Order dispatching and vacant vehicles rebalancing for the first-mile ride-sharing problem

Ye, Jinwen; Pantuso, Giovanni; Pisinger, David

Published in:
EURO Journal on Transportation and Logistics

DOI:
10.1016/j.ejtl.2024.100132

Publication date:
2024

Document version
Publisher's PDF, also known as Version of record

Document license:
CC BY-NC-ND

Citation for published version (APA):

Download date: 18. maj. 2024
Order dispatching and vacant vehicles rebalancing for the first-mile ride-sharing problem

Jinwen Ye a,b, Giovanni Pantuso a, David Pisinger b

a University of Copenhagen, Copenhagen, Denmark
b Technical University of Denmark, Copenhagen, Denmark

A B S T R A C T

Given a set of transport requests to a transit station and a set of homogeneous vehicle, both geographically dispersed in a business area, the First-Mile Ride-Sharing Problem (FMRSP) consists of finding least cost vehicle routes to transport passengers to the station by shared rides. In this paper we formulate the problem as a mathematical optimization problem and study the effectiveness of preventive movements of idle vehicles (i.e., rebalancing) in order to anticipate future demand. That is, we identify promising rebalancing locations based on historical data and give the model incentives to assign vehicles to such location. We then assess the effectiveness of such movements by simulating online usage of the mathematical model in a rolling-horizon framework. The results show that rebalancing is consistently preferable both in terms of profits and service rate. Particularly, in operating contexts where the station is not centrally located, rebalancing movements increase both profits and service rates by around 30% on average.

1. Introduction

Ride-sharing services, which are linked to a reduction of the number of private cars on the road, emissions and congestion (Al-Abbasi et al., 2019), have emerged as a potential solution to the increase in road congestion and air pollution generated by growing urban areas and population (Taniguchi et al., 2014). Such services have yet significant potential for development. As an example, according to the NYC taxi cab data (Commission and Limousine, 2023), during January 2020 there were 363,874 taxi trips to the Pennsylvania Station, a fairly busy transit station in New York City see Fig. 1(a), that is, on average 12,129 taxis trips daily to the station. Of these, only 6% were shared by multiple passengers, see Fig. 1(b), which leaves significant margins for more efficient connections to the station.

An effective implementation of ride-sharing services requires adequate responses to potentially frequent changes in demand patterns during the day that may determine geographical mismatches between demand as supply. Fig. 2 illustrates the location of the requests of transportation to Pennsylvania Station during January 2014, showing that the majority of the requests arrive from the North-East area, whereas much fewer requests arrive from the remaining zones of the city. This suggests implementing mechanisms that prepare the geographical distribution of the fleet in such a way to anticipate demand and perhaps reduce waiting and response times as well as service rate.

The existing literature study various aspects of ride-sharing services, including pricing mechanisms (Bian and Liu, 2019 a,b; Bian et al., 2020; Chen and Wang, 2018), integration with public transport (Shen et al., 2018), order dispatching and vehicle routes (Wang, 2019; Chen et al., 2020). Conversely, strategies for anticipating demand through, e.g. preventive or rebalancing movements (Wen et al., 2018), remain, to a large extent, an open research question. Particularly, efficient ways to simultaneously determine both dispatching and rebalancing movements have, to the best of our knowledge, been neglected.

We contribute to filling this gap by providing a mathematical programming model for joint order dispatch and rebalancing decisions in a first-mile ride-sharing service which transports passengers from their initial location to a common destination (e.g., a transit station). We will refer to this decision problem as the First-Mile Ride-Sharing Problem (FMRSP). In addition, we propose a strategy for identifying promising locations where to rebalance empty vehicles. The model and rebalancing strategies are tested in a rolling-horizon framework which simulates on-line usage.

The rest of this paper is structured as follows. In Section 2 we review the related literature and underline the contribution of this article. In Section 3 we formally introduce the problem and the corresponding mathematical programming model. In Section 4 we describe two methods for deciding where to relocate vehicles in anticipation of future
2. Literature review

The routing decisions considered in the FMRSP share similarities with those involved in well studied routing problems. Among these we find the Vehicle Routing Problem (VRP). Starting from the seminal paper of Dantzig and Ramser (1959), several exact and heuristics algorithms were proposed to solve VRPs (Bräysy and Gendreau, 2005; Bertsimas et al., 2019; Toth and Vigo, 2002) and several flavors of the problem have been studied, see e.g., the surveys (Kumar and Panneerselvam, 2012; Pillac et al., 2013; Lin et al., 2014; Ritzinger et al., 2016; Braekers et al., 2016). One of the major differences between the FMRSP and the different variants of the VRP is that VRPs typically consist of designing tours returning to the depot, while the FMRSP designs open paths from the vehicle’s origins to a common destination. Arguably, a (variant of the) VRP would resemble more closely a last-mile ride-sharing problem where a vehicle departs and returns to the station visiting the destinations of a number of customers. Furthermore, FMRSPs focus on transporting customers from multiple locations to the destination (station) while VRPs are typically concerned with the delivery of goods to customers. This impacts the types of restrictions imposed on the routes.

Particularly, the FMRSP shares features with the Dial-a-ride Problem (DARP) and the Pick-up-and-delivery Problem (PDP) (Cordeau and Laporte, 2003; Ropke and Cordeau, 2009; Berbeglia et al., 2010), which are generalizations of the VRP. A comprehensive review of DARP and PDP can be found in Ho et al. (2018). The goal of the DARP is to minimize the cost/time to transport a set of passenger by means of a fixed fleet of vehicles. Requests have different pickup and delivery locations, and the vehicles can pick up more than one passengers at a time. Also for the DARP different variants can be found, such as where the objective is to minimize the detour for the customers on board the vehicles (Pfeiffer and Schulz, 2022). The DARP can be considered as a variant of the PDP. The PDP typically deals with the transportation of goods while the DARP deals with passenger transportation (Parragh et al., 2008). Thus, the difference between DARP and PDP is usually expressed in terms of additional constraints or objectives that explicitly take user (in)convenience into account (e.g., time window and vehicle capacity constraints). The FMRSP can be seen as a special case of DARP where passengers travel to a common depot (station) and with additional service-specific constraints. In particular, the FMRSP takes the desired arrival time of accepted customers as constraints. This, in turn, implicitly shapes feasible time window for the other customers on board the same vehicle and for the newly arrived customers in a rolling horizon optimization framework. Particularly, in this paper, we study on-line dispatch and rebalancing decisions. That is, we consider the allocation of customers requests to vehicles as they arrive and while vehicles are busy with other transportation requests. This entails dealing two types of customers. First, we find customers whose request has been accepted in previous decision epochs and have not yet picked up. These customers requests must be satisfied. Second, we find new customers whose request may or may not be accepted, similarly to a
consider a set of passengers requiring a ride and study the problem via periodic re-optimization. Lotfi and Abdelghany (2022) study the problem of assigning customers to vehicles. The authors include service-specific constraints, including time windows and latest time for the problem of assigning customers to vehicles. Wang et al. (2018) consider a ride-share setting in which a service operator collects requests from individual travelers. Based on every request’s origin and destination as well as the desired pick-up time, the service provider matches vehicles and travelers and builds a route plan for each vehicle. Shen et al. (2018) study the integration of a FMRS service based on autonomous vehicles (AVs) with public transportation. The idea is to preserve high demand bus routes while using shared AVs as an alternative for low demand routes. In a simulation framework they use simple heuristics to match passengers to vehicles and define routes. Chen et al. (2020) provide a mixed-integer linear programming (MILP) model to decide the assignment of request groups to AV in FMRS service. The objective is that of minimizing operational costs. The authors devise a cluster-based solution method to deal with large-scale instances. Zhao et al. (2018) address the joint problem of optimally matching passengers and vehicles and that of routing each vehicle. The problem is formulated as a PDP with the addition of space–time windows. Wang et al. (2018) consider a ride-share setting in which a ride-share provider receives trip requests over time from potential participants. A trip can be either a driver or a rider. This process generates two disjoint sets of trip requests. The authors focus on finding a stable match between the two sets. Ma et al. (2019) study the problem of assigning customers to vehicles. The authors include service-specific constraints, including time windows and latest time for accepting or rejecting a transportation request. The authors address the problem via periodic re-optimization. A ride can be either a driver or a rider. This proce...
A number of studies address the problem of rebalancing vehicles. Wen et al. (2018) propose a reinforcement learning method to move idle vehicles in a shared mobility-on-demand systems. They test their solution method on a first-mile ride-sharing service in the city of London. Mao et al. (2020) consider a taxi sharing systems with AVs. They study the problem of determining the number of AVs to send from a zone of the city to another in order to minimize the expected cost of repositioning AVs. They compare a reinforcement learning algorithm with an integer programming model that assumes full knowledge of future demand. Wallar et al. (2018) propose algorithms for partitioning the operating area into zones, estimating the real-time demand and rebalancing idle vehicles. Sayarshad and Chow (2017) propose a queue-based model for matching and rebalancing decisions. They assume that the number of idle vehicles is known in advance. Alonso-mora et al. (2018) design a matching algorithm for on-demand ride-sharing. The method incorporates rebalancing decisions for idle vehicles. The authors describe a MILP formulation for the problem and solve the problem via a specialized procedure that begins by assigning passengers to vehicles and finding feasible trips, and terminates by rebalancing idle vehicles. Ma et al. (2019a) study a more involved system in which a ride-sharing fleet is operated jointly with public transport services in order to arrange complete on-demand journeys for their customers. The authors consider also the rebalancing of idle vehicles. They propose a queueing-theoretic model for the problem.

