Journal of Systems Engineering and Electronics

Vol. 20, No. 5, 2009, pp.1012–1016

Quantum-inspired ant algorithm for knapsack problems*

Wang Honggang, Ma Liang, Zhang Huizhen & Li Gaoya

Business School, Univ. of Shanghai for Science and Technology, Shanghai 200093, P. R. China

(Received March 11, 2008)

Abstract: The knapsack problem is a well-known combinatorial optimization problem which has been proved to be NP-hard. This paper proposes a new algorithm called quantum-inspired ant algorithm (QAA) to solve the knapsack problem. QAA takes the advantage of the principles in quantum computing, such as qubit, quantum gate, and quantum superposition of states, to get more probabilistic-based status with small colonies. By updating the pheromone in the ant algorithm and rotating the quantum gate, the algorithm can finally reach the optimal solution. The detailed steps to use QAA are presented, and by solving series of test cases of classical knapsack problems, the effectiveness and generality of the new algorithm are validated.

Keywords: knapsack problem, quantum computing, ant algorithm, quantum-inspired ant algorithm.

1. Introduction

The knapsack problem is a well-known combinatorial optimization problem which belongs to the typical NP-hard problem[1] and has a comprehensive variety of applications, such as budget controlling, decision making and material cutting. In addition, it is also studied as the sub-problem of other complex problems.

A knapsack problem is generally described as: given a knapsack with limited capacity $C \in Z^+$ and n items, each having profit $p_i \in Z$ and weight $w_i \in Z$ $(i = 1, 2, \ldots, n)$, select a subset or subsets of the n items such that their total weights doesn't overweight the given capacity C but as large profit as possible. In this paper, we mainly discuss the 0-1 knapsack problem formulated as

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

$$\text{s. t.} \quad \sum_{i=1}^{m} w_i x_i \leqslant C, \quad \forall i = 1, \ldots, n \tag{2}$$

$$x_i \in \{0, 1\}, \quad \forall\, i = 1, \ldots, n \tag{3}$$

The objective of the above model is to maximize the sum of the items' profit in the knapsack, as expressed in Eq. (1). Constraints (2) ensure that the total weight in the knapsack cannot overload. Constraints (3) impose the binary nature of the decision variables, $x_i = 1$ if the item i is selected for the knapsack, otherwise $x_i = 0$.

Because of the computational complexity of solving the knapsack problem, many algorithms have been advanced on it in the past years, which have been classified as exact algorithms such as dynamic programming, back-tracking, branch and bound algorithm, and heuristic algorithms such as ant algorithm, genetic algorithm, simulated annealing algorithm, particle swarm algorithm, tabu search algorithm, etc.

In this paper, we present a new heuristic algorithm called quantum-inspired ant algorithm (QAA) based on the ant algorithm and quantum computing and solve several knapsack problem cases with it. With the purpose of moderating the complexity of solving the knapsack problem with ant algorithm, constraints (2) are treated as a penalty function to the objective function (1), then the objective function $f(x)$ can be described as

* This project was supported by the National Natural Science Foundation of China (70871081) and the Shanghai Leading Academic Discipline Project (S30504).

$$f(x) = \sum_{i=1}^{m} p_i x_i - Pen(x)$$
$$Pen(x) = M \cdot \left| \min\{0, C - \sum_{i=1}^{m} w_i x_i\} \right| \tag{4}$$

where $Pen(x)$ and M are penalty function and penalty parameter respectively.

2. Model formulation

2.1 Quantum algorithm

The quantum algorithm is based on the concepts of qubits and the superposition of quantum mechanics' states. The smallest unit of information is stored in a two-state quantum computer, which is called quantum bit or qubit[2−7]. It may be in state 1, state 0 or any superposition of them and can be represented as

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{5}$$

where α and β are complex numbers that specify the probability amplitudes of the corresponding states. $|\alpha|^2$ explains the probability that the qubit will be in state 0 and $|\beta|^2$ shows the probability that the qubit will be in state 1. Normalization of the state to unity guarantees Eq. (6)

$$|\alpha|^2 + |\beta|^2 = 1 \tag{6}$$

A system with m qubits can present 2^m states at the same time, however, it collapses to a single state[3−7] in the act of observing a quantum state.

An m-qubits representation is defined as

$$\left[\begin{array}{c|c|c|c} \alpha_1 & \alpha_2 & \cdots & \alpha_m \\ \beta_1 & \beta_2 & \cdots & \beta_m \end{array} \right] \tag{7}$$

where $|\alpha_i|^2 + |\beta_i|^2 = 1\,(i = 1, 2, \ldots, m)$ which has the advantage of representing a linear superposition of the states probabilistically.

