Available online at www.sciencedirect.com

SciVerse ScienceDirect

Energy Procedia 17 (2012) 319 – 325

Energy Procedia

2012 International Conference on Future Electrical Power and Energy Systems

Application of an Improved Ant Colony Optimization on Generalized Traveling Salesman Problem

Kan Jun-man,Zhang Yi

Department of Computer Science,Jilin Business and Technology College,Changchun 130062, China

Abstract

In this paper, we present an improved ant colony optimization (ACO) and we use it to solve the generalized traveling salesman problem (GTSP). We design a novel optimized implementation approach to reduce the processing costs involved with routing of ants in the conventional ACO, and we also improved the performance of the ACO by using individual variation strategy. Simulation results show the speed and convergence of the ACO can be enhanced greatly, and we also get the best results in some instance of GTSP.

Keywords: ant colony optimazation; Generalized Traveling Salesman Problem; route optimazation

1.Introduction

The generalized traveling salesman problem (GTSP) is an extension of the well-known traveling salesman problem. GTSP is a very important combinatorial optimization problem and is known to be NP-hard. In the GTSP, the set of nodes is divided into clusters; the objective is to find a minimum-cost tour passing through one node from each cluster. Many applications of the GTSP exist in many fields. But researches still did not pay enough attention to GTSP specific local search and mostly use simple TSP heuristics with basic adaptations for GTSP [1].

M. Dorige presented the Ant Colony Optimization (ACO) in 1991[2], some important strategies such as positive feedback and hidden parallel were proposed. By using positive feedback strategy, the ACO can find the better result through parallel pheromone exchanging between ants. And by using hidden parallel strategy, jumping into the optimal solution can be prevented and the ACO is also very e□cient. The researches and applications on ACO algorithm have made great progresses in the past years. Many scholars presented some efficient methods to solve these problems [3-13]. However, it still has some basic problems that have only been partially solved, such as searching time is too long and it may easily jump into local optimal solution [10]

 doi:10.1016/j.egypro.2012.02.101

In this paper, two improvements on Ant Colony Optimization (ACO) algorithm are presented. We design an individual variation strategy to improve the convergence of the ACO.

In the proposed algorithm, the ants use different strategies to get the better results; and we also design a novel optimized implementation approach to reduce the processing costs involved with routing of ants in the conventional ACO. The results of the simulated experiments show that the improved algorithm not only reduces the number of routing in the ACO but also surpasses existing algorithms in performance. More precisely, the simulations show that the stability and the speed of convergence of the improved ACO algorithm can be enhanced greatly. All the instances used in experiments are from TSPLIB (ftp://ftp.iwr.uni-heidelberg.de/iwr/comopt/soft/TSPLIB95/TSPLIB.html). Fischetti et al. [14] provided a partition algorithm to convert the instances use in TSP to those which could be used in the GTSP. Simulation results show that our results are always better than the results provided in ref. [15] [16].

The rest of the paper is organized as follows. Section 2 introduces the GTSP problem and ant colony optimization. In Section 3, the improvements of the ant colony algorithm are presented, and we provided the codes of the improved algorithm. Some simulation results are provided in Section 5. The paper conclusion and future research are presented in Section 6.

2.The Ant Colony Optimization for GTSP

2.1.Ant Colony Optimization (ACO)

Ant colony algorithm can be described briefly as follow. As the typical case, traveling salesman problem (TSP) is to be used to show the ACO algrothm generally. Let $V = \{1, \ldots, n\}$ be a set of cities, $A = \{(i,j) : i,j \in V\}$ be the edge set, and $d(i,j) = d(j,i)$ be a cost measure associated with edge$(i,j) \in A$. The TSP is the problem of finding a minimal cost closed tour that visits each city once.

The object function of the traveling salesman problem is:

$$Min \qquad D = \sum_{i=1}^{n-1} d(i,i+1) + d(n,1)$$

where $d(i,j)$ $(i,j=1,2,\ldots,n)$ is the tour length form city i to city j.

