A study in Tabu Search Algorithm to Solve a Special Vehicle Routing Problem

Xingrong Yan\textsuperscript{a}, Hongan Dong\textsuperscript{b}

\textsuperscript{a}Department of business science, Binzhou Polytechnic College, Shandong, China
\textsuperscript{b}Department of computer science, Binzhou Polytechnic College, Shandong, China

Abstract

In this paper, a kind of special vehicle routing problem based on reality-- vehicle routing problem with facultative demands is presented. The attributes of the problem and the optimization target are described. The mathematical model of the problem is set up. To solve the problem, A meta-heuristic approach called tobu search (TS) is put forward. The neighborhood structure and the parameters of TS algorithm are designed respectively. The proposed algorithm is successfully applied to a case and the result indicates the TS algorithm is practicable and valid.

Index Terms: Vehicle Routing Problem, Tabu Search, Meta Heuristic Algorithm

© 2012 Published by MECS Publisher. Selection and/or peer review under responsibility of the Research Association of Modern Education and Computer Science.

1. Introduction

Vehicle routing problem (VRP) is an optimization problem of vital importance. A typical VRP problem is usually described just like working out a vehicle routine with the shortest distance, where the origin and terminal point of vehicles and the capacity of them is the same and each vehicle visits every customer only once. This kind of problem is firstly put forward by Dantzig\textsuperscript{[1]}. Garey\textsuperscript{[2]} has proved TSP problems to be NP-HARD problems, which means VRP problems are also NP-HARD problems.

Now we consider a kind of problem, where the vehicles set out in the same parking lot and arrive at their terminal after visiting every point. However, in this problem there is no restriction on the number of visiting times at each point and demands at points can surpass the unit vehicle capacity. Under the condition of satisfying the restriction on the capacity of vehicles and minimum cost, how should we arrange vehicle routings with the shortest distance and the least number of vehicles such that all vehicles can reach the destination in the limit time? This problem is an open VRP problem whose routing is not a closed cycle and is allowed to have partial repetition.

* Corresponding author.
E-mail address: \textsuperscript{a}fahongw@126.com; \textsuperscript{b}donghong_an@163.com
The problem that has some similarity to the problem studied here is the Vehicle Routing Problem with Simultaneous Delivery and Pick-up (VRPSDP), which introduced into the literature first by Min[3]. Till 2001, Dethloff[4] considered this problem again. However, during these literatures, the pick-up of the returned products cannot be split. Another related situation to our problem is the Vehicle Routing Problem with Backhauling (VRPB)[5]-[8] in which the customers are divided into two exclusive sets, the ‘pure delivery’ and ‘pure pick-up’ customers. Obviously, there is no correspondence between forward logistics and reverse logistics. Furthermore, it was pointed out by Dethloff[4] that none of the solution approaches towards the VRPB can be used directly for the strict VRPSDP.

To solve this NP-HARD problem, the typical operational research methods are not valid or efficient anymore. Here, a meta-heuristic approach called tabu search (TS) is designed.

2. Problem description and model formulation

To the above problem, denote and description as follows:

Let $G = (V, E)$ is an undigraph, therein $V = \{v_i \mid i = 0,1,2,\ldots, t \}$ denotes the vertex set, $E = \{(v_i, v_j) \mid v_i, v_j \in V, i \neq j \}$ denotes the edge set. The vertexes $v_i (i \neq 0, i \neq t)$ are customers in which their demands $c_j$ are satisfied by the unit capacity vehicles which denote by $m$. $K = \{k \mid k = 1,2,\ldots,m \}$ is the set of all vehicles which determined by demands of all the customers. $C$ is the vehicle capacity and $L$ is the maximum drive time limit. $d_{ij}$ is the distance between vertexes $(v_i, v_j)$. $s_{ik}$ is demands of $v_i$ which satisfied by vehicle $k$.

