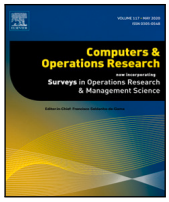
Computers & Operations Research 168 (2024) 106668

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor



Efficiently routing a fleet of autonomous vehicles in a real-time ride-sharing system

M. Bruglieri [a,*], R. Peruzzini [c], O. Pisacane [b]

[a] Dipartimento di Design, Politecnico di Milano, Italy
[b] Dipartimento di Ingegneria dell'Informazione, Università Politecnica delle Marche, Italy
[c] Independent Researcher

ARTICLE INFO

Keywords:
Vehicle Routing Problem
Pick up time decision
Mixed Integer Linear Programming formulation
Rolling horizon
Local Search

ABSTRACT

The advent of new communication technologies (e.g., smartphone) and autonomous vehicles (AVs) is enabling real-time ride-sharing systems where the travel requests arrives in the system on very short notice or even en-route, i.e., when AVs are already serving other users. Each request specifies an origin, a destination and a time window of pick-up, and must be immediately either accepted or rejected. Each request accepted must be assigned to an AV and both scheduled and inserted in its route considering the other possible requests already assigned to the same AV. In a lexicographic way, first we want to maximize the total number of new requests accepted, then we want to minimize the total traveled distance and finally, the total time of serving the requests. The problem is formulated as a Mixed Integer Linear Program and solved by a rolling horizon approach (MILP-RH). To efficiently address medium/large-sized instances, a rolling horizon Local Search (RHLS) is also designed, with moves properly tailored for the problem Numerical comparisons show that, on both the small-sized and some medium-sized instances, the RHLS outperforms the MILP-RH concerning the total computational time. Instead, on some medium-sized and on the large-sized instances, the RHLS is the only viable method since the MILP-RH is not able to even find a feasible solution in the given time limit. A sensitivity analysis on possible variation of some parameters is also performed deriving some useful managerial insights.

1. Introduction

In the last decades, in both Europe and US, the private car occupancy rates (i.e., the number of travelers per vehicle trip) have decreased. Indeed, according to the European Environment Agency (EEA), occupancy rates of passenger cars in Europe fell from 2.0–2.1 in the early 1970s to 1.5–1.6 in the early 1990s (European Environment Agency, 2020). Moreover, for urban trips, the occupancy rate is only 1.3, on average, and it also varies for travel purpose between 1.5, for household trips, and 1.1, for job trips. A similar trend also holds in US where in 2019 the average car occupancy rate is about 1.5 persons per vehicle (Center for Sustainable Systems, 2022). The decrease in the occupancy rate is mainly due to the increase in car ownership, extended use of cars for commuting and a continuous decrease in household size. Such a trend, in which the use of private cars for individual trips increases and, above all, the average car occupancy decreases, inevitably leads to an increase in traffic congestion, especially in urban contexts and, consequently in both environmental and noise pollution. To this purpose, Akimoto et al. (2022) demonstrate that both car and ride-sharing systems can certainly help in reducing the environment pollution.

In ride-sharing systems, travelers with similar itineraries and time schedules share a vehicle (Agatz et al., 2012). Therefore, such systems can combine both the flexibility and the speed provided by more common transportation modalities (e.g., the taxis) with the cheaper costs of the public transport. In addition, the use of such systems is also currently encouraged by the technological advances. In fact, through new communication technologies (e.g., smartphone and global positioning system), real-time ride-sharing systems (RRS) are becoming more and more widespread (Amey et al., 2011). On the contrary of the traditional ride-sharing systems, RRSs support an automatic ride-matching process among travelers on very short notice or even en-route. For example, the company *Carticipate* (Carticipate, 2008) proposed a dynamic ride-sharing on IPhone and it is currently developing a social network specifically for sharing real-time information on rides. However, while these dynamic services provide more flexibility for users, they pose new real-time decision-making problems that must be properly addressed.

* Corresponding author.
E-mail address: maurizio.bruglieri@polimi.it (M. Bruglieri).

https://doi.org/10.1016/j.cor.2024.106668
Received 2 July 2023; Received in revised form 11 April 2024; Accepted 13 April 2024
Available online 7 May 2024

Besides this, another significant technological advance reached in the automotive industry concerns the use of autonomous vehicles (AVs) that is becoming a reality (Faisal et al., 2019). Both Duarte and Ratti (2018) and Jones and Leibowicz (2019), for example, highlight that AVs have the potential to become a major catalyst for urban transformation and can contribute to reduce environmental pollution as well as the climate changes, being usually electric (then, without harmful emissions and less noisy). In addition, they can drive in a more efficient way, avoiding traffic congestion and charging in alignment with renewable electricity generation (Jones and Leibowicz, 2019). Moreover, AVs do not have a maximum travel time unlike the case in which drivers are employed. It is worth noting that, compared to carsharing systems, ride-sharing systems with AVs guarantee that the vehicle moves toward the users, not vice versa. Moreover, there is no need for a vehicle relocation since the AVs can directly move where they are necessary.

However, if on the one hand, the use of AVs for this type of service guarantees the significant advantage that no participant is required acting as a driver, on the other hand, a central system, acting as a dispatcher, is necessary to properly manage the dynamic assignment of the requests to the AVs, considering also the passengers already on board. In fact, the assignment has to respect the level of service to prevent that passengers on board make a long detour to serve new requests. Moreover, this dispatcher has to assign the requests in such a way that different objectives, e.g., the total number of passengers served and the total traveled distance, are simultaneously optimized.

The resulting optimization problem, i.e., the Real-time Ride-sharing Problem with AVs, Time Windows and Level Of Service Constraint (RRPAV-TW-LOSC), is introduced and addressed in this work. It is a very challenging problem belonging to the class of Vehicle Routing Problems. In particular, the constraint on the LOS imposes that the travel time of each user i must be not greater than L (user-defined threshold, greater than 1) times the shortest time to go from the origin of i, O_i, to her destination, D_i. We assume that a working day, divided into *time slot*, lasts about 16 h and thus, the vehicles can recharge overnight at a station which it returns to, at the end of the working day. In fact, the recent advances reached in the automotive industry guarantee higher drive ranges and then, in an urban context, an AV does not need recharging en-route. It is also assumed that at the beginning of a working day, an AV starts from the recharging station which it returned to the previous day. Each user specifies in her request i an origin O_i and a destination D_i as well as a time window $[e_i, l_i]$ within she has to be picked up. RRPAV-TW-LOSC consists in deciding for each new travel request i, arrived at time slot t, if either to reject or to serve it. In the latter case, it must be decided also which AV serves the request and when, considering the requests already assigned to the vehicle, without exceeding its capacity and LOS. In a lexicographic order, the total number of travel requests served is first maximized and then, the total traveled distance and the total time to serve the requests at their earliest convenience are minimized.

We assume that the rejection of new requests arrived must be immediately communicated to the users, i.e., within a given threshold of few seconds (e.g., 10 s). The exact time at which the user i is picked up is instead communicated later, i.e., within her earliest pick up time e_i minus 30 min. Indeed, with this degree of freedom we may obtain better solutions because the routes are not constrained to respect a fixed pick-up time for a accepted request until the time $e_i - 30$ is reached.

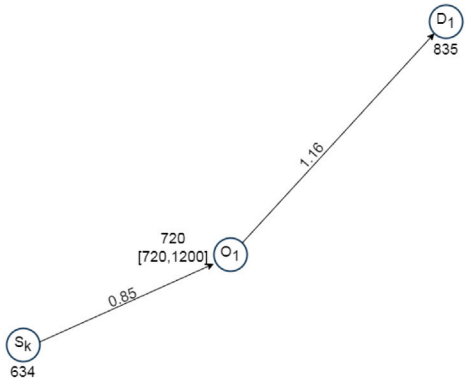
The real-time ride-sharing system considered in this work presents several differences with regard the classic one, also because we are assuming a fleet of AVs. First, the route of each AV does not end at the driver's destination (since no driver exists) implying that the routing problem to be solved is harder because the number of passengers that a vehicle may visit is generally by far higher being not limited to those whose destination is in a neighborhood of the driver's destination. Second, we are also separating the decision regarding the acceptance of a request from the decision regarding the time at which the pick-up occurs, which is communicated later. To the best of our knowledge, this aspect has never been considered in the literature and then represents another original contribution of our work.

The example depicted in Fig. 1 shows an instance with three requests arriving into the system at different time slots. Each node represents the real geographical location of a pick-up or a delivery request and the values in square brackets beside the pick-up nodes represent their time windows in seconds. Only the arcs traveled are shown and their length in km is indicated. We consider a constant vehicle speed of 40 km/h and a max value of LOS equal to 1.5. If the first request, with origin O_1 and destination D_1, arrives at time 0, it is scheduled and assigned to a vacant AV as in Fig. 1.a. If a second request, with origin O_2 and destination D_2, arrives at time 20 s, being its time window compatible with the previous request, it is served before the latter to minimize the total traveled distance (Fig. 1.b). When, at time 30 s, a third request arrives, with origin O_3 and destination D_3, it must be served as the first due to its time window $[130, 430]$. Indeed, after the pick-up of the corresponding customer, it is not possible to serve the second request (neither the first one) since the LOS of the third user would be violated being its earliest time 600 s and then, a waiting time of almost 170 s (i.e., $600 - 430$ minus the time to go from O_3 to O_2) would be necessary. Then, the route in Fig. 1.b is re-optimized as in Fig. 1.c.

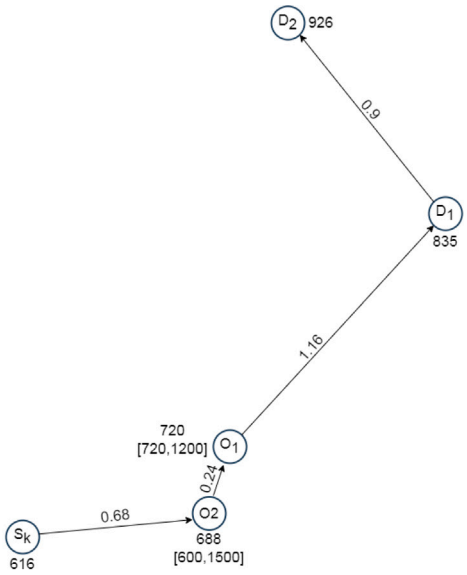
If we reconsider this example supposing that the vehicle starting from s_k is not an AV but a vehicle driven by a driver with destination in D_3, then the first two requests are not anymore compatible with this vehicle, but only the third one. Indeed, since the distance between s_k and D_3 is 1.4 Km, it is traveled in 126 s. Therefore, being the LOS 1.5, the maximum time the driver can be aboard (also to serve other users) is $126 \cdot 1.5 = 189$ s. Thus, the request of user 1 cannot be served by this driver since the total distance of the path s_k, O_1, D_1, D_3 is 2.51 Km (being the distance between D_1 and D_3 equal to 0.5 km) and then it requires 225.9 s. Similarly, the request of user 2 cannot be satisfied since the total distance of the path s_k, O_2, D_2, D_3 is 0.68+1.6+1.1=3.38 Km and then it requires 304.2 s.

The main original contributions of our work are detailed in the following:

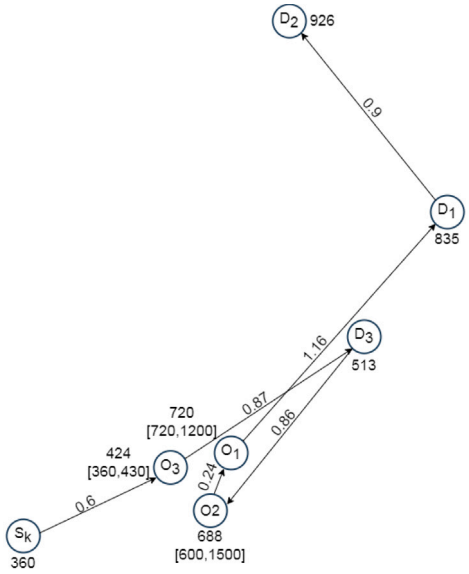
1. introducing a new challenging decision problem, i.e., the Real-time Ride-sharing Problem with AVs, Time Windows and Level Of Service Constraint (RRPAV-TW-LOSC) considering different objective functions in lexicographic way (first the number of users served and then, with the same priority, the total travel distance and the total serving time);
2. ensuring, through the LOSCs, that the travel time of each user is not greater than L (≥ 1) times the shortest time to reach his/her destination;
3. separating the decision to accept/reject a user request from the decision on when the user is picked up, being communicated later (30 min before the earliest time window of the user). Moreover, the solution approaches proposed, specifically exploit the degree of freedom provided by this assumption. The latter makes the RRPAV-TW-LOSC harder to be solved since the AV which the request is assigned to and the pick up time can be reconsidered along the optimization process being not fixed when the request is accepted;
4. designing a Mixed Integer Linear Programming based Rolling Horizon (MILP-RH) solution approach consisting in solving, at the beginning of each time slot, a MILP model in order to determine partial solutions with all the requests arrived up to the previous time slot;
5. developing a rolling horizon Local Search (RHLS) with tailored moves for efficiently addressing also medium/large-sized instances of the problem;
6. an extensive experimental campaign aimed not only to compare the performances of both MILP-RH and RHLS, but also to show, through a sensitivity analysis, the benefit of each original element, i.e., of LOSC (varying L), of separation of the acceptance



a. Only one request in the system



b. Two requests in the system



c. Three requests in the system

Fig. 1. An illustrative example.

decision from the pick up time decision, of time slot width and of lexicographic objective function.

The rest of the paper is organized as in the following. In Section 2, the main literature contributions are presented and discussed, emphasizing at the end the main differences with our work. In Section 3, the statement of the problem as well as the notation used in this paper are both introduced. Sections 4 and 5 are aimed to introduce, respectively, the MILP-based Rolling Horizon approach and the RHLS designed. The instances generated and the numerical results obtained are both described and discussed in Section 6. Finally, Section 7 draws some conclusions and provides some future research directions worthy of investigation.

2. Related work

RRPAV-TW-LOSC belongs to the more general class of Vehicle Routing Problems, VRPs, (Toth and Vigo, 2002). Indeed, it can be seen as a particular Dynamic VRP (Pillac et al., 2013), i.e., a Dynamic Pick-up and Delivery Problem (DPDP) in which some inputs may be unknown and revealed dynamically during the design/execution of the routes (Berbeglia et al., 2010). In this case, the routes are planned in an ongoing fashion, through also a real-time communication between vehicles and decision maker.

Traditionally, ride-sharing systems are divided into *unorganized* and *organized*. The former generally involve users who are from the same family, friends, neighbors etc. The latter instead involve a wider user base and then, pose more complex decision-making problems. In addition, in organized systems, a crucial aspect is how to assign requests to vehicles. Such an assignment can be done directly by the company providing the service or through agencies. In the latter case, there must be a matching between the drivers' route plan and the users' requests. The management of systems through ad-hoc agencies also pose economic issue to consider in order to properly incentivize both drivers and users.

Examples of systems in which passengers share a vehicle are *car-pooling systems*. But, in such systems, the passengers share the vehicle to go to the same destination (e.g., the office) (Teal, 1987, Spielberg and Shapiro, 2000 and Bruglieri et al., 2011).

Regarding the patterns, ride-sharing systems can be classified as proposed in (Furuhata et al., 2013): (1) *Identical ride-sharing*, in which driver and passenger share the same origin and the same destination; (2) *Inclusive ride-sharing*, in which both origin and destination of the passenger is different from those of the driver but on the way; (3) *Partial ride-sharing*, in which either the origin or the destination of the passengers is not on the way; (4) *Detour ride-sharing*, in which either the origin or the destination or both are not on the way.

In this work, we deal with a more general situation, in which also detour ride-sharing patterns are possible and we focus on dynamic ride-sharing systems. Moreover, for the purposes of this paper, we believe useful to propose the following classification of the literature contributions on dynamic ride-sharing systems: *category 1* refers to papers focused on dynamic ride/taxi-sharing dealing with also the problem of routing the drivers; *category 2* is related to papers focused on ride/taxi-sharing in which AVs are used.

Concerning *category 1*, Zhao et al. (2018) address both the problem of determining optimal matching and that of routing each driver. The resulting routing problem is solved by considering the location of flexible pickup and delivery points. In other words, the pickup and/or delivery point is re-positioned based on coordination between drivers and users. By introducing the concept of space–time windows, the problem is formulated as a classical PDP but with space–time windows on a space–time network. To efficiently and accurately solve it, the authors develop a Lagrangian relaxation based approach. The work of Zhang et al. (2018) addresses three, among the most common decision problems in a dynamic ride-sharing system. In particular, the problem of dynamically assigning riders to drivers is addressed as an online weighted matching problem considering a graph that changes over time. In order to define the prices to be charged to users, a sectional charging method is developed where each active passenger shares the cost of each section of distance proportional to their requested distances. A request is considered active between the time when it is accepted or assigned to a driver and the time when the rider is on the way to the own destination. Thus, the passengers' prices are mainly decided on the basis of the distances traveled. Finally, the problem of routing drivers is solved heuristically through an algorithm based on the nearest neighborhood search. Lotfi and Abdelghany (2022) assume that the travel time history on each link in the road network is known and that there is a set of passengers requiring a ride (not known in advance). Each passenger indicates both an origin and a destination as well as the earliest time to be picked up and the latest time to be dropped off. Moreover, they indicate their willingness to share the ride with others and/or have transfer along their route to receive discounts on their trip fare. The first objective to be maximized represents the total profit whereas, the second one to be maximized is the passengers' travel experience. The latter is achieved taking into account first of all solutions without transfer and then the rider is assigned to the vehicle with the minimum travel time. If this is not possible, the rider is assigned to two vehicles with the total minimum travel time with transfer. For this problem, a hybrid approach is designed, i.e., a myopic heuristic based on a first come first served policy and a greedy heuristic to re-optimize the requests already accepted. In Santos and Xavier (2015), both dynamic ride-sharing and taxi-sharing are modeled. In particular, the users decide to share either a car or a taxi, specifying the pickup and the delivery location, the earliest departure time, the latest arrival time and the maximum cost for the ride. Both the car owners and the taxi drivers specify their current location and a price per kilometer. In addition, the former also specifies the leaving time and the maximum accepted delay whereas, the latter report the time at which the service starts and the time at which the service ends. In this dynamic context, the authors solve the problem of matching users to vehicles and of determining the routes. In the case of the ride-sharing service, some constraints are also imposed, e.g., the maximum trip cost of each passenger and the maximum feasible delay. By dividing the working day into some time periods, they solve an instance of a static problem in each period through a greedy randomized adaptive search procedure. Although they introduce assumptions that are more general than ours and they also take into account money as incentive to use a specific service (ride or taxi sharing), it is worth noting that their solutions are fixed since once the user is assigned to a route from there he/she is no longer removed, without allowing a reoptimization. In addition, detours to serve new users are not allowed. Elting and Ehmke (2021) study request management policies to evaluate the economic potential of shared taxi services, taking into account also the LOS and implementing a rolling horizon approach. Unlike our work, once a new request arrives, it is immediately notified the time at which the user is picked up. Moreover, the authors assume that the service provider collects requests from users dispersed around a limited geographical area. The heterogeneity of the requests assumed in our work, instead, makes the problem of assigning users to AVs harder. Finally, on the contrary of our work, they optimize only the number of users served. Haferkamp and Ehmke (2022) investigate the effectiveness of demand and fulfillment control in dynamic fleet management of ride-sharing systems. In particular, unlike our paper, they assume that, as soon as a request arrives, the user has to be ready to be picked up at his/her origin without managing pre-booking requests (i.e. requests made a long time in advance). Moreover, they do not take into account the vehicle capacity. This assumption is not very realistic and holds only under specific instance conditions, i.e., when there are not too many concentrated requests. Finally, their objective function (to maximize) represents only the number of requests satisfied. Very recently, Zheng and Pantuso (2023) have addressed the first-mile ride-sharing problem, minimizing the total cost due to the transport of the users and maximizing the service rate, i.e., the number of users served over the total requests. However, unlike our work, they assume that all users share the same destination (i.e., the transit station) and therefore, that each route terminates at the transit station. In addition, each user has an origin and a preferred arrival time at the station. For this problem, they have designed an evolutionary algorithm based on efficient non-dominated solution sorting.

