hEART 2019 - 8th Symposium of the European Association for Research in Transportation
September 4-6, 2019, Budapest, Hungary

A Learning Large Neighborhood Search for The Dynamic Electric Autonomous Dial-A-Ride Problem

Claudia Bongiovanni [1] Mor Kaspi [2] Jean-François Cordeau [3]
Nikolas Geroliminis [1]

[1]Urban Transport Systems Laboratory, School of Architecture, Civil & Environmental Engineering
École Polytechnique Fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland
{claudia.bongiovanni,nikolas.geroliminis}@epfl.ch
[2]Department of Industrial Engineering
Tel-Aviv University, Tel-Aviv 69978, Israel
{morkaspi}@tauex.tau.ac.il
[3]Department of Logistics and Operations Management
HEC Montréal, H3T2A7 Montréal, Canada
{jean-francois.cordeau}@hec.ca

ABSTRACT

In the dynamic Dial-a-Ride Problem (DARP) a fleet of vehicles provide door-to-door transport to on-line requests. The problem typically aims at minimizing the total operational cost while maximizing the company revenue, measured by means of the number of served trip requests. In its standard version, the dynamic DARP considers time-window, capacity, maximum ride time, and maximum route duration constraints. The dynamic electric Autonomous Dial-a-Ride Problem (e-ADARP) extends the dynamic DARP by considering the employment of electric autonomous vehicles (e-AVs). Differently from human-driven vehicles, e-AVs can be diverted as often as desired in the course of the operations and operated on a non-stop schedule. Given the electric nature of the vehicles, the planning process needs to continuously re-optimize the vehicle battery levels, decisions regarding detours to charge stations, recharge times, together with the classic dial-a-ride features.

In this work, we propose a two-phase heuristic approach to solve the dynamic e-ADARP. The first phase consists of an insertion heuristic that efficiently modifies both vehicle routes and schedules with the arrival of new transportation requests. We propose an exact scheduling algorithm for the e-ADARP, which efficiently provides optimal vehicle schedules in quadratic time. The second phase introduces a new Learning Large Neighborhood Search (LLNS) algorithm that re-optimizes both vehicle plans and schedules through intra- or inter-route customer exchanges. The LLNS utilizes multiple neighborhoods defined from problem-specific characteristics. We formulate the choice of the operator by a classification problem, where the operator represents a class and selected characteristics of the problem instances or solutions represent the features. Numerical results are produced from an event-based simulation based on existing benchmark instances and real-world data from ride-hailing services.

Keywords: *Dial-a-Ride Problem, Dynamic problem, Electric Autonomous Vehicles, Metaheuristics, Machine Learning, Classification And Regression Trees*

1. INTRODUCTION

In the Dial-a-Ride Problem (DARP), minimum cost routes and schedules are defined for a fleet of vehicles exiting known depots and serving a set of customers with given pickup and dropoff locations [*1*]. The optimization can take into consideration multiple criteria and can include multiple type of users and destination depots (e.g. [*2*],[*3*]). Typical operational constraints include vehicle capacities, maximum vehicles-route duration, and maximum user ride times. In addition, service start times at pickup and dropoff locations are usually limited by time-window constraints. Recently, multiple companies and research institutions started developing test tracks using electric autonomous vehicles (e-AVs) and considering their use to provide transportation services (e.g. [*4*]). A recent work in [*5*], has considered a new mathematical model for the DARP with the use of e-AVs (e-ADARP). The objective of the e-ADARP minimizes a weighted-sum objective consisting of the total vehicle travel time and user excess ride time. Integrated operational constraints include battery management, decisions regarding detours to charging stations, recharging times, and destination depots.

The DARP literature can be divided into two main streams, namely, static and dynamic DARP. In the first case, demand is fully known in advance, whereas, in the second case, demand is revealed online. This study focuses on the development of a metaheuristic approach for the dynamic e-ADARP. For a review on dynamic DARP studies, the reader is referred to Section 6 in [*6*].

The operation of e-AVs introduces new opportunities that must be taken into account in real-time planning processes. First, e-AVs offer more flexibility to modify vehicle plans in real-time according to changing conditions. Such changes do not only correspond to the arrival of new transportation requests but also to unexpected increases in traffic conditions and modified availabilities at recharging facilities. That is, differently from human-driven vehicles, the dispatching system can easily divert e-AVs as often as desired in the course of operations. Second, e-AVs can operate non-stop. While this feature might help saving vehicle deadhead miles to decentralized depots, service quality aspects can arise in a multi-period context. The employment of e-AVs also introduces new challenges that need to be tackled on-line. That is, the planning process needs to continuously re-optimize the vehicle battery levels, decisions regarding detours to charging stations, recharging times, and destination depots together with the classic Dial-a-Ride features.

