TELKOMNIKA, Vol.11, No.1, January 2013, pp. 399~405
ISSN: 2302-4046

Grouping Heuristics on Vehicle Scheduling Problem Based on Changeable Expenditure Coefficient Model

Yong Zeng[*1,2], Da-Cheng Liu[1], Ju-Xuan Li[3],Xiang-Yu Hou[1]
[1]Dept. of Industrial Engineering, Tsinghua University, Beijing, P. R. China
519A, South of Shunde Building, Tsinghua University, Beijing, 100084, P. R. China
[2]Automobile N. C. O Academy of Bengbu, Bengbu, P. R. China
[3]Military Transportation Academy, Tianjin, P. R. China
*Corresponding author, e-mail: zengyong7505@163.com

Abstract

A kind of vehicle scheduling problem (VSP) with non-full load and combined pick-up and delivery is studied, a changeable expenditure coefficient model according to the actual load is made, and grouping heuristics algorithm under restrictions of vehicle load capacity, working time and mileage is designed to minimize the number of vehicle, the distance of empty load and the useless freight turnover, By programming and calculating, an example proves the algorithm is feasible and effectual.

Keywords: *scheduling optimization, combined pick-up and delivery, non-full load, change able expenditure coefficient, grouping heuristics algorithm*

1. Introduction

In daily transport activities, if the load of any task cannot exceed the capacity of the vehicle, and the task requires loading and unloading at certain places. That's to say, all the tasks has loading and unloading operations, and tasks can be combined. This kind of problem is called vehicle scheduling problem which is not fully loaded and combined pick-up and delivery. It's NP hard problem. Usually we try to solve this kind of problem based on heuristic algorithm, and there're a few literatures on this area. Some literatures have studied on VRP with non-full load [1][2] or with time windows [3], and the VRPs with pick-up and delivery already have been researched by different approaches [4]-[10].Our paper aimed at this kind of vehicle scheduling problem, using a changeable expenditure coefficient model rather than the former fixed expenditure model, and designed a grouping heuristic algorithm to minimize the number of vehicle, empty travel distance and useless freight turnover, providing a new method to solve this kind of task combination and vehicle scheduling problem.

2. Problem Description

The problem can be described as: l transport tasks r, denote as $r=1,\ldots,l$, any task r has its loading place $R^r{}_s$ and unloading place $R^r{}_t$, the volume of freight is g_r, the time of loading is $T^r{}_s$, and the time of unloading is $T^r{}_t$. These tasks are assigned to vn similar vehicles from a station, the capacity is q. The working time of this kind of vehicle $t_k$cannot exceed T, the working distance S_k cannot exceed S. Vehicles come back to the station after their tasks are finished. We already know that $g_r\leq q$ (r=1,…, l), a task cannot be divided, required a less than T time to finish, and the distance is less than $S.$ We want to determine the number of vehicle vnneeded, and the task assignment and the routine, to make the plan economic.

If all the tasks satisfy that $0.5q<g_r\leq q$, different tasks cannot be combined in one vehicle, we can only consider transport task by task. If $g_r<0.5q$, we can consider transport more than one task in a vehicle to improve the efficiency. If the goods allow mixed loading, we can divide the tasks into groups, the groups are assigned to different vehicles. The vehicle can have either loading or unload or both operations in one place, this is the situation discussed this paper.

Received August 15, 2012; Revised December 1, 2012; Accepted December 11, 2012

3. Model

For the convenience of model construction, we denote the station as 0, the tasks r is 1,…, *l*, the station and the places of loading and unloading can be expressed using *i*(*i*=0,1,…, *n*), then we have:

$$y_{rk} = \begin{cases} 1 & \text{task r is assigned to k} \\ 0 & \text{otherwise} \end{cases}$$

$$x_{ijk} = \begin{cases} 1 & \text{vehicle k goes from i to j} \\ 0 & \text{otherwise} \end{cases} .$$