2.2 Ant colony optimization algorithm

The ant algorithm is a developed population-based approach and has been successfully applied to several NP-hard combinatorial optimization problems. As the name suggests, the ant algorithm has been inspired by the behavior of real ant colonies, in particular, by their foraging behavior[8−9]. Every variables can only be 0 or 1 in the 0-1 knapsack problem discussed in this paper, so we can set 2 ants for every variables (it can also be more than 2 ants)[10], and define η_{ij} as

$$\eta_{ij} = f_j - f_i \tag{8}$$

where the profit $f(x)$ is expressed as Eq. (4), and η_{ij} is the difference between the objective profits f_i and f_j.

The transition probability is defined as

$$P_{ij} = \frac{\tau_j \cdot \eta_{ij}}{\sum\limits_k \tau_k \cdot \eta_{ik}} \tag{9}$$

where τ_j is the neighbor attraction of ant j, updated by Eq. (10)

$$\tau_j^{\text{new}} = (1 - \rho) \cdot \tau_j^{\text{old}} + \sum_k \tau_j^k \tag{10}$$

where ρ is the evaporation rate of the ant's pheromone, set between 0.3 and 0.5, and τ_j^k is increased by positive number Q.

Optimization process is based on ants' tour from their initial states (0 or 1): When $\eta_{ij} > 0$, ant i changes state i to state j by probability p_{ij}, namely the corresponding variable x transforms into $1 - x$; When $\eta_{ij} \leqslant 0$, ant i keeps the current state.

2.3 Quantum-inspired ant algorithm

The QAA generated by merging the ant algorithm and the quantum algorithm maintains the two algorithms' excellences and overcomes their disadvantages. The quantum-inspired ant algorithm can balance between exploration and exploitation compared with that of the basic ant algorithm.

The structure of QAA is described as follows: In the initial step qubits, α_i^0 and β_i^0 $(i = 1, 2, \ldots, m)$ of all q_j^0 $(j = 1, 2, \ldots, n)$ are initialized with $1/\sqrt{2}$, which means all the possible states with the same probability.

$$\left|\varphi_{q_j^0}\right\rangle = \sum_{k=1}^{2^m} \frac{1}{\sqrt{2^m}} \,|X_k\rangle \tag{11}$$

where X_k is the kth state represented by the binary string $(x_1\ x_2\ \ldots\ x_m)$, and $x_i\,(i = 1, 2, \ldots, m)$ is 0 or 1 according to the probability of either $\left|\alpha_i^0\right|^2$ or $\left|\beta_i^0\right|^2$.

It is noted that the performance of QAA can be influenced by the initial value. In each iteration, pheromones and quantum are updated to improve and reach the fitter states of the qubit binary. The whole algorithm can be described as follows.

```
procedure QAA
begin
  nc ← 0;(nc is the number of iterations)
  Set all the parameters and initialize pheromone
  trails and qubits;
  Repeat
    Set each ant on the item's qubit randomly;
    Move each ant by the probabilistic selection rule;
    Pheromones and quantum gates updating;
    Store the best solution among ants;
  Until nc is greater than the predetermined number
of iterations or termination-condition is satisfied;
end
```

The appropriate quantum gates can be designed in compliance with different problems. For example, rotation gates are set as Eq. (12) for the instances of the knapsack problem in this section[4].

$$U(\theta)=\begin{bmatrix}\cos\ \theta & -\sin\ \theta\\ \sin\ \theta & \cos\ \theta\end{bmatrix} \tag{12}$$

where θ is the rotation angle. The best solution among $P(t)$ will be kept until the next step and replace the stored best solution of the last step if it is fitter which makes the qubit converge to the fitter states. The binary solutions $P(t)$ are discarded at the end of the loop.

An m-length qubit string represents a linear superposition of solutions to the problem as shown in Eq. (7). The length of a qubit string is the same as the number of items. The ith item can be selected for the knapsack with probability $|\beta_i|^2$ or $1-|\alpha_i|^2$, thus an m-length binary string can be formed from the qubit string. For every bit in the binary string, we set $x_i=1$ if $r>|\alpha_i|^2$, where r is generated randomly and $r\in[0,1]$. The binary string X_j^t $(j=1,2,\ldots,n)$ of $P(t)$ represents the jth solution to the problem. For notational simplicity, X is used instead of X_j^t in the following. The ith item is selected for the knapsack if $x_i=1$. This procedure can be described as follows.

```
Procedure GetLocal (x)
begin
  i ← 0
  while (i < m) do
  begin
    i ← i + 1
    if random > |α_i|^2 then
      x_i ← 1
    else
      x_i ← 0
  end
end
```

