TELKOMNIKA, Vol. 11, No. 9, September 2013, pp. 5308~5315
ISSN: 2302-4046

Vehicle Routing Optimization in Logistics Distribution using Hybrid Ant Colony Algorithm

Chengming Qi
Beijing Union University, Beijing, China
e-mail: qicm@163.com

Abstract

The Vehicle Routing Problem (VRP) is an important management problem in the field of physical distribution and logistics. Good vehicle routing can not only increase the profit of logistics but also make logistics management more scientific. The Capacitated Vehicle Routing Problem (CVRP) constrained by the capacity of a vehicle is the extension of VRP. Our research applies a two-phase algorithm to address CVRP. It takes the advantages of Simulated Annealing (SA) and ant colony optimization for solving the capacitated vehicle routing problem. In the first phase of proposed algorithm, simulated annealing provides a good initial solution for ant colony optimization. In the second phase, Iterative Local Search (ILS) method is employed to seeking the close-to-optimal solution in local scope based on the capacity of the vehicle. Experimental results show that the proposed algorithm is superior to original ant colony optimization and simulated annealing separately reported on partial benchmark problems.

Keywords*: ant colony system, capacitated vehicle routing problem, iterative local search, simulated annealing*

1. Introduction

The Ant Colony system (ACS) [1] is a meta-heuristic which is inspired by the trail following behavior of real ant colonies. One of the most efficient Ant Colony Optimization (ACO) based implementations is based on ACS, which introduced a particular pheromone trail updating procedure for intensifying the search in the neighborhood of the best computed solution.Recently, some improved ant colony algorithm was used to solve the problems of error compensation for a high-precision numeral control machining system [2] and fuzzy controller problem of the autonomous parking [3].

Vehicle Routing Problem (VRP) is an NP-complete problem and has important practical value. More and more of the VRP research is attention because many real-life problems can be attributed to VRP. The capacitated vehicle routing problem (CVRP) is one of the elemental problems in supply chain management. The objective of CVRP is to provide each vehicle with a sequence of delivers so that all customers are serviced, and the traveling cost of vehicles is minimized [4–7]. It is hard to solve this problem directly when the number of customers is large [8, 9]. Silvia Mazzeo et al. [10] have improved Capacitated Vehicle Routing Problem (CVRP) by means of an ACO algorithm. A deoxyribonucleic acid computing and modified Adleman-Lipton model accelerates the search on large nodes CVRP in a decentralized model [11].

Local search is a generally applicable approach that can be used to find approximate solutions to hard optimization problems. The basic idea is to start from an initial solution and to search for successive improvements by examining neighboring solutions. In this paper, we apply iterated local search (ILS) [12] in second stage. ILS is a very simple and powerful meta-heuristic that has proved to be the best performing approximation algorithms for the well known Traveling Salesman Problem [12].

Nevertheless, most solutions obtained are worse than the best solution found so far. In this paper, we create a hybrid algorithm (HACS-SA) that combines the strengths of both search heuristics. It has both the advantage of SA, the ability to find feasible solutions, and that of ACO, the ability to avoid premature convergence and then search over the subspace. Finally our HACS-SA is tested by partial benchmark problems and compared the performance with original ant colony optimization and simulated annealing separately.

Received April 7, 2013; Revised June 10, 2013; Accepted June 21, 2013

The paper is organized as follows. Section 2 introduces the CVRP and the solution construction mechanism used by the ACS. Section 3 presents hybrid HACS-SA algorithm. In Section 4 we provide experimental results of HACS-SA on a set of benchmark problems and compare to the performance of ACS and SA. We conclude in Section 5 with a brief summary of the contributions of this paper.

2. CVRP and ACS

2.1. Capacity Vehicle Routing Problem

The vehicle routing problem deals with a single depot distribution system servicing a set of customers using a homogeneous fleet of vehicles. It is a very complicated combinatorial optimization problem that has been worked on since the late fifties, because of its central meaning in distribution management.

The vehicle routing problem can be described as follows [13]: *n* customers must be served from a (unique) depot. Each customer *i* ask for a quantity q_i of goods. A fleet of *v* vehicles, each vehicle *a* with a capacity Q_a, is available to deliver goods. A service time s_i is associated with each customer. It represents the time required to service him/her. Therefore, a VRP solution is a collection of tours.