With regard to *category 2*, Fagnant and Kockelman (2018) describes an agent and network based AV simulations by enabling dynamic ride-sharing, optimizing fleet sizing. In order to model dynamic services, they propose a method for matching passengers to AVs based on some rules. Among these rules, the most used one is that according to which each passenger must be served by the nearest AV. Lokhandwala and Cai (2018) propose an agent-based simulation approach for quantifying both the environmental and the energy benefits of a ride-sharing system with AVs and traditional taxis. In particular, they consider the preference of each user to share a vehicle as well as the maximum delay accepted. Moreover, they assume to know the fleet size as well as the percentage of users that is willing to share vehicles. The simulation model proposed returns the number of users served as well as the times at which the status of each user changes. Mao et al. (2020) focus on designing reinforcement learning strategies to solve a dynamic decision problem in taxi sharing systems with AVs. They divide the reference region into zones and time into time slots. Their goal is to determine, for each time period, the number of AVs to send from a zone to another such that the total expected operational cost (i.e., the cost of repositioning AVs and the one due to users that are still waiting for the service) is optimized. In Turan et al. (2020), the routing, the battery charging and the pricing are addressed for maximizing the total profit. The authors deal with the static planning problem and determine the optimal static policy. Whereas, the stochastic nature of the problem due to the trip demands, renewable energy availability, and electricity prices is addressed through a real-time policy that requires the development of a deep reinforcement learning based approach.

The authors demonstrate the effectiveness of the proposed approach using two case studies in San Francisco and Manhattan. Pouls et al. (2021) address the real-time dispatching requests in a dynamic ride-sharing system, especially from the methodological point of view by designing a local search based heuristic. Unlike our work, they do not have a period during which the accepted request may wait for being notified regarding the pickup time. In particular, when a new request arrives, they try to assign it, in a feasible way, to a vehicle (even an idle one). If it is not possible, it is rejected and then, the user is not served. In addition, they assume that at most 2 users per car may share the same trip. Therefore, the feasible solutions in their problem are by far less than those in ours. Moreover, unlike our work, they also assume that, when a new request arrives into the system, only the vehicle nearest to it may provide the service. Finally, the authors propose a hierarchical objective function in which the number of users served is firstly maximized and then, the total travel time of the vehicle fleet is minimized without considering also the minimization of the total time for serving the requests (i.e., also possible waiting times) like us. The work of Noruzoliaee and Zou (2022) faces with the problem of assigning the users to AVs in a network equilibrium setting with mixed AV and human-driven vehicle traffic. First, the authors investigate the assignment of one AV to many users in order to analyze the waiting times of both, i.e., when either the origin of the user is the same of the AV or it is on the way. Then, the authors introduce a section-based formulation allowing that an AV traveler's itinerary (i.e., origin destination pair) is different from that of the serving AV and other riders in the vehicle. Then, the concept of *section* is introduced and considered in order to both avoid undesired rider en-route transfer(s) and allow users of multiple origin/destinations to share the AV. Beirigo et al. (2022) study the dynamics of a ride-sharing system with a fleet of AVs where different types of users coexist, i.e., users distinguished according to service quality expectations in terms of responsiveness, reliability, and privacy. In order to support the ride-sharing company to meet the service quality expectation of the users, the authors assume possible that privately-owned freelance AVs can be hired on short notice. The problem is then formulated through multi-objectives MILP, optimizing simultaneously the vehicle occupancy, the number of AVs used, the service level violations and the waiting times. The dynamic version of the problem is then solved through a meta-heuristic approach. Very recently, Bongiovanni et al. (2022) propose a two-phase matheuristic in which, in the first phase, through a greedy heuristic, the new requests are assigned to the AVs. In the second stage, instead, through a local search based matheuristic, the previous AV-trip assignments are iteratively reoptimized through intra and inter-vehicle route moves. A machine learning approach, trained offline on a large set of instances, is implemented and applied for selecting a pool of destroy-repair moves.

Our problem also falls into the category of door-to-door (also referred as demand-responsive bus system or dial-a-ride) dynamic online system where a fleet of public vehicles is used to dynamically serve a set of trip requests that can come during the planning horizon as last minute bookings or also be booked long time in advance. According to the recent survey (Vansteenwegen et al., 2022), in this context, most papers only minimize either the total passenger travel time or the total vehicle travel time. Only two papers optimize at the same time the total number of requests accepted, the total passenger travel time and the total vehicle travel time. They consider a fleet of AVs and then, they also fall into *category 2*. The first one is van Engelen et al. (2018) where in their problem definition no LOS constraint is considered and an insertion algorithm based on demand forecasts for performing rerouting is proposed. The second one is Hyland and Mahmassani (2020) where a maximum user detour distance constraint is introduced, but no time window is associated with users and then, it is assumed that they can indefinitely wait for the arrival of a vehicle. This assumption makes the problem less realistic than the one we address since for instance it does not allow the request pre-booking (i.e., making a request also long time in advance as in our work). Moreover, the maximum user detour distance constraint is weaker than our LOS constraint since the former does not prevent users to be on board by far longer than the minimum time to directly reach their destinations. The problem is modeled using an agent-based stochastic dynamic simulation framework and a re-optimization-based heuristic solution approach is proposed. Moreover, in both the papers, there is no gap between when the request is accepted and when it is scheduled, unlike us. Vice versa, this aspect is considered in Melis and Sorensen (2022), but they only optimize the total waiting time and the total user travel time since all the requests are considered mandatory. Moreover, no LOS constraint is considered. The problem is solved by applying on all the known requests a Large Neighborhood Search already developed for the static version of the same problem. Afterwards, the dynamic requests are added one by one and the heuristic executes a real-time optimization

To the best of our knowledge, our work is the first contribution belonging to *category 2* in which several aspects are considered together, guaranteeing a higher degree of flexibility. Indeed, we separate the moment at which the user's request is accepted from that at which the time of pick-up is notified. Thanks to this, we allow changing the AV that serves a user until the moment in which the user is onboard, thus allowing better solutions. Moreover, we optimize in lexicographic way first the number of users served and then, the total traveled distance and the total time for serving the requests. This is a novelty, compared to the objective functions considered in the related literature, since as shown above, most papers maximize the total revenue associated with the served requests. Instead, we maximize the total number of requests served, regardless of its revenue, because if a request is rejected, the related user will probably not use this service operator (but another one), in the future. Therefore, this objective allows modeling the user loyalty (i.e., a non-rejected user will most likely continue to use the same service also in the future). For the same reason, the second objective (minimization of the total traveled distance) is considered with lower priority in our model, on the contrary of the literature where it is typically monetized and subtracted to the total revenue of the requests served. Finally, the third objective (total time for serving the requests) is a further original component of our objective function aimed at serving the users as soon as possible compatible with their time windows. This has been introduced for two main reasons. The first one is obviously the user satisfaction. The second one is related to the dynamic arrival of the requests to the system, and then the sooner we can satisfy current requests, the more likely the vehicles will be free to satisfy future requests. A further discussion on the objective function is reported in Section 6.5 in which we show the advantages of minimizing a hierarchical objective function instead of minimizing only one component or two components at time.

3. Statement of the problem and notation

We assume to have the following three categories of users:

- *Users ACC*: whose request is already accepted by the central system, they are already assigned to an AV and they also know the time for the pick-up;
- *Users TEMP*: whose request is already accepted by the central system, they are temporary assigned to an AV but they do not know the time of the pick-up;
- *Users NOT*: whose request is not accepted yet.

We assume to discretize time in time slots of fixed duration. For each request i, p_i denotes its service time and ω_{o_i} is the time which i with origin o_i has been generated at. If a request is generated during a time slot, we assume that it arrives at the beginning of the next time slot. Namely, if for example each time slot lasts 5 minutes and a request is generated at the third min of time slot 10 then it is assumed the request is generated at the beginning of time slot 11. Moreover, for each request i with origin o_i, the two parameters, e_{o_i} and l_{o_i}, denote the time

window. In particular, e_{o_i} and l_{o_i} are, respectively, the earliest and the latest time for picking up i at o_i. The time at which the AV assigned to i arrives at its destination d_i is taken under control by imposing the LOS constraint.

The set K denotes the available AVs, each one with a number of seats (i.e., vehicle capacity) equal to C. Vehicle speed is assumed constant and equal to v. In order to properly manage cases in which the central system only later realizes that it is not convenient to serve some users NOT, a *virtual pool* is also introduced. This *virtual pool* represents a fictitious AV with an unlimited number of available seats.

The planning horizon T contains at most t_{max} periods during which n requests arrive.

At each period t, I'_t denotes the set of the requests of users ACC whereas $\overline{I'_t} \subseteq I'_t$ is the subset of these requests already on board and finally, I''_t and I'''_t are the set of the requests of users TEMP and of users NOT, respectively. Therefore, an instance of RRPAV-TW-LOSC at a given period t is represented on a direct weighted graph $G = (V_t, A_t)$ where V_t is the set of nodes at period t whereas A_t denotes the set of arcs between each pair of nodes associated with the requests at period t. The set of nodes V_t contains the set of origins O'_t, O''_t and O'''_t of all the requests at period t of users ACC, TEMP and NOT, respectively. It also contains the set of destinations D'_t, D''_t and D'''_t of all the requests at period t of users ACC, TEMP and NOT, respectively. We also introduce the set $V'_t \subset V_t = O'_t \cup D'_t$, $V''_t \subset V_t = O''_t \cup D''_t$, $V'''_t \subset V_t = O'''_t \cup D'''_t$ and $O_t \subset V_t = O'_t \cup O''_t \cup O'''_t$, $D_t \subset V_t = D'_t \cup D''_t \cup D'''_t$ and finally, $I_t \subset V_t = I'_t \cup I''_t \cup I'''_t$. Among all the AVs, $K^0_t \subset K$ denotes AVs never used until period t whereas $K'_t \subset K$ represents AVs serving customers already on board. An AV assigned to a customer $i \in \bar{I}_t$ already on board is indicated as k_i. We indicate by s_k the starting node of vehicle k. We indicate by $\tilde{V}_t$ the set of the nodes including also the starting node of all the vehicles, i.e., $\tilde{V}_t = V_t \bigcup_{k \in K \setminus K'_t} s_k$. Whereas, u^{start}_k represents the number of seats already occupied in the vehicle k at its starting node s_k. The time starting from which the vehicle k is available at its starting node s_k is denoted by τ^{start}_k.

A pre-processing procedure removes all the arcs (d_i, o_i), i.e., those linking the destination and the origin of a request i and also those that violate the time windows. Moreover, $\bar{A}_t$ contains all the *frozen arcs*, i.e., arcs already traveled by users on board (i.e., users in $\bar{I}_t$) at period t. In other words, a *frozen arc* is not completely traveled at that period but its tail is already traveled. Therefore, the set of all the arcs can be defined as $A_t = (\tilde{V}_t \times V_t) \setminus \{\{(d_i, o_i) : i \in I_t\} \cup \overline{A}_t\}$.

In addition, for each node i, $\delta^+(i)$ and $\delta^-(i)$ denote its forward and backward star, respectively. For each pair of nodes, (i, j), d_{ij} denotes their distance. Finally, L and Γ denote, respectively, the maximum LOS value and the upper bound on the maximum traveled distance. In particular, L represents the maximum tolerated ratio between the time required to serve a user and the minimum time to reach her destination by her own vehicle. Indeed, L is related to the travel time rather than to the traveled distance in order to take into account possible waiting times due to the respect of the time windows. In the case in which an AV has to wait at a node before serving the user due to her time window, it is more convenient that the AV waits at the previous node in order to possibly serve other users in the meantime. Therefore, in the following, it is assumed that the period at which the time of picking up a user with request i and origin o_i is calculated with respect to e_{o_i}. Tables 1 and 2 summarize the sets and the parameters used, respectively.

Table 1
Sets.

Name	Meaning
I'_t	Requests ACC at period t
$\overline{I'_t}$	Requests ACC aboard at period t
I''_t	Requests TEMP at period t
I'''_t	Requests NOT at period t
O'_t	Origins of requests ACC at period t
O''_t	Origins of requests TEMP at period t
O'''_t	Origins of requests NOT at period t
D'_t	Destinations of requests ACC at period t
D''_t	Destinations of requests TEMP at period t
D'''_t	Destinations of requests NOT at period t
V'_t	Origins and destinations of requests ACC at period t
V''_t	Origins and destinations of requests TEMP at period t
V'''_t	Origins and destinations of requests NOT at period t
V_t	Origins and destinations of all types of requests at period t
$\tilde{V}_t$	Set of all nodes including also the starting node of every vehicle
O_t	Origins of all types of requests at period t
D_t	Destinations of all types of requests at period t
I_t	Requests of all types of requests at period t
A_t	Arcs
$\overline{A}_t$	Frozen arcs
K	Available AVs
K^0_t	AVs never used until period t
K'_t	AVs serving customers in $\overline{I'_t}$

Table 2
Parameters.

Name	Meaning
o_i	Origin of user i
d_i	Destination of user i
p_i	Service time at user i
e_{o_i}	Earliest time for picking up user i at her origin o_i;
l_{o_i}	Latest time for picking up user i at her origin o_i;
ω_{o_i}	Time at which the request of user i with origin o_i is generated
t_{max}	Maximum time slot
T	Planning horizon
n	Number of requests
s_k	starting node of AV $k \in K$
k_i	AV serving user $i \in \overline{I'_t}$
C	AV capacity
u^{start}_k	Occupancy of AV k at its starting node s_k
τ^{start}_k	Period from which AV k is available at s_k
$\delta^+(j)$	Forward star of node j
$\delta^-(j)$	Backward star of node j
d_{ij}	Distance between nodes i and j
v	AV speed
L	Max LOS value
Γ	Upper bound on the maximum traveled distance

Table 3
Decision variables.

Name	Meaning
x^k_{ij}	1 if $(i,j) \in A_t$ is traveled by vehicle k
f_{ik}	1 if $i \in V_t$ is the last node served by vehicle k
τ_i	Arrival time at node $i \in V_t$
u_i	Occupancy of the vehicle at the node $i \in V_t$

4. A MILP-based rolling horizon approach

In this section, we propose a solution approach that consists in solving a MILP formulation at each period $t \in T$, in order to find partial feasible routes, i.e., a MILP-based Rolling Horizon approach (MILP-RH). Indeed, MILP-RH decides which users in I'''_t must be accepted or rejected and both the routing and scheduling of the users in I''_t . RRPAV-TW-LOSC is formulated by introducing the following decision variables: x^k_{ij} equal to 1 if arc $(i,j) \in A_t$ is traveled by vehicle k, 0 otherwise; f_{ik} equal to 1 if $i \in V_t$ is the last node served by vehicle k, 0 otherwise; z_i equal to 1 if the vehicle occupancy after visiting node $i \in V_t$ is ≥ 1, it is 0 otherwise; τ_i arrival time at node $i \in V_t$ also including possible waiting time to satisfy the time window of node i when i is the origin of a request; and finally, u_i represents the vehicle occupancy at the node $i \in V_t$. The decision variables of the MILP model are summarized in Table 3. Moreover, since for each user $i \in I'_t$, the routing of the serving vehicle has already been determined, the values of τ_i $\forall i \in O'_t \cup D'_t$ are given in input.

The MILP formulation of the problem, at time slot t, is the following:

$$\max \sum_{k\in K}\sum_{(i,j)\in A_t : i\in O_t'''} x_{ij}^k - \frac{1}{\Gamma}\sum_{k\in K}\sum_{(i,j)\in A_t} d_{ij}x_{ij}^k - \frac{1}{2t_{max}|I_t \setminus \overline{I}_t'|}\left(\sum_{i\in I_t\setminus\overline{I}_t'}\tau_{o_i} + \sum_{i\in I_t\setminus\overline{I}_t'}\tau_{d_i}\right) \quad (1)$$

$$\sum_{j\in\delta^-(o_i)} x_{jo_i}^k = \sum_{j\in\delta^-(d_i)} x_{jd_i}^k \quad \forall i \in I_t \setminus \overline{I}_t', \forall k \in K \quad (2)$$

$$\sum_{j\in\delta^-(d_i)} x_{jd_i}^{k_i} = 1 \quad \forall i \in \overline{I}_t' \quad (3)$$

$$\sum_{j\in\delta^-(i)} x_{ji}^k - \sum_{j\in\delta^+(i)} x_{ij}^k = f_{ik} \quad \forall i \in V_t, \forall k \in K \quad (4)$$

$$\sum_{j\in\delta^-(s_k)} x_{js_k}^k - \sum_{j\in\delta^+(s_k)} x_{s_kj}^k = -1 \quad \forall k \in K \quad (5)$$

$$\sum_{i\in V_t} f_{ik} \le 1 \quad \forall k \in K \setminus K_t^0 \quad (6)$$

$$\sum_{k\in K}\sum_{j\in\delta^+(i)} x_{ij}^k = 1 \quad \forall i \in O_t \setminus O_t''' \quad (7)$$

$$\sum_{k\in K}\sum_{j\in\delta^+(i)} x_{ij}^k \le 1 \quad \forall i \in O_t''' \quad (8)$$

$$\sum_{j\in\delta^+(s_k)} x_{s_kj}^k \ge \sum_{j\in\delta^+(i)} x_{ij}^k \quad \forall i \in V_t, \forall k \in K_t^0 \quad (9)$$

$$\tau_j \ge \tau_i + p_i + \frac{d_{ij}}{v} - (t_{max} + p_i + \frac{d_{ij}}{v})(1 - x_{ij}^k) \quad \forall (i,j) \in A_t, \forall k \in K \quad (10)$$

$$\tau_j \ge \omega_j + \frac{d_{ij}}{v} - (t_{max} + \frac{d_{ij}}{v})(1 - x_{ij}^k) \quad \forall (i,j) \in A_t : j \in O_t, \forall k \in K \quad (11)$$

$$\tau_{s_k} \ge \tau_k^{start} \quad \forall k \in K \quad (12)$$

$$\tau_j \ge t \quad \forall j \in V_t'' \cup V_t''' \quad (13)$$

$$\tau_{d_i} - \tau_{o_i} \le L\frac{d_{o_id_i}}{v} \quad \forall i \in I_t \quad (14)$$

$$\tau_{d_i} - \tau_{o_i} \ge \frac{d_{o_id_i}}{v} \quad \forall i \in I_t \quad (15)$$

$$e_{o_i} \le \tau_{o_i} \le l_{o_i} \quad \forall i \in I_t'' \cup I_t''' \quad (16)$$

$$u_{s_k} = u_k^{start} \quad \forall k \in K \quad (17)$$

$$u_j \le u_i - 1 + C(1 - \sum_{k\in K} x_{ij}^k) \quad \forall (i,j) \in A_t : j \in D_t \quad (18)$$

$$u_j \ge u_i - 1 - C(1 - \sum_{k\in K} x_{ij}^k) \quad \forall (i,j) \in A_t : j \in D_t \quad (19)$$

$$u_j \le u_i + 1 + C(1 - \sum_{k\in K} x_{ij}^k) \quad \forall (i,j) \in A_t : j \in O_t \quad (20)$$

$$u_j \ge u_i + 1 - C(1 - \sum_{k\in K} x_{ij}^k) \quad \forall (i,j) \in A_t : j \in O_t \quad (21)$$

$$u_i \le C \quad \forall i \in V_t \quad (22)$$

$$x_{ij}^k \in \{0,1\} \quad \forall (i,j) \in A_t, \forall k \in K \quad (23)$$

$$\tau_i \ge 0, u_i \ge 0 \quad \forall i \in \tilde{V}_t \quad (24)$$

The objective function (1) models the lexicographic optimization of the total number of new requests to be maximized and then, with lower importance, the minimization of both the total traveled distance and the total time of serving the requests. In order to obtain this, we consider the normalization factor of the second term, Γ, computed as

$$\Gamma = D|I_t \setminus \overline{I}_t| + (2|I_t| - 1)max_{(i,j)\in A_t} d_{ij} \quad (25)$$

where $D = max_{k\in K\setminus K_t', i\in I_t\setminus\overline{I}_t} d_{s_ko_i}$. It is worth noting that the third component is added due to the policy assumed in this work, i.e., we communicate to a user the time at which she is picked up half an hour before the earliest time of her time window. Therefore, due to the fact that the problem addressed is online, it is more advantageous to handle the requests at their earliest convenience in order to have a better chance of serving more.

