Solving the Travelling Salesman Problem by the Program of Ant Colony Algorithm

ZHU Ju-fang
Chinese Academy of Surveying & Mapping, CASM
Chinese University of Mining & Technolegy
Beijing, P.R.C
zhjf_sxau@163.com

LI Qing-yuan
Chinese Academy of Surveying & Mapping, CASM
Beijing, P.R.C
LiQY@casm.ac.cn

***Abstract*—Ant colony algorithm is a novel simulated ecosystem evolutionary algorithm, which is applied to solving complex combinatorial optimization problems. The basic principle and realization about ant colony algorithm are studied in this paper. The algorithm is realized under the Visual C++ compiler environment, and applied to solving the travelling salesman problem (TSP). The result is accordance with the best route solution. This algorithm has practical worth.**

Keywords-ant colony algorithm; combinatorial optimization; travelling salesman problem (TSP); pheromone; solution

I. INTRODUCTION

Ant colony algorithm is an intelligent optimization algorithm first proposed in 1991 by the Italian scholar M. Dorigo, also called ant colony system (ACS) [1] or ant colony optimization (ACO) [2]. It is mainly applied to solving kinds of combinatorial optimization problems, a range of heuristic approaches are developed to deal with these problems. The most promising approaches are: genetic algorithm, tabu-search, neural nets and ant colony system. Finding the best solution is an evolutional process of the candidate colony.

Travelling salesman problem (TSP) [3] is a typical combinatorial optimization problem, its general definition is the following. Given n cities, a salesman sets out from one city, visits all the cities, meanwhile, every city is visited only once, at last, the salesman goes back to the initial city, his way is a closed tour, its length is given by the sum of the lengths of all the arcs of which it is composed. The total number of possible paths grows exponentially with the number of cities, it is difficult to find the shortest path, so that, it is meaningful to find out efficient approximate algorithm. Now, heuristic algorithms are recognized and do well in this field.

Ant colony algorithm [4] is one kind of heuristic algorithms. Due to its advantage of strong robustness and the ability to find optimal solution, ant colony algorithm gives prominence to combinatorial optimization problem. This paper applies ant colony algorithm to the TSP problem to find an optimal solution, and implements it under the VC++ compiler environment by using C++ language.

II. BRIEF DESCRIPTION OF ANT COLONY ALGORITHM

Ant colony algorithm [5] is inspired by the ability of real ants to find the shortest route between food source and nest. Ant colony algorithm copies a natural ant colony's use of pheromone to share information about the routes found [6]. How much pheromone an ant deposit in its path is relative to the length of its path. The shorter the path, the higher the pheromone's concentration, at the same time, the pheromone will volatilize with time. Pheromone affects the path's future, ants have a tendency to select paths marked with more pheromone. The more pheromone, the more ants are apt to wander in that path. We can see that ants which have returned from the nest using shorter routes will arrive more quickly and deposit pheromone more and sooner. This means that the shorter routes will be strongly marked with pheromone and ants will be more likely to use those routes in future. At last, ants can find a shorter path between their nest and food. So that, this ant foraging mechanism is suitable for solving a range of path optimization problems.

III. ANT COLONY ALGORITHM FOR SOLVING TSP

TSP [7] is a typical non-determinstic polynomial complete problem, it is difficult to slove. At first, constructive algorithms is efficient, such as nearest neighbor, nearest merger, nearest insertion, farthest insertion, nearest addition, greedy insertion and so on. Then, heuristic algorithm becomes the mainstream, recently, ant colony algorithm is highlighted.

A. Mathematical model

The mathematical model of ant colony algorithm consists of solution space, objective function and initial solution. Solution space contains all the paths. Objective function reflects the total length of one path. Initial solution brings up randomly, it is the starting point of the algorithm iteration. The optimal solution corresponds with the path that has shorter length.

B. The production and receiving mechanism of new solution

Step1: Parameter initialization: The number of ants is m; The number of cities is n; The quantity of the pheromone one ant released in a cycle is Q; The pheromone's volatilization parameter is ρ; The pheromone affects ants' path selection in a degree, this degree can be expressed by α; The affection degree of path's length to the ants' path selection can be expressed by β; *Iter* max represents the max iteration times.