CVRP is the basic version of the VRP where the vehicles have limited carrying capacity of the goods that must be delivered. In other words, CVRP is similar to VRP, but CVRP has an additional constraint that every vehicle must have uniform capacity of a single commodity. Since CVRP contains one or more TSP as subproblems, it is more difficult to solve than TSP.

The classic CVRP can be described as follows: *N* customers geographically dispersed in a planar region must be served from a unique depot. Each customer asks for a quantity q_i *(i=1, 2,…, n)* of goods. The transport cost from node *i* to node *j* is c_{ij}. *m* vehicles with a fixed capacity *Q* are available to deliver the goods stored in the depot. Each customer must be visited just once by only one vehicle. The objective of the problem is minimizing the total cost of all routes without violating the individual capacity of each vehicle. The depot is denoted by *i=0*. The model can be written as follows:

$$min\sum_{k=1}^{m}\sum_{j=0,j\neq i}^{n}\sum_{i=0}^{n}c_{ij}x_{ij}^{k} \quad (1)$$

$$x_{ij}^{k}=\begin{cases}1 & if\ vehicle\ k\ goes\ from\ i\ to\ j\\ 0 & otherwise\end{cases} \quad (2)$$

$$s.t.\sum_{k=1}^{m}\sum_{j=1}^{n}x_{ij}^{k}\leq m \quad i=0$$

$$\sum_{j=1}^{n}x_{ij}^{k}=\sum_{j=1}^{n}x_{ji}^{k}\leq 1 \quad i=0,k\in\{1,2,...,m\} \quad (3)$$

$$\sum_{j=1}^{m}\sum_{j=0,j\neq i}^{n}x_{ij}^{k}=1 \quad i\in\{1,2,...,n\} \quad (4)$$

$$\sum_{j=1}^{m}\sum_{i=0,i\neq j}^{n}x_{ij}^{k}=1 \quad j\in\{1,2,...,n\} \quad (5)$$

$$\sum_{i=1}^{n}d_i\sum_{j=0,j\neq i}^{n}x_{ij}^{k}\leq Q \quad j\in\{1,2,...,n\} \quad (6)$$

The objective function Equation (1) is minimizing the total distance traveled. Constraint Equation (2) assures the number of vehicles originating from the depot is not more than *m*.

Constraint Equation (3) states that each of the *k* vehicles has to leave and go back to the depot. Constraints Equation (4) and Equation (5) assure that each customer is visited exactly once. Constraint Equation (6) is the capacity constraints.

2.2. Ant Colony System

ACS is based on the way real ant colonies behave to find the shortest path between their nest and food sources. It simulates the described behavior of real ants to solve combinatorial optimization problems with artificial ants. To solve the VRP, the artificial ants construct vehicle routes by successively choosing cities to visit, until each city has been visited. Whenever the choice of another city would lead to an infeasible solution for reasons of vehicle capacity or total route length, the depot is chosen and a new tour is started.

This heuristic uses a population of *m* agents which construct solutions step by step. When all the ants have constructed their tour, the best solution is rewarded so as to encourage the identification of ever better solutions in the next cycles.

Construction of vehicle routes: This process is responsible for the construction of new solutions. This is achieved using probabilistic stepwise solution construction. ACS goal is to find a shortest tour. In ACS *m* ants build tours in parallel, where *m* is a parameter. Each ant is randomly assigned to a starting node and has to build a solution, that is, a complete tour. A tour is built node by node: each ant iteratively adds new nodes until all nodes have been visited. When ant *k* is located in node *i*, it chooses the next node *j* probabilistically in the set of feasible nodes N_i^k (i.e., the set of nodes that still have to be visited). The probabilistic rule used to construct a tour is the following: with probability q_0 a node with the highest $[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}, j \in N_i^k$ is chosen, while with probability $(1-q_0)$ the node *j* is chosen with a probability p_{ij} proportional to $[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}, j \in N_i^k$.