In this work, we propose a two-phase heuristic approach to solve the dynamic electric Autonomous Dial-a-Ride Problem (e-ADARP). In the first phase, an insertion heuristic algorithm is designed to efficiently introduce new transportation requests into vehicle routes. The vehicle schedules and recharging times are obtained through a new exact scheduling algorithm for the e-ADARP, which solves in quadratic time. Current work is focusing on the implementation of the second phase, in which an improvement heuristic re-optimizes the vehicle plans after the insertion of new customers, and on computational experiments. For these latter, we introduce an event-based simulated environment to generate dynamic scenarios from existing benchmark instances and real-world data from ride-hailing services.

The rest of the paper is organized as follows: Section 2 presents the first phase and the exact scheduling algorithm for the e-ADARP, Section 3 conceptually describes the second phase, Section 4 presents our preliminary results.

2. FIRST PHASE: INSERTION HEURISTIC

A new transportation request is represented by a generation time, a pickup/dropoff location, and a time-window around the pickup or dropoff location. Note that the time-window around the pickup can be easily computed from its dropoff by consideration of the user maximum ride time. Furthermore, the time-windows can be further tightened by considering the vehicle current locations and times. As a result, the earliest arrival at the newly generated request is vehicle-dependent. Vehicles that cannot arrive to the customer pickup by its latest arrival time are not candidates for the insertion.

Given a number of candidate vehicles for the insertion, operational costs need to be computed for each vehicle. A first screening to find feasible insertions for the new transportation request (both in terms of its pickup and dropoff) and a particular vehicle can be performed through time-window considerations. That is, having computed the forward and backward time slacks for the users scheduled in a vehicle (as proposed in [*7*]), it is possible to identify segments of its static plan where the insertion of the pickup or the dropoff of the new request might be time-window feasible. In consideration of precedence and capacity constraints, such segments can be further restricted. That is, knowing the vehicle maximum capacity and loads from the static plan, the newly generated request cannot be inserted when the vehicle is planned to travel at capacity. Note that time-window and battery feasibility for the complete insertion of the newly generated request (i.e. both in terms of its pickup and dropoff location) cannot be ensured by the simple screening presented so far and instead needs more advanced scheduling algorithms, which are presented next. Finally, for each candidate vehicle, multiple insertions of the new transportation request can be evaluated against a given objective function. The chosen insertion is then the one resulting in the lowest operational cost.

2.1 Exact scheduling and battery management algorithm for the e-ADARP

Scheduling problems in the standard Dial-a-Ride Problem (DARP) are typically heuristically solved by employing the forward and backward time slacks and by delaying the pickup time of the customers [*8*]. Note that such a procedure does not provide excess-time optimal solutions and does not guarantee battery feasibility. In this work we propose an efficient procedure to optimize schedules based on the excess time objective subject to time-window and battery considerations.

Consider a candidate insertion satisfying pairing, precedence, and capacity constraints of a selected vehicle. The insertion determines a new sequence of locations $\mathcal{I} = \{1, \ldots, M\}$ that must be visited by the vehicle. Then, the remaining problem may be stated as a linear program aiming at minimizing the total user excess time while satisfying time-window and maximum ride time constraints (LP1). Denote by P the set of pickups and D the dropoffs of the pickups P in sequence $\mathcal{I}$. Let T_i represent the service-start time at node $i \in \mathcal{I}$, d_i represent the service duration time, $t_{i,j}$ the travel time between nodes i and $j \in \mathcal{I}$, $[arr_i, dep_i]$ represent the time-window of node $i \in \mathcal{I}$, and u_{P_i} represent the maximum ride time of customer $P_i \in P \subset \mathcal{I}$. Then, the remaining problem can be stated as:

$$(LP1) \quad \min \sum_{i \in \mathcal{P}} (T_{D_i} - T_{P_i} - d_{P_i} - t_{P_i,D_i}) \tag{1}$$

Subject to:

$$T_i + t_{i,i+1} + d_i \leq T_{i+1} \qquad \forall i \in \{1, 2, \ldots, M-1\} \tag{2}$$

$$T_{D_i} - T_{P_i} - d_{P_i} \le u_{P_i} \quad \forall i \in \mathcal{P} \tag{3}$$
$$arr_i \le T_i \le dep_i \quad \forall i \in \{1, 2, \ldots, M\} \tag{4}$$

