Clustered Tabu Search Optimization for Reservation–based Shared Autonomous Vehicles

Shun Su[a], **Emmanouil Chaniotakis**[b,*], **Santhanakrishnan Narayanan**[a], **Hai Jiang**[c], **Constantinos Antoniou**[a], **and** [2]

[a]University of Kassel
[b]University College London
[c]Tsinghua University
[*]Corresponding author email: m.chaniotakis@ucl.ac.uk

Abstract

This paper investigates the optimization of Reservation–based Autonomous Car Sharing (RACS) systems, aiming at minimizing the total vehicle travel time and customer waiting time. Thus, the RACS system and its routing are formulated with a consideration for system efficiency and passengers' concerns. A meta–heuristic Tabu search method is investigated as a solution approach, in combination with K–Means (KMN–Tabu) or K–Medoids (KMD–Tabu) clustering algorithms. The proposed solution algorithms are tested in two different networks of varying complexity, and the performance of the algorithms are evaluated. The evaluation results show that the TS method is more suitable for small scale problems, while KMD–Tabu is suitable for large scale problems. However, KMN-Tabu has the least computation time, although the solution quality is lower.

Keywords: Autonomous Vehicles (AV); Shared Autonomous Vehicle (SAV); Shared mobility; Car sharing; Vehicle assignment; Routing optimization; Tabu search; Clustering

1 Introduction

The emergence of Autonomous Vehicles (AVs) is expected to overcome challenges of current shared mobility services and revamp the system with advanced self–driving relocation and re–balancing technologies, using which mobility service providers could gain more potential benefits through reduction of operation costs and enhancement of the efficiency of service [7]. Although skepticism exists, Shared Autonomous Vehicles (SAVs) are widely being discussed in the literature [6, 27]. Existing studies in the pertinent literature usually focus on real-time/dynamic services [11, 15, 17]. However, the benefits of an SAV service can be increased by employing a reservation option [20]. Such a system is called Reservation–based Autonomous Car Sharing (RACS) system, wherein the demand requests of customers who have planned their trips in advance are fulfilled [16]. By employing a fleet of SAVs to serve the fixed pre-planned trips, reservation based systems usually can result in higher routing quality and better assignment solution, due to the possibility to have relatively larger searching space and longer computation time [16, 23].

RACS systems can provide mobility services based on the pickup and drop off points of the passengers. This is a typical problem in the literature of vehicle routing optimization, commonly known as the Dial–A–Ride–Problem (DARP) with time windows [1, 5]. Since reservation–based service assumes to obtain all the relevant information (demand and predicted travel–cost between the nodes) in advance, RACS optimization problem is static and deterministic. Static DARPs are typically viewed as combinatorial optimization problems, which possess NP hard properties, and use discrete variables [22]. Reservation–based car sharing systems (both autonomous and non–autonomous) are usually modelled based on Linear and Mixed–Integer programming [5, 9]. For example, Cordeau [3] proposes a mixed–integer program to model a non–autonomous system, and uses Branch and Cut (BC) algorithm to find a set of minimum–cost vehicle routes. His algorithm can only be applied to relatively small–scale problems. For autonomous systems, Ma et al. [16] apply a linear programming approach with a trip–chain concept to minimize the operation cost of the system.

However, in this paper, instead of just operating cost, we minimize a combination of two performance metrics, operation cost (total system travel time) and service discomfort (customer waiting time). We follow a mixed integer programming approach for the same.

In real life conditions, thousands of requests may be served by a RACS system per day. For higher numbers of requests, efficient algorithms are usually investigated, aiming to obtain a better quality solution within a reasonable computation time [18]. Although a wide range of optimization algorithms are seen in the literature, such as Genetic Algorithms [12] and Particle Swarm Algorithms [28, 10], the meta–heuristic Tabu–Search (TS) method remains as one of the most widely used algorithms. This method has been proved to be a powerful and efficient algorithm for DARPs, which can handle large scale problems to obtain high quality solutions. However,the computation time of TS algorithms can still be high [4, 13, 21]. Clustering methods, such as K–Means [24] and K–Medoids [18], can be used along with the TS method to obtain solutions for large scale DARPs, wherein the intent is to decompose the original problem into several independent smaller sub–problems. With the decomposition of the original problem into several smaller sub–problems, computation time can be dramatically reduced by utilising parallel computation techniques. Hence, in this research work, we explore the meta–heuristic TS method based on Cordeau [4], in combination with different clustering methods, aiming at extending the literature on efficiently solving the large scale RACS routing problem.