With $\Omega = \{v_j \in V | v_j \text{ is feasible to be visited}\} \cup \{v_0\}$, city v_j is selected to be visited after city v_i according to a *random-proportional rule* [14] that can be stated as follows:

$$p_{ij} = \begin{cases} \dfrac{[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}}{\sum_{h\in\Omega}[\tau_{ij}]^{\alpha}[\eta_{ij}]^{\beta}} & if \quad v_j \in \Omega \\ 0 & otherwise \end{cases} \qquad (7)$$

Where:

$$\eta_{ij} = 1/\max(1,(Dist_{ij} - IN_j))$$
$$Dist_{ij} = (l_j - f_i) * (\max(f_i + d_{ij}, b_j) - f_i)$$

And IN_j represents the number of customer C_j which has not been selected.

***Pheromone trail update*:** Once solutions have been evaluated, they can influence the pheromone matrix through a pheromone update process. After an artificial ant *k* has constructed a feasible solution, the pheromone trails are laid depending on the objective value L_k. For each arc (v_i, v_j) that was used by ant *k*, the pheromone trail is increased by $\Delta\tau_{ij}^k = 1/L_k$. In addition to that, all arcs belonging to the so far best solution (objective value L^*) are emphasized as if σ ants, so-called *elitist* ants had used them. One elitist ant increases the trail intensity by an amount $\Delta\tau_{ij^*}$ that is equal to $1/L^*$ if arc (v_i, v_j) belongs to the so far best solution, and zero otherwise. Furthermore, part of the existing pheromone trails evaporates (ρ is the trail persistence) [15]. Thus, the trail intensities are updated according to the following:

$$\tau_{ij}^{new} = \rho\tau_{ij}^{old} + \sum_{k=1}^{m}\Delta\tau_{ij}^k + \sigma\Delta\tau_{ij}^* \qquad (8)$$

Wwhere *m* is the number of artificial ants.

3. Hybrid Algorithm HACS-SA

3.1. Applied SA in the First Phase

Metropolis et al. introduced the original concept of simulated annealing (SA) [16]. Kirkpatrick et al. employed SA in solving the problems of combinatorial optimization [17]. Recently, Li et al. [18] combined ACO and SA algorithm to address the dual resource constrained job shop scheduling problem.

SA is a local search technique that has been successfully applied to many NP-hard problems. It makes use of search strategies where cost-deteriorating neighborhood solutions may possibly be accepted as candidates in the search process. For SA, it randomly generates the new solution by changing the current solution and uses an energy function in accepting worse-fitness solutions so as to possibly move out of the local optimum. The Boltzmann distribution is used to decide whether to accept worse fitness solutions or not, and it is defined as [19]:

$$P(\Delta E) = e^{(-\Delta E/KT)} \tag{9}$$

Where P(ΔE) is the Boltzmann probability, ΔE is the difference between the fitness of new solution and the original solution, *K* is the Boltzmann constant, *T* is the controlled temperature, and $e^{(-\Delta E/KT)}$ is the Boltzmann factor.

The *P* function is usually chosen so that the probability of accepting a move decreases when the difference*ΔE* increases-that is, small uphill moves are more likely than large ones. However, this requirement is not strictly necessary, provided that the above requirements are met.

In SA, based on the profile of the search path, we use an adaptive cooling schedule that adjusts the temperature dynamically. Such adjustments could be enhancing the possibility of reheating. The new generated solution is regarded as the next solution only when the value of $e^{(-\Delta E/KT)}$ is greater than a random value generated from a uniform distribution in the interval of [0, 1]. Thus, the solution is always updated. When the new solution is not better than its ancestor, the solution may still take place of its ancestor in a random manner.

3.2. Applied ILS in the Second Phase

The essence of the iterated local search meta-heuristic can be given in a nut-shell: one *iteratively* builds a sequence of solutions generated by the embedded heuristic, leading to far better solutions than if one were to use repeated random trials of that heuristic[20]. Many authors have lead to many different names for iterated local search like *iterated descent* [21, 22], *large-step Markov chains* [12], etc. But, there are two main points that make an algorithm an iterated local search: (i) there must be a single chain that is being followed (this then excludes population-based algorithms); (ii) the search for better solutions occurs in a reduced space defined by the output of a black-box heuristic. In practice, local search has been the most frequently used embedded heuristic, but in fact any optimizer can be used, be-it deterministic or not.