Given that the sequence is fixed, then it is possible to calculate the earliest time ET_i and latest time LT_i at which service can start at node i by using the following recursive formulas:

$$ET_i = \max\{arr_i, ET_{i-1} + t_{i-1,i} + d_i\} \quad \forall i \in \{2, \ldots, M\}, ET_1 = arr_1 \tag{5}$$
$$LT_i = \max\{dep_i, LT_{i+1} - t_{i,i+1} - d_i\} \quad \forall i \in \{1, \ldots, M-1\}, LT_M = dep_M \tag{6}$$

Hence, LP1 can be re-formulated into an equivalent but tighter linear program (LP2), as follows[1]:

$$(LP2) \quad \min \sum_{i=1}^{M} L_i w_i \tag{7}$$

Subject to:

$$\sum_{j=1}^{i} w_j \ge ET_i - \sum_{j=1}^{i-1} t_{j,j+1} + \sum_{j=1}^{i-1} d_j - ET_1 \quad \forall i \in \{1, 2, \ldots, M\} \tag{8}$$
$$\sum_{j=1}^{i} w_j \le LT_i - \sum_{j=1}^{i-1} t_{j,j+1} + \sum_{j=1}^{i-1} d_j - ET_1 \quad \forall i \in \{1, 2, \ldots, M\} \tag{9}$$
$$\sum_{j=i+1}^{D_i} w_j \le u_i - \sum_{j=i}^{D_i-1} t_{j,j+1} - \sum_{j=i+1}^{D_i-1} d_j \quad \forall i \in \mathcal{P} \tag{10}$$

Where L_i represents the vehicle load and w_i the waiting time at location $i \in \mathcal{I}$. Denote by Δ_i and by Θ_i the right-hand side of constraints (8) and (9) respectively. Then, the following procedure is proposed to solve (LP2):

(ALG1)

1. Set $w_i = 0 \ \forall i \in \{1, 2, \ldots, M\}$
2. Set $\Omega = \emptyset$
3. For $i = 1$ to M
   (a) Set $\Omega = \Omega \cup \{i\}$
   (b) While $\sum_{k=1}^{i} w_k < \Delta_i$
      i. Set $j = \operatorname{argmin}_{k \in \Omega} L_k$
      ii. Set $w_j = \min\{\min_{k \in \{P | k \le j \ \& \ D_k \ge j\}} u_k - \sum_{l=k}^{D_k-1} t_{l,l+1} - \sum_{l=k+1}^{D_k-1} d_l - \sum_{l=k+1}^{j-1} w_l - \sum_{l=j+1}^{D_k} w_l; \Theta_i - \sum_{k=1}^{j-1} w_k - \sum_{k=j+1}^{i} w_k; \Delta_i - \sum_{k=1}^{j-1} w_k - \sum_{k=j+1}^{i} w_k\}$
      iii. If $\sum_{k=1}^{i} w_k < \Delta_i$, set $\Omega = \Omega \setminus \{j\}$
4. Return w_i

[1]For brevity reasons, the steps demonstrating the equivalence between LP1 and LP2 are omitted in this short paper

The algorithm is initialized at a basic non-feasible solution (assuming $\Delta_M > 0$). Each execution of step 3.(b) represents a move to a neighboring basic solution. Constraints (9)-(10) are satisfied at any step of ALG1 and the algorithm terminates when all the constraints in (8) are satisfied. That is, the algorithm terminates with a basic feasible solution. In step 3.(b).i., if several values minimize L_k then the one with the lowest index can be taken. Note that the complexity of ALG1 is $O(M^2)$ since in the worst case, step 3. may be executed M times and the minimization term in 3.(b) contains less than M components[2].