Initially put m ants into n cities randomly and each edge has an initial pheromone τ_{ij}（0）between two cities. The first element of tabu table of each ant is initialized with the initial city of each ant, and then each ant begins to select a tour to be next city. According to the following probability function, ants select the next city.

$$p_{ij}^{k} = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{t \in allowed_t}[\tau_{it}(t)]^{\alpha}[\eta_{it}]^{\beta}} & j \in allowed_t \\ 0, & other. \end{cases} \tag{1}$$

where pkij is the probability with which ant k chooses to move from city i to city j, τ_{ij} is the pheromone, ηij=1/dij is the inverse of the distance, α and β are the parameters which determine the relative importance of pheromone versus distance (α,β>0). In this way we favor the choice of edges which are shorter and have a greater amount of pheromone. After n times circles, all ants' tabu have been filled, at this time, we compute every ant's tour length to find the shortest one and save and record it in order to change pheromones. Repeat this until the stop condition is reached.

In ant system, the global updating rule is implemented as follows. Once all ants have built their tours, pheromone is updated on all edges according to

$$\Delta\tau_{ij}(t+n) = \rho\tau_{ij}(t) + \Delta\tau_{ij} \tag{2}$$

$$\tau_{ij} = \sum_{k=1}^{m} \Delta \tau_{ij}^{k} \quad . \tag{3}$$

Where $\Delta\tau$ ij is the sum of new increased pheromones at this edge is degree of dissipate for pheromones，$\Delta\tau$ kij is the amount of pheromones of the kth ant at its edge between *t* and *t*+*n*. Calculate $\Delta\tau_{ij}^{k}$ using the following equation

$$\Delta\tau_{ij}^{k} = \begin{cases} \dfrac{Q}{L_K}, & \text{if k - thantusesedge(i,j)initstour(betweenitmetandt+n)} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Where *Q* is a const，L_k is the length of the tour performed by ant *k*.

Because the ACO algorithm is a typical random search algorithm, variables of ACO are often set by trial and error [11]. The chief variables are α and β which are related to pheromones and distance respectively. Both variables have effects on the quality of solutions. Variable ρ involves in the speed of dissipating of pheromones. The parameter τ_{ij} is also a key value. The way of changing pheromones in ACO has some influences on efficiencies and the ability of getting results of algorithm. Dorigo gave three models: ant-cycle system, ant-quantity system and ant-density system [11]. The difference of the three models is the function (4). In this paper, we use ant-cycle system model because it changes pheromones using whole information and it has the best result on TSP program. The complexity of this algorithm is O(NC•n2), where NC is the maximum cycle times.

2.2.ACO for the GTSP

The GTSP is a variation of the well-known TSP in which the set of nodes is divided into clusters; the objective is to find a minimum-cost tour passing through one node from each cluster. In GTSP, we are given n cities into m groups and we are required to find a minimum length tour that includes exactly one city from each group. Generally, we can mostly use simple TSP heuristics with basic adaptations for GTSP, but these convertions lead to the increasing of dimensions of the instance.

We use ACO to solve the GTSP without converting the instance.First, we introuduce the GTSP with the mathmatic model [15]: Let G= (*V*, *E*, *W*) be a completely weighted graph, in which

V= {v_1, v_2... v_n} (n≥3), *E*= {e_{ij} ｜ v_i, $v_j \in V$}, and $W = \{w_{ij} \geq 0 \; and \quad w_{ii} = 0, \forall i, j \in N(n)\}$ are vertex set, edge set and cost set, respectively. The vertex set V is partitioned into m possibly intersecting groups V_1, V_2…, V_m with $|V_j| \geq 1 \quad and \quad V = \cup_j^m V_j$ .The special Hamiltonian cycle is required to pass through all of the groups, but not all of the vertices. At present, there are two kinds of GTSP [15]: (1) the cycle passes exactly one vertex in each group and (2) the cycle passes at least one vertex in each group.

