Solving a Multi-Trip VRP with Real Heterogeneous Fleet and Time Windows Based on Ant Colony Optimization: An Industrial Case Study

Jihee Han
University of Calgary
Calgary, Canada
jihee.han1@ucalgary.ca

Arash Mozhdehi
University of Calgary
Calgary, Canada
arash.mozhdehi@ucalgary.ca

Yunli Wang
National Research Council
Ottawa, Canada
yunli.wang@nrc-cnrc.gc.ca

Sun Sun
National Research Council
Waterloo, Canada
sun.sun@nrc-cnrc.gc.ca

Xin Wang
University of Calgary
Calgary, Canada
xcwang@ucalgary.ca

ABSTRACT

This paper deals with optimizing a practical variant of Vehicle Routing Problem (VRP), namely multi-trip VRP with heterogeneous fleet and time windows (MTVRPHFTW). To be able to solve this problem for industrial applications, we proposed an efficient constructive-based algorithm based on ant colony optimization (ACO) meta-heuristic. Two additional heuristics are proposed to further improve the performance of the algorithm. For evaluation, the proposed algorithm in this paper, named ACO algorithm with improvement mechanisms (IACO), is tested based on data provided by a logistics company in Canada with real-world settings. Experimental results of IACO demonstrates superiority of the proposed algorithm in terms of travelling cost, number of trips per vehicle, number of total trips, and balancing the load between the drivers compared to existing methods including the actual route history.

CCS CONCEPTS

• **Applied computing** → **Transportation**; • **Theory of computation** → *Evolutionary algorithms.*

KEYWORDS

Vehicle routing problem, Multi-trip, Heterogeneous fleet, Ant colony optimization

ACM Reference Format:
Jihee Han, Arash Mozhdehi, Yunli Wang, Sun Sun, and Xin Wang. 2022. Solving a Multi-Trip VRP with Real Heterogeneous Fleet and Time Windows Based on Ant Colony Optimization: An Industrial Case Study. In *The 15th ACM SIGSPATIAL International Workshop on Computational Transportation Science (IWCTS '22), November 1, 2022, Seattle, WA, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3557991.3567776

IWCTS '22, November 1, 2022, Seattle, WA, USA

ACM ISBN 978-1-4503-9539-7/22/11…$15.00
https://doi.org/10.1145/3557991.3567776

1 INTRODUCTION

In this article, we focus on solving a practical variant of VRP, Multi-Trip VRP with Heterogeneous Fleet and Time Windows (MTVRPHFTW), for a logistics company in Canada. In comparison with basic VRP, MTVRPHFTW allows to have heterogeneous fleets with different capacity. Compared to many of studies in the literature, investigating the heterogeneous VRP-based variants that only consider weight as a constraint for the capacity of the vehicle, in this paper, we also take the constraint with regards to the number of skids into account. In MTVRPHFTW, each vehicle can operate multiple trips in a planning horizon. The MTVRPHFTW considers a time window for each customer within which the customers should be served.

Contrary to VRP, research on MTVRPHFTW is scarce in the literature. More specifically, there are very limited studies investigating the problem and benchmarking the performance with realistic settings and use a real industrial data. [5] studied the multi-trip VRP with a heterogeneous fleet for a real industrial case of a furniture manufacturer, and used a combination of constructive and improvement heuristics for solving it. This paper does not consider any time windows constraint for delivery. In [7], VRP with heterogeneous fleets, without considering time-window for the customers' serving, for two real case studies of a diary and a construction company is studied. [2] developed simulated annealing algorithm for MTVRPHFTW and tested it by artificially generated data instead of real industrial data. [1] studied a primal-dual formulation for an exact solution of MTVRPHFTWA rather than provided the practical algorithm for industrial applicability.

The solution methods for VRP can be mainly classified into exact algorithms and heuristics which is further classified into constructive and improvement approaches. As the MTVRPHFTW is an $\mathcal{NP}$-Hard problem, the exact algorithms are computationally intractable for real-world cases [6]. Improvement approaches can be useful for practical use but it requires initial feasible solution, which is difficult to find for most VRP variants [6]. To this end, in this paper, for solving MTVRPHFTW, a practical variant of VRP, we developed an algorithm, named IACO, based on ACO constructive-based meta-heuristic with two additional mechanisms for improving the algorithm's performance.