Constraints (2) ensure that the origin and the destination of each request are both served by the same AV. Constraints (3) ensure that, if the user i is on the AV k_i, k_i must also reach d_i. It is worth noting that k_i is known since i belongs to those users already on board. Constraints (4) guarantee that, if i is the final node of a route (i.e., $f_{ik} = 1$), one unit of flow must enter i. Otherwise, the total flow through i must be equal to 0. In similar way, Constraints (5) guarantee that for each starting node s_k one unit of flow must exit from it. Constraints (6) impose that, for each AV, at most one final node can exist. Such final node does not exist only in the case in which an AV is not used. The origin of each accepted request must be visited (7). Each user NOT must be served by at most one AV (8). If an AV k is used to serve a request, its starting node s_k must be used (9).

Constraints (10) ensure that the arrival time at node j is given by the arrival time at the previous visited node i increased by both the service time at i and the corresponding travel time from i to j.

Constraints (11) ensure that the arrival time to the origin j of a request cannot be lower than the time at which the user asked for the request, increased by the time necessary to the AV to go from its previous visited node i to j. Therefore, each of these constraints implicitly imposes that the arrival time at node j cannot be lower than the maximum between τ_i and ω_j.

The arrival time at each starting node is initialized by (12). Whereas, constraints (13) guarantee that the arrival time at each node cannot be lower than the time slot t considered in the MILP model.

Constraints (14) ensure that, for each user, the total travel time, including also possible waiting times, cannot be greater than L time the minimum travel necessary to go directly from her origin to her destination (i.e., *MaxLOS conditions*). It is worth noting that if the user is already on board (i.e., $i \in I_t'$), τ_{o_i} is known and not a variable since it has been already fixed by the previous optimization phase. In such a case, the MILP model has to decide the arrival time at her destination, i.e., τ_{d_i}.

For each request, the destination is visited after the origin (15). Instead, constraints (16) force the time window satisfaction for the departure time from the origin of each request.

The starting occupancy of each used AV is set by (17). Instead, constraints (18)–(19) update the AV occupancy, decreasing it by one unit, if it visits a destination node. Vice versa, constraints (20)–(21) increase it by one unit if it visits an origin node. It is worth noting that, according to the definition of the set A_t, there are no arcs entering origins of users already on board. The vehicle capacity must be never exceeded (22). Finally, constraints (23)–(24) model the nature of the variables.

At each $t \in T$, after solving the related MILP model, for each AV k used, the corresponding starting node s_k is updated setting $s_k = i$, where i is the node with maximum value of $\tau_i \le t$, in the route of AV k, since such a node represents the last node currently visited by k.

It is worth noting that, if we consider a shared fleet of traditional vehicles instead of AVs, the proposed mathematical model does not work since further constraints should be added to ensure that when a vehicle moves to a user there is always a driver on board or vice versa, when no driver is on board of it, a user acting as a driver moves to it.

5. A rolling horizon local search

This section aims at describing the rolling horizon Local Search (RHLS) designed for efficiently addressing also medium-sized and large-sized instances of RRPAV-TW-LOSC. The RHLS is summarized in Algorithm 1, where t^{step} denotes the time slot duration (in our tests $t^{step} = 10$ s) and the other parameters have been already defined in Section 4.

Starting from an initial solution, RHLS dynamically improves it through a *Local Search* procedure. Therefore, the main steps are the initialization phase and the local search.

The initialization phase to solve our problem consists of the **initSol** procedure which acts as follows. At each time slot, each travel request i is processed according to its arrival time in the system (*First In First Out* policy). For each AV, the best way to insert the origin and destination of i in the current route of the AV, i.e., at minimum detour among all the possible insertions that respect both the time windows and LOS, is computed in exhaustive way. The request i is then assigned to the AV with minimum detour. It is not guaranteed that all the requests can be served. Indeed, some of them may not be assigned to any AV due to their time windows or the violation of LOS. To give these requests the possibility to be served through the Local Search (LS), they are put into a virtual pool (VP), that represents a fictitious AV with an unlimited number of available seats. This way, some of the requests in the VP may be served moving possibly also some users NOT in the VP.

To manage this, together with the three types of users already described, in the RHLS, we also introduce a fourth type of users, i.e., *dummy users*, representing the empty seats on the AVs. This way, in the RHLS, the users ACC on board cannot be moved from an AV to another whereas the users both ACC not on board and TEMP can be moved only among the AVs of the real fleet. Finally, the users NOT can be moved among the AVs belonging to the real fleet united with the VP, in order to allow changing the choice of the accepted requests until the end of the time slot.

The LS phase consists in three procedures: **reOpt**, **allocateVP** and **swap**. They are all applied after the procedure **initSol** only when new requests arrive (i.e., **verifyNewRequest** returns *true*). Otherwise, only the **swap** procedure is called. Every procedure immediately stops when the time slot ends, i.e., when the $Time()$ function, representing the elapsed time, reaches the value $t + t^{step}$ representing the ending time of the current time slot. The **reOpt** procedure implements an intra-route local search. More specifically, for each route, it considers in exhaustive way all the feasible permutations of the nodes to minimize the total traveled distance. In order to reduce the computational time required by this procedure, it is invoked only in the case a predefined number of nodes is changed (in our tests, equal to 5) and considering only at most a predefined number of next nodes (in our tests, equal to 9). The **allocateVP** procedure tries to assign each request i of the VP, considered by increasing value of the latest TW, to an AV. It is tested in two different ways. First of all, moving i to a route whose total traveled distance has been reduced by **reOpt**, if the insertion at minimum detour is now feasible. Otherwise, considering all the detours, $d_{r'}$, obtained when the request i is inserted in the route r' in feasible way with respect to all the previous served users, although r' becomes unfeasible concerning the next ones. Then, the first unfeasible request of r' is inserted into another route r'' at minimum detour, $d_{r''}$, testing all feasible insertions, if any, otherwise $d_{r''} = \infty$. Then, the move is applied for the combination of routes r', r'' with minimum value of $d_{r'} + d_{r''}$, if it is $< \infty$. When the **allocateVP** procedure ends, all the possible requests still in the VP are rejected and the VP is emptied.

The **swap** move swaps each pair of requests (i, i') either ACC, not on board, or TEMP, belonging to different routes and inserts them at the minimum detour. This move aims at minimizing the total traveled distance and it is applied to the list of requests already accepted but not served yet, regardless of the AV. It is implemented also considering the case in which we simply move a request from a route to another one without the latter providing one of its served requests to the former. Such a move includes a memory mechanism to make it more efficient. Indeed, it keeps track of the last request which the swap move was applied to in the previous time slot and then it is applied from the next request. The global stopping criterion consists of reaching the end of the planning horizon T.

Algorithm 1 Outline of the proposed RHLS

```
Input: T, t^step, num_veic, K, I_t, s_k ∀k ∈ K
Output: best solution found sol^best
Solution sol^best := ∅;
t := 0;
while t ≤ T do
  if verifyNewReq() then
    sol^best :=initSol(sol^best, I_t);
    sol^best:=reOpt(Time(), sol^best);
    sol^best:=allocateVP(Time(), sol^best);
  end if
  sol^best:=swap(Time(), sol^best);
  t := t + t^step;
end while
```

Fig. 2 refers to a small instance with three requests and shows how the heuristic solution built with the **initSol** procedure can be improved invoking the **reOpt** move. Each node represents the real geographical location of a pick-up or a delivery request and the values in square brackets beside the pick-up nodes represent their time windows in seconds. Only the arcs traveled are shown and their length in km is indicated. We consider a constant vehicle speed of 40 km/h, a fixed time of 10 s for each request of pickup and of delivery, and a max value of LOS equal to 2. In particular, at time 1200, the first request, with origin O_1 and destination D_1, arrives and it is accepted according to the schedule shown in Fig. 2.a. At time 2400, the second request arrives and it is served in optimal way as shown in Fig. 2.b. Indeed, for example the sequence s_k, O_1, O_2, D_2, D_1 has total length 17.44 km vs 17.12 km of the current solution. At time 3000, the third request arrives and according to the **initSol** procedure, the best way to insert O_3, D_3 in the current route (i.e., at minimum detour) without changing the ordering of the nodes already served is as shown in Fig. 2.c. Being in the system a sufficient number of nodes (both origins and destinations), the intra-vehicle move **reOpt** is called, finding a shorter route as reported in Fig. 2.d. Indeed, this new solution has a total length of 26.39 km against 28.10 km of that illustrated in Fig. 2.c. We remark that this solution cannot be found by the **initSol** procedure since it requires to change the visit sequence of the nodes already inserted before (O_3, D_3). We also remark that if the announcement time of the second request would be delayed, for instance at time 5000, and that of the third request at time 5500, then since the difference between the earliest pickup time of the second request and the announcement time is lower than 30 minutes (being $6600 - 5000 = 1600$ s), the computed pick-up time (6985) must be frozen and immediately communicated to the user. Thus, when we apply the **reOpt** move after the arrive of the third request and we evaluate the travel time of the route depicted in Fig. 2.d, it becomes 1374.2 s (including the service time for pick-up and delivery) since the pick-up time of O_2 must be 6985 instead of 6769 s. Thus, this route becomes unfeasible since it exceeds the LOS constraint of the first user being the time of the direct route $O_1 - D_1$ equal to 661.7 s.

Figs. 3.a–3.c show how the **swap** move works. In particular, in the following, we are assuming that the fleet contains 2 AVs: $AV1$ starting from S_1 and $AV2$ starting from S_2. At time 21980, two requests are in the system and they can be served by $AV1$, in the sequence S_1, O_1, O_2, D_2, D_1 (see Fig. 3.a). Such a sequence takes into account the users' time windows, according to which, the request 1 is picked up first (being its time window $[23961, 25160]$) and then the request 2 (with time window $[25083, 26282]$). Both are then delivered at their destinations. At time 21990, the request 3 enters into the system and is then assigned to $AV2$ (Fig. 3.b). In fact, although it would have been more convenient to assign the request 3 to $AV1$, the LOS of the request 1 would have been violated otherwise. After this assignment,

the proposed heuristic invokes the **swap** move, in order to further optimize the solution. In particular, the **swap** procedure verifies that moving the request 1 from $AV1$ to $AV2$ leads to a better solution. In fact, being all the three requests served, the solution shown in Fig. 3.b has a total travel distance of 52.26 km against the 50.42 km of the solution in Fig. 3.c. It is also worth noting that the total time of serving the requests (that we are minimizing) of the solution in Fig. 3.c, obtained after the **swap** move, is less than that of the solution in Fig. 3.b, being 150,210 s and 150,474 s, respectively.

Figs. 4.a–4.b show the application of the **allocateVP** move. In particular, Fig. 4.a shows the initial solution when three requests are in the system and are served by two AVs, one traveling the route r' and the other one the route r''. Then, a new request i arrives into the system but it cannot be served by the **initSol** procedure since, if it was added to the current route r', the time window of request 2 would be violated. Moreover, request i cannot even be added to the current route r'' since otherwise the LOS of request 3 would be violated. Therefore, request i is inserted into the virtual pool. The **reOpt** procedure is not invoked since the routes are not changed compared to the previous time slot. Thus, **allocateVP** procedure is applied and request i is moved from the VP to route r'. Then, request 2 is assigned to route r'' being incompatible with request i, whereas it is compatible with request 3 (both for the time windows and LOS). This way, we obtain the new solution depicted in Fig. 4.b, where also request i is served, with total detour with respect the initial solution of $d_{r'} + d_{r''} = 2.5$ km.

6. Computational results

In this section, we describe the experimental campaign carried out on small, medium and large-sized instances, generated for RRPAV-TW-LOSC.

Both the MILP-RH and the RHLS were implemented in Java (in the Eclipse environment) and each MILP was solved by ILOG's CPLEX Concert Technology (version 20.1). It is noteworthy that we employed CPLEX default settings and let it use all cores/threads available. The experiments were run on a computer with a 64-bit operating system, 2.39 GHz processor and 32 GB of RAM.

We run our approaches on the benchmark instances proposed in the work of Najmi et al. (2017), related to a real-life case study in the city of Melbourne, specifically designed for the problem of matching users requests and vehicles in real-time ride-haring systems. Since they do not consider a fleet of AVs but of traditional internal combustion engine vehicles, we need to introduce some changes in their instances. In particular, to ensure that at the beginning the AVs are fully recharged, we randomly distribute them in the recharging stations according to a uniform probability distribution. Therefore, we integrated the instances with the information of the geographical coordinates (detected on the city map) of the RSs where the AVs are initially located. Moreover, we also assume that both the drivers and the riders, originally distinguished by the authors, are all riders. The time horizon T is 16 h and we discretize it in time slots of 10 s.

In order to have a significant set of instances to use in the experimental campaign, we started from those proposed in Najmi et al. (2017), i.e., three sets of instances (*Small* (S), *Medium* (M) and *Large* (L)). Since, the S instances are too large to test our MILP-RH, because they have 22,875 requests, we generate smaller instances from them. In particular, from the S instances, we derived two sets distributing the requests of each instance in different instances, preserving the distribution of their announcement times: *Very Reduced Small* instances (VRSI), with on average 104 requests, and *Reduced Small* instances (RSI), with on average 1144 requests.

The instances generated differ in difficulty. For the origin of each request, this can be evaluated by computing the number of potential compatible origins and the distance from the nearest one. In particular, on average, on the VRSI the percentage of compatible origins is only 21.2% and the nearest origin is 6.7 km far. On the RSI these values improve to 61.1% and 2.5 km. On the S instances they become 95.8% and 0.6 km, on the M instances 97.9% and 0.4 km, and finally on the L instances 98.6% and 0.4 km.

In the following subsections and in Appendix, the numerical results provided by the RHLS and the MILP-RH (when available) are discussed and compared. In particular, the following KPIs are introduced: the average level of service (LOS), the percentage number of requests served over the total number of requests arrived to the system (SER) and the average occupancy rate (OR). The latter is computed as the average number of users on board in the AVs weighted with the time in which they are on board. It is worth noting that LOS is also related to the average OR: LOS near 1 means that an AV serves the users almost as a taxi, i.e., going directly from the origin to the destination of each user and then they are served separately; whereas a LOS value near to its upper bound 1.5 indicates that the AVs serve more users together since there are detours from the origin to the destination of each user. In addition, we also report the total distance traveled (TD), the average variance of OR (σ^2), the total computational time (TCT) required in seconds, the number of vehicles in the fleet (NV) and the number of users (I), where IG indicates the instance group among L, M, S, RSI and VRSI. It is worth remarking that TCT in the case of the MILP-RH is computed as the average on the computational times obtained as in the following: the total time required by CPLEX for solving all the MILPs divided by the number of MILPs solved. Moreover, regarding both the VRSI and the RSI instances, all the KPIs aforementioned are computed as average values on each of their sub-groups as clarified in Section 6.1 and in Section 6.2.

6.1. VRSI instance group

In this section, the numerical results obtained on the VRSI instances are discussed. This set of instances is divided into three sub-groups as originally proposed in Najmi et al. (2017) which they were derived from (i.e., VRSI1, VRSI2 and VRSI3). The VRSI instances are a total of 393 and in each sub-group, the average number of users is equal to 104. All the details of the computational results are presented and described in Table A.9 of Appendix A.1 in which the average results, for each sub-group, are shown as the number of vehicles in the fleet changes.

Fig. 5 shows as the solutions found by the RHLS approach are very close, in terms of SER, to the optimal ones certified by the MILP-RH. In particular, the figure plots the comparison between the average SER, on the three sub-groups, of the MILP-RH and that one of the RHLS, in function of NV. The MILP-RH serves on average more users than the RHLS and this may be due to the fact that these instances are characterized by a relatively low number of requests that the MILP-RH can manage. The difference is very close to 0 when the fleet has only one AV. This is a reasonable behavior since, with only one AV, there are no so many possibilities to serve users differently. Instead, the difference increases (but no more than 3.50 users on average) as the number of AVs in the fleet increases too (until 5 AVs), meaning that with more AVs, the MILP-RH is able to meet the demand better than the heuristic approach. Finally, the difference decreases with both 10 and 20 AVs since the complexity of the MILP-RH substantially increases.

However, the fact that the MILP-RH serves more users on average also justifies the trend shown in Fig. 6, in which we can observe that its average TD is higher than that of the RHLS, in almost all the cases. It is worth noting that this difference becomes negative (i.e., the TD of RHLS is higher) only in the case in which the number of AVs in the fleet is 20 where the value of SER is on average quite the same. In this last case, both the approaches are suitable to meet the demand but the MILP-RH routes AVs better, thus leading to a lower average TD. In addition, both the approaches provide an average OR of 1.04 (see Appendix A.1).

Finally, on all the three sub-groups, as shown in Table A.9 in Appendix A.1, the RHLS requires an TCT that is on average by far lower than that of the MILP-RH. This is a significant aspect that allows us to address also medium and large sized instances, as shown in the following sub-sections.