The contributions of this work include: (i) Formulation of the RACS problem with consideration for both operation cost (total system travel time) and service discomfort (customer waiting time) and (ii) Exploration of efficient handling of the formulated RACS problem, through the use of the TS method in combination with the K–Means and K–Medoids clustering methods. The exploration of the K–Means and K–Medoids clustering methods will help one to better understand the effect of clustered optimization on the computation time reduction, and also make the handling of larger problems feasible. This paper is structured as follows: the problem formulation is presented in the next section (Section 2.1), followed by the description of the solution approach (Section 2.2). Then the case studies are presented (Section 3). Finally conclusions are drawn, along with a discussion on future research directions (Section 4).

2 RACS problem & solution

Consider a fleet of SAVs that has to serve a set of known requests, where several requests can be chained to form a route and assigned to an SAV. We study the problem of finding routes of a fleet of SAVs, which are formed by reserved requests, that achieve the objective of minimum operation cost and service discomfort. In this research work, RACS problem is formulated for a single horizon, and without ride sharing.

2.1 Problem formulation

For the formulation of the RACS problem, we adapt the mixed integer program of Cordeau [5], to optimally plan K routes with a fleet of SAVs. The objective is to minimize total travel time and customer waiting time, as shown in Equation 1.

$$\min\left(\sum_{k\in K}\sum_{i\in V}\sum_{j\in V}c_{ij}x_{ijk}+\sum_{k\in K}\sum_{i\in V}TW_{ik}\right) \qquad (1)$$

where x_{ijk} is a binary decision variable, which takes the value 1 if an SAV k traverses arc (v_i, v_j). $V = v_0, v_1, \cdots, v_{2*n}$ denotes the set of all nodes in the network $G = (V, A)$, with $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ representing the arc set. Vertice v_0 is the depot, $P = \{v_1, v_2, ...v_n\}$ is the set of pick–up vertices and $D = \{v_{n+1}, v_{n+2}, ...v_{2*n}\}$ is the set of drop–off vertices. Each vertex pair $(v_i; v_{n+i})$ represent a customer request from origin v_i to destination v_{n+i}. Each vertex in P has a corresponding ride duration L_i (with maximum ride duration L), a time window of (e_i, l_i), where e_i and l_i denote the earliest and latest pick–up time of request i. Pick–up and drop–off vertices and earliest and latest pick–up times together constitute a request matrix R. Homogeneous SAVs with a fixed load capacity and a fleet size of K serves the given demand, with k referring to a random SAV. While c_{ij} is the travel time for using an arc (v_i, v_j), TW_i is the waiting time corresponding to each customer request. The set of travel times between all the nodes in the network constitute the travel cost matrix C.

The aforementioned original problem (Equation 1), upon clustering, gets decomposed into the following sub–problems:

$$\begin{aligned} &\min\left(\sum_{k\in K}\sum_{i\in V}\sum_{j\in V} c^1_{ij}x^1_{ijk} + \sum_{k\in K}\sum_{i\in V} TW^1_{ik}\right), \\ &\quad\vdots \\ &\min\left(\sum_{k\in K}\sum_{i\in V}\sum_{j\in V} c_{ij}x^M_{ijk} + \sum_{k\in K}\sum_{i\in V} TW^M_{ik}\right) \end{aligned} \tag{2}$$

where M represents the number of clusters. The original problem and the sub–problems should conform to the following: (1) For each request i, both the pick–up and drop–off nodes (v_i, v_{n+i}) should belong to the same route, thus visited by the same vehicle, (2) Pickup node v_i should be visited before drop off node v_{n+i}, (3) Request–serving at node v_i needs to start its pickup activity within a time window (e_i, l_i), (4) Each vehicle starts and ends its route at the depot, (5) Ride time L_i is the difference between the pickup time and drop off time and it should not exceed the maximum trip duration L, and (6) Each vehicle entering a node, leaves it in a feasible sequence. Although these can be expressed mathematically, we refrain from doing so because of space constraints and our focus on meta–heuristics.

