New Optimization Guidance for Dynamic Dial-a-Ride Problems

Christian Ackermann(✉) and Julia Rieck

Institute for Business Administration and Information Systems, Operations Research Group, University of Hildesheim, Universitätsplatz 1, 31141 Hildesheim, Germany
ackermann@bwl.uni-hildesheim.de

Abstract. In the dial-a-ride problem, customers have to be transported from different pickup to drop-off locations. Various constraints such as time windows and a maximum ride time per passenger need to be considered. In the dynamic version of the problem, not all customer requests are known in advance, but arrive during the operation time. Therefore, the maximization of the number of served customers is usually set as the optimization goal. Nevertheless, in the vast majority of known heuristics, the total distance is used to guide the optimization. In this paper, we present different metrics that should enable the evaluation of the insertion potential of future customers and lead to a higher acceptance rate through their use in solution procedures. We show that even a single metric can provide better results than the distances and present a *Markov decision process*-based approach to enable an agent trained by *reinforcement learning* to perform even more anticipatory decision making by considering multiple metrics simultaneously.

Keywords: Dynamic dial-a-ride problem · Metrics · Reinforcement learning

1 Introduction

As a compromise between expensive cabs and unflexible public transport, on-demand ridesharing services continue to gain in popularity. In order to model sharing concepts, the dial-a-ride problem (DARP) as a variant of the vehicle routing problem with time windows can be used. In this problem, a set of requests is served by a fleet of vehicles. Each request consists of taking a customer from a pickup to a drop-off point. The customers specify time windows for pickup or drop-off as well as a maximum ride time (often a linear function depending on the direct travel time).

The DARP combines a *routing* and a *scheduling* part. First, the routing has to determine which vehicle serves which customer in what order. The subsequent scheduling determines the exact timing so that time windows and the maximum ride time are respected. The goal of the static problem, in which all requests are known in advance, is usually to minimize the total travel distance. In a

N. Trautmann and M. Gnägi (Eds.): OR 2021, LNOR, pp. 283–288, 2022.
https://doi.org/10.1007/978-3-031-08623-6_42

dynamic DARP, only some of the requests are known at the beginning of the planning horizon, the remainder are received during the operation. How many of the requests are dynamic is specified by the *degree of dynamism*. Due to the consideration of requests arriving at short notice, it may not be possible to serve all requests. Hence, the maximization of the number of requests that can be accepted is typically chosen as the optimization goal.

There are well known approaches that adapt the scheduling part to be able to respond to the dynamic aspect of the DARP (see, e.g., [1]). The idea is to generate a schedule that facilitates the insertion of future requests. However, the routing part is usually disregarded. Instead, the corresponding approaches focus on insertion heuristics that may be followed by local search procedures. These are executed for a certain number of iterations or until a new event (e.g., arrival of a new request) occurs (see, e.g., [2]). Nevertheless, the respective insertion and local search methods only minimize distances, which leads to an efficient use of vehicles but does not explicitly maximize the insertion potential for future requests.

In this contribution, we seek to find a metric whose optimization in the routing part of a solution procedure leads to a higher acceptance rate than just using the distances. To this end, we evaluate different metrics for estimating the insertion potential of future requests (cf. Sect. 2). In Sect. 3, we show the superiority of one of the metrics over the previously used distances. In Sect. 4, we additionally present a *reinforcement learning*-based approach that could enable even more anticipatory decision making. Section 5 contains a summary and provides an outlook on future research.

2 Metrics for Estimating the Insertion Potential

The metrics used to guide the optimization are intended to reflect the potential of a solution to allow future request insertions. According to the literature and own preliminary experiments, primarily the time windows and secondarily the maximum ride time are the critical constraints. Consequently, we incorporate the subsequent four metrics (it is preferable if the metrics have high values):

Mean Waiting Time: For each arc $\langle A, B\rangle$ in the current solution (A and B are customer nodes), it is determined how long the respective vehicle waits on that arc. Thereby, the waiting time between end of service and departure at A as well as between arrival and start of service at B are taken into account. To evaluate the entire solution, the average waiting time is calculated over all arcs that are still modifiable, i.e., have not yet been fully or partially traversed.

Ellipse Area: Let $\langle A, B\rangle$ again be an arc in the solution. In order to now maximize the probability of being able to insert a random next request between A and B, we calculate where the node to be inserted (pickup or drop-off) could be located (assuming suitable time windows). We specify the earliest departure time at node A (ED_A) and the latest arrival time at node B (LA_B). The difference of LA_B and ED_A is the total available time (at_{AB}) for the trip from A to B. Determining all points C that can be reached within this time via the route