3.1. Changeable Expenditure Coefficient Model

Using the function$c$ (q_{ijk}, d_{ij}) denote the transport cost from i to j, it can be any of distance, expense, time and so on, this is determined according to the data we have, and we choose expense as our cost. The expenses are different at the situation the vehicle is fully loaded comparing to the situation of empty load, and there's a fixed cost, we denote the following changeable expenditure coefficient model:

$$c(q_{ijk}, d_{ij}) = \begin{cases} c_0 + c_1 d_{0j} & q_{ijk} = 0; i = 0; j = 1, \cdots, n & (1-1) \\ c_1 d_{ij} & q_{ijk} = 0; i \neq 0; j = 0, 1, \cdots, n & (1-2) \\ (c_1 + \alpha q_{ijk}) d_{ij} & q_{ijk} \neq 0; i, j = 1, \cdots, n & (1-3) \end{cases} \quad (1)$$

Where q_{ijk} is the actual deadweight of vehicle k from i to j; d_{ij} is the distance between i and j; c_0 is the fixed cost when a vehicle is used; c_1 is the expenditure coefficient of empty load; αq_{ijk} is the expenditure coefficient when loading goods which weigh q_{ijk}.; and α is the coefficient which means the expenditure for unit weight.

We know (1-1) is the cost from station to the loading place, including fixed cost and empty load running cost; (1-2) means the cost of empty load from i to j; (1-3) means the cost when the vehicle is not empty, the cost of deadweight is added to the empty load cost.

The value of c_0、c_1、α determined the policy of vehicle scheduling.

- When c_0 is large, the goal is to minimize the number of vehicles.
- Whenc_1 is large, the goal is to minimize the distance of empty load running.
- when α is large, when different tasks are combined in one vehicle, we should try to reduce the additional travelling distance caused by the combination, meaning that we should try to make the loaded vehicle go to the unloading place directly, reduce the useless turnover. In some literature, the definition of cost didn't include the expenditure increment caused by deadweight, this will increase the useless turnover in real situation.

Based on the practical significance of the parameters and the real transport situation, we can set the value of the parameters, and try to minimize the cost by adjust the scheduling policy. This paper is to minimize the number of vehicles at first, then minimize the distance of empty load and the useless turnovers under the constraints of vehicle capacity, working time and working distance.

3.2. Changeable Expenditure Coefficient Model with Pick-up and Delivery Combined under Not Fully Loaded Situations

The optimization model is expressed as:

$$\min z = \sum_i \sum_j \sum_k c(q_{ijk}, d_{ij}) x_{ijk} \quad (2)$$

Equation (2) is the optimization goal to minimize the cost, (3) requires the deadweight cannot exceed the capacity, (4) represents the constraint of working time, (5) means the distance cannot exceed the vehicle constraint, (6) means any task can only be assigned to one vehicle; (7) says all vehicles come from the station; (8) represents the vehicles should come back after the tasks finished. And (9), (10) are the constraints for the value of variables.

$$s.t.\quad q_{ijk} \le q \qquad i, j = 0,1,\cdots,n; \forall k \qquad (3)$$

$$\sum_r (T_s^r y_{kr}) + \sum_r (T_t^r y_{kr}) + \sum_i \sum_j (t_{ij} x_{ijk}) \le T \qquad \forall k \qquad (4)$$

$$\sum_i \sum_j (d_{ij} x_{ijk}) \le S \qquad \forall k \qquad (5)$$

$$\sum_k y_{kr} = 1 \qquad \forall r \qquad (6)$$

$$\sum_i \sum_k x_{0ik} = vn \qquad (7)$$

$$\sum_i \sum_k x_{0ik} - \sum_j \sum_k x_{j0k} = 0 \qquad (8)$$

$$x_{ijk} = 0 or 1 \qquad i, j = 0,1,\cdots,n; \forall k \qquad (9)$$

$$y_{kr} = 0 or 1 \qquad \forall k, r \qquad (10)$$

4. A Grouping Heuristic Algorithm

4.1. Principles of the Algorithm

Since the workload of single task is not large, a vehicle can afford several tasks at the same time, pick up from several sources and deliver to different destinations. Intuitively, we suppose it will be more economic if several nearby sources and destinations are grouped as a task group. Under the time constraint, a vehicle can be used to deliver several task groups, we name this situation that a vehicle deliver several tasks in sequence as a task queue. Finally, we get the number of vehicles needed, the tasks and routines of all the vehicles.