The remainder of the paper is organized as follows. In Section 2, we present the problem definition and assumptions for MTVRPHFTW in this study. Section 3 and 4 provide the proposed IACO algorithm and the results of the numerical experiments. Conclusions and future work are presented in Section 5.

2 PROBLEM DEFINITION AND ASSUMPTIONS

The MTVRPHFTW can be described through a complete graph $G = (N, E)$, where the node set $\{0, 1, 2, ..., |N|\}$ is the set of geographically dispersed locations, where the member 0 in the set represents the depot and the rest of the members standing for the customers. Each customer i has a demand, denoted by d_i^{ws}, associated with weight and number of skids of the demand. Each customer should be served within a time window denoted by $[et_i, lt_i]$, where et_i and lt_i are the earliest and latest time, respectively. st_i represents the service time to take at each node i, where it is a loading time at depot for $i = 0$. The travel time from node i to node j is represented by tt_{ij}. We also define a set $H = \{0, 1, 2, ..., |H|\}$ to denote heterogeneous fleets in terms of capacity. C_f^{ws} represents the capacity of vehicles in terms of weight and number of skids, $f \in H$. f_i denotes the preference of the customer i with respect to the type of vehicle to be used for delivery. The value for f_i is 0 if the corresponding customer has no preference for the vehicle type. The maximum working hours in a day for each vehicle is denoted by T.Figure 1 illustrates a simple example for MTVRPHFTW, where multiple trips are performed by three heterogeneous fleets and customers are serviced while satisfying the time windows.

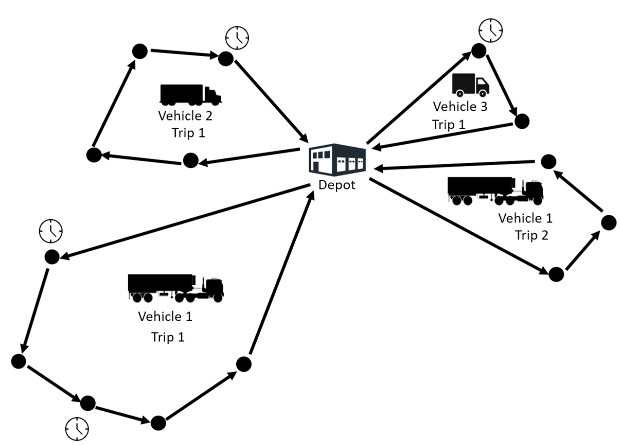

Figure 1: An example of MTVRPHFTW

3 METHODOLOGY

In this section, IACO algorithm for MTVRPHFTW is proposed for practical use in industry. The underlying ACO for IACO is first provided in Algorithm 1 (Section 3.1). Then how to incorporate multiple constraints into the underlying ACO is developed as well as two additional mechanisms to improve the solution quality in Function 1 (Section 3.2).

3.1 Underlying ACO

As the underlying method, this paper uses ACO algorithm which mimics the intelligence of ant colony in searching food sources by depositing pheromones on routes [3]. Algorithm 1 describes the overall underlying ACO algorithm with application to VRP. Given a set of ants denoted as M, each ant $k \in M$ constructs a solution by selecting a next node repeatedly, where a solution is represented as a sequence of node $i \in N$, starting and returning from depot. Then once a set of ants builds solutions, the pheromones get updated by deposition and evaporation. As in Equation (1), the pheromone from node i to j at iteration t, $\tau_t(i, j)$, gets evaporated with the rate of ρ ($0 < \rho < 1$). And it gets deposited by the amount of the pheromone by a set of ant $k \in M$, $\triangle\tau_{t-1}^k(i, j)$, which is obtained by the inverse of the length of its solution, L_k ($\triangle\tau_{t-1}^k(i, j) = \frac{1}{L_k}$).

$$\tau_t(i, j) = (1 - \rho) \cdot \tau_{t-1}(i, j) + \sum_{k=1}^{|M|} \triangle\tau_{t-1}^k(i, j) \tag{1}$$

Regarding how to choose a next node, ants select a next node based on the transition probability. As in Equation (2), the probability from node i to j at iteration t, $\Pr_t(i, j)$, is calculated based on the amount of pheromone and how close those two nodes are, which is called the visibility. Visibility from node i to j is denoted as $\eta(i, j)^\beta$ where β is a constant regarding importance of visibility. Q is a set of feasible neighbor nodes that can be visited from node i. Choosing a next node to visit is repeated until all customers are visited.

$$\Pr_t(i, j) = \frac{\tau_t(i, j) \cdot \eta(i, j)^\beta}{\sum_{q \in Q} \tau_t(i, q) \cdot \eta(i, q)^\beta} \tag{2}$$

The computational complexity of the proposed algorithm is $O(|N||M|\mathcal{T})$, where $\mathcal{T}$ stands for the number of iterations.