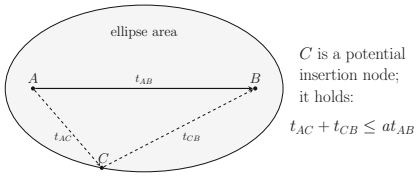

Fig. 1. Example of the ellipse area metric

$A \rightarrow C \rightarrow B$ results in an ellipse with focal points A and B (see Fig. 1). We take the area of the ellipse as an indicator of the probability of inserting a random request on $\langle A, B \rangle$. With (1), we obtain the area according to:

$$\text{ellipse area} = \frac{\pi}{4} \cdot a t_{AB} \cdot \sqrt{(a t_{AB})^2 - (t_{AB})^2} \tag{1}$$

with t_{AB} being the direct travel time from A to B. To evaluate the entire solution, a vehicle score S_k is determined by averaging the area of the ellipse over all arcs still to be traversed by vehicle k. Formula (2) is then used to combine the scores of each of the K existing vehicles into a joint score S:

$$S = \left(\frac{1}{K} \sum_{k=1}^{K} \sqrt{S_k} \right)^2 \tag{2}$$

Mean Ride Time Slack: We identify the average ride time slack (difference between actual and maximum ride time per customer) as a measure of how strict the current transportation plan is. The larger the average ride time slack of a customer not yet visited, the larger detours are possible without violating the customer's maximum ride time.

Mean Time Window Slacks: In addition, we integrate the *forward*, *central*, and *backward slack* times as proposed in [3]. These slack times consider the average difference between the planned start of service at a node and the opening (forward), center (central) or closing (backward) of the associated time window.

To evaluate how well these metrics can predict the insertion potential, we collected different information from temporary solutions during the application of a traditional distance-based solution procedure and noted whether a new request could be inserted. We trained a *random forest classifier* by *supervised learning* based on about 60,000 solutions to estimate whether the next request can be added to the current transportation plan or not. Input parameters were the current timestamp (as a measure of progress within the planning horizon), the number of requests accepted so far, and one of the collected metrics. The quality of the classification was measured by the area under the ROC curve (AUC). All metrics except the *mean ride time slack* were able to predict the insertion potential better than the pure distances. Particularly, the *ellipse area* achieved the best AUC with 0.9456 (in contrast to the distances with 0.9284).

3 Computational Study

Always taking the solution with the highest probability of inserting the next request is a very short-term and greedy strategy. In order to check whether this nevertheless leads to a higher acceptance rate in total, we have changed the acceptance criterion at all points in our optimization procedure. Instead of taking the solution with the shorter total distance, we choose the solution with the higher value of the metric under consideration.

The solution procedure used for the routing part is designed as follows: The initial insertion follows the ejection chain approach presented in [4]. All feasible insertion positions for the new request r are examined and the best one (according to the considered metric) is selected. If all insertion positions are infeasible, a request whose time windows overlap with r is removed from the current solution, the insertion procedure is repeated for r, and the removed request is reinserted again. This is iterated for all eligible requests. In case that no feasible solution is found, the request is rejected. Otherwise the best resulting solution is accepted and improved by a local search phase. This process essentially follows the approach in [5]. Neighborhood operators considered are the exchange of two requests, the relocation of one request, a 2-opt operator applied to route segments that do not have a customer in the vehicle at the beginning and end respectively, and a restricted 4-opt operator, where four consecutive arcs are eliminated. Once an improved solution is found, a simulated annealing criterion is applied to check whether it should be accepted or not. For the scheduling part and the feasibility check, the approach from [1] was used.

In the evaluation, we used 72 instances, which were generated based on realistic characteristics. The number of customer requests varied from 50 to 200, the time windows were 10 or 20 min each, the degree of dynamism ranged from 20% to 80%, and dynamic customers were known either 30 or 90 min before their pickup time window opened. The number of vehicles varied from 2 to 6 depending on the number of customers, the service area was 15×15 distance units, the duration of the planning horizon was 10 h, and the vehicles had a fixed capacity of 4 requests and a uniform speed of one distance unit per minute. The procedure was run 10 times with each metric and different durations of the local search for all instances, considering the average value over all 10 runs.

The *ellipse area* was the only metric that produced better results than the total distance. Table 1 shows the detailed results. While the first two columns display the properties of the instances, columns 3–5 and 6–8 compare the acceptance rate and the number of accepted customers for the *distance* (D) and the *ellipse area* (EA) metrics respectively, and provide the respective percentage differences. Column 9 presents a comparison of the number of instances in which the distance or the ellipse area led to a better solution as well as the number of instances in which the same results were achieved.

