A Deterministic Annealing Local Search for the Electric Autonomous Dial-a-Ride Problem

Yue Su, Nicolas Dupin, Jakob Puchinger

To cite this version:

Yue Su, Nicolas Dupin, Jakob Puchinger. A Deterministic Annealing Local Search for the Electric Autonomous Dial-a-Ride Problem. ROADEF2021, 21ème congrès annuel de la société Française de Recherche Opérationnelle et d'Aide à la Décision, Apr 2021, Mulhouse, France. hal-03197153

HAL Id: hal-03197153
https://hal.archives-ouvertes.fr/hal-03197153

Submitted on 13 Apr 2021

A Deterministic Annealing Local Search for the Electric Autonomous Dial-a-Ride Problem

Yue Su[1], Nicolas Dupin[2], Jakob Puchinger[1,3]

[1] Laboratoire Genie Industriel, CentraleSupélec, Université Paris-Saclay, France
`{yue.su, jakob.puchinger}@centralesupelec.fr`
[2] Laboratoire Interdisciplinaire des Sciences du Numérique, CNRS, Université Paris-Saclay, France
`nicolas.dupin@lisn.upsaclay.fr`
[3] IRT SystemX, Palaiseau, France
`jakob.puchinger@irt-systemx.fr`

Mots-clés : *dial-a-ride problem, electric autonomous vehicles, deterministic annealing, local search, meta-heuristic, vehicle routing.*

1 Introduction

Our study investigates the Electric Autonomous Dial-a-Ride Problem (E-ADARP) introduced by [1] which consists of scheduling electric autonomous vehicles (EAVs) to transport users from specific origins to specific destinations within predefined time windows. The E-ADARP consists of the following features that are different from the typical DARPs : (1) detour to recharging station on the route, (2) partial recharging at recharging station, (3) vehicle can locate at different origin depots andselect from a set of optional destination depots, (5) no restriction for route duration time. As with classical DARP, the exact methods cannot solve large-scale instances of E-ADARP within reasonable computational time [1]. We therefore establish a meta-heuristic approach, namely Deterministic Annealing (DA) heuristic enhanced by a Local Search (LS) for intensification.

2 Deterministic Annealing Heuristic

The algorithm input is the obtained solution from a parallel insertion heuristic. In the first step, we sort the requests according to their earliest time window and assign the first m requests to randomly generated m routes ($0 < m < K$, K is the total number of vehicles). Other requests are inserted into the route in which the last-assigned request has the minimum distance to this request. If the determined route cannot feasibly insert the request, then the second-minimum-distance route will be examined. The whole process will iterate until no more requests can be inserted into all the existing routes. If there are still requests in the list, a new route will be activated and the same insertion procedure is repeated. The solution cost of the obtained solution is denoted as $c(x)$, and the number of requests served in the solution is Nb_{req}. No infeasibility is tolerated during the initial solution construction process. There are basically two steps in the main algorithm : local search and threshold update. At the beginning of the algorithm, the threshold value T is set to T_{max}, and the best solution x_b and current solution x' are initialized to initial solution x_{init}. The number of iteration is denoted as Nb_{iter}. During the local search process, the local search operators are applied to alter the current solution. In the next step, the threshold value is updated and restart when the value is negative.

In the local search process, the number of operators used depends on the number of assigned requests in the initial solution. In case of existing un-inserted requests, an operator named "*AddNewRequest*" is designed to add these requests into the neighboring solution found by intra-route/inter-route operators. Once all the requests have been properly included in the routes, this operator is deactivated. During the local search process, when encountering battery-infeasible solutions, a bi-directional insertion algorithm is called to insert recharging stations at proper places. Each operator (intra- or inter-route) returns the best neighbor x' from current solution x if it exists. Solution x' is accepted to become the new current solution when the number of assigned requests increases or the total cost is less than that of the current solution plus the threshold value T. Once the new solution x' is accepted, we check whether it is the best solution x_b or not.

3 Numerical Experiments, Conclusions and Perspectives

In numerical studies, we assess the performance of the proposed algorithm on modified Cordeau instances ([1]) that have been supplemented with charging stations and battery specifications. The real open source data provided by Uber Technologies (`https://github.com/dima42/uber-gps-analysis/tree/master/gpsdata`) is also used in the experiments. To ensure a strong robustness of DA parameters, the sensitivity analysis is conducted on the exisiting instances to find good parameter settings (due to the limit of pages, this part will not be illustrated). Algorithm results are compared to the exact results presented in [1] on the modified Cordeau instances and Uber instances to evaluate its performance. After verifying the algorithm's effectiveness, we extend the model and consider the multiple visits on the recharging stations. The effect of allowing multiple visits to the same recharging station is investigated. Moreover, we establish the new Ropke instances ([2]) that have been supplemented with problem-related features to conduct large-scale experiments. All the algorithms are implemented in Julia 1.3.0 and performed on a 3.20 GHz Intel Core computer with 32G RAM. For each set of instances, we consider three different scenarios (i.e., low-energy, medium energy, and high-energy restriction). The three different scenarios represent the constraint on the minimum level of the battery capacity at the end of the route. That is, under low-/medium-/high-energy restriction, vehicle should return the depot with at least 10%/40%/70% of battery capacity.

Computational experiments prove the effectiveness of the proposed algorithm in solving E-ADARP. For each scenario, adapted instances from the literature are tested and an average gap of 0.58% (gaps ranging from 0.37% to 0.69%) is achieved compared to the best-known solutions for E-ADARP. Several new best solutions are found on previously solved and unsolved instances. The analysis on the effect of allowing multiple visits to the recharging stations shows that this operation can efficiently decrease the total cost and improve the solution feasibility. Furthermore, we establish new benchmark instances based on literature with up to 8 vehicles and 96 requests, with our algorithm providing feasible solutions that the exact method from the literature cannot solve in a given amount of time.

These results offer several new perspectives. The E-ADARP model may be improved to take into account more real-life characteristics such as time-dependent travel times. Besides, the objective function may consider users' convenience with less waiting or travel times which may be conflicting with the global financial optimization. Operators of our DA heuristic may be extended to consider additional constraints. Having several objective functions, a perspective is to design population multi-objective meta-heuristics, like evolutionary algorithms, to handle conflicting objectives. Lastly, the nature of the problem is strongly related to dynamic optimization, with new requests to serve or modifications due to uncertainties like traffic jams. Having quick and efficient heuristic algorithms for dynamic E-ADARP is there a crucial issue, where meta-heuristics are promising.

Références

[1] Claudia Bongiovanni, Mor Kaspi et Nikolas Geroliminis. “The electric autonomous dial-a-ride problem”. In : *Transportation Research Part B : Methodological* 122 (2019), p. 436-456.

[2] Stefan Ropke, Jean-François Cordeau et Gilbert Laporte. “Models and branch-and-cut algorithms for pickup and delivery problems with time windows”. In : *Networks : An International Journal* 49.4 (2007), p. 258-272.