ILS is a very simple and powerful stochastic local search method that has proved to be among the best performing approximation algorithms for the well-known Traveling Salesman Problem (TSP) [23] and a number of other problems [24]. The essential idea of ILS is to perform a biased, randomized walk in the space of locally optimal solutions instead of sampling the space of all possible candidate solutions. This walk is build by iteratively applying first a perturbation to a locally optimal solution, then applying a local search algorithm, and finally using an acceptance criterion which determines to which locally optimal solution the next perturbation is applied.

To apply an ILS algorithm, basically three procedures have to be specified. Given the current s_0, these are a procedure *Perturbation*, that perturbs the current solution *s* leading to same intermediate solution *s'*, a procedure *LocalSearch* that takes *s'* to a local optimum *s''*, and an *AcceptanceCriterion* that decides from which solution the next perturbation step is applied [25].

3.3. The Proposed Algorithm

In HACS-SA, Equation (7) and Equation (8) are used to construct solutions and update pheromone trails. In implementation, the value of τ_{min} is considered as $\tau_{min} = \frac{1}{2 \square N \square L_{best}}$ where N is the number of customer [26]. After updating pheromone trails, the Iterative local search is performed to find the best solution. The procedure of HACS-SA can be described as following:

Apply SA to generate the initial best solution:

Set parameters: *T,γ,m, M.*
If(random() < P(ΔE))
S_{new}:=S_{old}
While (not matched for the termination condition) do

Place each ant on a randomly selected node

Construct CVRP's route by Eq. (7)
//Apply ILS to Improve the CVRP solution

Generate some initial solution s_0
s = LocalSearch(s_0)
repeat{

s' = Perturbation(s,history)

s''= LocalSearch(s')

s = AcceptanceCriterion(s, s'',history)

}until termination condition met

Update the pheromone matrix by Eq. (8)

End

4. Experimental Results

In order to verify the effectiveness of HACS-SA, 12 instances of CVRP benchmark problems are selected from Augerat Set A (instances A32k5, A54k7, A60k9, A69k9 and A80k10), Augerat Set B (instances B57k7, B63k10 and B78k10) and Christofides and Eilon (instances E76k7, E76k8, E76k10 and E76k14). These include the best-known solutions to each problem. These problems range from 32 customers to 80 customers and from 5 vehicles to 14 vehicles for the solution. For each instance of the datasets, the number of customers is given by the first number on the instance name. The main difference between these sets of problems is their tightness (the ratio between demand and capacity) and the location of customers.

Experiments were run on a Pentium IV, 2GB of RAM, 2.8 GHz processor. Solutions are then averaged for each problem type and the result is reported in Table 1. We used n=15 artificial ants and set α= 1, q_0 = 0.8, β = 2 and ρ= 0.1. For all problems maximum iteration times are m=*30*.

It is noted that the parameters of ACS and SA are set as the proposed algorithm. Furthermore, we stop these algorithms after m=*30* continuous iterations if no improved solutions are found.

For large-scale benchmark problems, these instances are kelly01~kelly20 taken from Toth and Vigo [27]. Each instance is solved 15 times, and the best one among 15 runs is taken as the solution obtained [28]. The column *Optimum* indicates the best known solution which comes from Reimann [29].

The simulation results are listed in Table 1 and 2. From Table 1 and 2, HACS-SA has better performance than ACS and SA. In Table 1, we present the results achieved by HACS-SA. The table shows the best solutions found by the proposed algorithm as well as the averages of

the best solution found in each of the 30 runs. The column *Optimum* indicates the best known solution when our research started. The results reveal that HACS-SA was able to find the better solutions then ACS and SA for all instances. An interesting point is that HACS-SA was able to find best solutions (instances A32k5, A54k7, A69k9, B78k10 and E76k8).