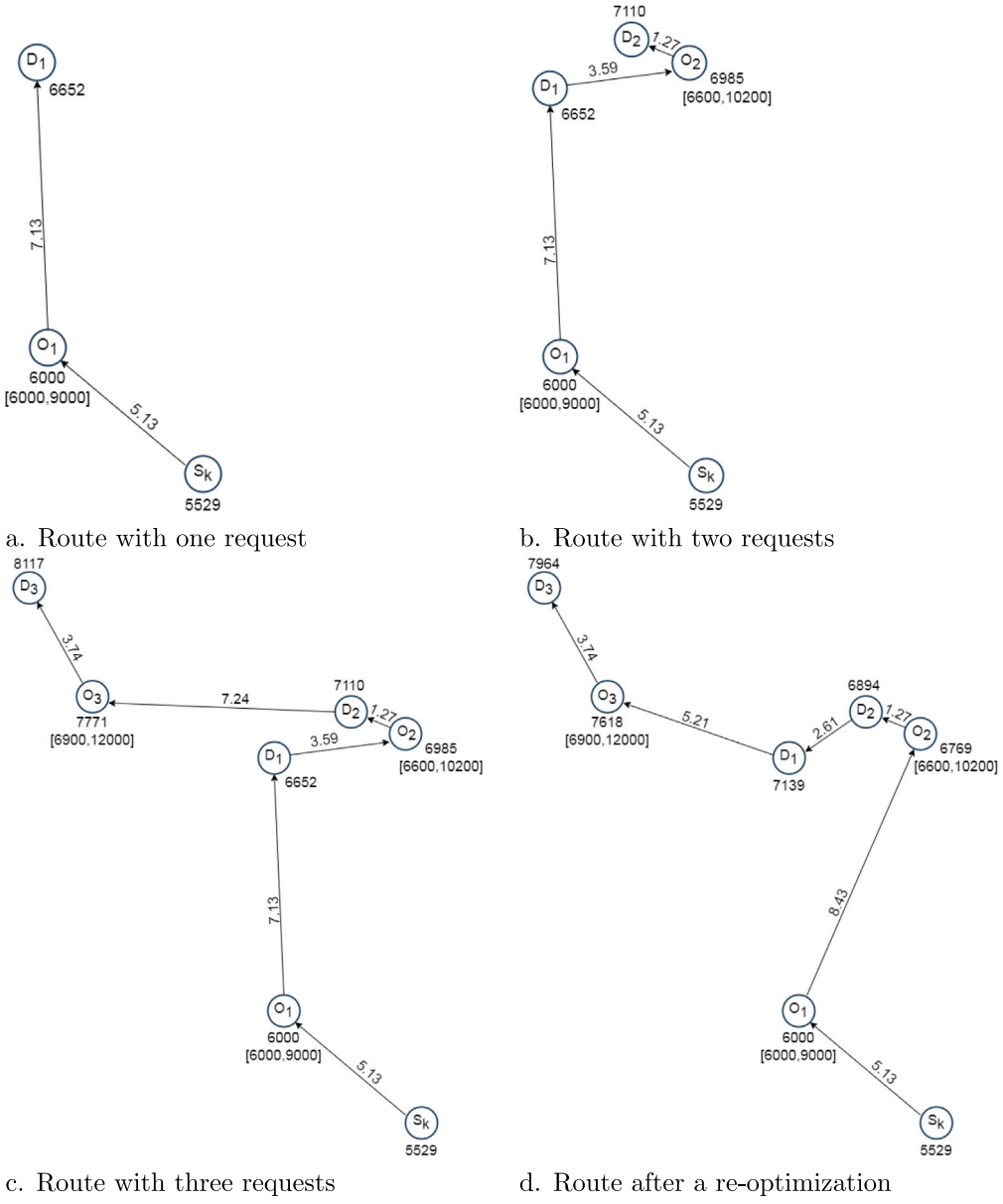
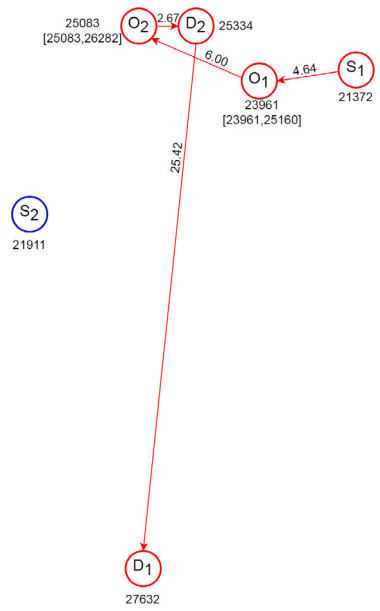


Fig. 2. An illustrative example showing the re-optimization of the route.

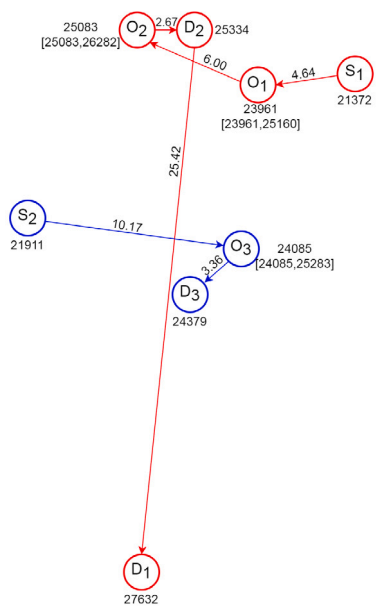
6.1.1. Comparisons between the MILP-RH and a bounding procedure

In order to determine a bound which the proposed MILP-RH can be compared with, an off-line MILP model is run. In other words, the solutions found by the MILP-RH are compared with those obtained running the MILP model under the assumption that all the requests are known a-priori. In the following, we refer to this bounding procedure as $MILP^{offline}$. The time limit given to CPLEX for solving the $MILP^{offline}$ is set to 72 h in order to find good quality bounds. To this aim, we selected the first 10 instances of each of the three VRSI groups. Table 4 reports, for each selected instance, the result comparisons between $MILP^{offline}$ and MILP-RH. In particular, for each approach, the percentage of users served (SER), the total traveled distance (TD) and the total computational time (TCT) in seconds are shown.

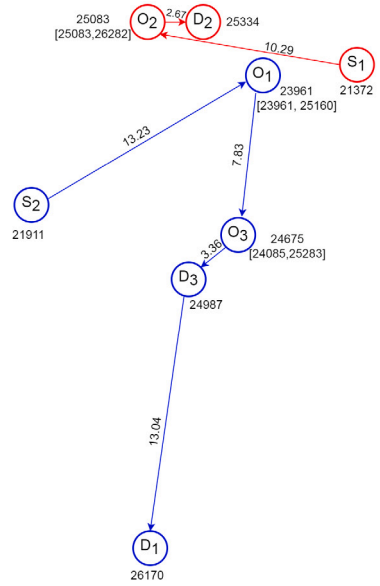
The results reported in the table show that, although working without knowing a-priori the requests to serve, the proposed MILP-RH has very small percentage gaps in terms of users served. Such percentages are on average equal to −8.75%, −7.69% and −7.88% respectively on the sets VRSI1, VRSI2 and VRSI3. The negative values indicate that the percentage of users served by the MILP-RH is less than that of the $MILP^{offline}$ since this gap is computed as the difference between SER of MILP-RH and that of $MILP^{offline}$. This behavior is reasonable because the MILP-RH has less degree of freedom regarding the acceptance of the next users, since it works without knowing a-priori all the requests to serve. Nevertheless, the average percentage gap on the number of requests served is not very high thus confirming the good quality solutions obtained by the MILP-RH. The percentage gap on the total traveled distance is computed dividing the difference between the TD of the MILP-RH and that of the $MILP^{offline}$ by the TD of the $MILP^{offline}$. In particular, the percentage gap on TD computed on the instances of the dataset VRSI1 is about 18.66% meaning that the solutions of MILP-RH require traveling on average longer distances that those of the $MILP^{offline}$ except for the instance VRSI1_6. On the instances of the dataset VRSI2, the average percentage gap on TD is about 14.99% whereas only on two instances, i.e., VRSI2_4 and VRSI2_6, the solutions of the MILP-RH require traveling a total distance shorter on average of 45.37%. Finally, analyzing the instances of the dataset VRSI3, the average percentage gap on TD is 15.76% whereas only on one instance, i.e., VRSI3_6, the solution found by the MILP-RH requires traveling a total distance shorter that of the $MILP^{offline}$ of 42.04%. It is worth remarking that as soon as a new request is accepted and



a. Route with the requests 1 and 2



b. Routes when the request 3 arrives



c. Routes after the swap move

Fig. 3. An illustrative example showing the application of the swap move.

becomes ACC, the assigned AV as well as its arrival time at the origin of the request cannot be changed anymore. This in some way reduces the degree of freedom of the MILP-RH that works within a rolling horizon. Nevertheless, such percentage gaps are not very high thus confirming again the good performances of the MILP-RH.

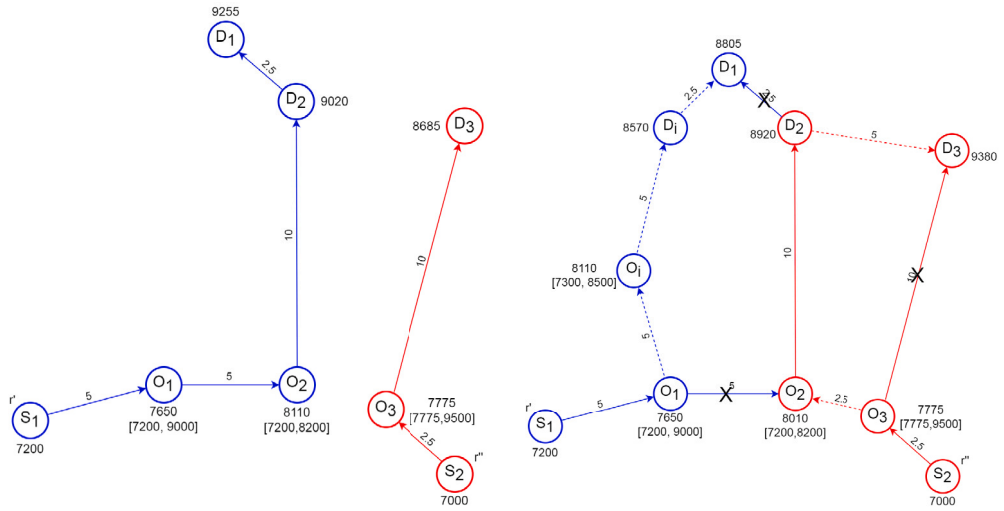
6.2. RSI instance group

In this section, the numerical results obtained on the RSI, divided into three sub-groups (i.e., RSI1, RSI2 and RSI3), are discussed. In total, the RSI are made up of 37 instances. The average number of users is equal to 1143.75 in each sub-group.

Table 5 shows the average results, for each sub-group and with a different number of AVs in the fleet. On the RSI, the MILP-RH is suitable to find solutions only when NV is equal to 2 and 5 whereas no feasible solution is found in the other cases. Comparing the two approaches, we observe that, for the solved instances, MILP-RH finds values of SER that are slightly better than those of RHLS (on average 0.4% higher), except for the instance RSI2 with 2 AVs. This can occur since sometimes, solving in heuristic way the problem can lead the AVs into a configuration that is more profitable for the next time slots. Concerning the traveled distance, for four cases, RHLS on average increases it of 0.35% and for two cases decreases it of 0.74%, on average.

On the instances solved by the MILP-RH, the average OR and the average σ^2 are almost the same for the two approaches (1.21 vs 1.20 and 0.21 vs 0.20). On all the instances solved by the RHLS, the average OR is 1.19 with an average σ^2 equal to 0.20.

The instances not solved by the MILP-RH, i.e., cases in which it is not able to even find a feasible solution, are highlighted in Table 5 through the † symbol. However, TCT taken by the MILP-RH still remains significantly higher than that of the RHLS (about 90% higher on



a. Initial solution with three requests

b. Solution after the allocateVP move

Fig. 4. An illustrative example showing the application of the allocateVP move.

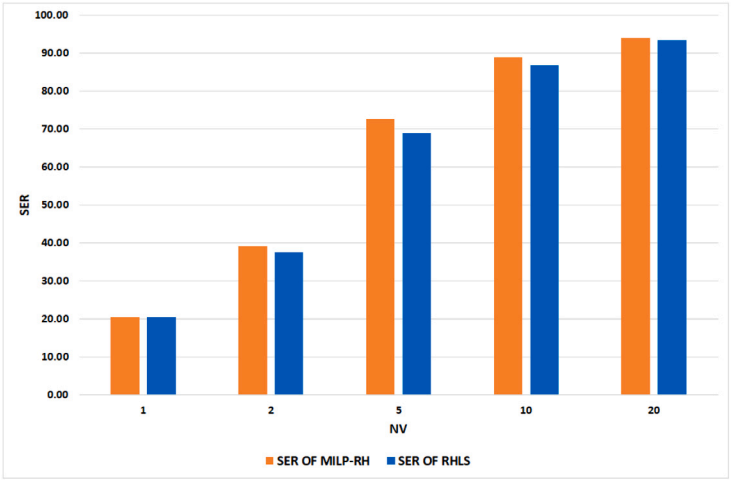


Fig. 5. Comparison between the average SER of MILP-RH and that of RHLS on the VRSI instances.

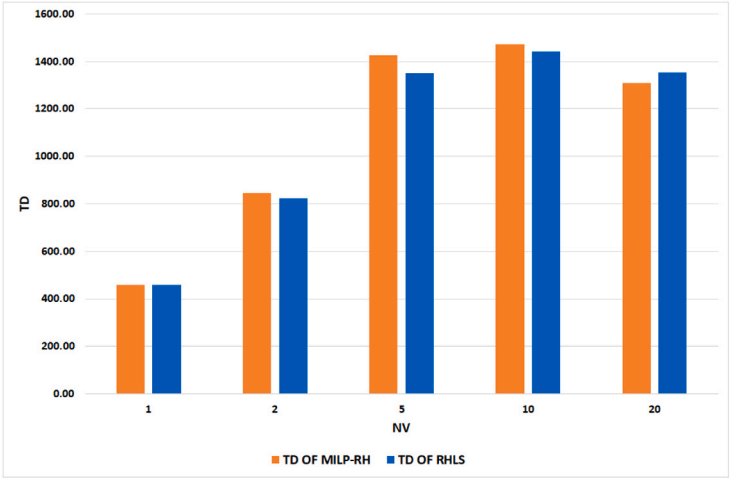


Fig. 6. Comparison between the average TD of MILP-RH and that of RHLS on the VRSI instances.

Table 4
Comparisons between the MILP-RH and the $MILP^{offline}$.

IG	MILP^{offline}			MILP-RH			GAPs (%)	
	SER	TD	TCT	SER	TD	TCT	SER	TD
VRSI1_0	27.88	388.24	11,318.68	17.31	450.59	1.84	−10.58	16.06
VRSI1_1	30.77	424.60	16,757.21	22.12	534.19	2.02	−8.65	25.81
VRSI1_2	30.77	378.38	89,464.53	22.12	498.21	9.84	−8.65	31.67
VRSI1_3	28.85	412.72	16,883.11	18.27	480.46	1.89	−10.58	16.41
VRSI1_4	26.92	387.36	26,149.00	20.19	471.61	1.84	−6.73	21.75
VRSI1_5	29.81	424.47	83,830.11	23.08	457.16	1.95	−6.73	7.70
VRSI1_6	29.81	833.95	259,205.67	18.27	434.27	1.94	−11.54	−47.93
VRSI1_7	31.73	433.70	107,999.48	24.04	503.40	3.72	−7.69	16.07
VRSI1_8	32.69	452.98	57,183.05	22.12	501.02	1.81	−10.58	10.61
VRSI1_9	26.92	327.46	15,298.97	21.15	399.31	1.84	−5.77	21.94
VRSI2_0	30.77	475.81	5753.14	25.00	494.95	1.84	−5.77	4.02
VRSI2_1	27.88	429.80	7498.29	17.31	449.87	1.77	−10.58	4.67
VRSI2_2	28.85	387.57	62,375.47	20.19	474.68	2.11	−8.65	22.48
VRSI2_3	28.85	317.04	5037.19	23.08	446.26	3.55	−5.77	40.76
VRSI2_4	27.88	604.59	86,430.47	11.54	307.70	1.77	−16.35	−49.11
VRSI2_5	25.96	381.85	558.11	19.23	413.64	1.77	−6.73	8.32
VRSI2_6	30.77	875.74	259,210.01	24.04	511.12	9.09	−6.73	−41.64
VRSI2_7	27.88	399.80	12,587.44	23.08	461.54	3.16	−4.81	15.44
VRSI2_8	29.81	411.66	25,556.53	23.08	477.41	2.23	−6.73	15.97
VRSI2_9	30.77	413.40	43,715.23	25.96	447.61	2.16	−4.81	8.27
VRSI3_0	28.85	362.61	23,662.45	25.00	463.04	1.98	−3.85	27.70
VRSI3_1	28.85	422.55	25,589.56	16.35	453.12	1.83	−12.50	7.24
VRSI3_2	28.85	436.96	25,437.44	21.15	472.54	1.84	−7.69	8.14
VRSI3_3	31.73	425.80	91,633.30	23.08	516.16	2.03	−8.65	21.22
VRSI3_4	30.77	391.91	155,050.79	18.27	429.47	1.80	−12.50	9.58
VRSI3_5	27.88	367.98	29,125.41	25.00	444.24	2.03	−2.88	20.73
VRSI3_6	29.81	829.31	259,209.11	21.15	480.71	1.95	−8.65	−42.04
VRSI3_7	28.85	450.38	2264.42	22.12	523.24	2.20	−6.73	16.18
VRSI3_8	30.77	412.66	33,952.29	20.19	432.63	1.89	−10.58	4.84
VRSI3_9	25.96	347.80	777.79	21.15	439.09	1.77	−4.81	26.25

Table 5
Numerical results on the RSI instances group.

IG	NV	MILP-RH						RHLS						GAPs (%)		
		SER	LOS	TD	OR	σ^2	TCT	SER	LOS	TD	OR	σ^2	TCT	SER	TD	TCT
RSI1	2	7.57	1.08	1150.25	1.20	0.19	94.01	7.50	1.08	1155.21	1.19	0.18	10.75	−0.07	0.43	−88.56
RSI2	2	7.01	1.08	1134.17	1.21	0.23	89.11	7.05	1.08	1143.39	1.19	0.19	9.08	0.03	0.81	−89.81
RSI3	2	7.49	1.09	1159.84	1.20	0.20	102.47	7.10	1.09	1175.86	1.21	0.22	9.95	−0.38	1.38	−90.29
RSI1	5	18.15	1.08	2771.85	1.19	0.19	810.00	17.53	1.08	2768.46	1.19	0.19	19.14	−0.63	−0.12	−97.64
RSI2	5	17.38	1.09	2747.13	1.21	0.22	700.55	17.07	1.08	2750.97	1.21	0.21	19.14	−0.31	0.14	−97.27
RSI3	5	17.63	1.09	2810.93	1.24	0.25	722.22	17.02	1.09	2772.61	1.22	0.21	18.92	−0.61	−1.36	−97.38
RSI1	10	†	†	†	†	†	†	33.84	1.07	5169.75	1.19	0.19	30.24	–	–	–
RSI2	10	†	†	†	†	†	†	33.73	1.08	5150.43	1.21	0.21	31.51	–	–	–
RSI3	10	†	†	†	†	†	†	33.02	1.08	5214.80	1.21	0.21	30.08	–	–	–
RSI1	20	†	†	†	†	†	†	60.83	1.08	8875.69	1.18	0.18	38.39	–	–	–
RSI2	20	†	†	†	†	†	†	60.11	1.08	8952.11	1.21	0.21	38.48	–	–	–
RSI3	20	†	†	†	†	†	†	59.89	1.08	9014.24	1.21	0.21	35.60	–	–	–
RSI1	50	†	†	†	†	†	†	92.51	1.07	11,788.48	1.17	0.17	24.04	–	–	–
RSI2	50	†	†	†	†	†	†	92.58	1.07	11,884.79	1.18	0.19	26.10	–	–	–
RSI3	50	†	†	†	†	†	†	91.93	1.07	11,906.35	1.19	0.19	24.42	–	–	–
RSI1	100	†	†	†	†	†	†	97.36	1.06	11,236.05	1.16	0.17	19.02	–	–	–
RSI2	100	†	†	†	†	†	†	97.31	1.06	11,396.57	1.18	0.18	20.48	–	–	–
RSI3	100	†	†	†	†	†	†	97.59	1.07	11,499.36	1.18	0.19	20.82	–	–	–

average). In addition, the number of MILPs solved in the MILP-RH is about equal to 104 and always closed to the optimality.

