Hindawi Publishing Corporation
Discrete Dynamics in Nature and Society
Volume 2013, Article ID 418436, 6 pages
http://dx.doi.org/10.1155/2013/418436

Research Article

Mutation Ant Colony Algorithm of Milk-Run Vehicle Routing Problem with Fastest Completion Time Based on Dynamic Optimization

Jianhua Ma and Guohua Sun

School of Management Science and Engineering, Shandong University of Finance and Economics, Jinan 250014, China

Correspondence should be addressed to Jianhua Ma; jianhuama@126.com

Received 20 December 2012; Accepted 28 February 2013

Academic Editor: Xiang Li

The objective of vehicle routing problem is usually to minimize the total traveling distance or cost. But in practice, there are a lot of problems needed to minimize the fastest completion time. The milk-run vehicle routing problem (MRVRP) is widely used in milk-run distribution. The mutation ACO is given to solve MRVRP with fastest completion time in this paper. The milk-run VRP with fastest completion time is introduced first, and then the customer division method based on dynamic optimization and split algorithm is given to transform this problem into finding the optimal customer order. At last the mutation ACO is given and the numerical examples verify the effectiveness of the algorithm.

1. Introduction

The vehicle routing problem (VRP) was firstly brought forward by Dantzig and Ramser in 1960 [1]. With the development of VRP, there exist several variations and specializations. Mostly VRP aims to minimize the total travel distance (or travel time) and total cost. But for distribution of fast foods, express delivery, and emergency supplies, these objectives are not suitable and the completion time is more important.

Though the vehicle routing problem with time windows takes the service time into consideration [2], it cannot solve the problem when all customers want to be served as early as possible. Nikolakopoulou et al. solved a vehicle routing problem by balancing the vehicles time utilization [3]. But it maybe takes a long time for some vehicles to complete the distribution tasks when the vehicles time utilizations are balanced. The VRP with fastest completion time was studied to minimize the fastest completion time [4–6].

Traditional vehicle routing problem assumes that the total distribution can be completed by one vehicle in a round trip. In practice, the number of vehicles is limited and there are many customers to be served. So a round trip by one vehicle is impossible and more round trips and vehicles are needed, such as the tobacco distribution with thousands of customers. As JIT production, small-scaled and multiple-batch distributions are more popular, in which one vehicle is needed to collect goods on multiple round trips. This kind of vehicle routing problem is called milk-run vehicle routing problem (MRVRP).

For problem aiming to minimize total travel distance or total cost, MRVRP can be transferred into the traditional VPR by increasing the routes number and allowing a vehicle serving customers in different routes. But for problem to minimize the fastest completion time, the completion time in the first period affects that in the second period. It is multiperiod optimization problem, so the algorithm of the MRVRP with fastest completion time will be studied based on dynamic optimization.

VRP is an NP-hard problem [7]; heuristics and evolutionary algorithms are used to solve VRP. In this paper, mutation ant colony algorithm is used to solve MRVRP with fastest completion time. In the next section, we will give the description of the MRVRP with fastest completion time. In Section 3, optimal division methods of customer orders based on dynamic optimization are given. In Section 4, mutation

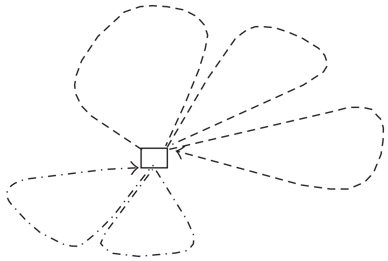

Figure 1: The milk-run vehicle routes of two vehicles.

ant colony algorithm is given to solve the problem. In the last section, a numerical example is given.

2. MRVRP with Fastest Completion Time

Suppose that there is one depot serving n customers, which have m vehicles with capacity w. The demand of customer i is q_i $(i = 1, \ldots, n)$ and satisfies $(k-1)w \leq \sum_{i=1}^{n} q_i < kw$, which means that for every vehicle, it needs k times to finish the distribution tasks on average.