Next, one needs to demonstrate that there is a schedule which maximizes battery levels and minimizes total user excess time. Observe that, if the schedule that maximizes battery does not satisfy battery restrictions, the given sequence is not feasible. However, if battery-feasible schedules do exist, at least one of them is excess-time optimal. In addition, note that battery levels can be seen as an inventory which can only decrease with traveling. Then, for feasibility purposes, it is always better to recharge as much as possible, as early as possible. Denote by N the number of charging stations contained in sequence $\mathcal{I}$. Let Q represent the nominal capacity of the e-AVs, α_s the charging rate and B_s the battery inventory level at charging facilities $s \in \{s_1, s_2, \ldots, s_N\}$. Then, the following procedure is proposed for battery management:

(ALG2)

1. Divide the sequence into H sub-sequences delimited by charging facilities $s_1, s_2, \ldots, s_N$
2. For each charging station i in $\{1, \ldots, N-1\}$
   (a) Compute the charging time at station i as $\min\{LT_j - ET_j; (Q - B_s)/\alpha_{s_i}\}$ where j is the first node of the following sub-sequence $\mathcal{H}_{i+1}$
   (b) Re-compute the earliest start times ET_j for all nodes following station i (i.e. $j \in \{i+1, \ldots, M\}$) by considering the computed recharging time
   (c) If the state of charge $B_{s_{i+1}}$ is negative, the sub-sequence up to node s_{i+1} is infeasible

In step 2.(c), the recharging times can be computed iteratively, starting from the first station. At each facility, recharging time is bounded by: i) the service start time at the preceding node and the latest service start time at the following node ii) the time needed to fully recharge. To ensure feasibility, the recharging time at the station needs to be set to the minimum of the two. Observe that for each station, if for the following sub-sequence we have $\Theta_1 \leq \Delta_M$, recharging more at the current station might imply that we reduce the available recharging time at the following station. If for the following sub-sequence $\Theta_1 > \Delta_M$, the earliest time in which recharging can begin at the next station is not influenced by the decisions made in the current station (i.e. recharging can begin as early as possible at the next station).

3. SECOND PHASE: IMPROVEMENT HEURISTIC

Given that no information about future demand is available during the first phase, vehicle plans might be improved once a number of myopic insertions have been performed. To this end, we are currently developing an improvement metaheuristic approach to re-optimize vehicle plans through intra-route and inter-route customer exchanges and charging plan modifications (i.e.

[2]For brevity reasons, the optimality proof for ALG1 is omitted from this short paper

the second phase). Multiple works in the dynamic DARP have considered metaheuristic solution approaches based on Large Neighborhood Search (LNS) ([*9*],[*10*],[*11*]). In this work, we propose a new metaheuristic approach for the dynamic e-ADARP in which multiple neighborhoods are defined from problem-specific characteristics and the search mechanism is ruled by a machine learning approach. During the search, the neighborhood operators are selected according to a prediction scheme derived from Classification And Regression Trees (CART) [*12*]. CART recursively partition the input space through a series of binary splits which are chosen through a mathematical model which maximizes a goodness of split function. The depth of the tree is typically controlled by a parameter which provides an upper bound on the maximal number of splits and, consequently, sub-regions. The class of each sub-region is decided by computing the largest number of representative samples in the sub-region (the "majority vote"). The predicted class for a new input point is then obtained by passing the point through the tree until a final node (or sub-region) is reached. In order to increase the accuracy of the prediction, multiple decision trees are typically combined through a technique called *bagging*, which essentially generates multiple models based on bootstrap samples of the input space. The final prediction is the aggregation of the predictions of all models. For a classification task, the aggregation corresponds to the most frequently-predicted class. In the context of the e-ADARP, we introduce the Learning Large Neighborhood Search (LLNS) metaheuristic. We formulate the choice of the operator by a classification problem (i.e. CART) in which each operator represents a class, while selected characteristics of the problem instances and solutions represent the features. Therefore, an instance of class i is an instance for which operator i is the best. Finally, the training data is a collection of examples of problem instances for which the best operator is known.

4. COMPUTATIONAL TESTS

Computational experiments are performed on benchmark instances from literature and instances based on ride-sharing data from Uber Technologies Inc. in 2011 [*5*]. In order to generate dynamic scenarios, an event-based simulated environment is employed. The simulated events consider vehicle departures, arrivals, recharges, customer pickups, dropoffs and generation times. Given a cumulative density function, customer generation times are drawn through the inverse transform method [*13*]. New visits to recharging stations can be triggered after the generation of new customer requests. We are currently testing the first phase of the two-phase heuristic approach (Section 2). In this short paper, we only provide preliminary results on a single instance composed of 10 vehicles and 100 dynamic requests received during a planning horizon of about 30 minutes. In particular, given a candidate insertion, we either employ a customary scheduling algorithm from literature ([*8*]), which minimizes completion time and the excess time of the inserted customer, or the scheduling algorithm ALG1 (Section 2). Note that ALG1 optimally solves LP2 and therefore provides equivalent results[3]. Figure1 shows the number of request arrivals, as well as the number of accepted requests through ALG LIT and ALG1. As it can be shown, even through a myopic insertion heuristic, ALG1 provides a higher acceptance rate with respect to ALG LIT. We expect the difference in the number of accepted requests between ALG LIT and ALG1 to increase as intra- and inter- route exchanges will allow to modify previously-made routing decisions (i.e. the second phase). The second phase is currently under development and preliminary results are therefore omitted from this short paper. For both the first and second phase, we are currently developing