6.2.1. Comparison between the offline and the online versions of the RHLS

The aim of this section is to discuss the comparison between the online RHLS, proposed in this work, and the offline variant in which we assume that all the requests are known in advance. This comparison shows that there are some cases in which the online procedure can perform better than the offline one, despite the a priori knowledge of the latter. For the aim of this analysis, we consider the RSI instances with 2 AVs because only for them this phenomenon occurs. In fact, as shown in Fig. 7, the online RHLS performs better, in terms of SER, than the offline one, in very few cases (i.e., 5 out of a total of 37 instances). This is not due to the fact that the online RHLS has more computation time available since it takes on average 13.77 s in total against the 449.85 s required in total by the offline RHLS. Indeed, at each time slot, very few requests arrive (sometimes even none) and therefore, the online RHLS stops not because the CPU time limit is reached (i.e., the duration of the time slot) but already after a few fractions of a second because it is unable to optimize further. Instead, in the offline RHLS, the requests have to be optimized all at once and this clearly requires more computation time, taking into account that it has been run ad libitum. On the five instances in which the offline RHLS performs slightly worse, the online RHLS serves on average 5.2 requests more (i.e. about 6.4% more). This may be justified by the fact that, in some time slots, the online RHLS can make choices that reveal to be more advantageous in the future. Instead, the offline version, that operates heuristically as the online one, sometimes may make worse choices despite the a priori knowledge of all the requests.

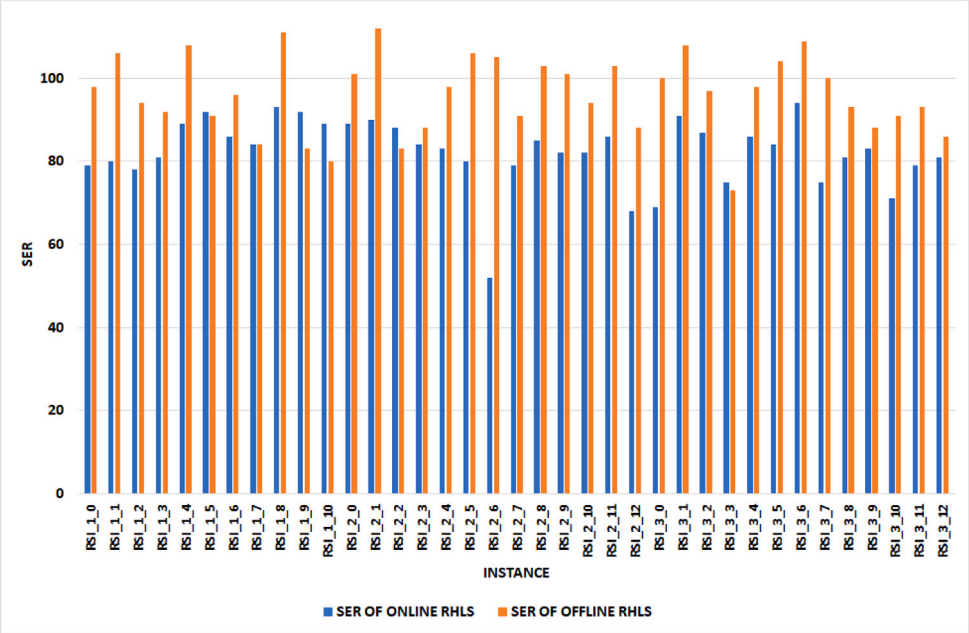


Fig. 7. Comparison between SER of RHLS offline and that of the RHLS online on the RSI instances with 2 AVs.

6.3. S, M and L instances

In this section, the numerical results obtained on the S, M and L instances are discussed. The number of requests is equal to 22875, 45750 and 68619 for the S, M and L, respectively. These instances are in total 3 for each group as originally proposed in Najmi et al. (2017).

Tables 6–8 show the numerical results, for each group and with a different number of vehicles in the fleet.

It is worth noting that the percentage of users served reasonably increases as NV increases too. In particular, on the S, it passes from 15.77% on average with 50 vehicles to 94.93% on average with 500 vehicles. Whereas, on the M, it goes from 17.63% on average with 100 vehicles to 96.25% on average with 500 vehicles. Finally, on the L, it varies from 25.33% on average with 200 vehicles to an average of 90.57% with 1000 vehicles.

Regarding LOS, it varies on average between 1.16 and 1.18 on the S, between 1.18 and 1.19 on the M and finally, between 1.19 and 1.20 on the L. In addition, the average OR is almost equal to 2 on all these instances. Finally, TCT is still reasonable, compared to the size of these instances. In fact, it varies between 5103.69 s and 26,858.37 s on the S, between 26,119.54 s and 43,395.81 s on the M and finally, between 44,862.61 s and 49,351.24 s on the L.

6.4. Sensitivity analysis varying the LOS and delaying the time windows

The aim of this section is to show some result comparisons when the time window of each request (associated with the pick-up) is delayed of a certain number of seconds. For the aims of this sensitivity analysis, we use the VRSI, the RSI and the S instances. Moreover, the S instances are tested with only some values of NV. This analysis originates from an observation regarding the announcement times of the requests. Indeed, we noted that the difference between the earliest pick-up time and the announcement time, is on average 28 min. Since the user must be notified of the actual pick-up time at own origin 30 min before the earliest pick-up time, the computed pick-up time is immediately frozen, for almost all the requests accepted. Therefore, the aim of delaying the time windows is to investigate whether the SER and the other KPIs can improve providing more available time between the announcement time and the earliest pick-up time.

Table 6
Numerical results on the S instances group.

IG	NV	RHLS					
		SER	LOS	TD	OR	σ^2	TTC
S1	50	15.80	1.17	28,795.64	1.64	0.66	5103.69
S2	50	15.73	1.17	28,725.47	1.69	0.69	5288.53
S3	50	15.77	1.18	28,734.72	1.72	0.73	5153.55
S1	100	31.94	1.17	54,574.01	1.69	0.71	14,604.14
S2	100	32.12	1.18	54,923.71	1.74	0.74	14,795.17
S3	100	32.07	1.18	54,699.18	1.74	0.75	14,434.93
S1	200	61.30	1.17	98,099.54	1.70	0.71	25,841.10
S2	200	61.07	1.17	99,213.28	1.72	0.72	25,698.60
S3	200	60.44	1.17	98,476.01	1.72	0.74	26,270.23
S1	300	81.46	1.16	129,180.54	1.67	0.70	25,869.82
S2	300	82.11	1.17	130,425.90	1.69	0.72	25,845.74
S3	300	81.15	1.17	129,995.31	1.70	0.72	26,858.37
S1	400	90.34	1.16	141,603.26	1.65	0.68	25,005.64
S2	400	91.34	1.16	144,039.25	1.67	0.70	25,151.79
S3	400	90.76	1.16	143,011.37	1.68	0.71	25,147.48
S1	500	94.67	1.16	147,124.42	1.65	0.68	24,083.56
S2	500	95.02	1.16	146,593.70	1.66	0.70	24,277.73
S3	500	95.11	1.16	149,485.06	1.66	0.70	24,350.97

Table 7
Numerical results on the M instances group.

IG	NV	RHLS					
		SER	LOS	TD	OR	σ^2	TCT
M1	100	17.82	1.19	57,333.54	1.85	0.83	27,764.99
M2	100	17.51	1.20	56,798.95	1.83	0.83	26,567.80
M3	100	17.55	1.19	56,902.89	1.86	0.83	26,119.54
M1	200	36.51	1.19	107,696.27	1.89	0.88	42,623.39
M2	200	35.71	1.19	107,380.95	1.87	0.85	42,535.75
M3	200	35.79	1.19	107,644.25	1.89	0.87	41,882.46
M1	500	79.48	1.18	225,943.96	1.86	0.86	43,183.42
M2	500	79.43	1.18	223,432.02	1.84	0.85	43,395.81
M3	500	79.21	1.18	225,221.04	1.86	0.87	43,191.13
M1	1000	96.62	1.18	268,237.13	1.80	0.82	43,077.22
M2	1000	95.96	1.18	266,099.34	1.79	0.82	43,230.55
M3	1000	96.16	1.18	267,278.58	1.81	0.83	43,084.41

The procedure implemented to delay the time windows is as follows. For each request, a random number between 0 and 7200 s is generated.

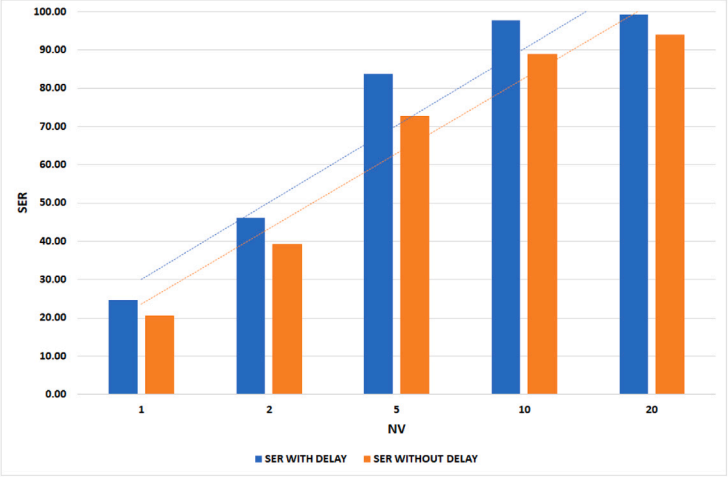


Fig. 8. Comparison of SER on the VRSI instances with and without delayed TWs.

Table 8
Numerical results on the L instances group.

IG	NV	RHLS					
		SER	LOS	TD	OR	σ^2	TCT
L1	200	24.90	1.20	168,634.69	1.97	0.93	44,862.61
L2	200	25.60	1.20	111,190.39	1.98	0.92	45,520.30
L3	200	25.51	1.20	111,552.40	1.97	0.92	45,342.61
L1	500	60.85	1.20	322,873.55	1.97	0.93	48,854.66
L2	500	61.13	1.20	249,257.01	1.98	0.93	49,282.26
L3	500	60.83	1.20	248,937.18	1.97	0.92	48,763.43
L1	1000	90.49	1.19	440,487.64	1.89	0.89	48,908.12
L2	1000	90.69	1.19	365,830.48	1.90	0.89	49,351.24
L3	1000	90.54	1.19	365,275.09	1.90	0.88	49,015.47

Then, a number of seconds equal to the extracted one is added to both the left and the right side of each time window. This way, the TW is delayed on average by one hour, without changing its width.

The detailed results are shown in Tables A.10–A.12 of Appendix A.2. Figs. 8, 10 and 12 show how, by increasing the number of AVs in the fleet, the value of SER increases too, in both the cases (with and without delayed TWs), for the VRSI, the RSI and the S instances, respectively. However, the increment is more significant when the TWs are delayed. In particular, the difference between the SER with delayed TWs and that one without delays is higher with 5 and 10 AVs in the fleet, for the VRSI instances, with 20 and 50 AVs, for the RSI, and with 200 AVs, for the S instances. These considerations also justify the trends in Figs. 9, 11 and 13 that show the average TD by varying NV, with and without the delays. With the same aforementioned values of AVs in the fleet, the average TD with delays is higher than that without delays and this is due to the fact that, with delays, we are able to serve more users. However, as NV increases, the average TD reasonable decreases in both the cases (i.e., with and without the delays) because there are more possible ways for handling the requests. In addition, the difference between the average TD with and without delays becomes lower.

Tables A.13–A.15, in Appendix A.3, detail the numerical comparisons, on the VRSI, RSI and S instances, respectively, when the LOS passes from 1.5 to 2 and the TWs are delayed as in the previous case. Figs. 14, 16 and 18 show the value of SER varying NV. We can observe that the increase of LOS does not affect SER for both the VRSI and RSI instances, but it slightly improves only the average TD for the RSI instances with $NV = 50,\ 100$ (see Figs. 15, 17 and 19). Whereas, it improves SER for the S instances with the higher values of NV (i.e. $NV = 100,\ 200$). Finally, it is worth noting that the increase of LOS significantly increases the OR, on average, of $4.50\%, 11.54\%$ and 19.50%, on the VRSI, RSI and S instances, respectively. This behavior is reasonable since increasing the LOS it is more likely that the users can share the same ride in feasible way.

A similar behavior can also be noted when we pass from a LOS equal to 2 to a LOS equal to 1.5, without delaying the TWs as shown in Figs. 20–25. The corresponding numerical comparisons are detailed in the Tables A.16–A.18 of Appendix A.4.

6.5. Analysis on the objective function terms and on the discretization step of the time horizon

The aim of this section is to analyze the impact of both the terms in the objective function and the step used for discretizing the time horizon T. For this purpose, we consider the RSI instances because they are in a statistically significant number, are characterized by a statistically significant number of requests and finally, with 2 and 5 AVs in the fleet, they are solved to the optimality. For the aim of our analysis, we focus on the case in which NV is equal to 2.

Fig. 26 shows how the different terms in the objective function affect the SER. In particular, we denote by f1 the number of users served (that we want to maximize with a higher priority), by f2 the total travel distance and by f3 the total time required for serving the requests (both to minimize with a lower priority). It is also worth noting that we cannot optimize only either f2 or f3 because otherwise we would get a solution with zero users served. This is why the cases in which f2 or f3 are optimized without f1 are not considered in our analysis. For the same reason, in our hierarchical objective function, we optimize f1 with higher priority compared to that of f2 and f3.

Analyzing the results shown in Fig. 26, we can note that the case in which only f1 is optimized performs the worst since, in each time slot, only the number of users satisfied is taken into account, neglecting both f2 and f3. Instead, the case in which f1 + f2 are both optimized performs better than the previous one since optimizing also f2, the detours to serve the users are minimized. Consequently, it is more likely that in the next time slots further users can be handled by the same AV without violating the LOS constraint or the time windows. Moreover, the case in which f1 + f3 are both optimized performs better than the two previous cases since minimizing also f3 allows serving, in the next time slots, a greater number of users that may be not possible to satisfy due to their time windows at their pickup points. Therefore, the best case is that in which all the three terms f1 + f2 + f3 are optimized.

Regarding the discretization step, for the aim of our analysis, we considered three possible values: 10 s (the original step), 60 s and finally, 600 s. First of all, it is worth noting that the proposed solution approaches work independently on the specific step chosen and that

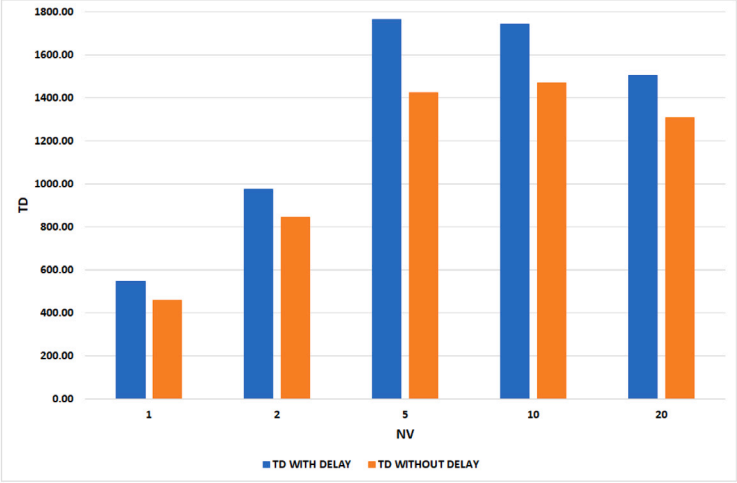


Fig. 9. Comparison of TD on the VRSI instances with and without delayed TWs.

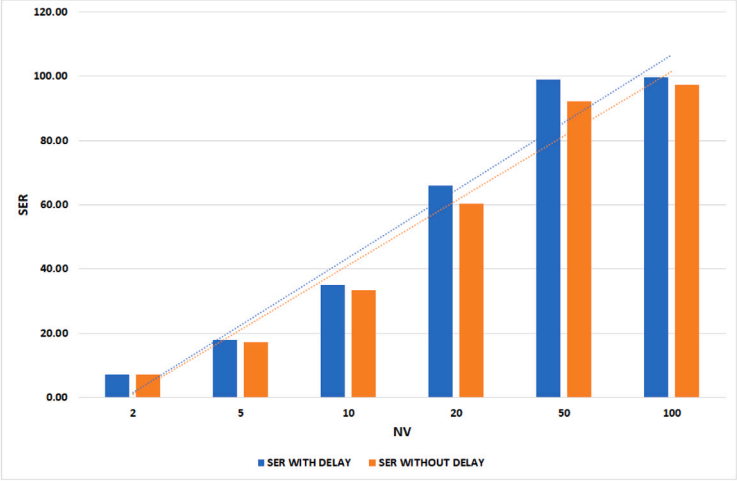


Fig. 10. Comparison of SER on the RSI instances with and without delayed TWs.

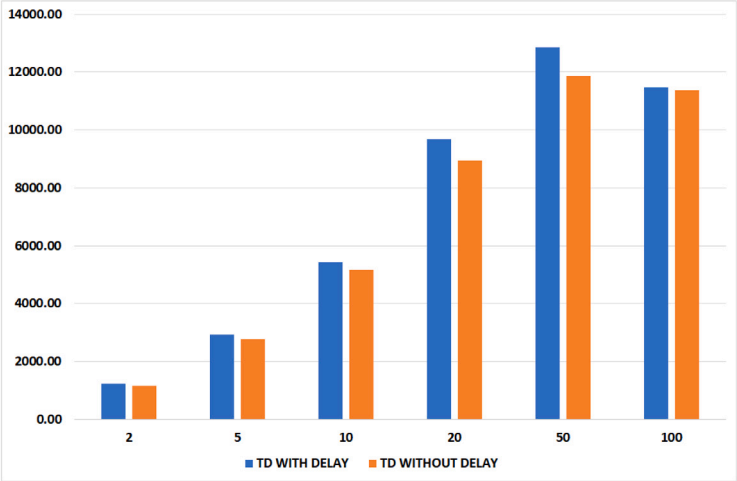


Fig. 11. Comparison of TD on the RSI instances with and without delayed TWs.

it can be varied according to the particular context. In addition, we have set the step equal to 10 s in our experiments because this value allows users to wait less for their acceptance or rejection. The results shown in Fig. 27 confirm a reasonable trend, i.e. as the discretization step increases, a better assignment of users to AVs can be achieved since, in larger time slots, the requests collected are larger and then, the decisions made are less myopic compared to those in smaller time slots. By consequence, SER increases too, passing from 83.67% with

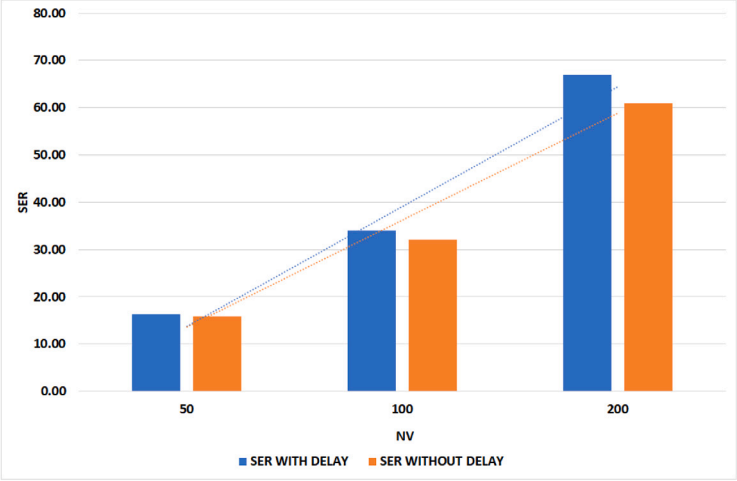


Fig. 12. Comparison of SER on the S instances with and without delayed TWs.