4.2. Design of the Algorithm and the Calculation Steps

To solve the scheduling problem mentioned above, we should select the main tasks from all the tasks firstly, and then group them as queues, and then calculate the number of vehicles and the routines. We define the main task as the task which has large load, has large influence and constraints on other tasks when considering mixed loading. It makes the combination of the main tasks and the incidental tasks much easier, reduce the feasible zones, and improve the efficiency of the algorithm.

Stage1: combine tasks and determine the routine in a group.

1: screen out the tasks required to be transported separately among the *l* tasks, these tasks composed a group, denote this group as a special group, meaning these tasks can only be executed in sequence, and each one is a main task. The set of the special group is DZ.

2: consider other tasks can be mixed loaded, select those tasks satisfy g_r≥0.5 as main tasks, denote as set Z.

3: consider the sequential loading and delivery process with a vehicle, which is, we should unload the former task before we load a new task, calculate $d(R_t^i R_s^j)$, i,j∈{1,…,l},i≠j, and the average $\overline{d}_1$. $d(R_t^i R_s^j)$ means the distance of task i from R^i_t, which means the destination of task i, to R^j_s, which means the source of task j.

4: consider mixed loading in a vehicle, calculate $d(R_s^i R_s^j) + d(R_t^i R_t^j)$, *i,j*∈{1,…,*l*},*i≠j* and the average $\overline{d}_2$. where $d(R_s^i R_s^j)$ means the distance of task i from R^i_s, which means the source of task i, to R^j_s, which means the source of task j.

5: the original policy is to give every task a vehicle, and calculate the value of objective function.

6: to optimize the solution, based on the location and the workload of tasks, sequential combination or mixed loading is applied alternately for the task pairs which meet combining conditions. The possible types of relative position of the tasks' source and destination is shown in Figure 1.

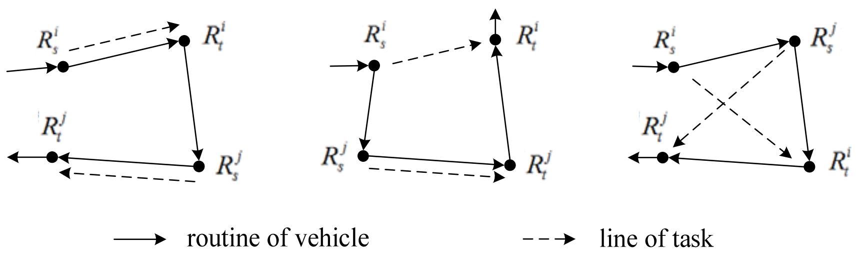

Figure 1.The possible relative position of tasks' source and destination and the routine of vehicle

Step 1: sort according to $d(R_t^i R_s^j)$. For those $d(R_t^i R_s^j) < \overline{d}_1$, get all pairs of tasks i and j. If $d(R_t^i R_s^j)$=0, consider sequential combination to i, j first. If both i and j aren't selected into the combination, go to step 3; if either i or j has already been selected into other combination, go to step 4; if both i and j are not selected, go to step 5. For $d(R_t^i R_s^j) \geq \overline{d}_1$, meaning for a long distance, the vehicle is empty, we don't consider the sequential combination at first, if possible, we consider that as a task queue in next stage. We denote those task group of i and j as (i、j), (*i*、*j*)∈G, where G is the set of task group.