As it is evident in Table 1, the available literature has typically addressed matching decisions (i.e., the assignment of passengers to vehicles) and rebalancing decisions (i.e., the assignment of vehicles to zones) separately. Furthermore, when rebalancing decisions are addressed, they concern mainly idle vehicles, that is vehicles which have not been dispatched to customer requests. Thus, matching and rebalancing decisions have been understood as sequential decisions. First, vehicles match current requests, then the remaining ones may be rebalanced. Finally, it is possible to notice that not all articles that study rebalancing decisions provide an optimization model for that. We extend the state-of-the-art in the following ways:

1. We address matching, routing and rebalancing decisions simultaneously. This entails that we do not necessarily rebalance only idle vehicles. In our approach, vehicles may move to promising demand areas in advance, even if this entails giving up the profit of a current request.  
2. We consider online optimization with binding acceptance of transportation requests. This entails that a subset of the customers (those whose request has been accepted in previous decision problems) must be serviced, while the remaining customers (those newly arrived) may be picked up if feasible and profitable. A side effect of this is that previously and newly accepted customers have an impact on the time window of the vehicle.
3. For this problem we provide an explicit mathematical model. The model includes service-specific constraints such as maximum waiting time, latest arrival time, and the necessity of fulfilling binding acceptance of transport requests.
4. We propose simple techniques to identify promising locations where to rebalance.
5. We test our model in a rolling-horizon simulation framework with periodic re-optimization based on randomly generated instances to assess the solutions delivered by the model and, particularly, the advantage provided by rebalancing activities.

It must be noted that rebalancing decisions have been extensively studied in other emerging problems in shared mobility. These include carsharing (Illgen and Höck, 2019; Folkestad et al., 2020; Pantuso, 2022), bike sharing (Faghihi-Imani et al., 2017; Liu et al., 2016; Chemla et al., 2013), scooter sharing (Osorio et al., 2021). Nevertheless, the relocation problem involved is significantly different. In the ride-sharing problem, a vehicle has to drive (with its own driver) to a more promising location. In the other vehicle-sharing problems, vehicles have to be picked up by drivers or service vehicles to be moved to more promising locations. The amount of work in the latter is typically much higher and the relocation problem alone may involve complex optimization problems. Similarities may emerge in the methods used to predict demand occurrence. However, we believe the methods proposed are not immediately applicable due to the inherent differences in the systems and types of demand.

3. The first-mile ride-sharing problem

In this section, we formally introduce the First-Mile Ride-Sharing Problem. We start, in Section 3.1, by providing a general introduction to the problem. Following, in Section 3.2, we introduce a mathematical model for the FMRSP. In addition, in Appendix A we provide a table that summarizes the notation and in Appendix B we provide a simple example that illustrates possible feasible solutions to the problem.

3.1. Problem statement

We consider the operator of a fleet of vehicles $\mathcal{X} := \{1, \ldots, K\}$ concerned with dispatch and relocation decisions in order to ensure a first-mile ride-sharing service. The fleet is homogeneous with capacity $Q$. We assume the operator makes dispatch and relocations decisions periodically, e.g., every 5 or 10 min, as a result of the arrival of new transportation requests. We refer to these decision times as “(re-)optimization phases”. At each re-optimization phase, the available customers can be partitioned in two sets, namely $\mathcal{A}_p := \{1, \ldots, N_p\}$ which contains the customers whose transportation request had already been accepted during a previous optimization phase, and $\mathcal{A}_c := \{1, \ldots, N_c\}$ which contains newly arrived customer requests which have not been considered in previous optimization phases. We assume that the customers in $\mathcal{A}_c$ may be either accepted (and thus assigned to a vehicle) or rejected, while the customers in $\mathcal{A}_p$ must be picked up (thus we assume acceptance decisions are binding). For convenience we set $\mathcal{A}_h := \mathcal{A}_c \cup \mathcal{A}_p$.

All customers travel to a common destination $d$ located in position $o(d)$ (e.g., a transit station) and for each customer $i$, the operator knows the requested pick-up time $T_i^p$, the requested arrival time $T_i^A$ and the origin $o(i)$. We let $\Delta$ be the maximum waiting time (i.e., difference between actual pick-up time and requested pick-up time). Similarly, at the beginning of the re-optimization phase, denoted $T_i$, each vehicle $k$ is located at $o(k)$ as a result of previous deployment or relocation decisions. The vehicle is either idle in its location, or traveling between customers or to the station. In addition, vehicles might initially have customers on board. We denote $V_i$ the number of customers on board of vehicle $k$ at the beginning of the re-optimization phase and $T_i$ the earliest arrival time of the passengers already on board vehicle $k$. The operator needs to ensure that vehicles with customers on board terminate their journey to the station. Conversely, vehicles with no customers on board may be sent to a rebalancing point or stay at their origin location $o(k)$. A set $\mathcal{R} := \{1, \ldots, r\}$ of potential rebalancing points in the operating area is available. For each rebalancing point $r$ we let $B_r$ denote an upper bound on the number of vehicles that can be dispatched to the rebalancing point.

The operator bears transportation costs generated by vehicle movements. Particularly, we assume travel times are known, with $T_{ij}^P$ being the travel time between locations $o(i)$ and $o(j)$ with $i, j \in \mathcal{X} \cup \mathcal{A}_p$, $j \in \mathcal{A}_c \cup \mathcal{R} \cup \{d\}$ and cost $C$ is born for each unit of travel time. The operator collects a revenue $P_{ij}$ when picking up customer $i$, for $i \in \mathcal{A}_c$. Note that the revenue is collected only when picking up new customers as we assume the revenue for the customers in $\mathcal{A}_p$ has been collected during previous optimization phases. Furthermore, $E_i$ denotes the expected revenue collected for each vehicle relocated to rebalancing center $i \in \mathcal{R}$. Parameter $E_i$ is calculated as $\bar{P}_i - CT_{id}$, where $\bar{P}_i$ is the
expected revenue obtained from dispatching a vehicle to rebalancing center \( i \in \mathcal{R} \). Expected future revenues from rebalancing activities are discounted using a parameter \( \beta \) that denotes the weight of the rebalancing reward.

The decisions made by the operator can be formalized as follows. We let \( x^k_{ij} \) take value 1 if vehicle \( k \) moves directly between \( (i) \) and \( (j) \), 0 otherwise, for all \( i \in \{k \cup N_U, j \in N_U \cup R \cup \{d\}, k \in \mathcal{X} \). Furthermore, we let \( t^k_i \) denote the actual arrival time of vehicle \( k \) to the station, for \( k \in \mathcal{X} \) and \( t^p_i \) denote the actual pick-up time of customer \( i \), for \( i \in N_U \).

Thus, we use a 3-index formulation of size \( O(|N_U|(|\mathcal{X}||\mathcal{R}||\mathcal{X}||\mathcal{R}|)) \). The FMRS is NP-hard, as it contains the prize-collecting TSP (Balas, 1989) as a special case.