Let:

$$x_{ijk} = \begin{cases} 1, & \text{if vehicle } k \text{ travels to customer } v_j \text{ directly from customer } v_i, \\ 0, & \text{others} \end{cases}, \quad y_{ik} = \begin{cases} 1, & \text{if customer } v_i \text{ is visited by vehicle } k \\ 0, & \text{others} \end{cases}$$

Then the problem can be formulated as follows:

$$\min \sum_i \sum_j \sum_k d_{ij}x_{ijk}$$

Subject to:

$$\sum_{k=1}^{m} y_{ik} = \begin{cases} \geq 1, & i = 1,2,\ldots,t-1 \\ m & i = 0,t \end{cases}$$
\[ \sum_{i=1}^{t-1} c_i y_{ik} \leq C, \quad k = 1, \ldots, m \]  

(3)

\[ \sum_{i,j=0}^{t} d_{ij} x_{ijk} \leq L \quad k = 1, \ldots, m \]  

(4)

\[ \sum_{k=1}^{m} x_{ik} y_{ik} = c_i \quad i = 1, 2, \ldots t - 1 \]  

(5)

\[ \sum_{j=1}^{t-1} x_{ijk} = \sum_{j=1}^{t-1} x_{jik} = y_{ik} \]  

(6)

In the model above, target (1) is to guarantee the minimum distance or least time; restriction (2) assures that each customer is visited at least once; restriction (3) is the capacity of vehicles; restriction (4) is the maximum distance of a single vehicle; restriction (5) guarantee the capacity of vehicles serving the same point can satisfy the demand at this point and the restriction (6) ensures the vehicle which must be leave the customer after serving it.

3. Algorithm Design

Considering the attributes of the problem, firstly, two lemmas are put forward.

**Lemma 1:** If there exists a customer point whose demand is larger than unit vehicle capacity, the minimum distance or cost of a vehicle k starting from the origin to its destination via this point is lower than that of several vehicles serving this customer point together.

Proof: to omit

**Lemma 2:** If there exists two or more intersection points between two routes in a feasible solution, this solution is not the optimization. We can get a better feasible solution after adjustment, where any two routes have one intersection point at most\(^9\).

Proof: to omit

3.1. TS Algorithm Design to the problem

**Neighborhood structure:**

The neighborhood structure of TS algorithm is based on three kinds of move operation: points inserting, points swapping and the mutual service point transferring. Let \(N_1(R), N_2(R)\) and \(N_3(R)\) denote neighborhoods made by three kinds of operations above respectively. Next we will introduce these operations in detail.
(1) Points inserting operation

For any point \( v_i \in R_k \), \( v_j \in R_i \), \( (R_k, R_i) \) are the any routings in routing set \( R \), move \( v_i \) to the situation behind the point \( v_j \) in route \( R_i \). Then, the route \( R_k \) and \( R_i \) turn into \( R'_k = \{v_0, \cdots, v_{i-1}, v_i, v_{i+1}, \cdots, v_j\} \) and \( R'_i = \{v_0, \cdots, v_{j-1}, v_j, v_{j+1}, \cdots, v_j\} \) respectively. If \( v_i \in R_k \cap R_i \), the point can be moved as long as the point satisfies the inserting condition. The condition can be described as follows:

1) The remainder capacity of vehicle \( l \) which serves the route \( R_i \) is full load, then the condition \( c_i = c_j \) as well as the restriction (4) must be meet.
2) \( c_i \leq C - \sum c_j \), in addition, the restriction (4) is still satisfied after the moving operation.

(2) Points swapping operation

For any point \( v_i \in R_k \), \( v_j \in R_i \), \( v_i \notin R_k \cap R_i \), \( v_j \notin R_k \cap R_i \), the points swapping operation means that \( v_i \in R_i / \{v_j\} \), \( v_j \in R_k / \{v_i\} \). And the condition as follows:

1) If vehicle \( k \) of route \( R_k \) and vehicle \( l \) of route \( R_i \) are full load, then the condition \( c_i = c_j \) as well as the restriction (4) must be meet.
2) If only one of the vehicle \( k \) of route \( R_k \) or vehicle \( l \) of route \( R_i \) is full load, might as well presumption that vehicle \( k \) is full load, then \( c_i \geq c_j \); Otherwise, if vehicle \( l \) is full load, then \( c_i \leq c_j \) must to be meet. All the swapping operation must be meet restriction (4).
3) If vehicle \( k \) of route \( R_k \) and vehicle \( l \) of route \( R_i \) are not full load, then the condition \( c_i \leq C - \sum c_i - c_j \), \( c_j \leq C - \sum c_k - c_i \) as well as the restriction (4) must be meet.