978-1-4244-4507-3/09/$25.00 ©2009 IEEE

Step2: The m ants are randomly distributed only to the n cities. Every ant visits these n cities, every city must be visited once and only once, when it finished a cycle, one solution brings up. The formation rule of the solution is following:

- The rule of path selection: Based on the concentration of pheromone on the ground, corresponding with probability, ant visits the next city that it never goes through in this cycle. It finishes a cycle after n times.

$$p_{ij}^{k}(t)=\frac{[\tau_{ij}(t)]^{\alpha}\cdot[\eta_{ij}(t)]^{\beta}}{\sum_{s\in allowedk}[\tau_{is}(t)]^{\alpha}\cdot[\eta_{is}(t)]^{\beta}} \quad (1)$$

When ant k transfers from city i to city j in t times, $p_{ij}^{k}(t)$ is the state transfer probability (1), $\tau_{ij}(t)$ means the quantity of pheromone it deposits on the ground, $\eta_{ij}(t)$ is a heuristic function (2).

$$\eta_{ij}(t)=1/d_{ij} \quad (2)$$

In the formula above, d_{ij} is the distance from city i to city j. The less d_{ij}, the greater $\eta_{ij}(t)$ and $p_{ij}^{k}(t)$. The $allowedk$ expresses a set of cities that ant k just has not went through.

- The rule of pheromone update: When one ant finished a cycle after n times, according to the total length, the whole path's pheromone needs to be updated, meanwhile, old pheromone reduces with time. So that, after $t+n$ times, in the path p_{ij}, the pheromone is updated as follow:

$$\tau_{ij}(t+n)=(1-\rho)\cdot\tau_{ij}(t)+\Delta\tau_{ij}(t) \quad (3)$$

$$\Delta\tau_{ij}(t)=\sum\Delta\tau_{ij}^{k}(t) \quad (4)$$

$$\Delta\tau_{ij}^{k}(t)=\frac{Q}{L_k} \quad (5)$$

In the formula above, ρ is the pheromone's volatilization parameter, so $1-\rho$ is its residual coefficient, $\Delta\tau_{ij}^{k}(t)$ expresses pheromone increment in the path p_{ij} in this cycle, $\Delta\tau_{ij}^{k}(t)$ means the quantity of pheromone that the ant k leaved in the path p_{ij} in this cycle, its value is computed by referencing ant-cycle model.

- Calculating the total length L_k that each ant went through, and updating the shortest path: A large number of ants are apt to select the paths that have high pheromone concentration and state transfer probability, At last, nearly all of them go along the short path, but a small number of them don't accord with the pheromone leading machnism, in order to find better solution, they find pathes randomly. In terms of the steps above, when the iteration times is equal to the max iteration times $Iter\max$, it is the time to terminate the cycle and get the shortest path as an optimal solution.

IV. ANT ALGORITHM'S IMPLEMENT UNDER VC

A. *Structure and class defined in the algorithm*

City structure declares the city number and city coordination; Ant class declares current city number, transfer probability, unpassed city array, passed city array, a path's total length and the shortest path; Info class declares pheromone increment array, pheromone array and distance array.

B. *Ant algorithm's main program in C++*

```
While(iter<Itermax)
{
  For(i=0;i<m;i++)//Ants' number is m
  {
    For(j=0;j<n-1;j++)
    //Each ant travelles from start city to other cities
    {
      ants[i].MoveToNextCity();
      //According to the state transfer probability, each
      ant selectes the next city from the unpassed city
      array, and updates the unpassed city array and
      passed city array
    }
  }
  For(i=0;i<m;i++)
  {
    Compare the path lengths of all the ant in this cycle;
    Record the shortest path,and its length;
  }
  //Compare the shortest path length in this cycle with the
  global shortest path length: if the fronter is shorter, then
  the global shortest path is updated by the shortest path
  length in this cycle
  If(curminlen<minlen)
  {
    minlen=curminlen;
    or(j=0;j<n;j++)
    {
      mintour[j]=curmintour[j];
    }
  }
  UpdateInfo();//Update pheromone
  For(i=0;i<m;i++)
    ants[i].Clear();
```

iter++;
}

V. SIMULATION EXAMPLES

The cities from eil51 file are chosen as objects, the number of ants is equal to the city number; Q 's value is 100; The values of these coefficients ρ , α , β , are respectively 0.3, 1, 5; The max iteration times' value is 500. Running the program and calculating, we can see the result path in Fig. 1. In Fig. 2, the blue line and green line respectively show the length of the shortest path and the average length of all the pathes in every iteration process.