### 3.2. Mathematical model

Having defined all decision variables and parameters, we may formulate the problem as follows:

\[
\begin{align*}
\max \sum_{k \in \mathcal{X}} \sum_{i \in \mathcal{R} \cup \{k\}} \sum_{j \in \mathcal{R} \cup \{d\}} P_{ij} x^k_{ij} - \sum_{i \in \{k \cup N_U, j \in \mathcal{R} \cup \{d\}, k \in \mathcal{X} \}} \sum_{k \in \mathcal{X}} CT_{ij} x^k_{ij} + \beta \sum_{i \in \mathcal{X}} \sum_{k \in \mathcal{X}} x^k_{ij} E_i & = \in \mathcal{X} \\
\text{s.t.} \quad \sum_{j \in \{k \cup N_U, j \in \mathcal{R} \cup \{d\}, k \in \mathcal{X} \}} x^k_{ij} & \leq 1 \quad \forall i \in N_C \quad (1a) \\
\sum_{j \in \{k \cup N_U, j \in \mathcal{R} \cup \{d\}, k \in \mathcal{X} \}} x^k_{ij} & = 1 \quad \forall i \in N_P \quad (1b) \\
\sum_{i \in \{k \cup N_U, j \in \mathcal{R} \cup \{d\}, k \in \mathcal{X} \}} x^k_{ij} & \leq 1 \quad \forall k \in \mathcal{X} \quad (1c) \\
\sum_{j \in \{k \cup N_U, j \in \mathcal{R} \cup \{d\}, k \in \mathcal{X} \}} x^k_{ij} = \sum_{i \in \{k \cup N_U, j \in \mathcal{R} \cup \{d\}, k \in \mathcal{X} \}} x^k_{ji} & \forall j \in N_U, k \in \mathcal{X} \quad (1d) \\
\sum_{i \in \{k \cup N_U, j \in \mathcal{R} \cup \{d\}, k \in \mathcal{X} \}} x^k_{ij} + V_k & \leq Q \quad \forall k \in \mathcal{X} \quad (1e) \\
\sum_{k \in \mathcal{X}} x^k_{ij} & \leq B_j \quad \forall j \in \mathcal{R} \quad (1f) \\
V_k & \leq Q \left(1 - \sum_{j \in \mathcal{R}} x^k_{ij}\right) \quad \forall k \in \mathcal{X} \quad (1g) \\
V_k & \leq Q \sum_{j \in \mathcal{R}} x^k_{ij} \quad \forall k \in \mathcal{X} \quad (1h) \\
t^p_i + T_{ij} & \leq t^p_i + T^L (1 - \sum_{k \in \mathcal{X}} x^k_{ij}) \quad \forall i, j \in N_U \quad (1i) \\
T + T_{kj} & \leq t^p_i + T^L (1 - x^k_{ij}) \quad \forall j \in N_U, k \in \mathcal{X} \quad (1j) \\
t^p_i - T^p_i & \leq \Delta \quad \forall i, j \in N_U \quad (1k) \\
t^k_i & \leq T^A_i + T^L (1 - \sum_{j \in \mathcal{R}} x^k_{ij}) \quad \forall i \in N_U, k \in \mathcal{X} \quad (1l)
\end{align*}
\]

Objective function (1a) represents the profit for the operator. The first term represents the revenue generated by picking up customers, the second term the total cost born of the vehicles movements and, finally, the third term is the discounted expected profit obtained in the rebalancing centers.

Constraints (1b) and (1c) state that new customers may be picked up at most once and customers already accepted must be picked up exactly once, respectively. Observe, in (1b) and (1c), that after visiting a customer \( i \in N_C \) or \( i \in N_P \), the vehicle can only move to another customer \( i \in N_P \cup R \cup \mathcal{X} \) or to the station \( d \). Constraints (1d) ensure that vehicles travel to the station at most once. Constraints (1e) state that whenever a vehicle arrives at a customer location, it must then move to another customer or to the station. Notice that a vehicle can arrive at a customer location \( j \) either from another customer or from the vehicle’s original location \( o(k) \). We remind the reader that variable \( x^k_{ij} \) must be understood as vehicle \( k \) moving from its original location \( o(k) \) to the location \( o(j) \) of customer \( j \). Constraints (1f) state that, if a vehicle departs from its original location \( o(k) \), i.e., \( x^k_{ij} = 1 \) for some \( j \), it must terminate its journey either at the station or at a rebalancing point. Also in this case, variables \( x^k_{ij} \) must be understood as the vehicle moving from its origin, \( o(k) \), see Section 3.1.

Constraints (1g) ensure that the capacity of the vehicles is not exceeded, while constraints (1h) ensure that the total number of vehicles dispatched to a rebalancing center will not exceed the upper bound on the vehicles dispatchable at the rebalancing center. Constraint (1i) states that only empty vehicles may be dispatched to rebalancing centers. For instance, if vehicle \( k \) is dispatched to one of the rebalancing center, the right-hand-side becomes 0, and the constraints can only be satisfied when \( V_k \) is equal to 0. If vehicle \( k \) is not dispatched to any rebalancing center, the right-hand-side reduces to the capacity of the vehicle, and the constraint holds with any value of \( V_k \). Notice that the movements between customer points \( N_U \) and rebalancing points are automatically forbidden by the absence of the corresponding \( x^k_{ij} \) variables. Constraints (1j) state that the vehicles that already have customers on board at the beginning of the period must be dispatched (i.e., cannot stay idle). If \( V_k \) is strictly positive, the constraint forces the right-hand-side to be strictly positive as well, and thus to dispatch the vehicle.

Constraints (1k) state that if customer \( j \) is picked up by vehicle \( k \) immediately after picking up customer \( i \), then the actual picking up time of customer \( i \) plus the travel time between customer \( i \) and \( j \) must be less or equal to customer \( j \)’s actual pick up time. Here \( T^L = \max\{T^A_k\} \) for \( i \in N_U \) is an upper bound on the requested arrival time. Similarly, constraints (1l) denote the pick-up time for the first customers in the route. Constraints (1m) ensure that the difference between the actual pick up time and the requested pick up time of the customer does not exceed the maximum waiting time \( \Delta \). Constraints (1n) ensure that the actual arrival time of vehicle \( k \) must be earlier than the requested arrival time of any of the customers on board of it. For instance, if customer \( i \) is picked up by vehicle \( k \), the right-hand-side becomes \( T^A_k \) enforcing that the actual arrival time of vehicle \( k \) is before \( T^A_k \). Constraints (1o) ensure that the actual arrival time of vehicle \( k \) is earlier than the earliest requested arrival time \( T_{ij} \) of the passengers.
on board at the beginning of the re-optimization phase. Constraints (1p) state the relationship between pick-up time and arrival time. For example, if \( j \) is the last customer picked up by vehicle \( k \) before the station \( s_{jd}^p \) takes value 1, the left-hand-side becomes \( t_k^p + T_{jd} \), and the right-hand-side becomes \( t_j^k \), enforcing that the actual pick-up time of customer \( j \) plus the travel time between customer \( j \) and station be less than or equal to the actual arrival time of vehicle \( k \). If \( j \) is not the last customer picked up by the vehicle \( k \) before arrive at the station, then the left-hand-side becomes \( t_k^p \), the right-hand-side becomes \( t_j^k + T_j^L \), which always holds. Finally, constraints (1q)-(1s) define the domain of the decision variables.

An illustrative example of possible solutions to the problem is provided in Appendix B.

4. Finding rebalancing centers

Identifying where to rebalance in order to anticipate demand, and how many vehicles to send to each rebalancing point is currently an open research question. Any such prediction model could be used to feed rebalancing centers to model (1). In this section we introduce a clustering-based method for identifying rebalancing centers. We refer to the method as the K-means Clustering (KC) method. The method identifies both the location and demand of the rebalancing centers, and in this turn allows us to determine the upper bound on the number of vehicles dispatchable to the different rebalancing centers.

Given a number \( k \) of rebalancing centers to find, the KC method finds rebalancing centers by partitioning all requests received in the current re-optimization phase into \( k = |\mathcal{R}| \) clusters. Clusters are created in such a way as to minimize the total distance between the points allocated to the cluster and the centroid of the cluster. The centroids of the clusters will then be used as rebalancing centers. The expected demand (number of requests) of the rebalancing centers will be set equal to the number of requests in the corresponding cluster. We let \( D_r \) be the demand of rebalancing point \( r \in \mathcal{R} \).