Table 1. Comparisons between HACS-SA and other Approaches for 12 Instances

Problem	Optimum	HACS-SA	ACS	SA
A32k5	784	**784**	792	795
A54k7	1167	**1167**	1180	1197
A60k9	1358	1365	1375	1389
A69k9	1167	**1167**	**1167**	1173
A80k10	1764	1776	1813	1842
B57k7	1153	1159	1163	1177
B63k10	1496	1507	1534	1534
B78k10	1266	**1266**	1275	1315
E76k7	682	687	707	707
E76k8	735	**735**	**735**	748
E76k10	832	846	861	867
E76k14	1032	1037	1043	1053

Form Table 2, HACS-SA also has the better solutions than ACS and SA. The results reveal that HACS-SA was able to find the better solutions (instances Kelly01, Kelly10, Kelly17 and Kelly19) then ACS and SA. In summary, HACS-SA outperforms these existing algorithms for CVRP.

Table 2. Comparisons between HACS-SA and other Approaches for Large-scale Instances

Problem	Optimum	HACS-SA	ACS	SA
Kelly01	5627.54	**5627.54**	5723.44	5846.28
Kelly02	8447.92	8717.36	8717.36	8789.4
Kelly03	11036.22	11470.4	11476.1	11615.1
Kelly04	13624.52	14487.1	14556.74	13973.0
Kelly05	6460.98	6546.18	6548.8	6546.19
Kelly06	8412.8	8674.74	8702.46	8755.48
Kelly07	10195.56	10650.5	10713.7	10706.8
Kelly08	11663.55	12170.92	12188.4	12347.6
Kelly09	583.39	583.79	603.64	603.01
Kelly10	742.03	**742.03**	775.9	779.80
Kelly11	918.45	926.7	961.47	968.80
Kelly12	1107.19	1122.44	1191.19	1193.7
Kelly13	859.115	875.13	895.47	893.65
Kelly14	1081.31	1102.9	1131.6	1137.63
Kelly15	1345.23	1369.70	1404.84	1409.70
Kelly16	1622.69	1638.10	1726.6	1741.1
Kelly17	707.79	**707.79**	726.95	731.18
Kelly18	998.73	1017.65	1038.50	1044.7
Kelly19	1366.86	**1366.86**	1425.6	1431.48
Kelly20	1821.15	1823.15	1919.6	1924.16

5. Conclusion

In this paper, a hybrid ant colony system is proposed for CVRP. It takes the advantages of simulated annealing and ant colony optimization. Twelve selected instances and twenty large-sized benchmark instances are used to verify the performance of the proposed algorithm. In two sets of benchmark instances, HACS-SA finds nine solutions which are equal to the best solutions found so far in literature.

Future work in this area may be dedicated to apply the proposed algorithm to further improve the solutions obtained through ACO algorithm. Another future direction the HACS-SA can be used to solve the vehicle routing problems with different limitation conditions.