Because the distances between the depot and the customers and the distances between the customers are different, the travel time between any two nodes is different. Supposing that the traveling time between node i and node j is t_{ij}, $i = 0, 1, \ldots, n$, $j = 0, 1, \ldots, n$, node 0 is depot and the times satisfy the triangle inequality.

The vehicle routes satisfy the following.

(i) Each customer is served by a certain vehicle.

(ii) One vehicle can serve many customers. When the demands of the customers exceed the vehicle capacity, the vehicle returns to the depot to unload and goes back to serve the next customer.

(iii) The demands of the customers served by a vehicle cannot exceed k times vehicle capacity.

(iv) All vehicles should depart from the depot and return to the depot.

When the demands of the customers exceed the vehicle capacity, the customers should be divided into different groups. The customers in the first group are served first, then the second group, until all the customers are served, which is shown in Figure 1.

For the MRVRP aiming to minimize total travel distance or total cost, because the traveling distance of a vehicle is the sum of the distances of each round and the total distance is the sum of all vehicle traveling distances, the MRVRP can be transferred as the VRP with km vehicles. The customers are divided into km routes and each vehicle serves k routes. The total traveling distance of m vehicles equals the total distance of km routes. Then if the total distance of km routes is minimized, the total traveling distance of m vehicles is also minimized.

But for the MRVRP with fastest completion time, the completion time of the whole distribution task is the completion time of the last vehicle. Suppose that the completion time of vehicle l is T_l, $l = 1, 2, \ldots, m$, and the completion time of the whole distribution task is T; then we have,

$$T = \max_{l=1,2,\ldots m} T_l. \tag{1}$$

For each vehicle, the distribution task needs to complete several routes and the completion time is determined by the time when last customer served. Because there is order of each route, the completion time of vehicle i is the sum of every route completion time. In the last route, the time from the customer to the depot is not considered.

If this problem is transferred to the problem with km routes in which each vehicle is in charge of k routes, the completion time is the sum of the travel times of previous $k-1$ routes and the completion time of the last routes. Supposing that the lth vehicle is in charge of the distribution task in k routes, the traveling time of the jth route is a_j, $j = 1, 2, \ldots, k$, and the traveling time from the last served customer to the depot is $\bar{t}$, the completion time of this vehicle is

$$T_l = \sum_{j=1}^{k} a_j - \bar{t}. \tag{2}$$

The difficulty of solving VRP lies in too many arrays of customer service order. It is hard to solve VRP by dynamic programming since there are too many states. For a given array of customers, the problem is transferred to how to divide the customers into groups and the customers in the same group are served by one vehicle. The feasible groups' division scheme is much less and it is easy to be solved by dynamic programming. In this paper, we will fix the serving array first and then give the optimal division scheme and calculate the fastest completion time. Taking this completion time as the objective, we will determine the final customer serving order by ACO.

3. Customer Division Method Based on Dynamic Programming

The problem of dividing customers into groups is how to divide a given serving array $i_1, i_2, \ldots, i_n$ into m groups to ensure that the completion time of all the distribution tasks is minimized and the total demands of the same group do not exceed kw. The customers belonging to the same group are served by one vehicle on the array order.

Usually there are a lot of division schemes and their completion times are different. Nikolakopoulou gave the split method to minimize the total traveling distances by transferring the dividing problem into the shortest-path problem [3]. We gave the improved split method to minimize the fastest completion time by transferring the division problem into the longest-edge shortest problem [5]. The improved split method spends a long calculation time, as it repeats to find the shortest paths. Then the customer division method based on dynamic programming is given which spends less time [6].

For the MRVRP, there are two divisions: the first is to divide the customers into m groups and the second is to divide the customers in the same groups into k routes. The aim of the first division is to minimize the longest completion time and is solved with dynamic programming. The aim of the second division is to minimize the total traveling time and is solved with split method.

3.1. The First Division of Customers. The problem of dividing customers into m groups is to determine $m-1$ cut points and can be looked as at m-period decision problem. In a period, the number of customers served by a vehicle is determined. So the state variable is the number of customers who have not assigned, denoted as s, and the number of vehicles which have not assigned, denoted as x. Then the state variable is (s, x), and the initial state is (n, m). The decision variable is the number z of customers served by the next vehicle.