The rational behind the clustering method is the following. Assume that the decision maker performs frequent re-optimizations and that the demand distribution changes slow enough. Then the geographical distribution of demand in the near future is approximately the same as the current demand. Current requests represent a sample from this (unknown) distribution. Thus, over many repetitions one expects to put rebalancing centers where there is actually more demand. We believe our study can provides evidences based on a more involved setup of the service.

The different configurations are tested on a set of random instances introduced in Section 5.2. All problems are solved with the Python libraries of GUROBI 9.5.0 and a server equipped with Intel Core i5 and 16 GB of RAM.

5. Numerical experiments

In this section we report the results of our numerical experiments. The scope of the experiments is to assess, in terms of profits and service rates, two different configurations of the service which we refer to as without rebalancing (woR) and with rebalancing (wR). The configuration woR refers to the situation where the service provider dispatches the vehicles only based on customer requests of the current re-optimization phase. For the model without rebalancing, we simplify our model in Section 3 by having an empty set of rebalancing centers. In the configuration wR, the service provider makes the dispatching decision based both on current customer requests and on predicted demand using rebalancing centers. In this case, we test the model with two different method for finding rebalancing centers. In the first case, which we refer to as wRKC, the company use KC to obtain the location and demand of the rebalancing centers. In the second case, which we refer to as wRRS, the company uses the RS method to find the locations and demands of the rebalancing centers.

Observe that the potential value of relocation activities in on-demand mobility has been the focus of other studies, such as Ma et al. (2019b), Jamshidi et al. (2021), Sayarshad and Chow (2017), Kash et al. (2022) and Danassis et al. (2022) for different configurations of the service. Particularly, our work shares similarities with Sayarshad and Chow (2017) who also explicitly model the decision of the service provider as an optimization model. However, with respect to Sayarshad and Chow (2017) we consider (i) binding previously accepted requests, (ii) customers desired arrival times and (iii) an upper bound on the maximum waiting time. We believe our study can provides evidences based on a more involved setup of the service.

5.1. Simulation framework

We test our model in a simulation framework, based on rolling-horizon optimization, with a planning horizon of one hour. We assume online re-optimization happens every 5 minutes. This means that every simulation requires the solution of 12 optimization problems (1). At each re-optimization we update the status of the system, and randomly generate (as explained in the next section) a number of new customers. Particularly,

- At the initial optimization phase, say \( T = 0 \), we assume that the \( X_p \) is empty. This means that there is no customer whose request had already been accepted in a previous optimization phase. We generate a number of new customers \( X_C \), rebalancing centers \( \mathcal{R} \), and initial vehicle positions (as explained in the next section) and solve the resulting model (1).

- We then step five minutes forward in time, say optimization phase \( T = 1 \), and assume the solution to the model for \( T = 0 \), has been implemented. This entails the vehicle followed the routes determined by the previous optimization model for five minutes, moving either to customers locations or to rebalancing centers. This provides their updated location for the new re-optimization phase. We then partition the customers of the previous optimization phase into three groups:

1. The first group contains those customers that had been assigned to a vehicle but have not yet been picked up in the five minutes interval between the two re-optimizations (i.e., the route of the vehicle assigned to the customer did not stop by the customer within the five-minute interval between re-optimizations). These customers form the set of mandatory customers \( X_p \) in the new re-optimization phase and must be picked up by some vehicle (possibly different from the one assigned in the previous re-optimization phase).
2. The second group contains those customers that had been assigned to a vehicle in the previous re-optimization phase and the route of the assigned vehicle stopped by the customer within the five-minute interval between re-optimizations. In the new re-optimization phase these customers represent occupied seats \( V_i \) in the vehicles to which they were assigned. Therefore, these customers represent fulfilled requests and do not appear in \( \mathcal{N}_p \) in the new re-optimization phase.

3. The third group contains those customers that had not been assigned to a vehicle in the previous re-optimization phase. These represent customers whose request has been rejected and will not show up in the new re-optimization phase.

Observe, that in the new re-optimization phase, vehicles may find themselves into one of the following situations. (A) The vehicle is empty at a given position and was on the way to pick-up customers or to a rebalancing centers. (B) The vehicle has passengers on board at a given position and was on the way to pick-up additional customers or to the station. In case (A), in the new re-optimization phase the vehicles may be assigned to new customers or to a rebalancing center, independently of the decision made in the previous re-optimization phase. That is, it is possible that the vehicle is assigned to a set of customers different from the ones previously assigned to the vehicle. In case (B) the vehicle cannot be sent to a rebalancing center but may be assigned to new customers or sent directly to the station. Following, we generate new customers \( \mathcal{N}_C \) for the new re-optimization phase and resolve a problem \((1)\).

The procedure continues stepping five minutes forward in time until the end of the one-hour planning horizon. Thus, we are able to collect statistics on the performance of the service. Particularly, the profit is computed as follows. At the end of each re-optimization phase, we collect the fee for all the customers that have been accepted (i.e., a vehicle has been assigned to them) and picked up (i.e., the vehicle has arrived at their location during the five-minute interval between re-optimizations) and subtract the cost of the movements the vehicles have done during the five-minute interval between re-optimizations. The final total profit is then the sum of the individual profits made during the one-hour planning horizon. The procedure is explained by the following example.

• Assume that at \( T = 0 \), \( \mathcal{N}_p \) is empty and \( V_i = 0 \) for all vehicles \( k \in \{ D_1, D_2, D_3 \} \). That is, there is no customer whose request had already been accepted in a previous optimization phase. We generate a number (say four) of new customers \( \mathcal{N}_C := \{ C^0_1, C^0_2, C^0_3, C^0_4 \} \), rebalancing centers \( \mathcal{R} := \mathcal{R}_0 \), and initial vehicle positions \( \{ D_1, D_2, D_3, D_4 \} \) and solve the resulting model \((1)\).

Assume that the solution determines the following routes for the three vehicles: \( \text{Route}^{10} := \{ D_1, C^0_1, d_1 \} \), \( \text{Route}^{20} := \{ D_2, C^0_2, d_2 \} \), \( \text{Route}^{30} := \{ D_3, C^0_3 \} \) and that customer \( C^0_4 \) is rejected.

The solution computed at \( T = 0 \) is implemented and the vehicle follow the respective routes for five minutes. This provides updated system information for optimization phase \( T = 1 \). That is, we updated the location of the vehicles \( \{ D_1, D_2, D_3, D_4 \} \). We observe that,

1. For \( \text{Route}^{10} \), the vehicle \( D_1 \) already picked up customer \( C^0_1 \) and is still on the way to pickup \( C^0_1 \), so we can delete \( C^0_1 \) from the system, update \( V_{D_1} = 1 \) and move \( C^0_1 \) to mandatory customer set \( \mathcal{N}_P := \{ C^0_1 \} \).

2. For \( \text{Route}^{20} \), vehicle \( D_2 \) already picked up customer \( C^0_2 \) and is still on the way to station, so we can delete \( C^0_2 \) from the system and update \( V_{D_2} = 1 \).

3. For \( \text{Route}^{30} \), vehicle \( D_3 \) arrived at the rebalancing center \( R^0 \), thus \( V_{D_3} = 0 \).

In the new optimization phase \( T = 1 \), since the vehicles \( D_1, D_2 \) already have customers on board, they cannot be sent to a rebalancing center but may be assigned to new customers or sent directly to the station. However, vehicle \( D_3 \) may be assigned to new customers or to a rebalancing center as it is still empty. Following, we generate new customers \( \mathcal{N}_C := \{ C_1, C_2, C_3, C_4 \} \) for the new re-optimization phase \( T = 1 \) and resolve the problem \((1)\).

5.2. Instance generation

We generate a number of artificial and randomly generated instances that mimic real-life operating scenarios for the service. Particularly, we assume a fleet of homogeneous vehicles of capacity \( Q = 4 \). The position of the vehicles for the first re-optimization phase is generated randomly in the business area (defined below) and initially vehicles are assumed to have no passenger on board. For the re-optimization phases other than the first, the position of the vehicles, and the number of
passengers on board is computed as the result of previous optimization phases.

We consider two different geographies of the business area. In the first geography, depicted in Fig. 3, the station is located at the center of the business area, and the business area itself is represented by a circle of radius $R = 4$ km. In the second geography, see Fig. 4, the station is located in a corner and the business area is a quarter of a circle of radius of $R = 8$ km. The second geography is meant to represent urban contexts where the demand is concentrated only on one side of the station due to, e.g., physical barriers such as rivers or harbors.