[3]A thorough comparison between LP2 and ALG1 is currently under testing

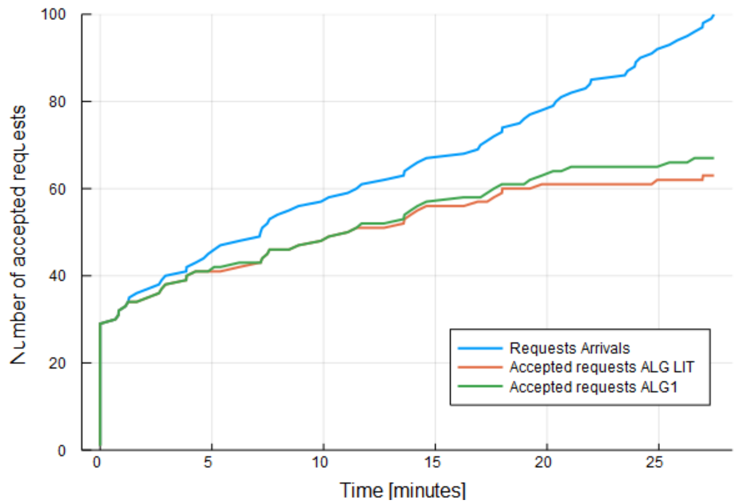

FIGURE 1 Accepted requests using ALG LIT or ALG1

tests (of variable size and up to 2500 requests) which consider several request generation times to produce statistically significant results. We are also planning to employ different cumulative density functions to analyze the impact of advanced time on the service quality and cost. Furthermore, different dynamic scenarios (i.e. ratio of static versus dynamic requests) will be employed to test the limits of the proposed framework. Collected statistics will include the ratio of accepted and rejected requests, static and dynamic customer excess ride times, vehicle idle times, recharging times, travel times, and end battery levels.

REFERENCES

1. J.-F. Cordeau, G. Laporte, "The dial-a-ride probelm: models and algorithms", *Annals of Operations Research* 153, 29-46 (2007).
2. S.N. Parragh, "Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem", *Transportation Research Part C: Emerging Technologies* 19(5), 912-930 (2011).
3. K. Braekers, A. Caris, G.K. Janssens, "Exact and meta-heuristics approach for a general heterogeneous dial-a-ride problem with multi depots", *Transportation Research Part B: Methodological* 67, 166-186 (2014).
4. Z. Szalay, T. Tettamanti, D. Esztergr-Kiss, I. Varga, C. Bartolini, "Development of a Test Track for Driverless Cars: Vehicle Design, Track Configuration, and Liability Considerations", *Periodica Polytechnica Transportation Engineering* 46(1), 29-35 (2018).
5. C. Bongiovanni, M. Kaspi, N. Geroliminis, "The Electric Dial-a-Ride Problem", *Transportation Research Part B: Methodological, under review* (2019).
6. G. Berbeglia, J.-F. Cordeau, G. Laporte, "Dynamic Pickup and delivery problems", *European Journal of Operational Research* 202, 8-15 (2010).
7. M.W.P. Savelsbergh, "The vehicle routing problem with time windows: Minimizing route duration", *ORSA Journal on Computing* 4, 146154 (1992).
8. J.-F. Cordeau, 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).

9. A. Attanasio, J.-F. Cordeau, G. Ghiani, G. Laporte, "Parallel Tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem", *Parallel Computing* 30(3), 377-387 (2004).
10. L. Coslovich, R. Pesenti, W. Ukovich, "A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem", *European Journal of Operational Research* 175(3), 1605-1615 (2006).
11. M. Schilde, K.F. Doerner, R.F. Hartl, "Metaheuristics for the dynamic stochastic dial-a-ride problem with expected return transports", *Computers & Operations Research* 38(12), 1719-1730 (2011).
12. L. Breiman, *Classification and regression trees*, Routledge, (2017).
13. S.M. Ross, *Simulation*, Academic Press, (2012).