In this paper, we add the parameter allowedt in the algorithm. The parameter allowedt denote the city in the group has not been visited by ant t. By using allowedt, the algorithm avoids visiting the cities in the same group.

$$allowed_t = \{c \mid c \in V \quad and \quad c \notin G, \forall G \in tabu_t\} \tag{5}$$

3.The Improved ACO and Its Implement

3.1.Individual Variation & Routing Strategies（IVRS）

Two improvements on Ant Colony Optimization (ACO) algorithm are presented in this paper [7], we named it IVRS algorithm. The first is a novel optimized implementing approach which is designed to

reduce the processing costs involved with routing of ants in the ACO. In this approach, only the ant who firstly finds the shortest route will travel all the cities, while others will stop when this ant finished travelling; Second, the individual variation is introduced to the ACO, which enables the ants have different route strategies. In this model, we adjust the routing strategy of the ant who worked better, that is to enhance the impact of pheromones in the route of this ant. As a result, the quality of solutions of ACO algorithm has also been enhanced. Simulation results demonstrate that our results are always better than or equal to the solutions in the TSPLIB.

In ACO, the parameters α and β set in (1) are static and all ants share the same value in the running of ACO. Dorigo [11] suggested that $\alpha = 1$ and $\beta = 5$ is a proper setting for many situations. For α and β are static, the relative importance of pheromone and distance on routinghold fixed according to (1). But it is not proper to use the same parameters in different phases of ACO. Initially, the distance has stronger impact on routing, because ants have little knowledge about the better path. After the algorithm runs for a long time, the impact of pheromone must be reinforced, because more information about the better path is stored in pheromone. From this observation, we introduce the Individual Variation to ACO. In this approach, each ant has different parameters which may change in the running of algorithm, which means that different ant has different routing bias. In the real world, individual variation makes the colony more robust. In IVRS, solutions worked out by ant colonies with different routing strategies are better than those of single strategy colonies. In experiments, we used an experiential rule of adjusting parameters, under which the algorithm finds the best solution faster in average: Increase the value of α and β at the same time decrease the value of until α=5 and β=1, when an ant becomes the winner at the first time in some cycle. In experiment, we find, for large-scale maps in which the number of cities is huge, the modified ACO converges not very fast. Therefore, the 2-opt [12] is employed in the end of each cycle of ACO to perform local optimization, and both the convergence speed and the quality of solution are enhanced.

In IVRS, we also optimized the ACO by reducing both the executing frequency and time complexity of routing algorithm. The improvements are given as follows.First, we design an approach in order to reduce the time complexity of routing: In contrast to select the next city from all the cities not visited, the set of candidates is limited to the nearest c cities. By this optimization, the time complexity is reduced greatly. The experiment results show that this improvement performed faster in large-scale TSP.Second, we introduce a method to reduce the routing frequency in each cycle of ACO. In one cycle of the algorithm, the ant that turns back to the source first wins and other ants stop routing and this cycle is stopped. In each step of a cycle, the ant whose path length is the shortest is chosen to route and moves to the next city and increases its path length. This procedure repeats until an ant turns back to the city where it started. Finally, a variable Lmin is employed to record the current minimum solution.

3.2.Pseudo-code of IVRS

In the pseudo-code, NC is the counter of cycles. Lmin is the current best solution. For ant k, ANT[k].start denote the starting city, ANT[k].current denote the current city ant i is in, ANT [i].tour denote the length of current tour, ANT [k].tabu denote the tabu table in which the cities had been visited, ANT [k].allowed denote the allowed table in which the cities in the groups haven't be visited. ANT[k].α and ANT[k].β denote the parameters α and β of ant k.