References

[1] M Dorigo, LM Gambardella. *Ant Colonies for the Traveling Salesman Problem.* BioSystems. 1997; 43: 73-81.
[2] Huanglin Zeng, Yong Sun, Xiaohui Zeng. A New Approach of Error Compensation on NC Machining based on Memetic Computation. *TELKOMNIKA Indonesian Journal of Electrical Engineering.* 2013; 11(4): 2148-2156.
[3] Zheng Guoqiang, Liang Zhao, Li Jishun. Optimization of an Intelligent Controller for Parallel Autonomous Parking. *TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2013; 11(2): 1069~1075.
[4] Hanshar FT, Ombuki-Berman BM. Dynamic vehicle routing using genetic algorithms. *Appl Intell,* 2007, 27: 89–99.
[5] Gillett BE, Miller LRA. Heuristic algorithm for the vehicle dispatch problem. *Oper Res.* 1974; 22: 340-349.
[6] Haimovich M, Kan AHGR. Bounds and heuristics for capacitated routing problems. Math Oper Res. 1985, 10: 527-542.
[7] Ho SC, Gendreau M. Path relinking for the vehicle routing problem. *J Heuristics.* 2006; 12: 55–72.
[8] Laporte G, Semet F. The vehicle routing problem. In: Toth P, Vigo D (eds) SIAM monographs on discrete mathematics and application. SIAM, Philadelphia. 2001: 109–125.
[9] Prins C. A simple and effective evolutionary algorithm for the vehicle routing problem. *Comput Oper Res.* 2004; 31: 1985–2002.
[10] Silvia Mazzeo, Irene Loiseau. An Ant Colony Algorithm for the Capacitated Vehicle Routing. *Electronic Notes in Discrete Mathematics.* 2004; 18: 181-186.
[11] Yuvraj Gajpal PL. Abad Multi-ant colony system (MACS) for a vehicle routing problem with backhauls. *European Journal of Operational Research*. 2009; 102–117.
[12] O Martin, SW Otto, EW Felten. *Large-Step Markov Chains for the Traveling Salesman Problem*. Complex Systems. 1991; 5(3): 299-326.
[13] R Montemanni, L Gambardella, A Rizzoli, A Donati. *A new algorithm for a dynamic vehicle routing problem based on ant colony system*. In Second International Workshop on Freight Transportation and Logistics. 2003.
[14] M Dorigo, LM Gambardella. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. *IEEE Transactions on Evolutionary Computation*. 1997; 1(1): 53-66.
[15] X Hu, J Zhang, Y Li. Flexible protein folding by ant colony optimization. In: Computational Intelligence in Biomedicine and Bioinformatics: Current Trends and Applications. Springer-Verlag, New York. 2008: 317- 336.
[16] Metropolis N, Rosenbluth AW, Rosenbluth MN, Teller AH, Teller E. Equations of state calculations by fast computing machines. *J Chem Phys.* 1953; 21: 1087–1092.
[17] Kirkpatrick S, Gelatt CD, Vecchi JMP. *Optimization by simulated annealing*. Science. 1983; 220: 671–680.
[18] J Li, S Sun, Y Huang. Adaptive Hybrid ant colony optimization for solving Dual Resource Constrained Job Shop Scheduling Problem. *Journal of Software*. 2011; 6(4): 584-594. doi:10.4304/jsw.6.4.584-594.
[19] Chou-Yuan Lee, Zne-Jung Lee, Shih-Wei Lin, Kuo-Ching Ying. An enhanced ant colony optimization (EACO) applied to capacitated vehicle routing problem. *Appl Intell.* 2010; 32: 88–95.
[20] HR Lourenc¸o, O Martin, T Stützle. Iterated local search. In F. Glover and G. Kochenberger, editors, Handbook of Metaheuristics. International Series in Operations Research & Management Science. Kluwer Academic Publishers, Norwell, MA, 2002; 57: 321–353.
[21] EB Baum. Towards practical "neural" computation for combinatorial optimization problems. In J. Denker, editor, Neural Networks for Computing. AIP conference proceedings. 1986: 53–64.
[22] EB Baum. Iterated descent: A better algorithm for local search in combinatorial optimization problems. Technical report, Caltech, Pasadena, CA, 1986. Manuscript.
[23] O Martin, SW Otto, EW Felten. Large-step Markov chains for the traveling salesman problem. *Complex Systems*. 1991; 5(3): 299–326.

[24] DS Johnson, LA McGeoch, Experimental analysis of heuristics for the STSP, in: G. Gutin, A. Punnen (Eds.), The Traveling Salesman Problem and Its Variations, Kluwer Academic Publishers, Dordrecht, The Netherlands. 2002; 369–443.
[25] T Stützle. Applying iterated local search to the permutation flow shop problem. Technical Report AIDA-98-04, FG Intellektik, TU Darmstadt. 1998.
[26] Shi YH, Eberhart RC. Empirical study of particle swarm optimization [A]. In: Proceedings of t he IEEE Congress on Evolutionary Computation. Piscataway, NJ, *IEEE Service Center*. 1999: 1945-1950.
[27] Hung KS, Su SF, Lee ZJ. Improving ant colony optimization algorithms for solving traveling salesman problems. *J Adv Comput Intell Intell Inform*. 2007; 11(4):433–442. JACIII.
[28] Toth P, Vigo D. The granular tabu search and its application to the vehicle-routing problem. *INFORMS J Comput. 2003.* 15: 333–346.
[29] Lin SW, Ying KC, Lee ZJ, His FH. *Applying simulated annealing approach for capacitated vehicle routing problems.* In: Proceeding of IEEE international conference on systems, man, and cybernetics. 2006; 639–644.