The profit of a binary solution X is evaluated by the profile, and it is used to find the best solution b after q_j is updated, which is updated by the rotation gate $U(\theta)$. The ith qubit value (α_i,β_i) is updated as

$$\begin{bmatrix}\alpha_i'\\ \beta_i'\end{bmatrix}=\begin{bmatrix}\cos\ \theta_i & -\sin\ \theta_i\\ \sin\ \theta_i & \cos\ \theta_i\end{bmatrix}\begin{bmatrix}\alpha_i\\ \beta_i\end{bmatrix} \tag{13}$$

In this paper, Q_i is given as $s(\alpha_i\beta_i)\Delta\theta_i$. The parameters used are shown in Table 1. For example, if $f(x)\geqslant f(b)$, x_i=1 and $b_i=0$, we can set $\Delta\theta_i=0.025\pi$

Table 1 Lookup table of $\Delta\theta_i$

x_i	b_i	$f(x)\geqslant f(b)$	$\Delta\theta_i$	$s(\alpha_i\beta_i)$			
				$\alpha_i\beta_i>0$	$\alpha_i\beta_i<0$	$\alpha_i=0$	$\beta_i=0$
0	0	False	0	0	0	0	0
0	0	True	0	0	0	0	0
0	1	False	0	0	0	0	0
0	1	True	0.05π	-1	$+1$	±1	0
1	0	False	0.01π	-1	$+1$	±1	0
1	0	True	0.025π	$+1$	-1	0	±1
1	1	False	0.005π	$+1$	-1	0	±1
1	1	True	0.025π	$+1$	-1	0	±1

and set $s(\alpha_i\beta_i)$ as $+1, -1$, or 0 according to the condition of $\alpha_i\beta_i$ so as to increase the probability of the state $|1\rangle$. The value of $\Delta\theta_i$ has an effect on the speed of convergence because the solutions may be divergent or convergent to a local optimum prematurely if it is too big. The sign $s(\alpha_i\beta_i)$ determines the direction of convergence to a global optimum. The lookup table can be used as a strategy for convergence. The updating procedure can be described as follows.

procedure update (q)
begin
  $i \leftarrow 0$
  while($i < m$) do
  begin
    $i \leftarrow i + 1$
    determine θ_i with the lookup table
    obtain (α_i', β_i') as:
    $[\alpha_i'\beta_i']^{\mathrm{T}} = U(\theta_i)[\alpha_i\beta_i]^{\mathrm{T}}$
  end
  $q \leftarrow q'$
end

Here f is the profit, $s(\alpha_i\beta_i)$ is the sign of θ_i, and b_i and x_i are the ith bits of the best solution b and the binary solution X respectively.

We can also replace the lookup table by the dynamic adjustment strategy to determine θ_i. In this work, we employ the following dynamic adjustment strategy

$$\begin{gathered} \theta_i = \Delta\theta \times \mathrm{sgn}(Y) \\ Y = (\alpha_i \times \beta_i) \times (f(R_0) - f(R_1)) \times (R_{0i} - R_{1i}) \\ i = 1, 2, \ldots, m \end{gathered} \tag{14}$$

Depending on the given problem, the updating procedure can be implemented in various methods with appropriate quantum gates.

3. Experimental results

We test the instances from the Ref. [4]. In all experiments the strongly correlated sets of data are considered

$$w_i = \text{uniformal} \quad \text{random}[1, 10]$$

$$p_i = w_i + 5$$

The average knapsack capacity is used

$$C = \frac{1}{2}\sum_{i=1}^{m} w_i$$

All the data files are unsorted. The procedure is programmed in Delphi 8 and is run on a personal computer Pentium IV 3.40 GHz for all experiments.

Figure 1 shows the evolution of the mean of the best profits and the mean of average profits of QAA over 25 runs in the three instances. It indicates the global search ability and the convergence ability of QAA.