3.2 Constraints Handling in Constructive Approach

As the MTVRPHFTW involves multiple constraints, this section deals with how to handle those constraints through a function ANTMOVEMENT (Function 1). Multi-trip constraint is handled by keeping track of vehicle index with trips as well as vehicle change decision. Heterogeneous fleet and time windows constraints are mainly handled by filtering out non-feasible customers.

As MTVRPHFTW has a time windows constraint, a variable for the accumulated working time (denoted as curr_time) is considered to not violate the time windows constraint. We need to also keep track of the current fleet type, denoted as curr_f, to meet the heterogeneous fleet constraint. As each vehicle performs multiple trips and vehicle is subject to be changed if there is no customer serviceable by the vehicle, a variable for identifying a vehicle (curr_vid) is needed to keep track of trips performed by vehicles (Line 5 in Function 1).

Additionally, since our industrial case allows customers to not have the fleet type restriction, which is labeled as 0, we let the fleet type be specified as 1 (largest fleet) if the curr_f remains not specified until a complete trip is made (Line 45-47 in Function 1). This assumption is to prevent the load from exceeding the capacity. The heterogeneous fleet constraint also makes a minor change on the capacity constraint (Line 33 in Function 1), due to the zero labeled fleet type and the inclusive relationship on fleet types.

Even though the constructive approach has an advantage in handling multiple constraints, it usually has limited search ability because solution is repeatedly constructed one at a time depending on the current state. To overcome the limitations, we developed two additional improvement mechanisms, which are the early vehicle change and the weighted probability for time-restricted customers.

Early vehicle change

With respect to the vehicle change in multi-trip VRP, the constructive approach reveals a disadvantage through the maximal usage of vehicle such as the last trip visiting only a small number of customers because of insufficient working time left. Regarding this case, changing a vehicle earlier than its maximal usage can be considered. That is, the vehicle is subject to be changed proportionally based on the remaining time (Line 19-23 in Function 1) where a flag variable (early) is defined to indicate whether the vehicle is changed to new one before the must-change. Once the vehicle is decided to be changed, the curr_time and curr_f are initialized as well as the curr_vid is increased by 1 (Line 24-31 in Function 1).

Weighted probability for time-restricted customers

Another mechanism focuses on the customers who requests the appointment time early in the morning. Those customers are likely to be filtered out during search and be covered later by new vehicles increasing the number of vehicles. To handle this situation, we let those customers have the weighted probability to give more importance in the beginning of each vehicle. ω_i is the weight to apply on customer i, calculated based on the maximum working hour T (=720 min) and the latest time of the customer i, represented as the appointment time (see Equation (3)). Also, the time threshold parameter TH is used to focus on the customers who request the early appointment time (Line 8-10 in Function 1).

$$\omega_i = \frac{T}{lt_i}, i \in N \tag{3}$$

ALGORITHM 1: IACO for MTVRPHFTW

```
Data: MTVRPHFTW instance
Result: Best solution
Initialize env        // (ACO environment) Pheromone and probability
colony ← ∅
for each ant k ∈ M do
    colony.append (InitializeSol())        // Sequence and fitness
end
best ← InitializeSol()
while termination criteria is not met do
    for each ant k ∈ M do
        sol ← InitializeSol()
        sol.route ← AntMovement(env)
        sol.fitness ← EvaluateSol (sol.route)
        Remove the worst solution from colony
        colony.append(sol)
    end
    Update env.pheromone                          // Equation (1)
    Update env.probability                        // Equation (2)
    if min(colony.fitness) < best.fitness then
        m ← argmin(colony.fitness)
        best ← colony[m]
    end
end
```

