is \(m\) and density is \(n\).

| Problem | BS     | BR     | AR     | WR     | NB     | Problem | BS     | BR     | AR     | WR     | NB     |
|---------|--------|--------|--------|--------|--------|---------|--------|--------|--------|--------|--------|
| r_100_50_1 | 83742  | 83742  | 83742  | 83742  | 50     | r_200_50_1 | 372097 | 372097 | 372097 | 372097 | 50     |
| r_100_50_2 | 104856 | 104856 | 104818 | 104792 | 20     | r_200_50_2 | 211130 | 211130 | 211123 | 211087 | 31     |
| r_100_50_3 | 34006  | 34006  | 34006  | 34006  | 50     | r_200_50_3 | 227185 | 227185 | 227185 | 227185 | 50     |
| r_100_50_4 | 105996 | 105996 | 105996 | 105996 | 50     | r_200_50_4 | 228572 | 228572 | 228572 | 228572 | 50     |
| r_100_50_5 | 56464  | 56464  | 56464  | 56464  | 50     | r_200_50_5 | 479651 | 479651 | 479651 | 479651 | 50     |
| r_100_50_6 | 16083  | 16083  | 16083  | 16083  | 50     | r_200_50_6 | 426777 | 426777 | 426771 | 426755 | 36     |
| r_100_50_7 | 52819  | 52819  | 52819  | 52819  | 50     | r_200_50_7 | 220890 | 220890 | 220890 | 220890 | 50     |
| r_100_50_8 | 54246  | 54246  | 54246  | 54246  | 50     | r_200_50_8 | 317952 | 317952 | 317952 | 317952 | 50     |
| r_100_50_9 | 68974  | 68974  | 68974  | 68974  | 50     | r_200_50_9 | 104936 | 104936 | 104936 | 104936 | 50     |
| r_100_50_10 | 88634  | 88634  | 88634  | 88634  | 50     | r_200_50_10 | 284751 | 284751 | 284751 | 284751 | 46     |
| r_300_50_1 | 513379 | 513379 | 513379 | 513379 | 50     | r_300_50_6 | 734053 | 734053 | 734053 | 734053 | 50     |
The results in Table 1 shown that for all 30 instances, AQEA can find the best solution, which shows its effectiveness, and in most of cases, the 30 optimization results can all find the best solution, which shows that the algorithm has good optimization performance and can solve QKP well.

5.2 Algorithm comparison

Next, we take 3 instances to compare AQEA with CGA and QEA. All tests are run 50 times by three algorithms, respectively. In them, the population size is 20, the maximum number of iteration is 10NV. The optimization results are shown in Table 2, and Figure1 show the average results of 50 tests.

| Problem    | Algorithm | BS     | BR     | AR     | WR     | NB |
|------------|-----------|--------|--------|--------|--------|----|
| r_100_25_1 | CGA       | 18558  | 18558  | 18448  | 18255  | 1  |
|            | QEA       | 18558  | 18558  | 18520  | 18422  | 6  |
|            | AQEA      | 18558  | 18558  | 18558  | 18558  | 50 |
| r_200_25_1 | CGA       | 204441 | 204441 | 204374 | 204161 | 16 |
|            | QEA       | 204441 | 204441 | 204353 | 204102 | 8  |
|            | AQEA      | 204441 | 204441 | 204439 | 204401 | 48 |
| r_300_25_1 | CGA       | 29140  | 29140  | 28763  | 28407  | 2  |
|            | QEA       | 29140  | 29140  | 29049  | 28745  | 18 |
|            | AQEA      | 29140  | 29140  | 29140  | 29140  | 50 |

![Figure 1. Optimization curves](a) r_100_25_1 (b) r_200_25_1 (c) r_300_25_1)

We can see that in Table 2, as three effective optimization algorithms, they can all solve the problem effectively, but in contrast, AQEA has better performance. Meanwhile, Figure1 show that AQEA has faster convergence speed than CGA and QEA. Totally, its performance is superior to the other two algorithms.

6. Conclusions

QKP is a typical combinatorial optimization problem. The research on QKP is of great significance both in theory and in practice. However, this problem is a NP complete problem, the difficulty of solving it will increase exponentially as problem size increases. On the basis of QEA, a series of improvements were carried out and AQEA is put forward for QKP. In this algorithm, the qubits are defined in angle and initialized hierarchically according to the value densities of their corresponding items, the rotation angle is determined by directly comparing the quantum chromosome and the best solution, \( H\) gate is taken to avoid premature convergence. For infeasible solution, a dynamic value density is introduced to choose item to be added or dropped. The simulation results for a set of
benchmark instances show that the algorithm can solve QKP successfully, and its performance is better than CGA and QEA.

References
[1] Gallo G., Hammer P., Simeone B.. (1980) Quadratic Knapsack Problem. Math. Programming Study. 12: 132-149.
[2] Laghunn D.L. (1970) Quadratic binary programming with applications to capital budgeting problems. Oper.Res. 18: 454-461.
[3] Ferreira C. E., Martin A., C.C.deSouza. (1996) Formulations and valid inequalities for node capacitated graph partitioning. Math. Program. 74: 247-266.
[4] C.Witzall. (1975) Mathematical methods of site selecting for electronic message system. NBS Int. Rep. Tech Rep.
[5] Patvardhan C., Bansal S., Srivastav A.. (2015) Solving the 0-1 quadratic knapsack problem with a competitive quantum inspired evolutionary algorithm. Journal of Computational and Applied mathematics. 285: 86-99.
[6] Julstrom B. A. (2005) Greedy, genetic and greedy genetic algorithm for the quadratic knapsack problem. Proceedings of 2005 Congress on Genetic and Evolutionary Computation. New York, USA: ACM. 607-614.
[7] Pulikanti S, Singh A. (2009) An artificial bee colony algorithm for the quadratic knapsack problem. Neural Information Processing, 58: 196-205.
[8] Azad Md. A. K., Rocha A. M. A. C., Fernandes E. M. G. P.. (2014) A simplified binary artificial fish swarm algorithm for 0-1 quadratic knapsack problems. Journal of Computational and Applied mathematics. 259: 897-904.
[9] Han K.H., Kim J–H. (2002) Quantum- inspired Evolutionary Algorithm for a Class of Combinatorial Optimization. IEEE Trans Evolutionary Computation. 6: 580–593.
[10] Wang Y, Feng X.Y, Huang Y.X. (2007) A novel quantum swarm evolutionary algorithm and its applications. Neurocomputing. 70: 633-640.
[11] Pavithr R.S., Gursaran. (2016) Quantum inspired social evolution(QSE) algorithm for 0-1 knapsack problem. Swarm and Evolutionary Computation. 29: 33-46.
[12] Luciano R, Ricardo T, Marley M B R V. (2017) Quantum inspired evolutionary algorithm for ordering problems. Expert Systems with Applications. 67: 71-83.
[13] Arpaia P, Maisto D, Manna C. (2011) A quantum-inspired evolutionary algorithm with a competitive variation operator for multiple-fault diagnosis. Applied soft computing. 11: 4655-4666.
[14] Billionet, E. Soutif, (2004) QKP instances.[Online]. http://cedric.cnam.fr/~soutif/QKP/QKP.html.