For each geography, and for each re-optimization phase, customer requests are generated in the following two different scenarios of demand distribution. In the first scenario, referred to as the uniform demand scenario, pickup locations are randomly scattered in the whole business area, see Figs. 3(a) and 4(a). In the second scenario, referred to as the non-uniform demand scenario, one third of the requests arrive, randomly, from inside the inner circle of radius $R$ (where $R$ is the radius of the outer circle), while the remaining requests arrive, randomly, from the outer portion of the circle, see Figs. 3(b) and 4(b). We obtain, in total, four configurations namely

1. station in the center and uniform demand (UCt),
2. station in the center and non-uniform demand (NUCt),
3. station in the corner and uniform demand (UCn),
4. station in the corner and non-uniform demand (NUCn).

For each request, the requested pickup time ($T_p^i$) is randomly generated uniformly between 0 and 3 minutes after the beginning of the planning horizon, and the requested arrival time ($T_a^i$) is set as the sum of the requested pickup time, $T_p^i$, travel time between customer $i$ and the station $d_i$, and a buffer time randomly generated between 5 and 8 minutes. Travel time $d_i$ are calculated using Euclidean distances and assuming an average speed of 36 km/h (Commission and Limousine, 2023). The unit transportation cost $C$ is set to $11.25$/h (English, 2008) and trip revenues $P$ are computed using a fare of $2.59$ per Km traveled plus $0.74$ per minute traveled, with a minimum fare of $8$ following the setting in INSHUR (2022).

We set the value of $\beta$ to 0.1 in the objective function, unless otherwise specified. Observe that $\beta$ determines the impact of rebalancing movements. High values will increase the potential benefit of rebalancing and might lead to rejecting current customers while low values might lead the model to provide myopic decisions. The impact of different values of $\beta$ will be assessed through sensitivity analysis in Section 5.5. The expected revenue for rebalancing a vehicle to center $i \in \mathcal{B}$, $E_i$, is calculated as $E_i = CT_{id}$, where $E_i$ is the revenue of dispatching a vehicle to rebalancing center $i \in \mathcal{B}$, and is calculated as $E_i = Q P_i^b$, where $P_i^b$ the average revenue for the requests in the cluster where $i$ is the centroid and $Q$ is the capacity of the vehicles. $C$ is the unit transportation cost, set as above, and $T_{id}$ is the distance between rebalancing point $i$ and the station. To obtain the upper bound on the number of vehicles dispatchable to a rebalancing center ($B_i$, see Section 4) we set the average number of customers on board during one trip of a vehicle $Q$ equal to half of the capacity $Q = 4$.

For each configuration, we generate different instances varying in the number of vehicles and customers that appear at each new re-optimization phase. Particularly, we create instance classes named $\mathcal{C}: \mathcal{A}: |\mathcal{X}| \in \mathcal{M}$ with number of customers $|\mathcal{A}| \in \{1, 2, 3, 4, 5\}$, and number of vehicles, $|\mathcal{Y}| \in \{10, 12, 14\}$. As an example, $\mathcal{C}: \mathcal{A}: |\mathcal{X}| \in \mathcal{M}$ indicates a class of instances with 8 new customers in each re-optimization phase and 10 vehicles available for dispatching for the whole planning period. For each instance class we randomly generate 3 different instances. Observe, however, that for each instance we solve 12 different optimization problems in our simulation framework.

We set the number of rebalancing centers $|\mathcal{B}|$ to 3 in all instances (later we perform sensitivity analysis with respect to this parameter.). This number is found using the Elbow Method (EM) and the Silhouette Analysis Method (SAM) (Mahendra, 2019). Particularly, we use the instances with $|\mathcal{A}| = 8$ as a reference case to find a suitable value for $k = |\mathcal{B}|$. First we randomly generate 8 customers in the operating area, then we use the EM approach to show the performance of the KC method for different values of $k$. For each $k$ we consider the Within-Cluster-Sum of Squared Errors (WSS). We then plot the WSS versus $k$, and choose the value of $k$ for which the WSS flattens. This point is referred to as an “Elbow”, see Fig. 5(a).

Nevertheless, in some cases, the EM does not give precise answers as it is not always clear for which value of $k$ the change of slope is significant. In such cases, we will use the SAM to make a decision. The silhouette score is a measure of how similar a customer request is to its own rebalancing cluster compared to other rebalancing clusters. To be more precise, the silhouette score of one customer request $i$ can be calculate as below:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

(2)
the instance, with the station located in the center. In this case, no rebalancing corresponds to leaving the vehicle close to the station, which is on average halfway to the next request in the worst case. Nevertheless, we observe that both the service rate and profits in the wRKC case are systematically higher than in the case woR, illustrating that rebalancing activities still pay off. Rebalancing according to the wRKC method is consistently preferable to rebalancing to random locations (wRRS) and rebalancing to random locations is not necessarily better than not rebalancing (woR). This is somewhat expected: When there is a clear pattern in the demand, sending vehicles to locations that are not carefully chosen represents, in general, a waste of resources. In general, when the station is in the center of the business area, we do not observe significant changes between the case with uniform demand (UCt) and non-uniform demand (NUCt).

When the station is located in a corner (configurations UCn and NUCn) we observe a much more marked difference in performance between the setting with and without rebalancing. Particularly, we notice that the setting wRKC improves both profits and service rates by approximately 30%, compared to woR in the case UCt. In the configuration NUCn the improvement is even more marked with approximately 50% higher in service rates and 60% higher profits. Random rebalancing (wRRS) is still slightly preferable to no rebalancing (woR) but lags significantly behind compared to the strategy of using clustering-based rebalancing centers (wRKC).

In Tables 2 and 3 it can be further observed that the gap between wRKC and woR tends to increase with the number of customers, keeping the fleet size fixed. In the NUCcase we observe that, in the wRKC case, both profits and service rates improve as the number of vehicles increases, keeping fixed the number of customers. Compared to the UCcase we notice that for the woR strategy, the highest service rate for the GBY* instances is lower than 60%, while the lowest service rate for the same instances in the UCn scenario is above 60%.

### Table 2
Profit [\$] in the simulations with \(|\mathcal{E}| = 3\).

<table>
<thead>
<tr>
<th></th>
<th>GSV10</th>
<th>GSV12</th>
<th>GSV14</th>
<th>C7V10</th>
<th>C7V12</th>
<th>C7V14</th>
<th>C8V10</th>
<th>C8V12</th>
<th>C8V14</th>
</tr>
</thead>
<tbody>
<tr>
<td>wRKC</td>
<td>57.52</td>
<td>59.06</td>
<td>58.20</td>
<td>68.15</td>
<td>68.31</td>
<td>71.14</td>
<td>79.16</td>
<td>81.46</td>
<td>85.17</td>
</tr>
<tr>
<td>NUCt</td>
<td>55.62</td>
<td>58.04</td>
<td>55.54</td>
<td>64.59</td>
<td>67.58</td>
<td>70.51</td>
<td>75.36</td>
<td>80.47</td>
<td>77.63</td>
</tr>
<tr>
<td>woR</td>
<td>56.37</td>
<td>56.61</td>
<td>56.42</td>
<td>63.96</td>
<td>67.08</td>
<td>65.72</td>
<td>74.71</td>
<td>77.52</td>
<td>78.71</td>
</tr>
<tr>
<td>UCt</td>
<td>54.35</td>
<td>57.53</td>
<td>54.90</td>
<td>60.64</td>
<td>64.79</td>
<td>65.54</td>
<td>72.78</td>
<td>74.55</td>
<td>76.47</td>
</tr>
<tr>
<td>wRRS</td>
<td>51.41</td>
<td>53.99</td>
<td>53.45</td>
<td>60.97</td>
<td>62.20</td>
<td>60.98</td>
<td>70.65</td>
<td>70.52</td>
<td>74.08</td>
</tr>
<tr>
<td>wRKC</td>
<td>50.88</td>
<td>55.03</td>
<td>54.35</td>
<td>61.46</td>
<td>61.68</td>
<td>65.22</td>
<td>66.79</td>
<td>71.78</td>
<td>68.77</td>
</tr>
<tr>
<td>NUCn</td>
<td>98.00</td>
<td>105.55</td>
<td>108.57</td>
<td>114.57</td>
<td>121.98</td>
<td>133.76</td>
<td>125.51</td>
<td>148.86</td>
<td>154.95</td>
</tr>
<tr>
<td>woR</td>
<td>97.16</td>
<td>72.65</td>
<td>85.50</td>
<td>97.42</td>
<td>101.69</td>
<td>93.04</td>
<td>97.07</td>
<td>101.03</td>
<td></td>
</tr>
<tr>
<td>UCn</td>
<td>95.42</td>
<td>100.10</td>
<td>103.81</td>
<td>108.41</td>
<td>119.32</td>
<td>118.73</td>
<td>114.99</td>
<td>128.97</td>
<td>146.30</td>
</tr>
<tr>
<td>wRRS</td>
<td>96.91</td>
<td>77.88</td>
<td>88.76</td>
<td>86.11</td>
<td>86.95</td>
<td>93.30</td>
<td>76.58</td>
<td>88.92</td>
<td>104.48</td>
</tr>
<tr>
<td>wRKC</td>
<td>70.33</td>
<td>69.94</td>
<td>89.82</td>
<td>77.23</td>
<td>94.43</td>
<td>86.60</td>
<td>90.63</td>
<td>106.04</td>
<td>91.04</td>
</tr>
</tbody>
</table>