FUNCTION 1: ANTMOVEMENT

```
1  Function AntMovement(env):
2      route ← ∅
3      unvisit ← [1, 2, ..., |N|]
4      visit ← 0
5      curr_loc, curr_time, curr_vid, curr_f, curr_load^{ws} ← 0
6      while unvisit is not empty do
7          pr ← env.probability[curr_loc, :]
8          if curr_time < TH then
9              pr ← pr · ω_i for ∀i ∈ N                    // Equation (3)
10         end
11         pr[visit] ← 0
12         pr[curr_time + tt_{curr_loc,j} > lt_j] ← 0 for ∀j ∈ N
13         pr[curr_time + tt_{curr_loc,j} + tt_{j,0} + st_j > T] ← 0 for ∀j ∈ N
14         if curr_f ≠ 0 then
15             pr[f_i > curr_f] ← 0 for ∀i ∈ N
16             pr[d_i^{ws} > C_{curr_f}^{ws}] ← 0 for ∀i ∈ N
17         end
18         early ← 0
19         if curr_loc == 0 then
20             if rand() > 1 - curr_time/T then
21                 early ← 1
22             end
23         end
24         if sum(pr) == 0 | early == 1 then
25             next ← 0
26             curr_load^{ws} ← 0
27             if curr_loc == 0 then
28                 curr_vid ← curr_vid + 1
29                 curr_time, curr_f ← 0
30             end
31         else
32             next ← SelectNode(pr)
33             if curr_load^{ws} + d_next^{ws} ≤ C_{max(f_next,curr_f)}^{ws} then
34                 if curr_f == 0 then
35                     curr_f ← f_next
36                 end
37                 unvisit.remove(next)
38                 visit.append(next)
39                 curr_load^{ws} ← curr_load^{ws} + d_next^{ws}
40                 curr_time ← curr_time + tt_{curr_loc, next} + st_next
41                 if curr_loc == 0 then
42                     curr_time ← curr_time + st_0
43                 end
44             else
45                 if curr_f == 0 then
46                     curr_f ← 1
47                 end
48                 next ← 0
49                 curr_load^{ws} ← 0
50                 curr_time ← curr_time + tt_{curr_loc, 0}
51             end
52         end
53         route.append(next)
54         curr_loc ← next
55     end
56 return route
```

4 EXPERIMENTS

4.1 Experimental Design

For the testing instances, we used 8 industrial instances from the logistics company in Canada, where the number of deliveries is 99.75 on average. The company has three different fleets that are heterogeneous in capacities (weight and skid). The largest truck is indexed as 1, and the mid and smallest trucks are indexed as 2 and 3, respectively. Each customer is labeled by one of the fleet type or 0 which represents no restriction in fleet type. Drivers start working

Table 1: Improvement from basic ACO

	ACO[b]	IACO[a]	IACO$_{w/1}$	IACO$_{w/2}$	Impv.[a,b]
Total distance	1168.80	1124.98	1134.48	1160.68	-3.75%
No. of vehicles	10.03	10.35	10.50	9.70	3.24%
No. of trips per vehicle	3.34	3.01	2.99	3.42	-9.78%
No. of total trips	33.28	31.13	31.30	33.03	-6.46 %
Load imbalance	1.42	1.01	1.04	1.33	-29.29 %

from 6:00 am and return to depot by 6:00 pm. Loading time at depot and service time at customers are set to 60 and 20 min, respectively. We also assume that truck speed is 0.7km/m (= 42km/h).

The objective function is to minimize the total distance. Other measures such as number of vehicles, number of trips per vehicle, number of total trips and load imbalance (standard deviation on the number of trips that each vehicle (driver) has in a day) are also obtained to discuss the practical applicability in real-life. In ACO, the number of ants, number of iterations, and evaporation rate on pheromone are set to 100, 300 and 0.1, respectively. Due to the randomness of ACO search, we ran 5 times per each instance for ACO-based algorithms and used the average value.

Experiments are first conducted within the ACO method to see how the improvement mechanisms are effective in enhancing the solution quality. Then, comparative experiments are conducted with the Sweep algorithm [4] as one of the traditional VRP algorithms and the actual industry solution obtained from the company.

4.2 Experimental Results

The results about the effectiveness of the additional improvement mechanisms are summarized in Table 1, where the remaining time based early vehicle change and the weighted probability for time-restricted customers are labeled as 1 and 2, respectively. Improvement from ACO is calculated as $\frac{\text{IACO-ACO}}{\text{ACO}} \times 100\%$. Table 2 summarizes the results where the improvements from Industry (actual industry history) and from the Sweep algorithm are calculated as $\frac{\text{IACO-Industry}}{\text{Industry}} \times 100\%$ and $\frac{\text{IACO-Sweep}}{\text{Sweep}} \times 100\%$, respectively.