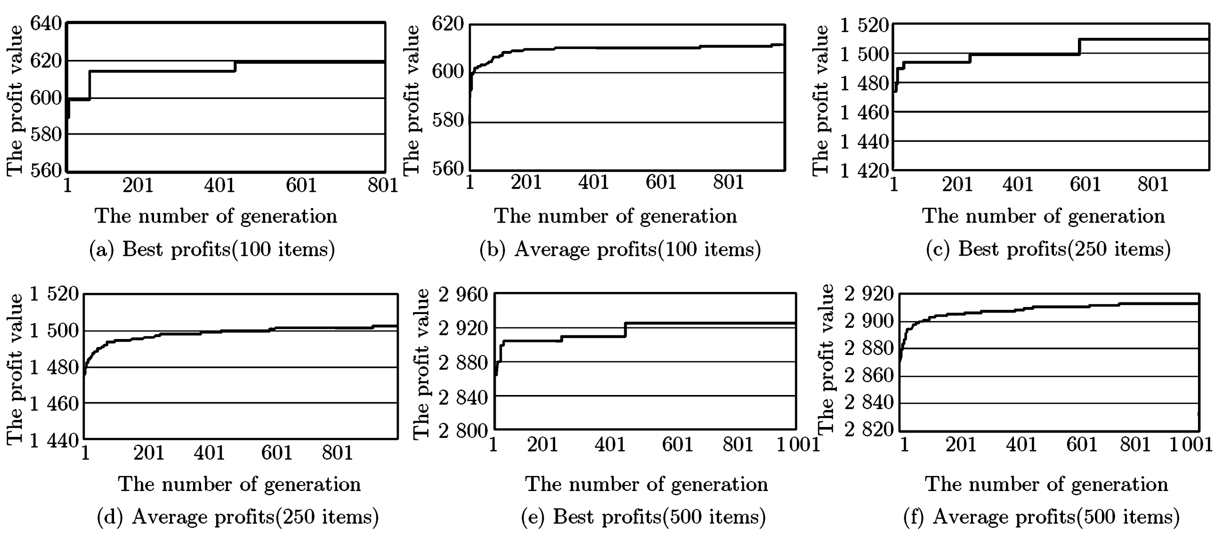

Fig. 1 QAA behavior

Table 2 shows the experimental results of the instances. For the instance 1, QAA yields superior results compared with that of ACO and GQA. Series of experimental results demonstrate the superiority and effectiveness of QAA. By comparing with that of Ref. [4], QAA can surely get better results and is convergent.

Table 2 Comparisons of simulation

Instances	Number of items	Algorithms	Best solution	Worst solution	Average
1	100	QAA	619	609	611.4
		ACO	614	604	607.6
		GQA	613	593	603.9
2	250	QAA	1 509	1 499	1 502.0
		ACO	1 480	1 489	1 485.9
		GQA	1 480	1 444	1 467.1
3	500	QAA	2 925	2 905	2 912.5
		ACO	2 884	2 895	2 886.5
		GQA	2 860	2 813	2 841.3

4. Conclusions

This paper proposes a novel ant algorithm—QAA based on the principles of quantum computing such as concepts of qubits and superposition of states. Owing to its excellent performance, favorable ability of global search and diversity caused by the probabilistic property, QAA can approach better solutions within more reasonable computing time than GQA and ACO. QAA's superior convergence compared with GQA's and ACO's is shown by solving the knapsack problem instances. The good quality solutions validate the effectiveness and applicability of QAA.

References

[1] Maciej M S, Narsingh Deo, Janusz S K. Discrete optimization algorithms. *Englewood Cliffs, New Jersey: Prentice-Hall*, 1983: 118–165.

[2] Hey T. Quantum computing: an introduction. *Journal of Computing & Control Engineering*, 1999, 10 (3): 105–112.

[3] Narayanan A. Quantum computing for beginners. *Proc. of the Congress on Evolutionary Computation*, 1999: 2231–2238.

[4] Han Kuk-Hyun, Kim Jong-Hwan. Genetic quantum algorithm and its application to combinatorial optimization problem. *IEEE Proc. of the Congress on Evolutionary Computation*, 2001: 1354–1360.

[5] Huang jianguo, Liu kewei, Sun Yi, et al. Quantum computing and its applications in signal and information processing. *Systems Engineering and Electronics*, 2003, 25 (7): 800–803. (in Chinese)

[6] Zhang Gexiang, Li Na, Jin Weidong, et al. A novel quantum genetic algorithm and its application. *ACTA Electronica Sinica*, 2004, 32 (3): 476–479.

[7] Gao Hui, Xu Guanghui, Wang Zhenren. An improved quantum evolutionary algorithm and its application to a real distribution routing problemp. *Control Theory & Applications*, 2007, 24 (6): 969–972.

[8] Dorigo M, Maniezzo V, Colorni A. The ant system: optimization by ant colony cooperating agents. *IEEE Trans. on Systems, Man, and Cybernetics*, 1996, 26 (2): 29–41.

[9] Dorigo M, Gambardella L. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Trans. on Evolutionary Computation*, 1997, 1 (1): 53–66.

[10] Ma Liang, Zhu Gang, Ning Aibin. Ant colony optimization algorithm. *Beijing: Science Press*, 2008: 168–178. (in Chinese)

Wang Honggang was born in 1984. He is a master of University of Shanghai for Science and Technology. His research interest is system management.
E-mail: honggang.wang@hotmail.com

Ma Liang was born in 1964. He is a professor and doctoral supervisor of University of Shanghai for Science and Technology. His research interest is intelligent optimization, operational research.
E-mail: maliang@usst.edu.cn

Zhang Huizhen was born in 1979. She is a Ph.D. of University of Shanghai for Science and Technology.

Li Gaoya was born in 1982. She is a master of University of Shanghai for Science and Technology.