where \(b(i)\) is the average of the minimum euclidean distance between customer \(i\) and the customers in clusters other than the one customer \(i\) belongs to. Parameter \(a(i)\) is the average euclidean distance from customer \(i\) to other customer requests inside customer \(i\)'s own rebalancing cluster. For each \(k\), we sum \(s(i)\) for all customer requests, and we plot it against \(k\), see a qualitative description in Fig. 8. We then choose the value of \(k\) for which the sum is the highest (the higher \(s(i)\) the higher is the difference between a point \(i\) and the clusters other than the one it belongs to). In our case, the best value of \(k\) was found to be \(k = 3\) and therefore we use this value in the computational study.

### 5.3. Managerial insights

In this subsection, we assess the solutions provided by the model in terms of service rates and profits. The service rate is computed as the ratio between the total number of customers transported during the whole planning period over the total number of requests received in the same period. The profit consists of the cumulative profit over the whole planning period over the total number of requests received during the entire simulated period (thus once hours with re-optimization every five minutes). Particularly, we compare three different strategies, namely no rebalancing (woR), rebalancing to random rebalancing centers (wRRS), and rebalancing to rebalancing centers found with the KC method (wRKC), see Section 4. These three strategies are assessed on different configurations of the service, namely UCt, NUCt, UCn, NUCn, see Section 5.2.

Tables 2 and 3 report the profit and service rate for the different strategies and configurations of the service. When the station is in the center (UCt and NUCt) we notice that all rebalancing strategies yield a 90% or higher service rate. The profits are likewise relatively similar across rebalancing strategies. This is mainly due to the geography of the instance, with the station located in the center. In this case, no
In conclusion, wRKC outperforms both woR and wRRS in all scenarios. When the station is in the center (UCt and NUCt), the model reaches a service rate higher than 90% regardless of whether and how rebalancing is done. Nevertheless, profits and service rates are consistently higher when rebalancing centers are chosen with the clustering method (wRKC). When the station is in a corner (UCn and NUCn), rebalancing to centers chosen with the clustering method (wRKC) yields a much more marked improvement both in profits and service rate, especially with the demand is not evenly spread in the service area (NUCn). Furthermore, as we observe in this case, keeping fixed the number of customers, the profit increases with the size of the fleet. As more vehicles provide better opportunities for rebalancing and anticipating future requests. This also shows that rebalancing to appropriately chosen locations (e.g., wRKC) can improve the utilization of empty vehicles.

As explained in Section 4, the strategy of finding rebalancing centers by clustering existing requests is expected to work well provided that the demand changes slower than our re-optimization frequency. Particularly, we studied the geographies NUCt and NUCn, where the demand is not uniformly distributed, and assumed the distribution changes between re-optimizations. In both cases, we start with having 10% of the demand in the inner zone at the first optimization phase, and we increase the demand in the inner zone with an additional 5% at every re-optimization. In this way, the distribution for last (the 12th) re-optimization sees 65% of the demand coming from the inner zone. Tables 4 and 5 report profits and service rates, respectively. Compared with the default setup where one third of the demand comes from the inner region (see Tables 2 and 3) we notice that the profit increases substantially. This is due to the fact that, with a changing demand pattern, there will be more requests coming from the inner zone, which in general means short travel distances to the station. Despite this, the service rate for the wRKC strategy is slightly lower, due to the reduced prediction ability. Nevertheless, the difference between the wRKC, the random and the no-rebalancing strategies is still visible and in some cases more marked.

5.4. Experiments on a real case

We performed additional tests on a real-life data set. The data obtained from the New York City Yellow Taxi data set (Commission and Limousine, 2023). These additional tests are meant to validate the results obtained on the randomly generated instances.

Particularly, we generate instances by selecting from the records of taxi rides occurred on February 1st 2023 between 16:00 and 17:00. We focus in particular on zones 140, 141 and 237 and set zone 236 as the common destination zone, see Fig. 6. Thus, the geography of the instances is similar to the one of the randomly generated instances with the station in a corner. We set the re-optimization frequency to 15 minutes and consider a planning horizon made of 12 re-optimizations, hence 180 minutes. We assume that the distribution of the demand is as described in Table 6.

The speed is calculated as the average speed of all requests in the focal data set. The required pickup time is generated randomly between 3 and 5 minutes. The requested arrival time is generated as the travel time from the customer pick-up location to the station, plus a buffer time randomly generated between 5 and 5 minutes. The fare for picking up new customers is obtained from the original data by subtracting the tip. The original locations of the vehicles are also generated randomly in the selected zones.

We compare the rebalancing strategy wRKC against no rebalancing. In addition, we test a rebalancing strategy based on the aggregation of historical data. Particularly, in a new rebalancing strategy named wH we generate rebalancing points by clustering all requests obtained during
the previous two weeks (i.e., the last two weeks of January 2023). The rationale behind this is that by considering more historical data, we obtain more precise estimates of where requests may occur. The results are presented in Table 7. Also in this case we observe a marked superiority of the proposed rebalancing strategy. The service rate is almost always above 90%. The difference with the no-rebalancing strategy is significantly larger than the ones we used. In this section, we assess how a higher number of rebalancing centers affects the results. Particularly, we run the tests with four and five rebalancing centers. We found no statistically significant differences which allow us to draw conclusions. In other words, our results suggest that adding potential rebalancing centers does not necessarily yield better results. The results are provided in Appendix C.

5.6. Complexity of the models

For the numerical experiments presented above, we choose to use relatively small instances in order to obtain results that are not affected by optimality gaps. All the instances presented could be solved, in all re-optimization phases, to provable optimality within less than a second (45 seconds in the worst case).

Nevertheless, in practical situations one may encounter instances significantly larger than the ones we used. In this section, we assess how solution times and optimality gaps scale with the size of the instance. Particularly, we create instance classes named \( C_i \), \( V_i \) and \( |X_i| \) for \( i \in \{40, 50, 60\} \). As an example, \( C60V40 \) indicates a class of instances with 60 new customers in each re-optimization phase and 40 vehicles available for dispatch for the whole planning period. Fig. 10 reports the progression of upper and lower bounds for instances \( C60V40, C60V50 \) and \( C60V60 \). The optimality gaps of the three instances are 21.1%, 71.7% and 26.0%, respectively, after 120 seconds (an amount of time which is sufficiently large if the operator re-optimizes, e.g., every 5 minutes). We observe significant optimality gaps and, particularly, that the primal bound improves slowly while the dual bound remains steady. We believe that finding optimal or close-to-optimal solutions is valuable in this context. In case of lack of performance (in some dimension), the decision maker would be able to rule out the possibility that this is caused by highly sub-optimal solutions, and understand that the reason should rather be found in choices in terms of system design.

Fig. 11 provides the same information for three smaller instances, namely \( C25V15, C25V20 \) and \( C25V25 \). In this case, the instances are small enough to observe an improvement of the upper bound. Nevertheless, while a good primal solution is found rather quickly, the upper
bound improves slowly. This suggests new lines of research that provide both tighter formulations and methods, perhaps heuristic, to quickly find primal solutions in large scale instances.