Step 2: sort according to the value of $d(R_s^i R_s^j) + d(R_t^i R_t^j)$. For those satisfy $d(R_s^i R_s^j) + d(R_t^i R_t^j) < \overline{d}_2$, select the task pair i and j, $i,j \notin DZ$. If $g_i+g_j > q$, ignore this pair, until the set is empty; if $g_i+g_j \leq q$, we can combine i and j into a vehicle: if both i and j are not into any combination, go to step 6; if either i or j is in a combination, go to step 7; if both iand j are in other combinations, go to step 8. If $d(R_s^i R_s^j) + d(R_t^i R_t^j) \geq \overline{d}_2$, meaning that the two tasks are apart, if mixed loading, it may cause a lot of useless turnover.

Step 3: the destination of i is close to the source of j, the routine is $R^i_s \rightarrow R^i_t \rightarrow R^j_s \rightarrow R^j_t$. go to step 2.

Step 4: put j into task group (…,*i*,…). The order in the group (…,R^i_s,…,R^i_t,…,) remains unchanged. Make i and j in sequential, so we have the order(…,R^i_s,…,R^i_t,…,R^j_s,…,R^j_t,…). Keep the four point relative rest, adjust the position of j, select one possible and good enough result, we get a new task group and the routine in the group, go to step 2.

Step 5: task group (…,*i*,…) and group (…,*j*,…) keep their own sequencing, combine i and j in sequential. Similar to step4, combine the two groups into one. And adjust the order in the new group, if no improvement is achieved, cancel the combination; otherwise, combine into one group and go to step2.

Step 6: the sources and destinations of two tasks are nearby, we have four possible routines when mixed loading: $R^i_s \rightarrow R^j_s \rightarrow R^j_t \rightarrow R^i_t$, $R^i_s \rightarrow R^j_s \rightarrow R^i_t \rightarrow R^j_t$, $R^j_s \rightarrow R^i_s \rightarrow R^j_t \rightarrow R^i_t$, $R^j_s \rightarrow R^i_s \rightarrow R^i_t \rightarrow R^j_t$. Check these four routines, for example, for the first one, we have:

$$c_{ijji} = c_0 + c_1 t_{0R_s^i} + (c_1 + \alpha g_i) t_{R_s^i R_s^j} + [c_1 + \alpha(g_i + g_j)] t_{R_s^j R_t^j} + (c_1 + \alpha g_i) t_{R_t^j R_t^i} + c_1 t_{R_t^i 0} \quad (11)$$

$$t_{ijji} = T_s^i + T_s^j + T_t^j + T_t^i + t_{0R_s^i} + t_{R_s^i R_s^j} + t_{R_s^j R_t^j} + t_{R_t^j R_t^i} + t_{R_t^i 0} \quad (12)$$

Where $t_{ijji}$is less than *T*, otherwise the routine is unfeasible. For c_{ijji}, the first i means the source of task i, the second i means the destination of task i, so is j. And there's the same meaning for t_{ijji}. Select the best one from all feasible routines as the routine for the group and go to step1.

Step 7: insert i into group (…,*j*,…). Keep the order (…,R^j_s,…,R^j_t,…,) unchanged. For the mixed loading combination of i and j, we can insert as the following four types(…,R^i_s,…,R^j_s,…,R^i_t,…,R^j_t,…) 、 (…,R^j_s,…,R^i_s,…,R^j_t,…,R^i_t,…) 、 (…,R^j_s,…,R^i_s,…,R^i_t,…,R^j_t,…)、(…,R^i_s,…,R^j_s,…,R^j_t,…,R^i_t,…).Adjust the position of i in each type, select a best one from all feasible routines as the new group's routine. If i has to be inserted in sequential, we only have two types: (R^i_s,R^i_t,…)、(…,R^i_s,R^i_t), so we don't regard i in the group. Then go to step1.

Step 8: for task group (…,*i*,…) and task group (…,*j*,…), keeping their own order, if combine i and j into one transport process, and combine the two groups into one similar to step7. Adjust the order, if there's no improvement, cancel the combination, otherwise combine into one group and go to step1.

Step 9: if all the task pairs are checked, for those cannot be combined in sequential or combined by mixed loading, set up one group for each task.

Stage 2: combine the task groups into task queues.