In the next period, the numbers of unassigned customers and vehicles are $s-z$ and $x-1$. The state variable is $(s-z,\ x-1)$. The state variable and decision variable are both confined by the vehicle capacity. For a given (s, x), the demands of assigned customer cannot exceed k times vehicle capacity. Supposing that d_i $(i = 1, \ldots, n)$ is the total demand from the first customer to the ith customer, we have

$$d_{n-s} \leq (m-x)\,kw. \tag{3}$$

When the decision variable in the present period is z and the unassigned customers need $x-1$ vehicles to distribute, the total demand of the z customers cannot exceed kw; that is,

$$\begin{aligned} d_n - d_{n-s+z} &\leq (x-1)\,kw, \\ d_{n-s+z} - d_{n-s} &\leq kw. \end{aligned} \tag{4}$$

Denote decision variable set as Ω; then

$$\Omega = \{z \in \mathrm{Z} \mid d_n - d_{n-s+z} \leq (x-1)\,kw,\ d_{n-s+z} - d_{n-s} \leq kw\}. \tag{5}$$

Supposing that $f(s, x)$ is the fastest completion time of serving the s customers with x vehicles and $h(s, z)$ is the completion time if the former z customers are served by a vehicle, we have

$$f(s, x) = \min_{z \in \Omega} \max\left(h(s, z), f(s-z, x-1)\right). \tag{6}$$

When there is a vehicle left, if the total demands of s customers do not exceed kw, the s customers are served by this vehicle and the completion time is $h(s, s)$; otherwise the completion time is infinite; that is,

$$f(s, 1) = \begin{cases} h(s, s), & \text{if } q_{n-s+1} + \cdots q_n \leq w, \\ +\infty, & \text{if } q_{n-s+1} + \cdots q_n > w. \end{cases} \tag{7}$$

The state variable and decision variable are discrete. It can be solved by enumeration method. In a period, the vehicle number decreases one unit and the unassigned vehicle is determined by the periods. Then the number of states is determined by the number of unassigned customers, the number of states of each period does not exceed n. Since the upper bound of vehicle capacity is kw, there is also an upper bound of the number of customers to be served, denoted as g. The maximum iteration time of the method is mng. In practice, since the state variable should satisfy the inequality (3) and the decision variable should satisfy the inequality (4), the actual iteration time is much less than mng.

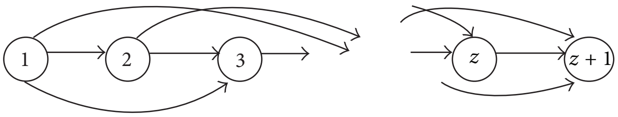

Figure 2: Split graph.

In each period, the completion time of every vehicle is needed to calculate and the second division of customers is needed.

3.2. The Second Division of Customers. Suppose that the customers served by a given vehicle are $i_1, i_2, \ldots, i_z$. Because the total demands exceed the vehicle capacity w, the route should be cut into smaller groups to ensure that the customers in the same group can be served by the same vehicle. This is also a division problem which is different from that in Section 3.1 in the scale. The scale is smaller and the objective is not to minimize the maximal completion but to minimize the total milk run completion time.

To get the division schemes, a directed graph is constructed. The vertex set has the depot and the customers served by the same vehicle. For nodes j_1 and j_2, supposing that i_{j_1} is in front of i_{j_2}, the total demand of all customers between i_{j_1} and i_{j_2} (Including customer i_{j_1} but not i_{j_2}) is calculated. If the total demand is less than w, an arc from j_1 to j_2 is drawn. The weight is the traveling time from the depot in turn reach all customers between i_{j_1} and i_{j_2} (including customer i_{j_1} but not i_{j_2}), and then return. The weighted direct graph is called split graph which is shown in Figure 2.

The shortest distance from 1 to $z+1$ is the shortest travel time of all distribution tasks and the fastest completion time is this time minus the return time from customer i_z to depot.

Suppose the capacity of a vehicle is 15 and the order of customers to be served by the vehicle is 1-2-5-3-4-7; the traveling times between the nodes are shown in Figure 3, where the numbers in the brackets are the demands.