Step1.
NC ← 0; L_{min}← ∞
***for* each edge (i, j) do**
$\tau_{i,j}$← c; $\Delta\tau_{i,j}$← 0; α←1,β←5,
end for
Step2.
if NC < NCM AX and no stagnation behavior ***then***

```
      for k ← 1 to m do
        ANT[k].tabu ← ∅ ∪ANT [k].start
        ANT[k].tour ← 0
      end for
    else
      Print shortest tour
      Stop.
    end if
Step3.
    Search for ant min whose tour is the minimum.
    if ANT [min].tour ≥ L_min                    then
      GOTO step 2
    end if
    if ant_min returns to the city where it started then
       GOTO step 5
    end if
    j ← Route(min)
    ANT[min].tour ←
    ANT [min].tour + distance(ANT[min].current, j)
    ANT[min].current ← j
    ANT[min].tabu ← ANT [min].tabu ∪j
    Compute ANT[min].allowed according to Eq.(5)
Step4.
    if ant min had visited all the cities then
      ANT [min].tour ←
      ANT [min].tour + distance(j,ANT [min].start)`
      GOTO step 3
    end if
Step5.
    if ANT [min].tour < L_min then
      L_min← ANT [min].tour
    Update the pheromone of the path of ant min according to Eq. (3)
    if ANT[min]. α≠5 and ANT[min]. β≠1 then
    ANT[min]. α+ +; ANT [min]. β--
    end if
    end if
Step6.
    for each edge (i, j) do
    Compute τi,j(t + n) according to Eq.(2)
    Δτ_i,j ← 0.
    end for
    t ← t + n; NC ← NC + 1
    GOTO step 2
```

4.Experiments and Computational Results

In order to testify our conclusion, we do a series of experiments. First, we compare the performance of the ACO algorithm and the IVRS algorithm in terms of performance, speed of convergence and quality of solution by several experiments.In this approch, we modify the algorithm (IVRS) for some instance in

TSPLIB. We also compare the solutions in GTSP of IVRS and other existing algorithms'result. Experiments are conducted on a 2.1GHz Celeron PC with 1G bytes memory under Windows 2003 using the VC6.0 compiler. All the maps used in experiments are from TSPLIB (ftp://ftp.iwr.uni-heidelberg.de/iwr/comopt/soft/TSPLIB95/TSPLIB.html).

First, we compare the solutions and the running time of basic ACO and improved algorithm IRVS, results show in Table 1.

In order to testify our improved algorithm in GTSP, we use the partition algorithm to convert the instances used in TSP to those which could be used in the GTSP [14]. All of the instances are computed by IRVA Simulation results show that our results are always better than or equal to the results provided in ref. [15], results show in Table 2.

5.Conclusion

Two improvements on Ant Colony Optimization (ACO) algorithm are presented in this paper. The first is a novel optimized implementing approach which is designed to reduce the processing costs involved with routing of ants in the ACO; second, the individual variation is introduced to the ACO, which enables the ants have different route strategies. In this model, we adjust the routing strategy of the ant who worked better, that is to enhance the impact of pheromones in the route of this ant. As a result, the quality of solutions of ACO algorithm has also been enhanced. Simulation results demonstrate that our results are always better than or equal to the solutions in the TSPLIB. We also apply the improved algorithm to GTSP, and get the better results.

References

[1] Lien, Y., Ma, E., Wah, B.W.S.: Transformation of the generalized traveling salesman problem into the standard traveling salesman problem, Information Science. 74 (1993) 177–189

[2] Dorigo M, Gambardella LM. A study of some properties of ant-Q. In: Voigt H-M, Ebeling W, Rechenberg I, Schwefel H-S, eds. Proceedings of the PPSN 44th International Conference on Parallel Problem Solving from Nature. Berlin: Springer-Verlag, 1996. 656~665.

[3] Gambardella LM, Dorigo M. An ant colony system hybridized with a new local search for the sequential ordering problem. INFORMS Journal on Computing, 2000, 12 (3): 237~255.