For x task groups *rr*(*rr*=1, 2, …, *x*), if possible, combine several groups in sequential as task queue. That's to say, connect the last destination of a task group with the first source of another task group. A task group is similar to a task, has its own source R^{rr}_s, destination R^{rr}_t, working time T^{rr}, and cost C^{rr}. The algorithm of connect task groups is similar to C-W algorithm. At last, the task queues we get require a same amount of vehicles.

5. Example

There are 9 tasks r(r=1,…,9), the capacity of a vehicle q=5t. The speed is 50km/h, the maximum working time for a vehicle T=6h. The average loading(unloading) time is 20min. A vehicle can run 200km at most one day. The fixed cost c_0=200yuan, c_1 =10yuan/km, α=20yuan/t. The position of the station, the source and destination for each task is shown in Table 1 and Figure 1. The dotted line in Figure 1 is the line connect the source and destination for each task.

Table 1. The transport tasks

Task*r*	1	2	3	4	5	6	7	8	9
Qantity(t)	0.9	3.9	4	2.2	2.7	1	3.7	2.5	2.1
Source and destination	2→10	3→4	3→6	4→5	5→11	7→5	8→12	9→1	12→8
Destination(km)	37.7	38.8	21.5	21.9	40.6	57.3	42.8	38.9	42.8

Considering the complexity of the problem, when programming the algorithm, we denote the task with the indicator to the record, which includes all the information about the task such as number, routine, distance, time, cost and so on. The process of combination is to merge two records into one.

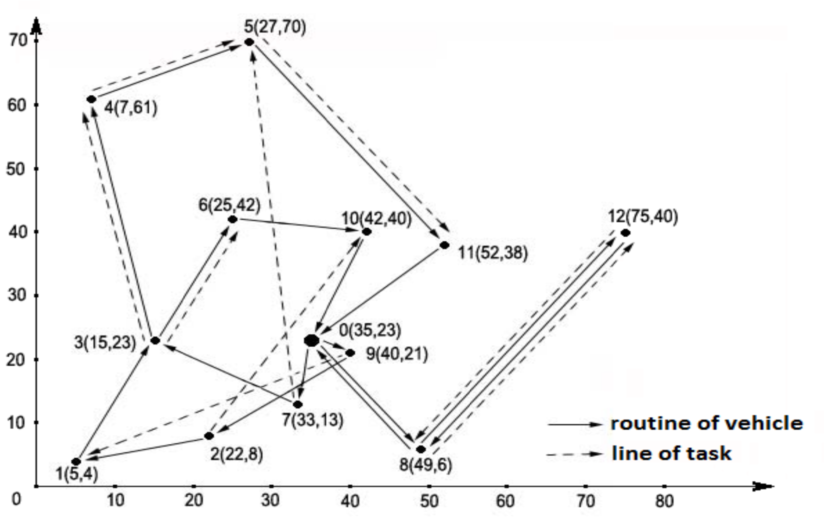

Figure 1. The tasks and the final optimal routines

In this example, we first select 2, 3, 5, 7, 8 as the main tasks. Considering combination in sequential, we have (2,4),(4,5),(6,5),(7,9). Considering mixed loading, we have combinations (1,3),(2,6),(4,6). Finally we get (8,1,3), (6,2,4,5), (7,9), and we need 3 vehicles, the value of objective function is 22555.

The task queues are:

Queue1: task group (8, 1, 3); the routine is 0→9→2→1→3→6→10→0; the distance is 123.6km, and the load is 0, 2.5, 3.4, 0.9, 4.9, 0.9, 0 successively. The working time is 4.47h.

Queue2: task group (6, 2, 4, 5); the routine is 0→7→3→4→5→11→0; the distance is 154.8km, and the load is 0, 1, 4.9, 3.2, 2.7, 0 successively. The working time is 5.76h.

Queue3: task group (7, 9); the routine is 0→8→12→8→0; the distance is 129.6km, and the load is 0, 3.7, 2.1, 0 successively. The working time is 3.93h, and the routine is shown in Figure1.