The total demand of customers 1, 2, and 5 is no more than 15; then an edge is drawn between node 1 and 3 with the weight 90. The other weights are got similarly. After calculation, the directed weighted graph is got in Figure 4.

The shortest path is from node 1 to node 3 plus from node 3 to node 0. The optimal distance is 180. Customers 1, 2, and 5 construct the first route. Customers 3, 4, and 7 construct the second route. The completion time is 155 (= 180 − 25).

The algorithm to solve the shortest path problem such as Dijkstra algorithm is polynomial. Its complexity is $O(n^2)$, where n is the number of nodes. For this problem, the maximum number of nodes is g and the complexity is $O(g^2)$. Because for every iteration the shortest path problem should be solved to calculate the completion time of a vehicle, the complexity of the whole algorithm is $O(mng^3)$.

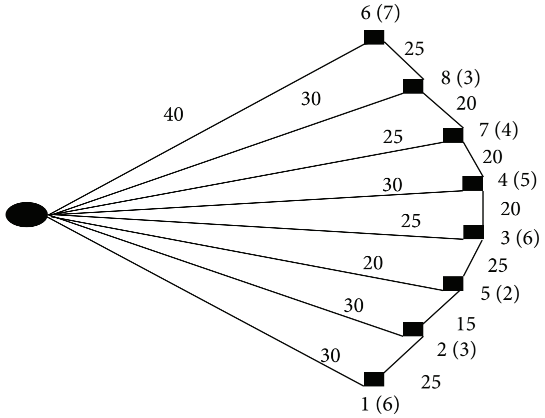

Figure 3: The distribution of customers.

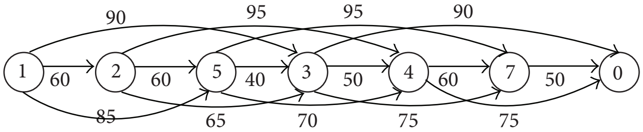

Figure 4: Split graph.

4. Mutation ACO Algorithm

In this section, we will give an improved ACO to solve the optimal customers array.

4.1. ACO Algorithm. The main idea of ant colony algorithms is to mimic the pheromone trail used by real ants as a medium for communication and feedback among ants. Basically, the ACO algorithm is a population-based, cooperative search procedure that is derived from the behavior of real ants. ACO algorithms make use of simple agents called ants that iteratively construct solutions to combinatorial optimization problems. The key problem to solve VRP with ACO is how an individual ant constructs a complete solution by starting with a null solution and iteratively adding solution components until a complete solution is constructed. The key problem of ACO is to determine the pheromone matrix. The pheromone matrix is $(n + 1) \times n$ when there are n customers, where the last row stands for the information from the depot to the customer and the ith row stands for the information from customer i to other customers. Initially, since for a given customer there is the same possibility following other customers, the pheromone matrix starts with equal probable matrix. When $t = 0$, the pheromone matrix is

$$B_{ij}(0) = \left\{ \begin{matrix} 0 & \frac{1}{n-1} & \cdots & \frac{1}{n-1} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{1}{n-1} & \frac{1}{n-1} & \cdots & 0 \\ \frac{1}{n} & \frac{1}{n} & \cdots & \frac{1}{n} \end{matrix} \right\}_{(n+1)\times n}, \quad (8)$$

where b_{ij}, $i = 1, 2, \ldots, n + 1$, $j = 1, 2, \ldots, n$ is the pheromone from i to j.

In the tth iteration, there are m ants and m customer arrays can be got. Supposing that the fastest completion time of each array is $L_k(t)$; $k = 1, 2, \ldots, m$, the increased value of pheromone in the tth iteration is $\Delta b_{ij}(t) = \sum_{k=1}^{m} \Delta b_{ij}^{k}(t)$, where

$$\Delta b_{ij}^{k}(t) = \begin{cases} \dfrac{Q}{L_k(t)}; & \text{if } j \text{ follows } i \text{ closely, where } Q \text{ is a constant,} \\ 0; & \text{otherwise.} \end{cases} \quad (9)$$

Local update is performed during the ant constructive procedure in the following way:

$$b_{ij}(t+1) = (1-\rho)\, b_{ij}(t) + \Delta b_{ij}(t), \quad (10)$$

where ρ is the evaporation coefficient.