The proposed method, IACO, was able to reduce the total distance from ACO about 3.75% (see Table 1). It also achieved a smaller number of trips per vehicle and of total trips as well as more balanced load for drivers, while the number of vehicles increased from ACO. Applying the early vehicle change, IACO w/1, generated larger number of vehicles (10.5) than the ones without it (10.03 and 9.70), but number of trips per vehicles and of total trips are reduced and overall load balance gets improved. It also appears to affect the distance reduction which can be seen as changing vehicles earlier than required makes better trip opportunity for subsequent vehicles. Applying the weighted probability for time-restricted customers, IACO w/2, seems to be effective reducing number of vehicles by showing the smallest number of vehicles (9.70).

Table 2 shows that the industry history had the 18.79% and 24.04% increases in the total distance and number of vehicles from the proposed method, respectively. However, it showed the lowest load imbalance compared to other algorithms which appears to result in the large number of vehicles and smaller number of trips per vehicle. This indicates that the company considers the load balancing important so that the drivers can have similar number of

Table 2: Summary of comparative experiment with existing methods

	IACO[a]	Industry[b]	Sweep[c]	Impv.[a,b]	Impv.[a,c]
Total distance	1124.98	1385.34	1482.99	-18.79%	-24.14%
No. of vehicles	10.35	13.63	10.88	-24.04%	-4.83%
No. of trips per vehicle	3.01	2.75	3.55	9.47%	-15.22%
No. of total trips	31.13	37.63	38.63	-17.28%	-19.42%
No. of customers per trip	3.24	2.67	2.60	21.11%	24.60%
Load imbalance	1.01	0.62	1.47	62.60%	-31.68%

trips per day. Sweep algorithm showed the largest distance among all algorithms. It also showed large number of trips per vehicle, which happened because the number of customers visited in each trip is relatively small. It indicates that choosing a next node only based on the angle order can incur inefficiently frequent trips since the selected next node might not be able to be visited due to the capacity constraint and the vehicle needs to be go back to the depot for capacity initialization. On the other hand, ACO-based algorithms appear to be more effective to avoid that situation by selecting a next node based on the transition probability obtained from the pheromone over iterations.

5 CONCLUSION

This paper focuses on solving the realistic VRP, namely MTVRPHFTW, for practical use in industry. The proposed method is ACO-based algorithm with additional mechanisms to improve the solution quality. Our work can be useful for the last-mile logistics companies, as the method was developed with the real industrial data and the results showed that it outperforms the existing methods. For the future work, developing a machine learning based algorithm for MTVRPHFTW can be studied as the machine learning technique would have an advantage over the meta-heuristics based on the trained model.

REFERENCES

[1] Marco Boschetti and Vittorio Maniezzo. 2015. A set covering based matheuristic for a real-world city logistics problem. *International Transactions in Operational Research* 22, 1 (2015), 169–195.

[2] François Despaux and Sebastián Basterrech. 2014. A study of the multi-trip vehicle routing problem with time windows and heterogeneous fleet. In *2014 14th International Conference on Intelligent Systems Design and Applications*. IEEE, 7–12.

[3] Marco Dorigo, Vittorio Maniezzo, and Alberto Colorni. 1996. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 26, 1 (1996), 29–41.

[4] Billy E Gillett and Leland R Miller. 1974. A heuristic algorithm for the vehicle-dispatch problem. *Operations research* 22, 2 (1974), 340–349.

[5] Christian Prins. 2002. Efficient heuristics for the heterogeneous fleet multitrip VRP with application to a large-scale real case. *Journal of Mathematical Modelling and Algorithms* 1, 2 (2002), 135–150.

[6] Martin WP Savelsbergh. 1985. Local search in routing problems with time windows. *Annals of Operations research* 4, 1 (1985), 285–305.

[7] Christos D Tarantilis and Chris T Kiranoudis. 2007. A flexible adaptive memory-based algorithm for real-life transportation operations: Two case studies from dairy and construction sector. *European Journal of Operational Research* 179, 3 (2007), 806–822.

ACKNOWLEDGMENTS

The authors would like to thank Artificial Intelligence for Logistics program at the National Research Council Canada and the Natural Sciences and Engineering Research Council of Canada (NSERC) for financial support.