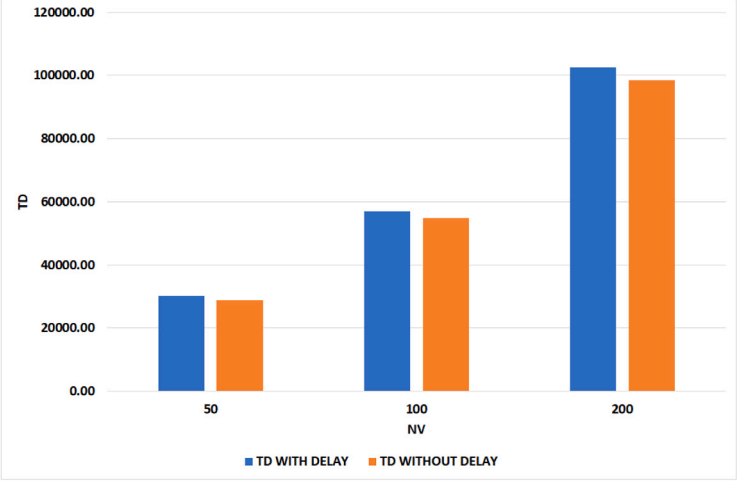


Fig. 13. Comparison of TD on the S instances with and without delayed TWs.

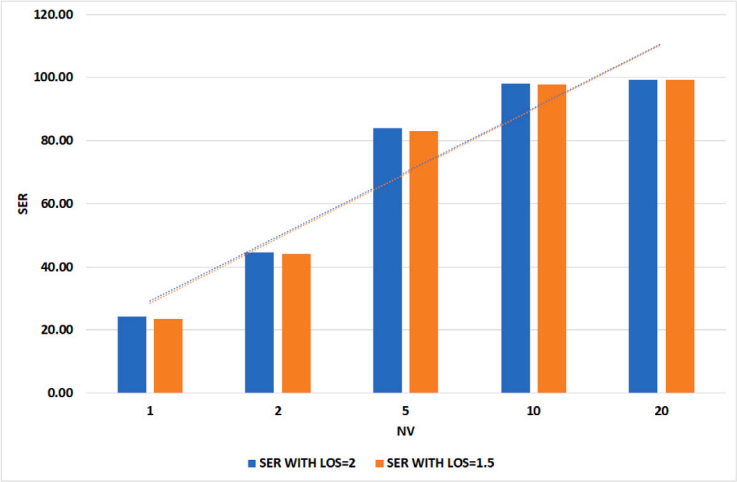


Fig. 14. Comparison of SER on the VRSI instances with LOS=2 and LOS=1.5, with delays.

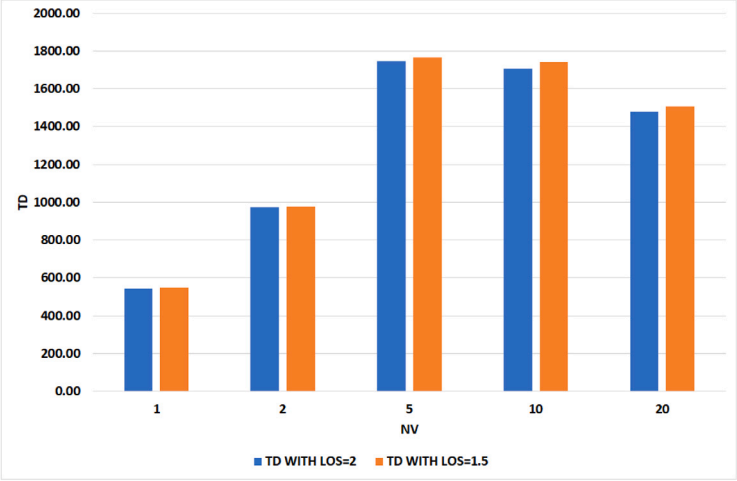


Fig. 15. Comparison of TD on the VRSI instances with LOS=2 and LOS=1.5, with delays.

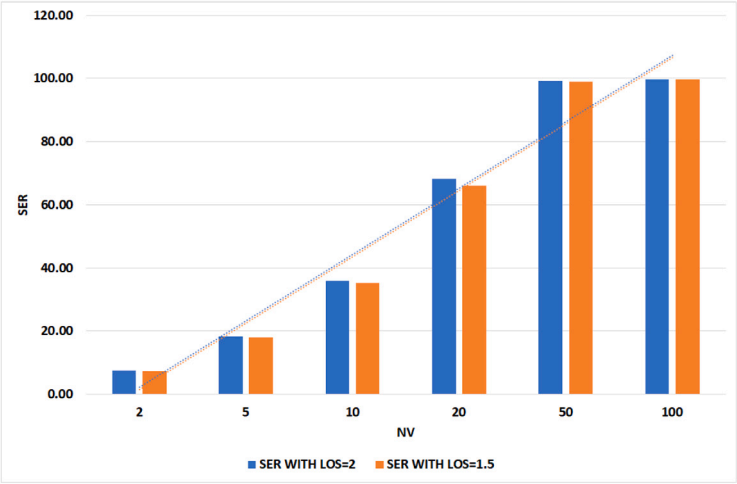


Fig. 16. Comparison of SER on the RSI instances with LOS=2 and LOS=1.5, with delays.

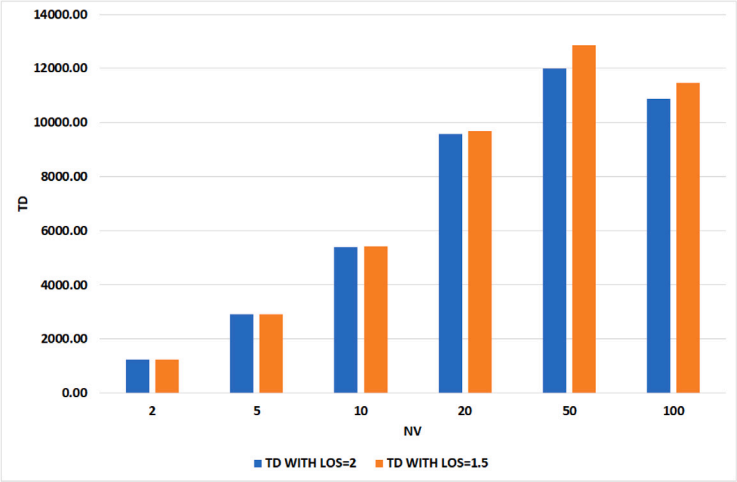


Fig. 17. Comparison of TD on the RSI instances with LOS=2 and LOS=1.5, with delays.

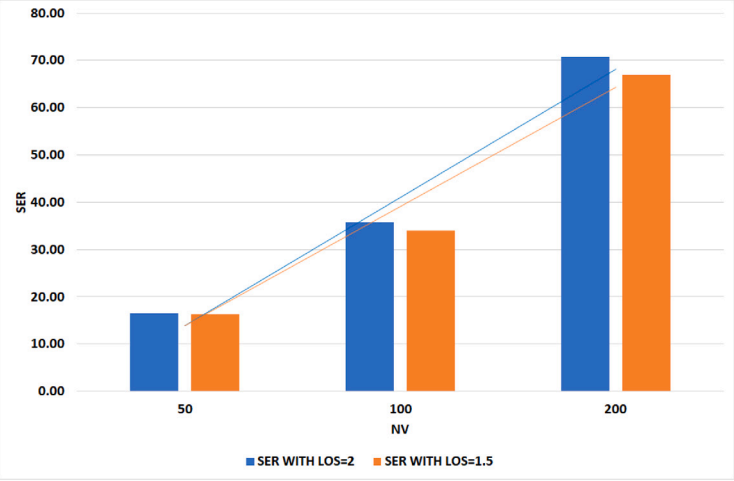


Fig. 18. Comparison of SER on the S instances with LOS=2 and LOS=1.5, with delays.

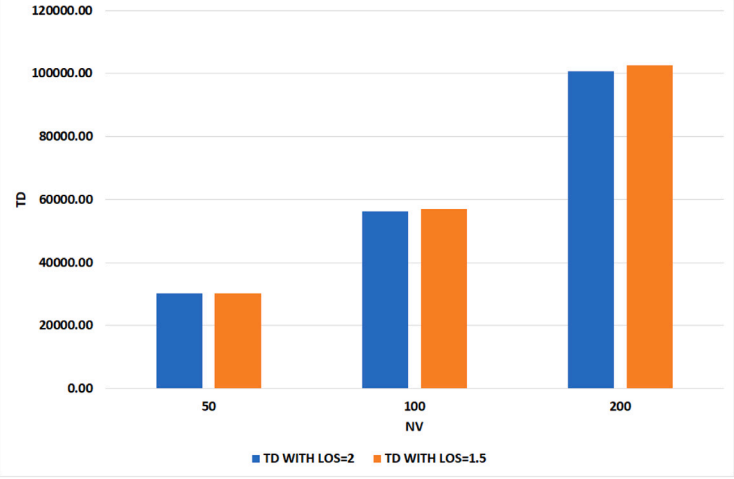


Fig. 19. Comparison of TD on the S instances with LOS=2 and LOS=1.5, with delays.

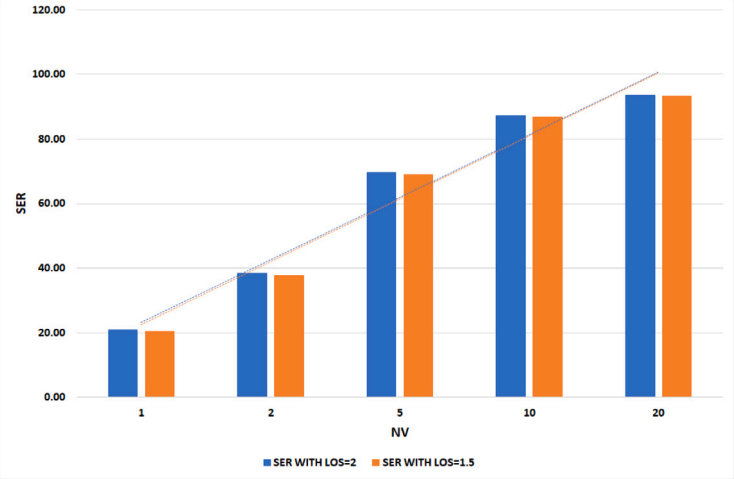


Fig. 20. Comparison of SER on the VRSI instances with LOS=2 and LOS=1.5.

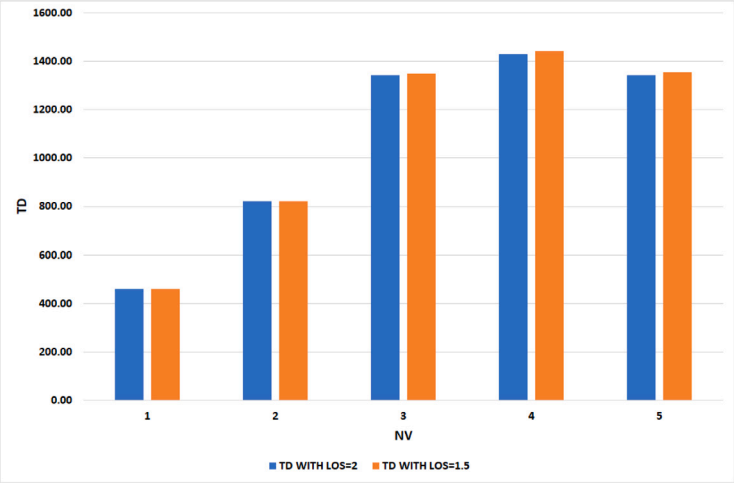


Fig. 21. Comparison of TD on the VRSI instances with LOS=2 and LOS=1.5.

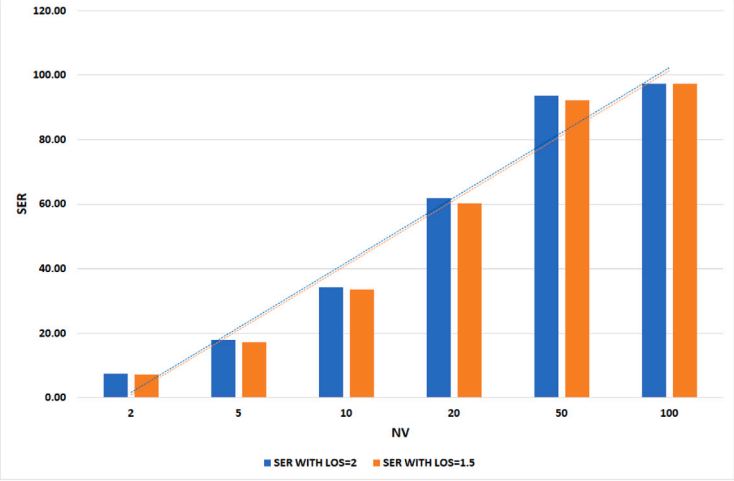


Fig. 22. Comparison of SER on the RSI instances with LOS=2 and LOS=1.5.

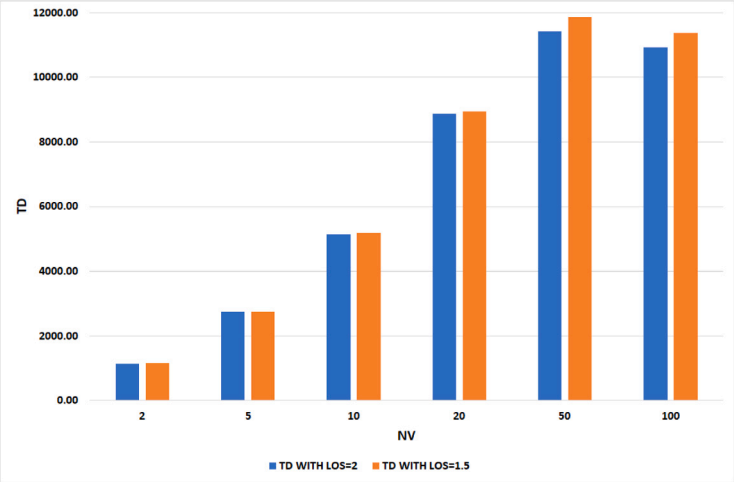


Fig. 23. Comparison of TD on the RSI instances with LOS=2 and LOS=1.5.

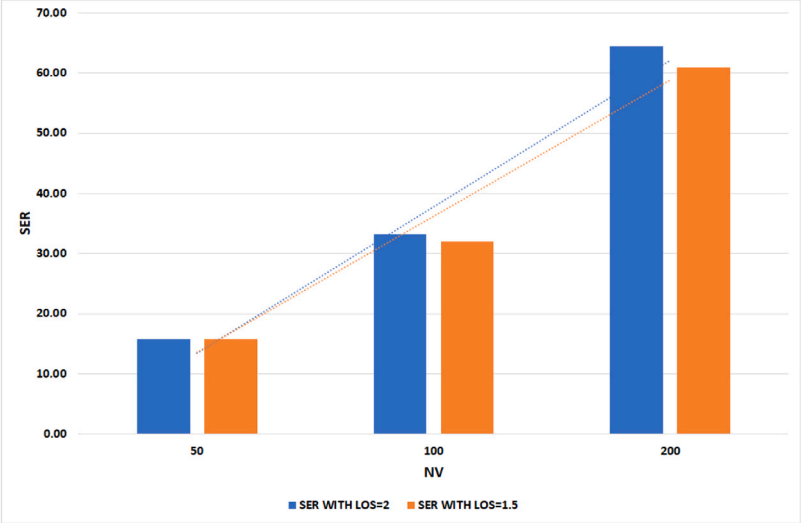


Fig. 24. Comparison of SER on the S instances with LOS=2 and LOS=1.5.

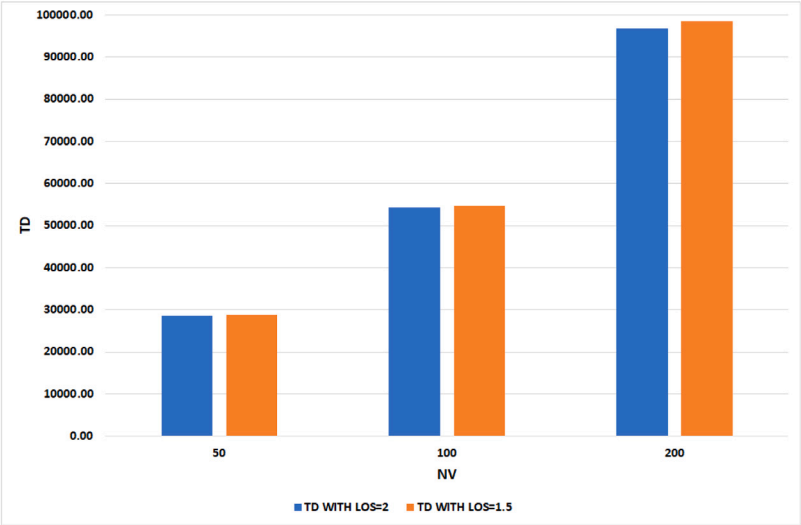


Fig. 25. Comparison of TD on the S instances with LOS=2 and LOS=1.5.

step equal to 10 s to 84.30% and 86.83% with steps equal to 60 and 600 s, respectively.

7. Conclusions and future works

In this work, we addressed the Real-time Ride-sharing Problem with Autonomous Vehicles, Time Windows and Level Of Service Constraint (RRPAV-TW-LOSC). The LOSC ensures that the travel time of each user is not greater than L times the shortest time to reach his/her destination (with $L \geq 1$). The objective function consists in the lexicographic optimization first of the number of users served and then, with the same priority, of the total traveled distance and the total time for serving the requests. This is a novelty, compared to the objective functions considered in most of the related literature where generally the total revenue associated with the served requests is maximized or the difference between it and the total cost to serve the requests. Instead, we maximize with higher priority the total number of requests served, regardless of its revenue, because if a request is rejected, the related user will probably not use the same service operator in the future. The other two objectives have been mainly introduced because, considering the dynamic arrival of requests to the system, the sooner we can satisfy current requests, the more likely the vehicles will be free to satisfy future requests.