The heuristic information is in an $(n+1) \times n$ matrix, with the last row $\eta_{n+1,j} = A/t_j$ and the other rows $\eta_{ij} = A/t_{ij}$, where t_{ij} is the traveling time between two nodes, t_j is the average traveling time from customer j to the depot, and A is a constant depending on the situations.

In a given service order, the first served customer is determined by probability. The next served customer is selected from the allowing set allowed$_i$, which is the customers set that can be served after i. The customer j is chosen with the probability p_{ij}, where

$$p_{ij}(t) = \begin{cases} \dfrac{(b_{ij}(t))^{\alpha} (\eta_{ij})^{\beta}}{\sum_{s \in \text{allowed}_i} (b_{is}(t))^{\alpha} (\eta_{is})^{\beta}}; & j \in \text{allowed}_i, \\ 0; & j \notin \text{allowed}_i. \end{cases} \quad (11)$$

α is the importance of pheromone information, and β is the importance of heuristics information. After calculation of the probability matrix, the node is selected by the probability p_{ij} and the node is deleted from the allowing set. Then a customer service order is got with the pheromone.

4.2. Mutation Operator. ACO algorithm possibly runs into prematurity just as other evolutionary algorithms. The main reason is the concentrations of pheromone which makes the same solutions be got. In reality, the pheromone may be changed by rain and other factors and this change may help the ants find a new route. A mutation operator will be introduced into the pheromone of ACO to escape from local optima and strengthen its global search ability.

The initial pheromone matrix's factors are the same. As time goes on, some elements become large but other elements become small in the same row. The pheromone is concentrated which obstructs the ability to find more optimal solutions. The concentration of a row is defined as the ratio of the maximum factor and the sum of the factors in the row. The ith row concentration $\mu_i(t)$ is

$$\mu_i(t) = \frac{\max_{j=1,2,\ldots,n} \tau_{ij}}{\sum_{j=1}^{n} \tau_{ij}}, \quad i = 1, 2, \ldots, n + 1. \quad (12)$$

The concentrations of different row are not the same. Whether to mutate a row should depend on the concentration of the row. When the concentrations of a row exceed the threshold value, the factors in the row should be mutated.

For the row mutation, the maximum factor decreases randomly and the decreased value is assigned to other positions. Suppose j_0th factor is the maximum factor in the ith row, and given a random number r $(0 \leq r \leq 1)$ and a random vector $R_{1\times n}$. A new row is got with the j_0th factor multiplied by r, and the decreased value is assigned to other positions; that is,

$$\tau_i' = \tau_i' + \frac{\tau_{ij_0}(1-r)}{\sum_{j=1}^{n} R_j} R. \tag{13}$$

Row mutation is a local mutation and only affects the choice of routes partly. When the matrix has high concentration, local mutation cannot ensure escaping the local optima and matrix mutation is necessary. The minimum concentration of the row is the matrix concentration, which is $\nu(t)$:

$$\nu(t) = \min_{i=1,2,\ldots,n+1} \mu_i(t). \tag{14}$$

Whether to mutate a matrix should also depend on the concentration of the matrix. When the concentrations of a matrix exceed the threshold value, the matrix should be mutated. For the matrix mutation, every element in the matrix decreases randomly and the decreased value is again randomly assigned. A random number $\overline{r}$ $(0 \leq \overline{r} \leq 1)$ and a random vector $R_{n+1\times n}$ are got. New values are got with every factor multiplied by $\overline{r}$. Then the decreased value is assigned randomly' that is,

$$\begin{gathered} \tau' = \overline{r} * \tau, \\ \tau' = \tau' + \frac{\sum_{i=1}^{n+1}\sum_{j=1}^{n}\left(\tau_{ij} - \tau_{ij}'\right)}{\sum_{i=1}^{n+1}\sum_{j=1}^{n} R_{ij}} R. \end{gathered} \tag{15}$$

Then a new pheromone matrix is got and the mutation can ensure that the total pheromones do not change.