[4] Parpinelli RS, Lopes HS, Freitas AA. Data mining with an ant colony optimization algorithm. IEEE Trans. on Evolutionary Computation, 2002,6(4): 321~328.

[5] Wu QH, Zhang JH, Xu XH. An ant colony algorithm with mutation features. Journal of Computer Research & Development, 1999, 36 (10): 1240~1245 .

[6] Chen L, Shen J, Qin L. An adaptive ant colony algorithm based on equilibrium of distribution. Journal of Software, 2003, 14 (8): 1379~1387 (in Chinese).

[7] Yi Zhang, Zhi-Li Pei, Jinhui Yang, Yanchun Liang: An Improved Ant Colony Optimization Algorithm Based on Route Optimization and Its Applications in Traveling Salesman Problem. BIBE 2007: 693-698

[8] Wu B, Shi ZZ. An ant colony algorithm based partition algorithm for TSP. Chinese Journal of Computers, 2001, 24 (12): 1328~1333 (in Chinese).

[9] Zhang JH, GAO QS, Xu XH. A self-adaptive ant colony algorithm. Control Theory and Applications, 2000, 17 (1): 1~3 (in Chinese).

[10] Colorni A, Dorigo M. Heuristics from nature for hard combinatorial optimization problems. International Trans Operational Research, 1996, 3 (1): 1~21.

[11] Dorigo M, Vittorio Maniezzo, Alberto Colorni. The Ant System: optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics, Part B, 1996, 26 (1): 1~13

[12] Huang L, Zhou CG, Wang KP，Hybrid ant colony algorithm for traveling salesman problem, Progress in Natural Science, 2003, 13 (4): 295~299.

[13] Chengming Qi An Ant Colony System Hybridized with Randomized Algorithm for TSP Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD2007. Eighth ACIS International Conference on Volume 3, Issue , July 30 2007-Aug. 1 2007 Page(s):461 - 465

[14] Fischetti, M., Salazar, J.J., Toth, P. Branch-and-cut algorithm for the symmetric generalized traveling salesman problem, Operations Research. 45(3) (1997) 378-394

[15] Han Huang, Xiaowei Yang, et. Hybrid chromosome genetic algorithm for generalized traveling salesman problems. 1st International Conference on Natural Computation (ICNC 2005), Changsha, China, 2005. Lecture Notes in Computer Science, 2005, Vol. 3612: 137–140.

[16] 16. Chunguo Wu, ET. A generalized chromosome genetic algorithm for generalized traveling salesman problems and its applications for machining. Physical Review E, 2004, (70): 1-13.

TABLE 1. COMPARISON OF THE RUNNING TIME (S) AND SOLUTION OF ACO AND IRVS IN SOLVING DIFFERENT TSP MAPS

Instance	TSPLIB solution	IRVS+2opt			ACO+2opt		
		Best solution	*Average solution*	*Run time/s*	*Best solution*	*Average solution*	*Run time/s*
Eil51	429.9833	429.1179	431.1054	11.32	431.2568	439.2571	18.35
Berlin52	7544.3659	7544.3659	7547.2365	8.23	7549.5264	7556.5897	14.69
KroA100	21285.4432	21309.429	21498.6153	17.83	22006.3968	23441.804	23.06
Eil101	629	631.2879	648.6725	9.45	654.4980	672.3756	13.58
D198	15780	15842.523	15972.473	21.47	16054.4731	16252.928	28.38

TABLE 2. COMPARISON OF SOLUTIONS FOR BENCHMARK TEST PROBLEM

Instance	IRVS+2opt		HCGA		GCGA	
	Best solution	*Run time/s*	*Best solution*	*Run time/s*	*Best solution*	*Run time/s*
31Pr152	51573	1.89	51573	0.89	51586	4.31
40KroB200	13009	5.97	13113	8.00	13120	6.78
64Lin318	20719	14.12	20788	18.34	20977	16.81
88Pr439	60184	7.24	60184	10.87	61373	6.89