2.2 Solution methodology

The RACS model consists of K vehicle routes with corresponding requests on network G. By assigning SAV trips to the network, a set of SAV chains is obtained as output, with minimum total travel time and customer waiting time. Three different algorithms are investigated in this research, namely Tabu–Search (TS), K–Means Tabu–Search (KMN–Tabu) and K–Medoids Tabu–Search (KMD–Tabu) methods. TS is an efficient meta–heuristic method to escape the local optimal solutions, which usually occurs when solving DARPs. In this research, we apply TS algorithm according to Cordeau et al. [4], with soft time–window constraints. The type of TS implemented is the reactive TS, which is a robust search technique that enhances classical TS, by automatically adjusting the search parameters based on the state and quality of the search. Reactive TS enables obtaining near optimal solutions with relatively small computational effort [19]. The method includes the following: a relaxation mechanism, neighbourhood structure, diversification strategy, initial solution construction, TS iteration, neighborhood evaluation and neighborhood reduction. The TS algorithm uses simple insertion, i.e., remove a vertex pair $(v_i; v_{i+n})$ from its current route and reinsert it in a different route. Each simple insertion, is added to a Tabu list and will be removed from it after few iterations ($tabu_{max}$). A Tabu check is used, wherein the existence of the request node selected in a route is checked in previous solutions, stored in the Tabu list. If the selected move results in a route already existing in the Tabu list, then the move is skipped. If not, the route cost of this move is evaluated and if it reduced, the move is realised. For a detailed description of the TS algorithm, the reader is referred to Cordeau et al. [4]. In addition, α, is introduced as a self adjustment parameter, allowing the violation of the time window constraints. This parameter is multiplied with the total waiting time in the objective function, i.e., $\alpha \sum\sum TW_{ik}$. After each TS iteration, the value of α is modified by a factor of $1 + \sigma$, where $\sigma > 0$. If the current solution is feasible with respect to time window constraint, the value of α is divided by $1 + \sigma$; otherwise, it is multiplied by $1 + \sigma$.

The TS solution procedure used is shown in Algorithm 1. First few iterations ($iter_s$ set to 30 iterations, upon results of a sensitivity analysis) are considered to be a warm-up phase to move across the solution space and hence, Tabu check is carried out after that. Adding solutions in the tabu list takes place after some iterations ($iter_k$ set to 25). The number of iterations after which the insertions are removed from the list ($tabu_{max}$) is determined upon a sensitivity analysis to ensure better solution quality (values between 1-10). The value used for the parameter σ is 0.0005 and the initial value of α is set to 1. The algorithm runs for a maximum number of iterations ($iter_{max}$) determined upon a sensitivity analysis as with $tabu_{max}$.

Algorithm 1 TS method

```
Input: $G(V, A)$, $R$, $K$, $C$, $\alpha$, $\sigma$, $iter_s$, $iter_k$ $tabu_{max}$, $iter_{max}$
Construct a random initial solution $S_0$.
while iteration < $iter_{max}$ do
    for all $K$ do
        for all requests served by $k$ do
            Select insertion move $m$
            if iteration > $iter_s$ then
                Perform Tabu check
                if Tabu check != True then
                    Calculate route cost with the move ($Cost_k^m$)
                    if Route cost is lower than previous ($Cost_k^m < Cost_k$) then
                        Make the move, adjust costs
                    end if
                end if
            end if
        end for
    end for
    Calculate the new objective value and compare it with the previous best. If the new value is better, update it as the previous best.

    if iteration > $iter_k$ then
        Add the move to tabu list and remove the same from the list after $tabu_{max}$ iterations
    end if
    Adjust $\alpha$
end while
```

Concerning clustering approaches, K–Means is an unsupervised clustering approach, which finds clusters based on observations' similarity. The approach is an iterative process, consisting of the following steps: (a) Selecting cluster centroid, (b) (Re)assigning points to the closest centroid and (c) Estimating cluster variation using a distance measure between the cluster points and the centroid. The iteration occurs until there is no significant change in the cluster variations. For detailed information on K–Means clustering, the reader is referred to the work of Forgy [8]. Similar to K–Means, K–Medoids is also an unsupervised clustering approach, which is based on the Medoid shift algorithm [18]. Compared to the K–Means, K–Medoids is more robust to noise and outliers, as it minimizes pairwise dissimilarity distance, rather than squared distances. We perform K–Medoids clustering using Partitioning Around Medoids (PAM) algorithm. PAM is an iterative building and a swapping procedure, based on incremental stepping heuristics. For details about this algorithm, the reader is referred to Muelas et al. [18]. In this research work, we cluster the original demand (customer requests) based on the earliest pick–up and drop–off time of the requests (each request is represented as a node based on these two). Clustered TS optimization solution procedure is shown in Algorithm 2.