The mutation ACO algorithm is as follows.

Step 1. Initiation: Determine the parameters m, T, α, β, ε, ϕ, and p. Input the initial pheromone $\tau_{ij}(0)$ and heuristics matrix η_{ij} and get the initial ants. Give m customers arrays randomly as

$$A(0) = \{A_1(0), A_2(0), \ldots, A_m(0)\}. \tag{16}$$

Step 2. For a given array, assign the customers to the vehicle and get the corresponding completion time by dynamic programming and split algorithm. Record the present best route $besttrip$ and the fastest completion time $finishtime$.

Step 3. The local pheromone update is performed by all the ants after each construction step on the formulas (9) and (10).

Step 4. Row Mutation: Inspect whether the pheromone concentration of the row in the pheromone matrix is more than ε; if yes, a random number r is got. If $r \leq p$, a new row is got by (13).

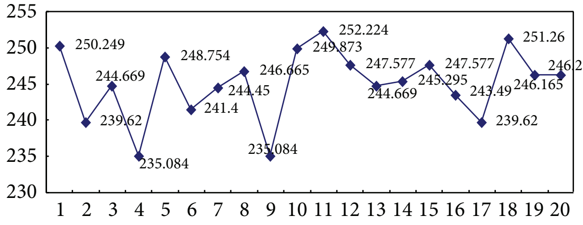

Figure 5: 20 times calculation results.

Step 5. Matrix Mutation: Inspect whether the pheromone matrix concentration is more than ϕ; if yes, a random number $\overline{r}$ is got. If $\overline{r} \leq p$, a new pheromone matrix is got by the formula (15).

Step 6. According to the new pheromone matrix $B_{ij}(t)$ and the heuristics information matrix η_{ij}, Give m customers arrays randomly as

$$A(t+1) = \{A_1(t+1), A_2(t+1), \ldots, A_m(t+1)\}. \tag{17}$$

Step 7. Let $t = t+1$. ACO procedure stops if $t = T$ and output the *finishtime* and besttrip; otherwise return to Step 2.

5. Numerical Examples

In this section, we will consider the emergency supplies distribution after the earthquake in Wenchuan. After the earthquake, the roads on earth are not fluent. The emergency supplies are mainly transported by helicopter. As the number of helicopters is limited comparED to the broad place, the helicopter s is needed for distribution many times. The problem is a typical MRVRP with fastest completion time.

Suppose there are 3 helicopters in charge of the distribution of emergency supplies to 20 settlements. The location of material distribution center is (30, 40), the demands and location of the settlements are given in Table 1.

The capacity of a helicopter is $12t$. The total demand is $92t$. Every plane is needed to fly 3 times on average and distribute at most 5 settlements every time.

The algorithm is realized by Scilab. The parameters are as follows: the ants number is 10 and $\alpha = 1.2$, $\beta = 0.5$, $P = 0.15$, $\varepsilon = 0.8$, $\phi = 0.75$. The iteration time is 400. Run the program 20 times for the same problem, and get the fastest completion which times are as shown in Figure 5.

Average fastest completion time is 244.9978, their maximum gap is 7.291, and the gap is 3.1014% of the best scheme. It shows that the algorithm has good convergence.

Best distribution route is as shown in Figure 6.

Its fastest completion time is 235.08428 and total travel time is 865.66435. A new scheme to minimize total travel time is as shown in Figure 7.

Its shortest total travel time is 845.89676, the fastest completion time of this customer array is 273.379, and it is 38.295 minutes more than the first scheme. And the total

Table 1: Locations and demands of settlement.