By comparing with genetic algorithm or basic ant colony algorithm, we can find the one designed in this article has higher search efficiency and can get better results as shown in Table 2.

Table 2. The comparison of experiment result

Experiment Algorithm	average success rate	average total cost
Genetic Algorithm	17%	33095
Ant Colony Algorithm	27%	29265
Grouping HeuristicsAlgorithm	48%	24685

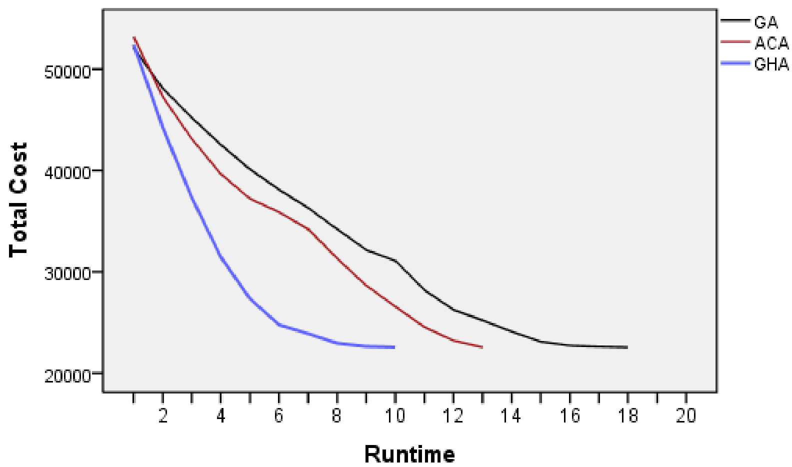

Figure 2. The comparison of three algorithms

6. Conclusion

For the complex transport scheduling problem, it’s hard to design efficient exact algorithm, so it’s necessary to find approximate algorithm. Constructing a proper heuristic algorithm and applying it to the problem to reduce the feasible zones, is a important problem for us. This paper aimed at a non-fully loaded, combined pick-up and delivery VSP, set up a more suitable changeable expenditure coefficient calculation model. Under the capacity and working time constraints, designed a grouping heuristic algorithm to solve this kind of complex VSP.

References

[1] Li J. Vehicle Routing Problem with Non-Full Load of Less Freight Quantity. *Journal of Southwest Jiaotong University*.1995; 3(1): 41-50.

[2] Xie BL, Li J, Guo YH. Genetic algorithm for vehicle scheduling problem of non-full loads with time windows. *Journal of Systems Engineering.* 2000; 13(3): 290-294.

[3] Li J. A Heuristic for Vehicle Routing with Time Windows.*Systems Engineering.*1996; 14(5): 45-50.

[4] Mosheiov, Gur. Vehicle Routing with Pick-up and Delivery: Tour-partitioning Heuristics. *Computer, Industrial Engineering*. 1998; 34(3): 669-684.
[5] Swihart MR, Papastavrou JD. A stochastic and dynamic model for the single-vehicle pick-up and delivery problem.*European Journal of Operational Research*. 1999; 114(3): 447-464.
[6] Chen JM, Zhang XM, Xiao J, Dan B. Region partitioning policy for stochastic dynamic pick-up and delivery problem. *Journal of Management Sciences in China*. 2012; 15(1): 43-53.
[7] Zhang XM, Chen JM, Xiao J Stochastic dynamic multi-vehicles pick-up and delivery problem with heavy traffic and its solution policy. *Journal of Systems Engineering.* 2012; 27(1): 61-68.
[8] N Velasco, P Dejax, C Guéret, C Prins. A non-dominated sorting genetic algorithm for a bi-objectivepick-up and delivery problem.*Engineering Optimization*. 2012; 44(3): 305-325.
[9] ZhaoK.The Research of Gray Algorithm and Information Entropy in Route Planning Optimization. *TELKOMNIKA*. 2012; 10(8): 145-154.
[10] XS Yan, QH Wu, HM Liu. An Improved Robot Path Planning Algorithm Based on Genetic Algorithm. *TELKOMNIKA*. 2012; 10(8): 28-36.