Algorithm 2 Tabu-KMN/Tabu-KMD method

```
Input: $G(V, A)$, $R$, $K$, $C$, $\alpha$,$\sigma$, $Tabulist_{max}$, $iter_{max}$
Create clusters (sub–problems) from the original request matrix using the K–Means [8] / K–Medoids [18] algorithm
Distribute each cluster to a computing node and carry out parallel execution of the TS algorithm (Algorithm 1)
Receive the routes from all the computing nodes and join them together to obtain the final solution.
```

3 Case studies & results

In order to explore the efficiency of the proposed solution algorithms (TS, KMN–Tabu and KMD–Tabu), we utilise two different networks of increasing size and complexity. We use an Intel Core i5–7400CPU 3.00 GHZ computer, which can execute parallel computation with a maximum of 4 cores. For each algorithm, three runs are performed to ascertain the mean and standard deviation of the objective value. The results from the case studies (objective function, total travel cost and waiting time) are shown in Figure 1.

3.1 Sioux–Falls

The small scale problem of Sioux–Falls [26] comprises of 24 network nodes. 100 trips requests are randomly generated using a 15min time window, and are assumed to be served by 40 vehicles in the service horizon of 300 minutes (5 hours). Travel time matrix is constructed based on the data from [26]. For KMN–Tabu and KMD–Tabu, the requests are clustered into four groups.

Objective function, total travel cost and waiting time values based on the evaluation results of the three algorithms are shown in Figure 1a. Based on the evaluation results, it is found that the mean of the objective function is 4113.7, 5068 and 4970 for TS, KMNTabu and KMDTabu respectively. From the results, it is obvious that the TS method produces the highest solution quality among the three. By applying the KMN–Tabu and KMD–Tabu algorithm, the initial problem is decomposed into four sub–problems. Thus, the solution quality of KMN–Tabu and KMD–Tabu are lower than the TS method, which is because of the limited searching space. However, the computation time for TS method is found to be five times higher than that of the other two methods (For 1000 iterations: 545, 77 and 86 seconds for TS, KMN–Tabu and KMD–Tabu respectively). From the above, TS method appears to be more appropriate for small scale RACS problems, as it results in a better solution.

3.2 New York City

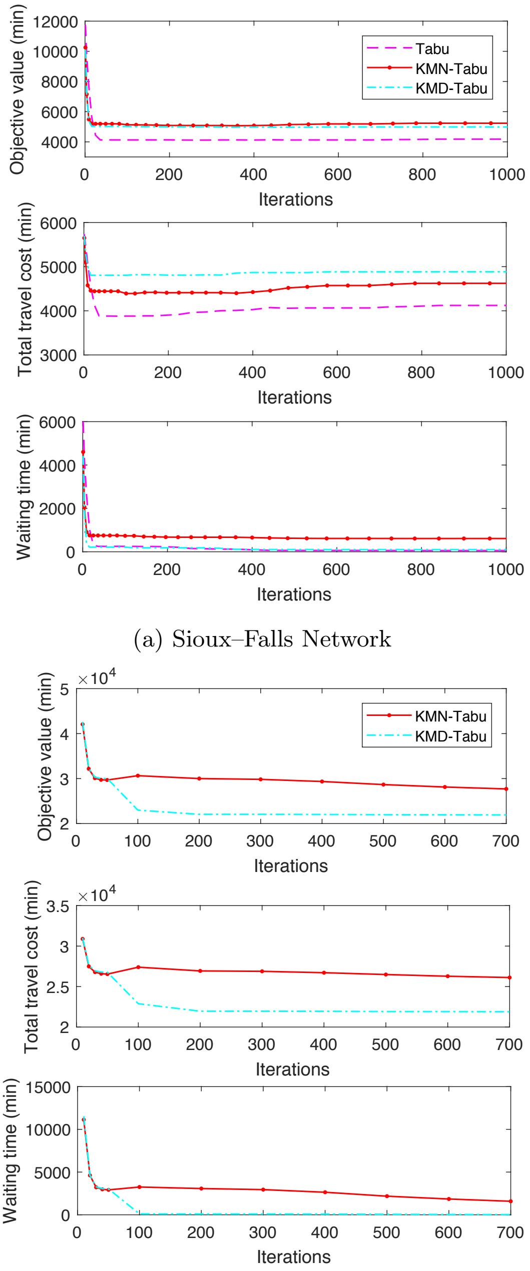
For the New York city case study, we use a sample of 3 hour demand from the Green Taxi dataset, published by the Taxi and Limousine Commission (TLC) [25]. This sample contains 3,968 requests. The TLC zoning system is used, with centroids representing pick–up and drop–off points. The maximum waiting time for the customers is assumed to be 15 minutes, and the travel time matrix is estimated based on city block distance metric. Attempts to run the TS algorithm failed, with computation times exceeding 15 days. Hence, only KMN–Tabu and KMD–Tabu algorithms are evaluated for this case study. The requests are clustered in 100 groups, where the customer requests of each group are assumed to be served by 8 SAVs.