(3) Mutual service point transferring

For \( v_i \in R_k \cap R_i \), \( v_j \in R_i \), let the point \( v_j \) as the mutual service point instead of point \( v_i \) which removes from \( R_k \) into \( R_i \). Then, \( v_j \in R_k \cap R_i \), \( v_i \in R_i \). The conditions must to be meet as follows:

1) If vehicle \( k \) of route \( R_k \) and vehicle \( l \) of route \( R_i \) are full load, then \( c_j \geq s_{ki} \) and restriction (3) as well as restriction (4) must to be meet.
2) If the vehicle \( k \) of route \( R_k \) or vehicle \( l \) of route \( R_i \) is not full load, then operation can be hold only as restriction (3) as well as restriction (4) to be meet.

3.2. Design Of Other Parameters

(1) Adaptive value function: Using target function as adaptive value function.
(2) Selection of tabu object, tabu list, tabu length and candidate:

In the algorithm, Tabu object is the change of the status-self and \( p=5 \) is the tabu length. Tabu list is designed as follows:

\[ R \] is one of the feasible solution set, \( R = \{R_1, R_2, \cdots, R_m\} \).
1) Points inserting operation:
For any point \( v_i \in R_k \), \( v_j \in R_i \), move \( v_i \) to the place behind the point \( v_j \) in route \( R_j \). The point \( v_i \) is forbidden to be back to \( R_k \). Then let us denote the tabu list \( T_i(v_i, R_k) = p \).

2) Points swapping operation:
After the points swapping operation, denotes \( T_2(v_i, R_j) = p \), \( T_2(v_j, R_i) = p \).

3) Mutual service point transferring:
After the mutual service point transferring operation, denotes \( T_3(v_i, R_k) = p \).

The candidate solution are all of the solutions in the current status neighborhood.

3) Candidate list strategy: if the fit value of a tabu object’s candidate solution is better than the “best so far” status, then let the candidate solution as the current status and updates the “best so far” status.

4) End condition: under the condition of giving tabu length, the end condition adopts the traversal of status space or the best fit value is unchanged in a continue five steps.

5) Initial solution: the solution obtained by the shortest routing algorithm is our initial solution.

4. A case to Study

Given a case owned some vehicles serve 23 dispersed customers(points) and marked from 0 to 22. 0 is the parking lot as well as 22 is destination lot and 1-21 is customers. \( D = (d_{ij})_{23 \times 23} \) and \( c_j \) are known. \( C = 50 \), \( L = 30 \).

Initial solution: the solution obtained by the shortest routing algorithm is our initial solution.

\( R_1: (0,6,22) \quad \sum c_i = 50 \quad \sum d_{ij} = 19.5 \quad R_2: (0,1,2,3,4,5,6,7,8,9,21,22) \quad \sum c_i = 50 \quad \sum d_{ij} = 22 \).

\( R_3: (0,10,11,12,18,19,13,14,15,22) \quad \sum c_i = 50 \quad \sum d_{ij} = 28 \).

\( R_4: (0,16,17,18,13,19,20,21,22) \quad \sum c_i = 43 \quad \sum d_{ij} = 29 \).

The target function value is: \( 19.5 + 22 + 28 + 29 = 98.5 \)

Therein: customer 21 is the mutual service point of \( R_2 \) and \( R_4 \); customer 19 is the mutual service point of \( R_3 \) and \( R_4 \).

Optimization solutions by TS algorithm:

\( R_1: (0,6,22) \quad \sum c_i = 50 \quad \sum d_{ij} = 19.5 \).

\( R_2: (0,1,2,3,4,5,6,7,8,20,22) \quad \sum c_i = 49 \quad \sum d_{ij} = 20.5 \).

\( R_3: (0,11,12,18,19,13,14,21,9,22) \quad \sum c_i = 49 \quad \sum d_{ij} = 27.5 \).

\( R_4: (0,10,.16,17,15,22) \quad \sum c_i = 45 \quad \sum d_{ij} = 28 \).

The target function value is: \( 19.5 + 20.5 + 27.5 + 28 = 95.5 \).

It is shown by this case that the algorithm is valid not only in the total target length but also in eliminating the mutual service points. Because there is no mutual service points anymore.

References

A study in Tabu Search Algorithm to Solve a Special Vehicle Routing Problem