6. Conclusions

In this paper we developed a MILP model for online order dispatching and vehicle rebalancing in a first-mile ride-sharing service. The model was used in a rolling-horizon simulation framework based on constructed instances. We assessed whether rebalancing is advantageous over leaving idle vehicles in their positions. We have done this using a clustering method to identify promising rebalancing locations. The results show that rebalancing using a clustering-based strategy consistently outperforms strategies based on rebalancing to random locations or not rebalancing at all. Particularly, rebalancing strategies perform dramatically better in a context where the station is not
centrally located (e.g., the business area is entirely on one side of the station).

The proposed method to find promising re-balancing locations based on current requests has some inherent limits. Particularly, it is founded on the assumption that current demand represents future demand. This is true in contexts where re-optimization is performed frequently and/or demand distribution changes slowly enough to make the difference between two re-optimizations negligible. Performing re-optimization frequently requires the ability of solving the proposed model efficiently also for large-scale instance. However, our tests show that finding high quality solutions and bounds to the problem presented becomes problematic as the size of the problem increases. This suggests future research on both efficient solution methods and tighter formulations. Furthermore, additional research is needed to develop methods that can accurately predict future occurrence of demand and promising re-balancing locations in more general conditions than the ones we assume in this article. Finally, the assumptions made in the optimization model could be relaxed to improve the model in a number of ways. As an example, we assume that a vehicle with passengers onboard cannot drive to a re-balancing center. However, it is reasonable to imagine a scenario where a vehicle drops off the passengers on board and then moves to a re-balancing center rather than waiting at the station. These solutions are currently infeasible in our model.

### Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Jinwen Ye reports financial support was provided by European Union.

### Acknowledgment

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 801199.

### Appendix A. Notation

**Sets:**

- \( A_P := \{1, \ldots, N_P\} \) is the set of customers that have been assigned to vehicles but have not been picked up yet in previous re-optimization phase, in which customers can be reassigned but cannot be rejected in current re-optimization phase.
- \( A_C := \{1, \ldots, N_C\} \) is the set of new customers for current re-optimization phase, in which customers can be accepted or rejected.
- \( A_U := A_C \cup A_P \) is the combined set of assigned and new customers.
- \( \mathcal{R} := \{1, \ldots, R\} \) is the set of rebalancing centers.
- \( \mathcal{X} := \{1, \ldots, K\} \) is the set of vehicles.

**Parameters:**

- \( d \): Denotes the destination/station node
- \( \alpha(k) \): Denotes the original location of vehicle \( k \), for \( k \in \mathcal{X} \)
- \( \alpha(i) \): Denotes the location of request \( i \in A_U \)
- \( P_i \): Denotes the profit of picking up customer \( i \), for \( i \in A_U \)
- \( C \): Denotes the unit travel cost of the vehicle.
- \( \beta \): Denotes the weight parameter of rebalancing reward.
- \( V_i \): Denotes the number of customers on board of vehicle \( k \) at the beginning of the operational period, for \( k \in \mathcal{X} \)
- \( Q \): Denotes the capacity of the vehicle.
- \( T_i \): Denotes the start time of the operational period.
- \( T_{ij} \): Denotes the travel time between locations \( o(i) \) and \( o(j) \), for \( i, j \in \mathcal{X} \cup A_U \), \( j \in A_U \cup \mathcal{R} \cup \{d\} \)
- \( T_{ij}^o \): Denotes the requested arrival time of customer \( i \), for \( i \in A_U \)
- \( T_{ij}^2 \): Denotes the upper bound of the requested arrival time of all the customers.
- \( T_i^p \): Denotes the requested arrival time of vehicle \( k \), for \( k \in \mathcal{X} \)
- \( T_i^P \): Denotes the requested pick up time of the customer \( i \), for \( i \in A_U \)

**Decision variables:**

- \( x_{ij}^o \): Equal to 1 if vehicle \( k \) moves directly between \( o(i) \) and \( o(j) \), 0 otherwise, for \( i \in \mathcal{X} \cup A_U \), \( j \in A_U \cup \mathcal{R} \cup \{d\} \), \( k \in \mathcal{X} \).
- \( t_{ij}^o \): Denotes the actual arrival time of vehicle \( k \) to the station, for \( k \in \mathcal{X} \)
- \( t_{ij}^P \): Denotes the actual pick up time of customer \( i \), for \( i \in A_U \)

### Appendix B. Illustrative example

In what follows we provide two examples to illustrate how the model in Section 3.2 works. The online FMRSP model re-optimizes the vehicles dispatch and rebalancing decisions at fixed time intervals. Assume that the current re-optimization phase start at \( T = T_1 \) and a snapshot of the system, which includes vehicles and customers positions, is shown in Figs. 12(a) and 13(a). The blue circles denote customers while the yellow squares denote vehicles. Assume all vehicles have a capacity \( Q = 2 \), and that are all empty except for D3 that has currently one customer on board (\( V_{D3}_{1} = 1 \)). The triangle denotes the station. Particularly, Fig. 12 describes a scenario without rebalancing, while Fig. 13 describes a scenario where rebalancing is permitted.

The solution for the case without rebalancing, is illustrated in Fig. 12(b) where the orange circles denote rebalancing centers and their size reflects the expected demand at their location. The solution for the scenario with rebalancing is provided in Fig. 13(b). Consider the solution to the first scenario. We observe that the model provides three routes and that the rebalancing centers are not visited. The route for vehicle D1 (Route 1) starts from the current location of the vehicle, \( o(D1) \), and visits customer 1 (thus \( x_{D11}^{P1} = 1 \)) and customer 2 (\( x_{D12}^{P1} = 1 \)), in this order, before arriving at the station (\( x_{D13}^{P1} = 1 \)). In this solution, vehicles D4, D5 and D6 stay idle at their current position and wait for the next re-optimization phase.

Fig. 12(c) depicts a new snapshot of the system at the new re-optimization phase (say \( T = T_2 \)), and the corresponding solution. Customers 1 through 5 from the previous optimization phase are not in the system anymore because they have been by the time picked up. Similarly, vehicles D1, D2 and D3 are not in the system because they are currently on the way to the station and have no extra capacity (D1 and D2 have filled their two seats and D3 has the remaining seat available). New customers N1-N4 appear in the system. The solution suggests picking up only customer N1. The remaining customers are, in fact, too distant from the available vehicles.

---

Fig. 9. Proportion of vehicles dispatched to rebalancing centers.
The solution for the scenario with rebalancing (Fig. 13(b)) differs only in the fact that vehicles \(D_4\), \(D_5\), and \(D_6\) are dispatched to rebalancing centers. Therefore, at the next re-optimization phase (Fig. 13(c)), \(D_4\) and \(D_6\) are much closer to the new requests \(N_2\), \(N_3\), and \(N_4\), and can pick them up ensuring their requested arrival time is satisfied.

The example in Fig. 14(a) illustrates how capacity constraints work. The original number of customers on board are shown in green boxes next to the vehicle. Assume the capacity of each vehicle is \(Q = 4\). We observe that routes 1 and 3 are feasible with respect to the capacity constraints as in both cases the number of customers on board does not exceed \(Q\). Route 2, on the other hand, will be infeasible, since there are already 3 customers on board vehicle \(D_2\) at the beginning of the optimization phase, and the route assigns two additional customers to the vehicle, thus violating the capacity constraints.

Appendix C. Results for different numbers of rebalancing centers

See Tables 9–12.
Fig. 12. Example problem in a scenario where rebalancing is not permitted.
Fig. 13. Example problem in a scenario where rebalancing is permitted.
Fig. 14. Illustration of capacity and arrival time constraints.

Table 10
Service rate [%] in the simulations with $|\mathcal{R}| = 4$.