It is clearly evident that, on average, the ellipse area metric gives better results than the distances (regardless of the instance properties) and also solves most instances better with respect to the objective function. Larger time windows seem to amplify the difference due to the higher degree of freedom. How short-term the

Table 1. Comparison of distance and ellipse area metrics for optimization guidance

Instance property		Acceptance rate [%]			Accepted customers			InstComp
		D	EA	Incr [%]	D	EA	Incr[%]	D/EA/=
Time windows	10 min	90.88	91.79	1.00	129.30	130.46	0.90	32/70/6
	20 min	93.27	94.93	1.78	133.13	135.10	1.48	19/89/0
Booking in advance	30 min	91.97	93.26	1.40	130.83	132.47	1.25	29/78/1
	90 min	92.19	93.46	1.38	131.60	133.09	1.13	22/81/5
# Requests	50	90.26	92.28	2.24	45.13	46.14	2.24	7/26/3
	100	89.57	91.36	2.00	89.57	91.36	2.00	8/28/0
	150	92.70	93.85	1.24	139.05	140.78	1.24	16/54/2
	200	93.62	94.41	0.84	187.24	188.81	0.84	20/51/1
Degree of dynamism	20%	91.38	93.14	1.93	130.16	132.35	1.68	21/51/0
	50%	91.62	92.38	0.83	130.91	131.54	0.48	21/49/2
	80%	93.24	94.56	1.42	132.58	134.45	1.41	9/59/4
Local search duration	500 ms	91.92	93.03	1.21	130.94	132.24	0.99	21/50/1
	1000 ms	92.14	93.42	1.39	131.32	132.89	1.20	15/54/3
	2000 ms	92.18	93.63	1.57	131.38	133.20	1.39	15/55/2

requests become known does not seem to have a decisive influence. For smaller instances, the advantage of the ellipse area is stronger than for larger instances. Regarding the proportion of dynamic requests, the worst results are obtained when the ratio of static and dynamic requests is balanced. When the degree of dynamism is high, the optimization potential is greater and this obviously affects the results. Please note that the advantage of the ellipse area metric becomes more salient when increasing the duration of the local search.

4 MDP-Based Approach

As a first step towards an approach that can also account for combinations of multiple metrics, we modeled the scenario as a *Markov decision process* (MDP). The MDP serves as a basis for training a *reinforcement learning* (RL) agent. A *decision point* occurs when a decision must be made whether to choose a new solution over the previous one. The *actions* indicate which of the two solutions is chosen. The *states* include for both solutions the current timestamp, the number of accepted customers as well as the values of the metrics to be used. A unit *reward* is given if the solution with more customers than the other is selected. In case that both solutions have the same number of customers, no reward is given. The *objective* is to maximize the sum of future rewards, i.e., the sum of requests to be accepted. This is to enable the model to learn to make decisions that lead to states in which a request can be integrated. The *transition function* is non-deterministic in that the next state is formed from the chosen solution and the next solution found.

In a proof-of-concept, we used Q-learning and a classical Q-table to validate the MDP-based approach for general functionality. Due to the use of Q-tables, the state space had to be kept very small. We strongly discretized the time and only used $\{-1, 0, 1\}$ to indicate whether the first solution was worse, as good,

or better than the second solution with respect to the considered metric. The results show that in the case of different numbers of requests, the solution with more requests should be chosen and otherwise the solution with the higher *ellipse area metric*. This confirms the observations from Sect. 3. To exploit the full potential of the approach, *deep reinforcement learning* should be used by replacing the Q-table with a neural network so that the magnitude of the differences in the metrics can also be taken into account. Alternatively, *approximate dynamic programming* techniques can be used (see, e.g., [6]).

Please note that a complex MDP brings the following advantage: The common greedy approach of accepting a request whenever possible can be suboptimal. As long as the optimization goal is to maximize the number of accepted requests and not, e.g., the revenue generated, short or "easy" requests are more lucrative. An anticipatory RL-trained agent could accordingly identify a non-lucrative request and reject it despite an insertion opportunity, thus maintaining the possibility of accepting several more requests in the future instead.

5 Conclusion and Outlook

We developed different metrics and analyzed their suitability for predicting the insertion potential of future requests in dynamic dial-a-ride problems. The computational study showed that the presented *ellipse area*, which considers the number of reachable points between two nodes, was able to increase the customer acceptance rate by over 1.5% in the used method. In addition, we presented an MDP-based approach that could enable an RL agent to learn a combination of metrics and thus be even better to decide which solution actually leads to a higher acceptance rate. Further research will address the difficulties of the MDP-based approach (e.g., very similar Q-values for the states due to sparse reward and non-deterministic transition function).

References

1. Berbeglia, G., Cordeau, J.F., Laporte, G.: A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem. Informs J. Comput. **24**(3), 343–355 (2012)
2. Vallee, S., Oulamara, A., Cherif-Khettaf, W.R.: New online reinsertion approaches for a dynamic dial-a-ride problem. J. Comput. Sci. **47**, 101199 (2020)
3. Chassaing, M., Giboulot, V., Lacomme, P., Quilliot, A., Ren, L.: Determination of robust solutions for the dynamic dial-a-ride problem. In: CIE45 Proceedings (2015)
4. Luo, Y., Schonfeld, P.: Online rejected-reinsertion heuristics for dynamic multivehicle dial-a-ride problem. Transp. Res. Rec. **2218**(1), 59–67 (2011)
5. Braekers, K., Caris, A., Janssens, G.K.: Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots. Transp. Res. Part B **67**, 166–186 (2014)
6. Ulmer, M.W.: Anticipation vs. reactive reoptimization for dynamic vehicle routing with stochastic requests. Networks **73**(3), 277–291 (2018)