We formulated the problem through Mixed Integer Linear Programming (MILP) and solved it by a rolling horizon approach (MILP-RH) assuming to discretize time into slots of fixed duration. We also designed a rolling horizon Local Search (RHLS) approach in order to efficiently solve medium and large-sized instances of RRPAV-TW-LOSC with even more 68,000 requests.

Another important gap in the current literature filled by our work is that the only few papers that also consider LOSC do not address pre-booking requests (i.e. requests made also long time in advance) and assume that the requests must be served as soon as they arrive, without considering time windows associated with them.

A further original aspect we consider is the assumption that the request acceptance is separated from the pick-up time decision, since we assume that the latter must be communicated to the users only 30 min before their earliest pick-up time. This degree of freedom, very little investigated in the related literature, is exploited in our solution approach and allows obtaining better solutions. Indeed, with an extended experimental campaign we show that anticipating on average the booking requests of 1 h significantly improves the percentage of accepted requests between about 6% (for the larger-sized instances with

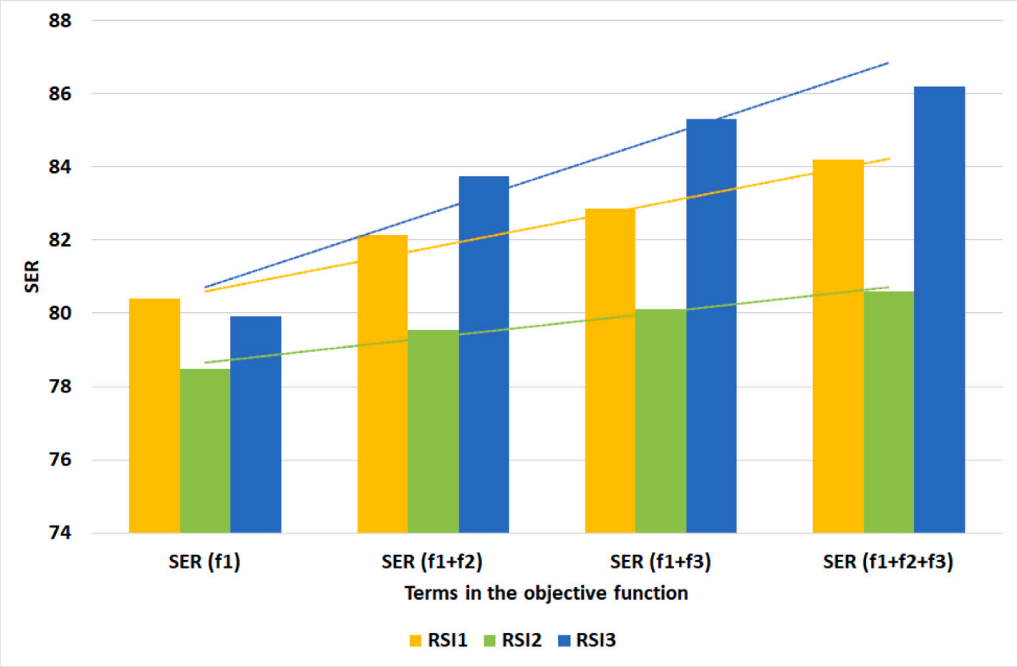


Fig. 26. Impact of the terms of the objective function on the RSI instances.

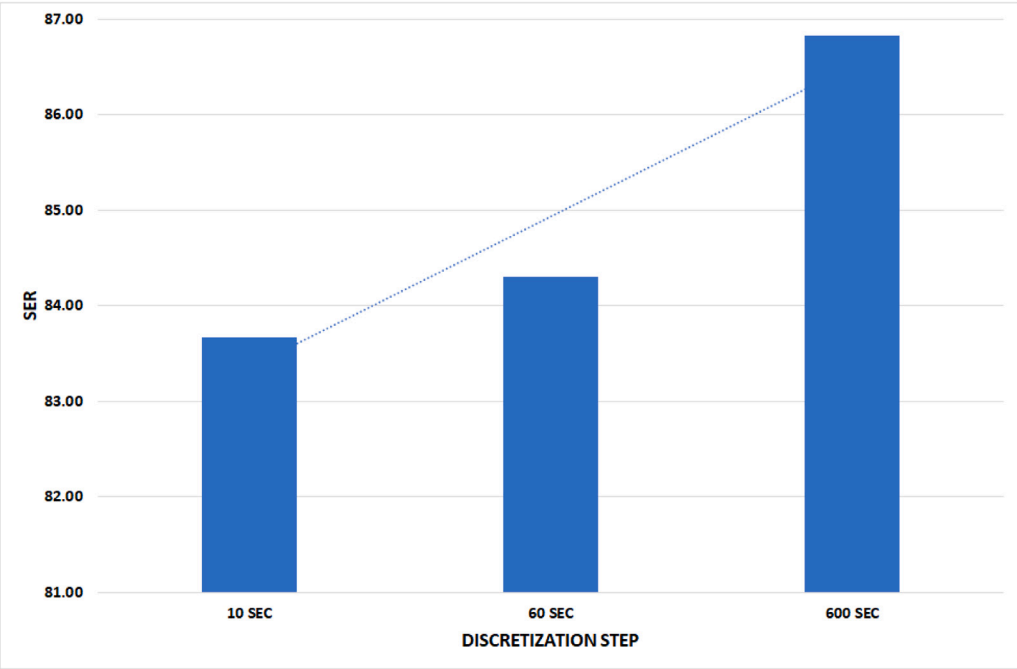


Fig. 27. Impact of the discretization step on the RSI instances.

200 AVs) up to about 10% (for the smaller instances with 5 AVs). On the other hand, this assumption makes the problem harder to be solved since the vehicle to which the request is assigned and the pick up time can be reconsidered along the optimization process being not fixed when the request is accepted. Despite this, our RHLS approach is very efficient since at each time slot it is able to provide the answer about the acceptance or rejection of the new requests arrived within 10 s, i.e., the duration of a time slot.

Another aspect that makes RRPAV-TW-LOSC more challenging, compared to other real-time ride-sharing optimization problems, is the use of a fleet of AVs. The impact of this, in the mathematical model and then, in the solution approach that is based on it, is significant. Indeed, unlike the classic ridesharing problems with manned vehicles, the route of each AV does not end at the driver's destination (since no driver exists) thus the number of passengers that a vehicle may visit is generally by far higher being not limited to those whose destination is in a neighborhood of the driver's destination.

We also performed a sensitivity analysis on the LOS, increasing it from 1.5 to 2. We observed that it does not affect the average percentage of accepted requests but it significantly affects the average occupancy rate with an average improvement of about 12% (for the smaller instances) up to about 19% (for the larger instances). Moreover,

Table A.9
Numerical results on the VRSI instances group.

IG	NV	MILP-RH						RHLS						GAPs (%)		
		SER	LOS	TD	OR	σ^2	TCT	SER	LOS	TD	OR	σ^2	TCT	SER	TD	TCT
VRSI1	1	20.47	1.02	458.88	1.04	0.04	2.62	20.46	1.02	458.67	1.04	0.04	0.72	0.01	0.05	72.47
VRSI2	1	20.59	1.02	458.10	1.04	0.04	2.70	20.59	1.02	458.17	1.04	0.04	0.73	0.00	−0.02	73.02
VRSI3	1	20.26	1.02	463.15	1.05	0.04	2.45	20.21	1.02	463.02	1.05	0.04	0.73	0.04	0.03	70.08
VRSI1	2	39.72	1.02	849.80	1.04	0.04	9.18	37.80	1.02	819.05	1.04	0.04	0.88	1.92	3.62	90.39
VRSI2	2	39.11	1.02	845.09	1.04	0.04	9.01	37.89	1.02	827.58	1.04	0.04	0.87	1.22	2.07	90.36
VRSI3	2	38.52	1.02	845.44	1.04	0.04	8.50	37.07	1.02	822.05	1.04	0.04	0.86	1.44	2.77	89.83
VRSI1	5	72.67	1.02	1407.76	1.03	0.03	31.63	68.82	1.02	1334.20	1.03	0.03	1.11	3.85	5.23	96.48
VRSI2	5	72.68	1.02	1437.63	1.04	0.04	31.30	69.31	1.02	1360.77	1.04	0.03	1.09	3.37	5.35	96.51
VRSI3	5	72.57	1.02	1430.92	1.04	0.04	30.89	68.85	1.02	1352.54	1.04	0.04	1.09	3.71	5.48	96.48
VRSI1	10	89.01	1.01	1460.64	1.03	0.03	49.24	86.77	1.02	1428.36	1.03	0.03	1.23	2.24	2.21	97.50
VRSI2	10	88.96	1.01	1481.99	1.03	0.03	50.20	86.88	1.01	1450.10	1.03	0.03	1.23	2.08	2.15	97.55
VRSI3	10	88.92	1.01	1472.09	1.03	0.03	49.41	86.83	1.02	1450.01	1.03	0.03	1.23	2.09	1.50	97.50
VRSI1	20	94.15	1.01	1299.96	1.02	0.02	99.10	93.63	1.01	1343.20	1.04	0.04	1.54	0.52	−3.33	98.45
VRSI2	20	93.72	1.01	1313.02	1.02	0.02	99.08	93.18	1.01	1357.50	1.04	0.04	1.55	0.54	−3.39	98.44
VRSI3	20	94.07	1.01	1318.09	1.02	0.02	100.14	93.54	1.01	1360.66	1.05	0.04	1.56	0.53	−3.23	98.45

we also performed a sensitivity analysis on the different terms of the objective function showing that the best value of average percentage of served requests is obtained when all the terms are optimized at the same time in lexicographic way (with a relative average improvement of about 8% compared to the case in which only the number of accepted request is maximized). Finally, we performed a sensitivity analysis also on the impact of the discretization step increasing it from 10 s to 60 and 600 s. This analysis shows that as the discretization step increases, a better assignment of users to AVs can be achieved since, in larger time slots, the requests collected are larger and then, the decisions made are less myopic than those in smaller time slots.

Hence, the main insights we can draw from our experimental campaign are that providing user requests as far in advance as possible is useful for increasing user acceptance, but not the vehicle occupancy rate. Whereas, increasing the LOS is advantageous for increasing the vehicle occupancy rates, but not the percentage of accepted requests. This behavior is reasonable since increasing the LOS makes more likely that the users can share the same ride in feasible way. Finally, increasing the discretization step is also useful to increase the percentage of accepted requests since it allows making less myopic decisions, but it increases the waiting time for users to know their acceptance or rejection.

As future research directions, it would be interesting to consider the extension of the proposed MILP-RH model to a stochastic environment, where stochasticity might be related to both the geographical distribution of future requests and to their time windows. Thus, the MILP-RH model presented in Section 4 could be adapted to embed uncertainty either through stochastic programming or robust optimization methods.

CRediT authorship contribution statement

M. Bruglieri: Writing – review & editing, Writing – original draft, Validation, Supervision, Software, Methodology, Investigation, Conceptualization. **R. Peruzzini:** Writing – review & editing, Writing – original draft, Validation, Supervision, Software, Methodology, Investigation, Conceptualization. **O. Pisacane:** Writing – review & editing, Writing – original draft, Validation, Supervision, Software, Methodology, Investigation, Conceptualization.

Data availability

Data will be made available on request.

Acknowledgments

This paper was partially founded by the Università Politecnica delle Marche, under RSA-B 2020 project.

Appendix. Detailed computational results

A.1. VRSI instances

Table A.9 shows the average results, for each sub-group of type VRSI and with a different number of vehicles in the fleet. Moreover, in each sub-group, the average number of users is equal to 104.

For 6 instances over 15, the RHLS finds solutions with a value of SER near to the optimal one certified by the MILP-RH. Indeed, for them the percentage gap on SER is lower than 0.54%, but the MILP is able to reduce the TD even of 3.39%. For the other instances, MILP-RH is able to find solutions with a percentage improvement on SER up to 3.85% but often with an increasing of TD also of 5.48%. However, the total computational time required by the RHLS is always by far less than that of the MILP-RH, on average of about 91%.

Finally, for both the approaches, the average OR is equal to 1.04 meaning that on average almost one user is on board. Moreover, the average σ^2 is very low, i.e., equal to 0.03 and 0.04, respectively, for the MILP-RH and the RHLS, meaning that the OR is almost stable in their solutions.

A.2. Sensitivity analysis delaying the time windows

In the following tables, we compare the original scenario with that in which the time windows are delayed as described. For the three sets of instances (VRSI, RSI and S), the results are reported in Table A.10, A.11 and A.12, respectively. Table A.10 shows that, for all the three subsets, SER significantly increases when the time windows are delayed, passing from a minimum average increase of about 3%, in the case with one vehicle, to a maximum of about 10%, obtained with 5 vehicles. This trend is confirmed also on the RSI set as reported in Table A.11, where the average increase is up to 6.57% with 50 vehicles. Finally, also on set S (Table A.12) SER significantly improves when the TWs are delayed, with an average increase up to 5.98% for 200 vehicles. Of course, in the meantime TD worsen since on average it increases of about 18.26% on set VRSI, 5.73% on RSI and 4.46% on S. Whereas, OR slightly improves on VRSI and RSI with an average increase of 1.89% and 1.23%, respectively, and slightly worsens on S with an average decrease of −1.80%.

A.3. Sensitivity analysis varying the LOS and delaying the time windows

Tables A.13–A.15 compare the numerical results obtained on the three sets of instances VRSI, RSI and S, respectively, when the LOS passes from 1.5 to 2 and the time windows are delayed as before. In this case, the increase of SER is not significant on the sets VRSI and RSI being on average only 0.63% and 0.41%, respectively. It is pretty

Table A.10
Numerical comparisons on the VRSI instances with and without delays.

IG	NV	With delay					Without delay				
		SER	LOS	TD	OR	σ^2	SER	LOS	TD	OR	σ^2
VRSI1	1	23.63	1.03	548.71	1.06	0.05	20.47	1.02	458.88	1.04	0.04
VRSI2	1	23.83	1.03	545.14	1.06	0.05	20.59	1.02	458.10	1.04	0.04
VRSI3	1	23.48	1.03	544.94	1.05	0.05	20.26	1.02	463.15	1.05	0.04
VRSI1	2	44.46	1.03	975.54	1.06	0.05	39.72	1.02	849.80	1.04	0.04
VRSI2	2	43.82	1.03	970.01	1.06	0.06	39.11	1.02	845.09	1.04	0.04
VRSI3	2	43.82	1.03	985.63	1.06	0.06	38.52	1.02	845.44	1.04	0.04
VRSI1	5	83.39	1.03	1755.01	1.05	0.05	72.67	1.02	1407.76	1.03	0.03
VRSI2	5	82.94	1.03	1769.38	1.05	0.05	72.68	1.02	1437.63	1.04	0.04
VRSI3	5	82.78	1.03	1766.01	1.05	0.05	72.57	1.02	1430.92	1.04	0.04
VRSI1	10	97.93	1.02	1725.06	1.05	0.05	89.01	1.01	1460.64	1.03	0.03
VRSI2	10	97.67	1.03	1743.90	1.05	0.05	88.96	1.01	1481.99	1.03	0.03
VRSI3	10	97.71	1.03	1760.04	1.05	0.05	88.92	1.01	1472.09	1.03	0.03
VRSI1	20	99.31	1.02	1490.28	1.05	0.05	94.15	1.01	1299.96	1.02	0.02
VRSI2	20	99.12	1.02	1510.19	1.05	0.05	93.72	1.01	1313.02	1.02	0.02
VRSI3	20	99.23	1.02	1519.80	1.05	0.05	94.07	1.01	1318.09	1.02	0.02

Table A.11
Numerical comparisons on the RSI instances with and without delays.

IG	NV	With delay					Without delay				
		SER	LOS	TD	OR	σ^2	SER	LOS	TD	OR	σ^2
RSI1	2	7.33	1.07	1252.29	1.17	0.16	7.50	1.08	1155.21	1.19	0.18
RSI2	2	7.45	1.07	1227.52	1.18	0.19	7.05	1.08	1143.39	1.19	0.19
RSI3	2	7.05	1.08	1223.53	1.20	0.21	7.10	1.09	1175.86	1.21	0.22
RSI1	5	18.06	1.07	2927.67	1.17	0.16	17.53	1.08	2768.46	1.19	0.19
RSI2	5	18.15	1.08	2908.56	1.21	0.21	17.07	1.08	2750.97	1.21	0.21
RSI3	5	17.68	1.08	2911.79	1.20	0.21	17.02	1.09	2772.61	1.22	0.21
RSI1	10	35.57	1.08	5426.14	1.18	0.17	33.84	1.07	5169.75	1.19	0.19
RSI2	10	35.21	1.08	5407.69	1.21	0.21	33.73	1.08	5150.43	1.21	0.21
RSI3	10	34.64	1.08	5447.64	1.22	0.22	33.02	1.08	5214.80	1.21	0.21
RSI1	20	65.69	1.08	9663.80	1.21	0.21	60.83	1.08	8875.69	1.18	0.18
RSI2	20	66.46	1.08	9696.05	1.22	0.23	60.11	1.08	8952.11	1.21	0.21
RSI3	20	65.68	1.08	9697.27	1.22	0.23	59.89	1.08	9014.24	1.21	0.21
RSI1	50	98.90	1.08	12,671.02	1.21	0.22	92.51	1.07	11,788.48	1.17	0.17
RSI2	50	98.88	1.09	12,895.32	1.23	0.23	92.58	1.07	11,884.79	1.18	0.19
RSI3	50	98.94	1.08	12,992.67	1.23	0.24	91.93	1.07	11,906.35	1.19	0.19
RSI1	100	99.79	1.08	11,293.79	1.22	0.23	97.36	1.06	11,236.05	1.16	0.17
RSI2	100	99.72	1.08	11,594.03	1.23	0.24	97.31	1.06	11,396.57	1.18	0.18
RSI3	100	99.69	1.08	11,532.55	1.23	0.24	97.59	1.07	11,499.36	1.18	0.19

Table A.12
Numerical comparisons on the S instances with and without delays.

IG	NV	With delay					Without delay				
		SER	LOS	TD	OR	σ^2	SER	LOS	TD	OR	σ^2
S1	50	16.22	1.15	30,342.32	1.59	0.62	15.80	1.17	28,795.64	1.64	0.66
S2	50	16.61	1.15	30,183.64	1.61	0.62	15.73	1.17	28,725.47	1.69	0.69
S3	50	16.31	1.15	30,404.98	1.62	0.64	15.77	1.18	28,734.72	1.72	0.73
S1	100	33.87	1.16	57,203.26	1.68	0.70	31.94	1.17	54,574.01	1.69	0.71
S2	100	34.31	1.16	56,934.14	1.70	0.71	32.12	1.18	54,923.71	1.74	0.74
S3	100	33.66	1.16	56,582.24	1.69	0.70	32.07	1.18	54,699.18	1.74	0.75
S1	200	67.03	1.16	102,488.82	1.72	0.73	61.30	1.17	98,099.54	1.70	0.71
S2	200	67.27	1.16	102,747.24	1.74	0.74	61.07	1.17	99,213.28	1.72	0.72
S3	200	66.46	1.16	102,302.06	1.74	0.76	60.44	1.17	98,476.01	1.72	0.74

significant on set S where is on average 1.91%. Whereas, it is very significant the OR improvement being on average of 4.50% on VRSI, 11.54% on RSI and 19.50% on S. Moreover, there is a mild decrease of TD, being on average of −1.17% on VRSI, of −2.31% on RSI and −1.02% on S.