<table>
<thead>
<tr>
<th></th>
<th>C6V10</th>
<th>C6V12</th>
<th>C6V14</th>
<th>C7V10</th>
<th>C7V12</th>
<th>C7V14</th>
<th>C8V10</th>
<th>C8V12</th>
<th>C8V14</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUCt</td>
<td>wRKC</td>
<td>91%</td>
<td>98%</td>
<td>98%</td>
<td>95%</td>
<td>97%</td>
<td>97%</td>
<td>93%</td>
<td>96%</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>95%</td>
<td>94%</td>
<td>93%</td>
<td>91%</td>
<td>92%</td>
<td>91%</td>
<td>89%</td>
<td>93%</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>94%</td>
<td>94%</td>
<td>94%</td>
<td>89%</td>
<td>94%</td>
<td>91%</td>
<td>90%</td>
<td>90%</td>
</tr>
<tr>
<td>Uct</td>
<td>wRKC</td>
<td>97%</td>
<td>97%</td>
<td>96%</td>
<td>95%</td>
<td>96%</td>
<td>98%</td>
<td>94%</td>
<td>98%</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>92%</td>
<td>94%</td>
<td>95%</td>
<td>92%</td>
<td>93%</td>
<td>94%</td>
<td>92%</td>
<td>94%</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>91%</td>
<td>95%</td>
<td>94%</td>
<td>92%</td>
<td>95%</td>
<td>91%</td>
<td>96%</td>
<td>92%</td>
</tr>
<tr>
<td>NUCn</td>
<td>wRKC</td>
<td>79%</td>
<td>84%</td>
<td>86%</td>
<td>79%</td>
<td>84%</td>
<td>86%</td>
<td>77%</td>
<td>84%</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>59%</td>
<td>62%</td>
<td>77%</td>
<td>56%</td>
<td>63%</td>
<td>65%</td>
<td>62%</td>
<td>57%</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>63%</td>
<td>66%</td>
<td>72%</td>
<td>48%</td>
<td>60%</td>
<td>62%</td>
<td>47%</td>
<td>48%</td>
</tr>
<tr>
<td>Ucn</td>
<td>wRKC</td>
<td>86%</td>
<td>87%</td>
<td>87%</td>
<td>81%</td>
<td>88%</td>
<td>87%</td>
<td>82%</td>
<td>87%</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>67%</td>
<td>70%</td>
<td>72%</td>
<td>72%</td>
<td>72%</td>
<td>78%</td>
<td>64%</td>
<td>62%</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>66%</td>
<td>69%</td>
<td>78%</td>
<td>63%</td>
<td>76%</td>
<td>68%</td>
<td>66%</td>
<td>72%</td>
</tr>
</tbody>
</table>

Table 11
Profit [$] in the simulations with $|\mathcal{R}| = 5$.

<table>
<thead>
<tr>
<th></th>
<th>C6V10</th>
<th>C6V12</th>
<th>C6V14</th>
<th>C7V10</th>
<th>C7V12</th>
<th>C7V14</th>
<th>C8V10</th>
<th>C8V12</th>
<th>C8V14</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUCt</td>
<td>wRKC</td>
<td>55.59</td>
<td>58.84</td>
<td>57.78</td>
<td>68.43</td>
<td>67.63</td>
<td>70.83</td>
<td>76.62</td>
<td>81.41</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>56.72</td>
<td>55.37</td>
<td>53.75</td>
<td>64.77</td>
<td>68.90</td>
<td>66.49</td>
<td>73.98</td>
<td>79.11</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>56.37</td>
<td>56.61</td>
<td>56.42</td>
<td>63.96</td>
<td>67.08</td>
<td>65.72</td>
<td>74.71</td>
<td>77.52</td>
</tr>
<tr>
<td>Uct</td>
<td>wRKC</td>
<td>54.50</td>
<td>51.97</td>
<td>56.89</td>
<td>64.16</td>
<td>64.61</td>
<td>63.51</td>
<td>67.93</td>
<td>74.05</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>53.76</td>
<td>51.60</td>
<td>53.94</td>
<td>61.15</td>
<td>60.48</td>
<td>62.18</td>
<td>67.57</td>
<td>67.84</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>50.88</td>
<td>55.03</td>
<td>54.35</td>
<td>61.46</td>
<td>61.68</td>
<td>65.22</td>
<td>66.79</td>
<td>71.78</td>
</tr>
<tr>
<td>NUCn</td>
<td>wRKC</td>
<td>96.28</td>
<td>89.64</td>
<td>97.71</td>
<td>117.68</td>
<td>120.27</td>
<td>128.11</td>
<td>127.05</td>
<td>149.52</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>73.74</td>
<td>79.37</td>
<td>87.74</td>
<td>86.42</td>
<td>77.28</td>
<td>97.84</td>
<td>91.69</td>
<td>103.02</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>71.16</td>
<td>72.65</td>
<td>85.50</td>
<td>64.70</td>
<td>82.34</td>
<td>88.40</td>
<td>73.28</td>
<td>76.91</td>
</tr>
<tr>
<td>Ucn</td>
<td>wRKC</td>
<td>90.33</td>
<td>86.50</td>
<td>94.46</td>
<td>101.61</td>
<td>112.41</td>
<td>106.93</td>
<td>114.81</td>
<td>123.93</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>61.13</td>
<td>84.17</td>
<td>81.85</td>
<td>72.83</td>
<td>78.28</td>
<td>91.78</td>
<td>86.65</td>
<td>88.83</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>70.33</td>
<td>69.94</td>
<td>89.82</td>
<td>77.23</td>
<td>94.43</td>
<td>86.60</td>
<td>90.63</td>
<td>106.04</td>
</tr>
</tbody>
</table>

Table 12
Service rate [%] in the simulations with $|\mathcal{R}| = 5$.

<table>
<thead>
<tr>
<th></th>
<th>C6V10</th>
<th>C6V12</th>
<th>C6V14</th>
<th>C7V10</th>
<th>C7V12</th>
<th>C7V14</th>
<th>C8V10</th>
<th>C8V12</th>
<th>C8V14</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUCt</td>
<td>wRKC</td>
<td>93%</td>
<td>99%</td>
<td>96%</td>
<td>94%</td>
<td>95%</td>
<td>97%</td>
<td>89%</td>
<td>95%</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>93%</td>
<td>94%</td>
<td>91%</td>
<td>91%</td>
<td>94%</td>
<td>92%</td>
<td>90%</td>
<td>93%</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>94%</td>
<td>94%</td>
<td>94%</td>
<td>89%</td>
<td>94%</td>
<td>91%</td>
<td>90%</td>
<td>90%</td>
</tr>
<tr>
<td>Uct</td>
<td>wRKC</td>
<td>98%</td>
<td>94%</td>
<td>98%</td>
<td>95%</td>
<td>98%</td>
<td>95%</td>
<td>93%</td>
<td>99%</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>94%</td>
<td>94%</td>
<td>97%</td>
<td>95%</td>
<td>93%</td>
<td>94%</td>
<td>92%</td>
<td>90%</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>91%</td>
<td>95%</td>
<td>94%</td>
<td>92%</td>
<td>95%</td>
<td>91%</td>
<td>96%</td>
<td>92%</td>
</tr>
<tr>
<td>NUCn</td>
<td>wRKC</td>
<td>78%</td>
<td>78%</td>
<td>81%</td>
<td>81%</td>
<td>83%</td>
<td>86%</td>
<td>78%</td>
<td>86%</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>63%</td>
<td>68%</td>
<td>75%</td>
<td>61%</td>
<td>59%</td>
<td>70%</td>
<td>59%</td>
<td>63%</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>63%</td>
<td>66%</td>
<td>72%</td>
<td>48%</td>
<td>60%</td>
<td>62%</td>
<td>47%</td>
<td>48%</td>
</tr>
<tr>
<td>Ucn</td>
<td>wRKC</td>
<td>81%</td>
<td>75%</td>
<td>82%</td>
<td>81%</td>
<td>87%</td>
<td>83%</td>
<td>77%</td>
<td>82%</td>
</tr>
<tr>
<td></td>
<td>wRRS</td>
<td>61%</td>
<td>78%</td>
<td>76%</td>
<td>62%</td>
<td>66%</td>
<td>76%</td>
<td>64%</td>
<td>68%</td>
</tr>
<tr>
<td></td>
<td>woR</td>
<td>66%</td>
<td>69%</td>
<td>78%</td>
<td>63%</td>
<td>76%</td>
<td>68%</td>
<td>66%</td>
<td>72%</td>
</tr>
</tbody>
</table>


Toth, Paolo, Vigo, Daniele, 2002. The Vehicle Routing Problem. SIAM.


Xu, Zhe, Li, Zhixin, Guan, Qingwen, Zhang, Dinghsui, Li, Qiang, Nan, Junxiao, Liu, Chunyang, Bie, Ye, Jieping, 2018. Large-scale order dispatch in on-demand ride-hailing platforms: A learning and planning approach. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. KDD ’18, Association for Computing Machinery, New York, NY, USA, pp. 905–913.