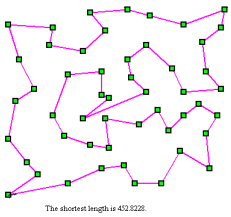

Figure 1. The shortest path

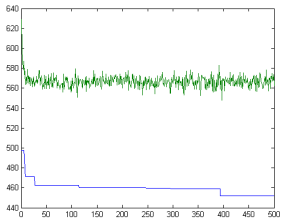

Figure 2. The iteration show

See from the sample, the ant algorithm is efficient to solve the travelling salesman problem, meanwhile, we find that the optimization solution depends on parameters at a certain extent. It is necessary to adjust those parameters for getting the best solution.

TABLE I. THE AFFECTION OF THE NUMBER OF ANTS TO THE SOLUTION

CityNum	51	51	51	51	51	51	51
AntNum	10	20	50	55	100	200	300
PathLength	481	468	466	465	460	457	457

α =1; β =5; ρ =0.95; Itermax=500; Q=100;

TABLE II. THE AFFECTION OF THE PHEROMONE IMPORTANCE PARAMETER

α	0.1	0.2	0.3	0.5	0.7	0.8	0.9	1
PathLength	481	479	467	466	463	461	461	460

β =5; ρ =0.95; Itermax=500; Q=100;AntNum=100;

First of all, the relationship between the number of ants and the number of cities is considered in Table I. When the number of ants is the number of cities one or two times, it is more likely to find shorter path. If the number of ants is small, the solution is not optimal enough. Note that given a large number of ants , the time efficiency of the algorithm will decrease. Secondly, the optimal solution changes with the value of the pheromone importance parameter α from Table Ⅱ. The best value of α is 1. In addition, a lot of experiments testify that the solution flunctuates little with the pheromone volatilization coefficient ρ or heuristic improtance parameter β .

VI. CONCLUSION

This paper successes to apply the ant algorithm to solving the travelling salesman problem under the VC++ compiler environment. The simulation example reflects the ant colony algorithm's good optimization ability. By analizing a large number of experiment data, different parameters' value methods are promised, they are meaningful to the future study of the algorithm. In addition, we also find that, the path length quickly goes down to be the shortest in the first twenty steps, it proves that the ant algorithm has the disadvantage of stagnation, how to solve this problem is worthful to study.

REFERENCES

[1] M. Dorigo and L. Gambardella, "Ant colony system: a cooperative learning approach to the travelling salesman problem." IEEE Transactions on Evolutionary Computation, vol. 1, pp. 53-66, April. 1997.

[2] M. Dorigo, G. Di Caro, and L. M. Gambardella, Ant algorithms for discrete optimization. Artificial Life, 1999,vol. 5, No. 2, pp. 137-172,.

[3] Liu Yong, Kang Lishan, Chen Yuping. Non-numerical parallel algorithms-Genetic Algorithm, 2nd ed, Science Press, 1995, pp. 165-171.

[4] Huang Xiyue, Hu Xiaobing. An application of ant colony algorithm in the K-TSP problem. Computer simulation. 2004.12, Vol. 21, NO. 12, pp. 162-164.

[5] Duan Heibing, The principle and application of ant colony algorithm. Beijing: Science Press, December 2005.

[6] Zhou Wei, Liu Fenlin, Wu Hao and Wang Qingxian. Simple simulation analysis of ant colony algorithm. Control and decision, 2003. 5, Vol. 18,No. 3, pp. 317-323.

[7] Cai Xiaochen,Qi Yuxing. The study of the modern optimization algorithms about the travelling salesman problem(TSP). Warships Electronic Engineering, 2008, No. 4, pp. 114-117.