A.4. Sensitivity analysis varying the LOS, without delaying the time windows

We also performed a sensitivity analysis by varying the LOS from 1.5 to 2, without delaying the time windows. The corresponding results

Table A.13
Numerical comparisons on the VRSI instances with delays varying the LOS.

IG	NV	With delay and LOS=2					With delay and LOS=1.5				
		SER	LOS	TD	OR	σ^2	SER	LOS	TD	OR	σ^2
VRS1	1	24.27	1.09	549.22	1.10	0.10	23.63	1.03	548.71	1.06	0.05
VRS2	1	24.20	1.10	544.01	1.11	0.10	23.83	1.03	545.14	1.06	0.05
VRS3	1	23.92	1.09	542.62	1.11	0.10	23.48	1.03	544.94	1.05	0.05
VRS1	2	45.11	1.09	969.42	1.11	0.11	44.46	1.03	975.54	1.06	0.05
VRS2	2	44.19	1.09	968.31	1.11	0.11	43.82	1.03	970.01	1.06	0.06
VRS3	2	44.37	1.09	978.73	1.11	0.11	43.82	1.03	985.63	1.06	0.06
VRS1	5	84.21	1.08	1741.45	1.10	0.10	83.39	1.03	1755.01	1.05	0.05
VRS2	5	83.78	1.08	1746.55	1.10	0.10	82.94	1.03	1769.38	1.05	0.05
VRS3	5	83.62	1.08	1745.83	1.10	0.10	82.78	1.03	1766.01	1.05	0.05
VRS1	10	98.11	1.07	1692.05	1.10	0.09	97.93	1.02	1725.06	1.05	0.05
VRS2	10	97.87	1.08	1704.25	1.10	0.10	97.67	1.03	1743.90	1.05	0.05
VRS3	10	97.95	1.08	1715.44	1.10	0.10	97.71	1.03	1760.04	1.05	0.05
VRS1	20	99.37	1.07	1464.38	1.09	0.09	99.31	1.02	1490.28	1.05	0.05
VRS2	20	99.11	1.07	1481.66	1.09	0.09	99.12	1.02	1510.19	1.05	0.05
VRS3	20	99.23	1.07	1491.12	1.09	0.09	99.23	1.02	1519.80	1.05	0.05

Table A.14
Numerical comparisons on the RSI instances with delays varying the LOS.

IG	NV	With delay and LOS=2					With delay and LOS=1.5				
		SER	LOS	TD	OR	σ^2	SER	LOS	TD	OR	σ^2
RSI1	2	7.50	1.19	1240.86	1.29	0.27	7.33	1.07	1252.29	1.17	0.16
RSI2	2	7.76	1.20	1224.01	1.32	0.32	7.45	1.07	1227.52	1.18	0.19
RSI3	2	7.36	1.20	1230.23	1.35	0.34	7.05	1.08	1223.53	1.2	0.21
RSI1	5	18.36	1.20	2942.56	1.32	0.33	18.06	1.07	2927.67	1.17	0.16
RSI2	5	18.39	1.21	2911.12	1.35	0.36	18.15	1.08	2908.56	1.21	0.21
RSI3	5	18.23	1.21	2906.45	1.35	0.36	17.68	1.08	2911.79	1.2	0.21
RSI1	10	36.27	1.20	5412.90	1.33	0.33	35.57	1.08	5426.14	1.18	0.17
RSI2	10	35.99	1.21	5357.96	1.36	0.38	35.21	1.08	5407.69	1.21	0.21
RSI3	10	35.32	1.21	5405.68	1.38	0.39	34.64	1.08	5447.64	1.22	0.22
RSI1	20	68.05	1.20	9586.56	1.35	0.36	65.69	1.08	9663.8	1.21	0.21
RSI2	20	68.43	1.21	9524.40	1.37	0.39	66.46	1.08	9696.05	1.22	0.23
RSI3	20	67.75	1.21	9582.77	1.37	0.38	65.68	1.08	9697.27	1.22	0.23
RSI1	50	99.20	1.20	11,842.28	1.34	0.36	98.9	1.08	12,671.02	1.21	0.22
RSI2	50	99.30	1.21	12,017.59	1.36	0.38	98.88	1.09	12,895.32	1.23	0.23
RSI3	50	99.15	1.20	12,124.97	1.36	0.38	98.94	1.08	12,992.67	1.23	0.24
RSI1	100	99.76	1.18	10,731.91	1.33	0.35	99.79	1.08	11,293.79	1.22	0.23
RSI2	100	99.72	1.19	10,941.92	1.35	0.38	99.72	1.08	11,594.03	1.23	0.24
RSI3	100	99.74	1.19	10,941.61	1.36	0.39	99.69	1.08	11,532.55	1.23	0.24

Table A.15
Numerical comparisons on the S instances with delays varying the LOS.

IG	NV	With delay and LOS=2					With delay and LOS=1.5				
		SER	LOS	TD	OR	σ^2	SER	LOS	TD	OR	σ^2
S1	50	16.70	1.33	30,551.80	1.91	0.88	16.22	1.15	30,342.32	1.59	0.62
S2	50	15.97	1.33	30,108.71	1.90	0.87	16.61	1.15	30,183.64	1.61	0.62
S3	50	16.78	1.34	30,195.59	1.94	0.89	16.31	1.15	30,404.98	1.62	0.64
S1	100	35.91	1.33	56,544.83	2.00	0.92	33.87	1.16	57,203.26	1.68	0.70
S2	100	35.83	1.34	56,282.13	2.04	0.96	34.31	1.16	56,934.14	1.70	0.71
S3	100	35.28	1.34	55,969.94	2.03	0.95	33.66	1.16	56,582.24	1.69	0.70
S1	200	71.56	1.33	100,039.46	2.05	0.95	67.03	1.16	102,488.82	1.72	0.73
S2	200	70.73	1.33	100,701.91	2.07	0.96	67.27	1.16	102,747.24	1.74	0.74
S3	200	70.19	1.33	101,085.93	2.07	0.97	66.46	1.16	102,302.06	1.74	0.76

are reported in Tables A.16–A.18 for the sets VRSI, RSI and S, respectively. In this case, we can observe that SER only slightly improves with an average increase of 0.51% on set VRSI, 0.75% on RSI and 1.57% on S. Also TD mildly improves with an average decrease of −0.50%

Table A.16
Numerical comparisons on the VRSI instances varying the LOS.

IG	NV	LOS=2					LOS=1.5				
		SER	LOS	TD	OR	σ^2	SER	LOS	TD	OR	σ^2
VRSI1	1	21.00	1.08	462.60	1.09	0.08	20.46	1.02	458.67	1.04	0.04
VRSI2	1	20.96	1.09	456.33	1.10	0.09	20.59	1.02	458.17	1.04	0.04
VRSI3	1	20.78	1.09	460.84	1.10	0.10	20.21	1.02	463.02	1.05	0.04
VRSI1	2	39.09	1.07	828.24	1.09	0.09	37.80	1.02	819.05	1.04	0.04
VRSI2	2	38.24	1.08	818.25	1.09	0.09	37.89	1.02	827.58	1.04	0.04
VRSI3	2	37.90	1.08	820.62	1.09	0.09	37.07	1.02	822.05	1.04	0.04
VRSI1	5	69.60	1.06	1333.42	1.07	0.07	68.82	1.02	1334.20	1.03	0.03
VRSI2	5	69.87	1.06	1344.60	1.08	0.08	69.31	1.02	1360.77	1.04	0.03
VRSI3	5	69.82	1.06	1346.63	1.08	0.08	68.85	1.02	1352.54	1.04	0.04
VRSI1	10	87.14	1.05	1422.04	1.06	0.06	86.77	1.02	1428.36	1.03	0.03
VRSI2	10	87.23	1.05	1431.25	1.06	0.06	86.88	1.01	1450.10	1.03	0.03
VRSI3	10	87.28	1.05	1432.99	1.07	0.07	86.83	1.02	1450.01	1.03	0.03
VRSI1	20	93.72	1.04	1332.87	1.05	0.05	93.63	1.01	1343.20	1.04	0.04
VRSI2	20	93.29	1.04	1343.88	1.05	0.05	93.18	1.01	1357.50	1.04	0.04
VRSI3	20	93.61	1.04	1347.53	1.05	0.06	93.54	1.01	1360.66	1.05	0.04

Table A.17
Numerical comparisons on the RSI instances varying the LOS.

IG	NV	LOS=2					LOS=1.5				
		SER	LOS	TD	OR	σ^2	SER	LOS	TD	OR	σ^2
RSI1	2	7.61	1.22	1152.21	1.31	0.31	7.50	1.08	1155.21	1.19	0.18
RSI2	2	7.10	1.24	1132.91	1.36	0.36	7.05	1.08	1143.39	1.19	0.19
RSI3	2	7.28	1.23	1155.88	1.35	0.34	7.10	1.09	1175.86	1.21	0.22
RSI1	5	18.08	1.22	2752.40	1.34	0.35	17.53	1.08	2768.46	1.19	0.19
RSI2	5	17.61	1.23	2708.83	1.38	0.38	17.07	1.08	2750.97	1.21	0.21
RSI3	5	17.85	1.23	2778.30	1.36	0.36	17.02	1.09	2772.61	1.22	0.21
RSI1	10	34.29	1.21	5130.28	1.34	0.34	33.84	1.07	5169.75	1.19	0.19
RSI2	10	34.21	1.22	5097.81	1.36	0.36	33.73	1.08	5150.43	1.21	0.21
RSI3	10	34.15	1.22	5179.15	1.36	0.37	33.02	1.08	5214.80	1.21	0.21
RSI1	20	61.89	1.20	8836.15	1.33	0.34	60.83	1.08	8875.69	1.18	0.18
RSI2	20	61.96	1.20	8889.16	1.36	0.37	60.11	1.08	8952.11	1.21	0.21
RSI3	20	62.11	1.21	8899.38	1.35	0.37	59.89	1.08	9014.24	1.21	0.21
RSI1	50	93.93	1.17	11,356.86	1.30	0.31	92.51	1.07	11,788.48	1.17	0.17
RSI2	50	93.68	1.17	11,367.82	1.32	0.34	92.58	1.07	11,884.79	1.18	0.19
RSI3	50	93.31	1.17	11,575.28	1.31	0.34	91.93	1.07	11,906.35	1.19	0.19
RSI1	100	97.55	1.15	10,813.64	1.27	0.30	97.36	1.06	11,236.05	1.16	0.17
RSI2	100	97.30	1.15	10,942.33	1.29	0.32	97.31	1.06	11,396.57	1.18	0.18
RSI3	100	97.62	1.15	11,070.73	1.30	0.33	97.59	1.07	11,499.36	1.18	0.19

Table A.18
Numerical comparisons on the S instances varying the LOS.

IG	NV	LOS=2					LOS=1.5				
		SER	LOS	TD	OR	σ^2	SER	LOS	TD	OR	σ^2
S1	50	15.52	1.37	28,820.58	1.96	0.88	15.80	1.17	28,795.64	1.64	0.66
S2	50	15.92	1.37	28,578.56	2.00	0.92	15.73	1.17	28,725.47	1.69	0.69
S3	50	15.80	1.36	28,615.27	2.04	0.96	15.77	1.18	28,734.72	1.72	0.73
S1	100	33.24	1.36	54,307.57	2.03	0.96	31.94	1.17	54,574.01	1.69	0.71
S2	100	32.70	1.36	54,011.17	2.04	0.93	32.12	1.18	54,923.71	1.74	0.74
S3	100	33.63	1.36	54,363.35	2.09	0.97	32.07	1.18	54,699.18	1.74	0.75
S1	200	65.22	1.34	96,742.18	2.03	0.97	61.30	1.17	98,099.54	1.70	0.71
S2	200	64.61	1.34	97,027.13	2.06	0.97	61.07	1.17	99,213.28	1.72	0.72
S3	200	63.75	1.34	96,773.86	2.07	0.99	60.44	1.17	98,476.01	1.72	0.74

on VRSI, −1.77% on RSI and −0.99% on S. Whereas, OR significantly improves with an average increase of 3.58% on VRSI, 11.76% on RSI and 19.14% on S.

References

Agatz, N., Erera, A., Savelsbergh, M., Wang, X., 2012. Optimization for dynamic ride-sharing: A review. European J. Oper. Res. 223 (2), 295–303.

Akimoto, K., Sano, F., Oda, J., 2022. Impacts of ride and car-sharing associated with fully autonomous cars on global energy consumptions and carbon dioxide emissions. Technol. Forecast. Soc. Change 174, 121311.

Amey, A., Attanucci, J., Mishalani, R., 2011. Real-time ridesharing: opportunities and challenges in using mobile phone technology to improve rideshare services. Transp. Res. Rec. 2217 (1), 103–110.

Beirigo, B.A., Negenborn, R.R., Alonso-Mora, J., Schulte, F., 2022. A business class for autonomous mobility-on-demand: Modeling service quality contracts in dynamic ridesharing systems. Transp. Res. C 136, 103520.

Berbeglia, G., Cordeau, J.-F., Laporte, G., 2010. Dynamic pickup and delivery problems. European J. Oper. Res. 202 (1), 8–15.

Bongiovanni, C., Kaspi, M., Cordeau, J.-F., Geroliminis, N., 2022. A machine learning-driven two-phase metaheuristic for autonomous ridesharing operations. Transportation Research Part E: Logistics and Transportation Review 165, 102835.

Bruglieri, M., Ciccarelli, D., Colorni, A., Luè, A., 2011. PoliUniPool: A carpooling system for universities. Procedia-Soc. Behav. Sci. 20, 558–567.

Carticipate, https://www.carticipate.com.

Center for Sustainable Systems, C.S.S., 2022. Personal Transportation Factsheet. Technical Report, vol. CSS01-07, University of Michigan.

Duarte, F., Ratti, C., 2018. The impact of autonomous vehicles on cities: A review. J. Urban Technol. 25 (4), 3–18.

Elting, S., Ehmke, J.F., 2021. Potential of shared taxi services in rural areas–A case study. Transp. Res. Procedia 52, 661–668.

European Environment Agency, E.E.A., 2020. Are We Moving in the Right Direction? Indicators on Transport and Environmental Integration in the EU: TERM 2000. Technical report, URL https://www.eea.europa.eu/publications/ENVISSUENo12/page029.html.

Fagnant, D.J., Kockelman, K.M., 2018. Dynamic ride-sharing and fleet sizing for a system of shared autonomous vehicles in Austin, Texas. Transportation 45 (1), 143–158.

Faisal, A., Kamruzzaman, M., Yigitcanlar, T., Currie, G., 2019. Understanding autonomous vehicles. J. Transp. Land Use 12 (1), 45–72.

Furuhata, M., Dessouky, M., Ordóñez, F., Brunet, M.-E., Wang, X., Koenig, S., 2013. Ridesharing: The state-of-the-art and future directions. Transp. Res. B 57, 28–46.

Haferkamp, J., Ehmke, J.F., 2022. Effectiveness of demand and fulfillment control in dynamic fleet management of ride-sharing systems. Networks 79 (3), 314–337.

Hyland, M., Mahmassani, H.S., 2020. Operational benefits and challenges of shared-ride automated mobility-on-demand services. Transp. Res. Part A: Policy Pract. 134, 251–270. http://dx.doi.org/10.1016/j.tra.2020.02.017, URL https://www.sciencedirect.com/science/article/pii/S0965856419307888.

Jones, E.C., Leibowicz, B.D., 2019. Contributions of shared autonomous vehicles to climate change mitigation. Transp. Res. Part D: Transp. Environ. 72, 279–298.

Lokhandwala, M., Cai, H., 2018. Dynamic ride sharing using traditional taxis and shared autonomous taxis: A case study of NYC. Transp. Res. C 97, 45–60.

Lotfi, S., Abdelghany, K., 2022. Ride matching and vehicle routing for on-demand mobility services. J. Heuristics 1–24.

Mao, C., Liu, Y., Shen, Z.-J.M., 2020. Dispatch of autonomous vehicles for taxi services: A deep reinforcement learning approach. Transp. Res. C 115, 102626.

Melis, L., Sorensen, K., 2022. The real-time on-demand bus routing problem: The cost of dynamic requests. Comput. Oper. Res. 147, 105941. http://dx.doi.org/10.1016/j.cor.2022.105941, URL https://www.sciencedirect.com/science/article/pii/S0305054822001940.

Najmi, A., Rey, D., Rashidi, T.H., 2017. Novel dynamic formulations for real-time ride-sharing systems. Transp. Res. Part E: Logist. Transp. Rev. 108, 122–140.

Noruzoliaee, M., Zou, B., 2022. One-to-many matching and section-based formulation of autonomous ridesharing equilibrium. Transp. Res. B 155, 72–100.

Pillac, V., Gendreau, M., Guéret, C., Medaglia, A.L., 2013. A review of dynamic vehicle routing problems. European J. Oper. Res. 225 (1), 1–11.

Pouls, M., Meyer, A., Glock, K., 2021. Real-time dispatching with local search improvement for dynamic ride-sharing. In: International Conference on Computational Logistics. Springer, pp. 299–315.

Santos, D.O., Xavier, E.C., 2015. Taxi and ride sharing: A dynamic dial-a-ride problem with money as an incentive. Expert Syst. Appl. 42 (19), 6728–6737.

Spielberg, F., Shapiro, P., 2000. Mating habits of slugs: Dynamic carpool formation in the I-95/I-395 corridor of northern virginia. Transp. Res. Rec. 1711 (1), 31–38.

Teal, R.F., 1987. Carpooling: who, how and why. Transp. Res. Part A: Gen. 21 (3), 203–214.

Toth, P., Vigo, D., 2002. An overview of vehicle routing problems. Veh. Routing Probl. 1–26.

Turan, B., Pedarsani, R., Alizadeh, M., 2020. Dynamic pricing and fleet management for electric autonomous mobility on demand systems. Transp. Res. C 121, 102829.

van Engelen, M., Cats, O., Post, H., Aardal, K., 2018. Enhancing flexible transport services with demand-anticipatory insertion heuristics. Transp. Res. Part E: Logist. Transp. Rev. 110, 110–121. http://dx.doi.org/10.1016/j.tre.2017.12.015, URL https://www.sciencedirect.com/science/article/pii/S1366554517307810.

Vansteenwegen, P., Melis, L., Aktaş, D., Galarza Montenegro, B.D., Sartori Vieira, F., Sörensen, K., 2022. A survey on demand-responsive public bus systems. Transp. Res. C 137, 103573.

Zhang, C., Xie, J., Wu, F., Gao, X., Chen, G., 2018. Algorithm designs for dynamic ridesharing system. In: International Conference on Algorithmic Applications in Management. Springer, pp. 209–220.

Zhao, M., Yin, J., An, S., Wang, J., Feng, D., 2018. Ridesharing problem with flexible pickup and delivery locations for app-based transportation service: Mathematical modeling and decomposition methods. J. Adv. Transp. 2018.

Zheng, M., Pantuso, G., 2023. Trading off costs and service rates in a first-mile ride-sharing service. Transp. Res. C 150, 104099.