Settlement	1	2	3	4	5	6	7	8	9	10
Location	(74, 29)	(64, 26)	(67, 80)	(88, 15)	(21, 65)	(72, 42)	(92, 80)	(46, 38)	(76, 86)	(30, 46)
Demand	3	7	5	6	2	9	7	4	4	3.5
Settlement	11	12	13	14	15	16	17	18	19	20
Location	(63, 48)	(23, 11)	(36, 72)	(29, 54)	(66, 16)	(10, 10)	(15, 50)	(10, 10)	(20, 70)	(70, 12)
Demand	4	3	8	6	5	1.5	5	2.5	3	3.5

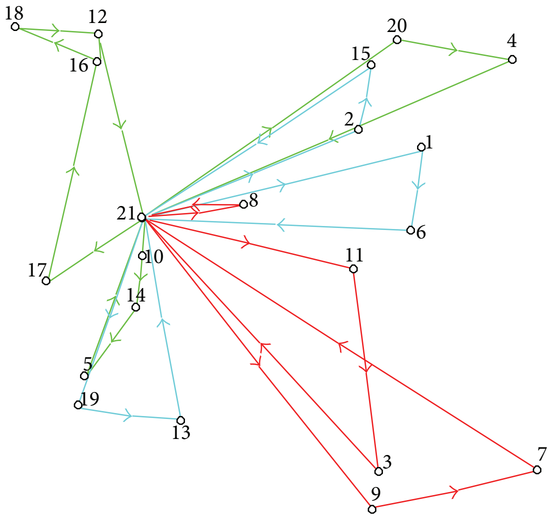

Figure 6: Optimal scheme of MRVRP with fastest completion time.

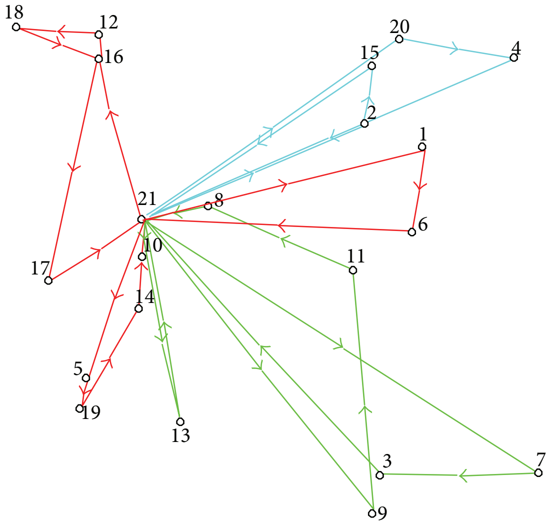

Figure 7: Optimal scheme.

flight time is reduced by 19.76759 minutes. In emergency management, effectively shortening the completion time is very necessary, and therefore the first scheme is more reasonable.

6. Conclusions

In this paper, the MRVRP with fastest completion time is proposed, which has many applications in fast foods distribution, express delivery, and emergency supplies. Solving the problem is more difficult than the general VRP. The key problem to solve MRVRP with fastest completion time is to give the division method for customer array. The customer division method based on dynamic programming and split method is given in this paper, which can transfer MRVRP with fastest completion time into the problem of finding the optimal customer service order. Then the problem is solved with mutation ACO.

References

[1] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management Science*, vol. 6, pp. 80–91, 1960.

[2] Q. L. Ding, X. P. Hu, and Y. X. Li, "Hybrid ant colony system for vehicle routing problem with time windows," *System Engineering Theory and Practice*, vol. 27, no. 10, pp. 98–104, 2007.

[3] G. Nikolakopoulou, S. Kortesis, A. Synefaki, and R. Kalfakakou, "Solving a vehicle routing problem by balancing the vehicles time utilization," *European Journal of Operational Research*, vol. 152, no. 2, pp. 520–527, 2004.

[4] X. Zhang, J. Ma, W. Liu, and F. Jin, "Ant colony algorithm for vehicle routing problem with shortest completion time," in *Proceedings of the 13th International Conference on Industrial Engineering and Engineering Management*, vol. 1–5, pp. 12928–2933, 2006.

[5] J. Ma and J. Yuan, "Ant colony algorithm for multiple-depot vehicle routing problem with shortest finish time," *Communications in Computer and Information Science*, vol. 113, no. 1, pp. 114–123, 2010.

[6] J. Ma, Y. Fang, and J. Yuan, "Mutation ant colony algorithm for multiple-depot multiple-types vehicle routing problems with shortest finish time," *Systems Engineering-Theory & Practice*, vol. 31, no. 8, pp. 1508–1516, 2011.

[7] J. K. Lenstra and A. H. G. Rinnooy Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.