Objective function, total travel time and waiting time values based on the evaluation results of the two algorithms are presented in Figure 1b. The results show that the objective value is 27,042 and 21,923 for KMN–Tabu and KMD–Tabu respectively. Thus, KMD–Tabu outperforms KMN–Tabu, by producing a better solution with lower objective function value, within fewer iterations. However, the computation time of KMD–Tabu is much higher than that of KMN–Tabu (For 700 iterations: 2,643 and 301 seconds for KMD–Tabu and KMN–Tabu respectively), although the computation time is still reasonable. From the above, for large scale RACS problems, KMD–Tabu appears to be suitable. If the computation time is of prime importance, then KMN–Tabu is more suitable, though the solution quality will get compromised.

4 Conclusions & future works

This research aims at contributing to the RACS routing problem by formulating the problem as a mixed–integer program. The meta–heuristic method of Tabu search has been utilized as a solution procedure. Furthermore, K–Means and K–Medoids clustering methods are combined with Tabu search to decompose the initial problem into smaller sub problems, in order to ascertain the possibility of reducing computation time. Although meta–heuristics do not guarantee a global optimal solution, Tabu–search and its extensions are found to be effective in producing good quality solutions [21, 14, 9]. Our findings suggest that the Tabu–search method is suitable for small scale problems, while it is not appropriate for large scale problems, because of very high computation time. K–Means Tabu (KMN–Tabu) and K–Medoids Tabu (KMD–Tabu) algorithms are found to be better for large scale problems. From a solution quality perspective, KMD–Tabu proves to be better than KMN–Tabu. However, KMN–Tabu outperforms KMD–Tabu, when computation time is considered.

Future research should focus on multi–planning horizons with more complex problem settings. Concerning complex problem settings, inclusion of operation–related factors in the optimization formulation, such as the cost of temporary vehicle parking and electric vehicle charging [2], could be explored. Additionally, since our model assumes



(a) Sioux–Falls Network

(b) New York taxi network

Figure 1: Case study results

perfect traffic information, future work should also focus on optimization with consideration for congestion effects, which would better represent the reality, and make the estimations more robust. Finally, our RACS system is operated without rejection of customer requests, which assumes an increase in the system capacity according to the demand. Thus, future research should also consider a scenario wherein requests of some of the customers are unserved.

Acknowledgements

This research has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No. 815069 [project MOMENTUM (Modelling Emerging Transport Solutions for Urban Mobility)], the German Research Foundation - Deutsche Forschungsgemeinschaft (DFG) [Project number 392047120, Research grants in collaboration with China] and the International Graduate School of Science and Engineering - IGSSE (MO3 Project).

References

[1] A. Ali and F. Hamed. The closeopen mixed multi depot vehicle routing problem considering internal and external fleet of vehicles. *Transportation Letters*, 11(2):78–92, 2019.

[2] C. Bongiovanni, M. Kaspi, and N. Geroliminis. The electric autonomous dial-a-ride problem. *Transportation Research Part B: Methodological*, 122:436 – 456, 2019.

[3] J. F. Cordeau. A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. *Operations Research*, 54(3):573–586, 2006.

[4] J. F. Cordeau and G. Laporte. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6):579–594, 2003.

[5] J. F. Cordeau and G. Laporte. The dial-a-ride problem: Models and algorithms. *Annals of Operations Research*, 153(1):29–46, 2007.

[6] J. Farhan and T. Donna Chen. Impact of ridesharing on operational efficiency of shared autonomous electric vehicle fleet. *Transportation Research Part C: Emerging Technologies*, 93:310 – 321, 2018.

[7] H. Fazlollahtabar. Lagrangian relaxation method for optimizing delay of multiple autonomous guided vehicles. *Transportation Letters*, 10(6):354–360, 2018.

[8] E. Forgy. Cluster analysis of multivariate data: Efficiency versus interpretability of classification. *Biometrics*, 21(3):768–769, 1965.

[9] S. C. Ho, W. Y. Szeto, Y. H. Kuo, J. M.Y. Leung, M. Petering, and T. W.H. Tou. A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, 0:1–27, 2018.

[10] S. Hong, L. Zhou, S. Ma, N. Jia, L. Zhang, and B. Yao. The optimization of bus rapid transit route based on an improved particle swarm optimization. *Transportation Letters*, 10(5):257–268, 2018.

[11] M. Hyland and H. S. Mahmassani. Dynamic autonomous vehicle fleet operations: Optimization-based strategies to assign avs to immediate traveler demand requests. *Transportation Research Part C: Emerging Technologies*, 92:278–297, 2018.

[12] R M Jorgensen, J. Larsen, and K B Bergvinsdottir. Solving the dial-a-ride problem using genetic algorithms. *Journal of the Operational Research Society*, 58(10):1321–1331, Oct 2007.

[13] Y. Kergosien, C. Lenté, and J. C. Billaut. A tabu search algorithm for solving a transportation problem of patients between care units, 2008.

[14] D. Kirchler and R. Wolfler Calvo. A Granular Tabu Search algorithm for the Dial-a-Ride Problem. *Transportation Research Part B: Methodological*, 56:120–135, 2013.

[15] M. W. Levin. Congestion-aware system optimal route choice for shared autonomous vehicles. *Transportation Research Part C: Emerging Technologies*, 82:229–247, 2017.

[16] J. Ma, X. Li, F. Zhou, and W. Hao. Designing optimal autonomous vehicle sharing and reservation systems: A linear programming approach. *Transportation Research Part C: Emerging Technologies*, 84:124–141, 2017.

[17] N. Masoud and R. Jayakrishnan. A decomposition algorithm to solve the multi-hop Peer-to-Peer ride-matching problem. *Transportation Research Part B: Methodological*, 99:1–29, 2017.

[18] S. Muelas, A. LaTorre, and J. M. Peña. A distributed VNS algorithm for optimizing dial-a-ride problems in large-scale scenarios. *Transportation Research Part C: Emerging Technologies*, 54:110–130, 2015.

[19] W. P. Nanry and J. Wesley Barnes. Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research Part B: Methodological*, 34(2):107–121, February 2000.

[20] S. Narayanan, E. Chaniotakis, and C. Antoniou. Shared autonomous vehicle services: A comprehensive review. *Transportation Research Part C: Emerging Technologies*, 111:255–293, 2020.

[21] J. Paquette, J. F. Cordeau, G. Laporte, and M. M. B. Pascoal. Combining multicriteria analysis and tabu search for dial-a-ride problems. *Transportation Research Part B: Methodological*, 52:1–16, 2013.

[22] S. N. Parragh, K. F. Doerner, and R. F. Hartl. Variable neighborhood search for the dial-a-ride problem. *Computers and Operations Research*, 37(6):1129–1138, 2010.

[23] V. Pimenta, A. Quilliot, H. Toussaint, and D. Vigo. Models and algorithms for reliability-oriented dial-a-ride with autonomous electric vehicles. *European Journal of Operational Research*, 257(2):601–613, 2017.

[24] P. Singanamala, K D. Reddy, and P. Venkataramaiah. Solution to a Multi Depot Vehicle Routing Problem Using K-means Algorithm , Clarke and Wright Algorithm and Ant Colony Optimization. 13(21):15236–15246, 2018.

[25] NYC Taxi and Limousine Commission. Trip record data. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

[26] Transportation Networks for Research Core Team. Transportation networks for research, 2019.

[27] X. Wang, M. Dessouky, and F. Ordonez. A pickup and delivery problem for ridesharing considering congestion. *Transportation Letters*, 8(5):259–269, 2016.

[28] A. K. Yazdi, M. A. Kaviani, A. Emrouznejad, and H. Sahebi. A binary particle swarm optimization algorithm for ship routing and scheduling of liquefied natural gas transportation. *Transportation Letters*, 0(0):1–10, 2019.