Prepositioning can improve the performance of a dynamic stochastic on-demand public bus system

Prepositioning can improve the performance of a dynamic stochastic on-demand public bus system

Ying Lian[a,*], Flavien Lucas[b], Kenneth Sörensen[a]

[a] *ANT/OR - Operations Research Group, Department of Engineering Management, University of Antwerp, Belgium*
[b] *IMT Nord Europe, Institut Mines-Télécom, Univ. Lille, Centre for Digital Systems, F-59000 Lille, France*

Abstract

This paper presents the first application of *prepositioning* in the context of the dynamic stochastic on-demand bus routing problem (DODBRP). The DODBRP is a large-scale dial-a-ride problem that involves bus station assignment and aims to minimize the total user ride time (URT) by simultaneously assigning passengers to alternative stations and determining optimal bus routes.

In the DODBRP, transportation requests are introduced dynamically, and buses are dispatched to stations with known requests. This paper investigates the concept of prepositioning, which involves sending buses not only to currently known requests but also to requests that are likely to appear in the future, based on a given probability.

To solve this dynamic and stochastic ODBRP, the paper proposes a heuristic algorithm based on variable neighborhood search (VNS). The algorithm considers multiple scenarios to represent different realizations of the stochastic requests.

Experimental results demonstrate the superiority of the prepositioning approach over the DODBRP across various levels of forecast accuracy, lengths of time bucket, and probabilities of realization. Furthermore, the paper shows that removing empty stations as a recourse action can further enhance solution quality. Additionally, in situations with low prediction accuracy, increasing the number of scenarios can lead to improved solutions. Finally, a combination of prepositioning, empty station removal, and the insertion of dynamic requests proves to be effective.

Overall, the findings of this paper provide valuable insights into the application of prepositioning in the dynamic stochastic on-demand bus routing problem, highlighting its potential for addressing real-world transportation challenges.

Keywords: routing, on-demand bus, stochastic requests, dynamic requests, prepositioning

1. Introduction

Public transit systems traditionally fall into two categories: line-based, scheduled public transport—exemplified by most bus services—and flexible, on-demand transport—like taxi services. Scheduled buses operate on fixed routes and timetables, offering cost efficiency and environmental

*Corresponding author.
Email addresses: ying.lian@uantwerpen.be (Ying Lian), flavien.lucas@imt-nord-europe.fr (Flavien Lucas), kenneth.sorensen@uantwerpen.be (Kenneth Sörensen)

friendliness, yet they necessitate passengers to tailor their travel plans to align with these fixed parameters. Conversely, taxis offer flexibility regarding routes and schedules, but impose a significant expense on both passengers and the environment.

Recently, on-demand public transit services have gained traction as a cost-effective and adaptable solution. Historically, these services primarily catered to people with disabilities and the elderly, but the proliferation of advanced technologies (such as mobile location devices) and environmental concerns have spurred interest in their application to broader public transportation. Academically, these vehicle routing and scheduling challenges have typically been modeled as a variant of the dial-a-ride problem (DARP), which aims to transport passengers from a point of origin to a destination (door-to-door transportation). However, in an urban setting, it is not feasible to pick up or drop off passengers at any arbitrary location, limiting stops to designated bus stations. Hence, the ODBRP incorporates bus station assignment (BSA), wherein each passenger has multiple alternative boarding and alighting stations (within a maximum walking distance from their origin and destination). The algorithm then determines the specific stations for each passenger's pick-up and drop-off, allowing for passenger pooling and improving the total user ride time (URT) significantly (Melis and Sörensen, 2022b). URT refers to the duration a passenger spends on board—the time difference between arrival at the drop-off station and departure from the pick-up station.

Within the ODBRP framework, each passenger creates a *request*, consisting of potential departure and arrival stations, coupled with a desired time window—the earliest pick-up time and the latest drop-off time.

Existing literature on the ODBRP only directs buses to stations assigned to actual, received requests. However, it might prove advantageous to dispatch buses to stations where *future requests* are highly probable, termed as *potential requests* in this paper. We refer to the action of dispatching buses to potential request locations as *prepositioning*.

This paper develops a heuristic algorithm to address the ODBRP with prepositioning, leveraging information on stochastic potential requests within the dynamic and stochastic ODBRP context. Our approach is inspired by the advent of accurate demand forecasts, such as those by Mariñas-Collado et al. (2022); Liyanage et al. (2022); Banerjee et al. (2020).

In our prepositioning model, certain requests—static and deterministic—are known in advance, while others are dynamic and stochastic, holding a specific probability of future realization. Under this problem setting, instead of merely reacting to stochastic and dynamic requests once they are realized, we proactively plan bus routes and schedules based on the potential requests' probabilities.

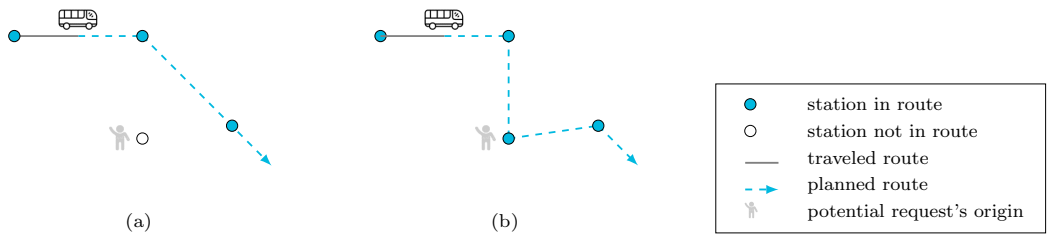

Figure 1: Conceptual illustration of prepositioning

The primary objective of this paper is to explore the benefits of *prepositioning*—dispatching vehicles to stations with potential requests. A conceptual illustration of prepositioning is shown in

Figure 1: (a) detection of a potential request is followed by (b) the proposed algorithm's attempt to devise a routing and scheduling solution for it. In the dynamic on-demand bus routing problem (DODBRP) *without* prepositioning, the potential request in (a) is overlooked, and bus rerouting only transpires once the request is realized.

Our methodology for evaluating the benefits of prepositioning closely resembles that of Li et al. (2019), who investigate prepositioning within the context of a door-to-door dial-a-ride problem. Like Li et al. (2019), we introduce multiple scenarios to represent possible realizations of the stochastic request. However, our approach differs from the approach of Li et al. (2019) in several ways. First, we apply prepositioning to the ODBRP, and not the DARP. In contrast to the DARP, the ODBRP naturally has a discrete and finite set of locations at which stochastic requests may appear (i.e., the bus stations). Second, bus station assignment, a feature exclusive to the ODBRP, adds an additional layer of complexity not present in the DARP. By comparing our algorithm with and without bus station assignment, we can investigate its benefits in the context of a stochastic dynamic ODBRP with prepositioning. Third, unlike Li et al. (2019), we do not consider time-dependent or stochastic travel times in this paper.

In addition to the above, our research offers several further contributions when juxtaposed with existing literature, including the modeling of realization time, the introduction of a recourse action, and a comprehensive sensitivity analysis. We model realization time as a normal distribution within a specified interval. Moreover, we introduce a recourse action—removal of empty stations (those with no realized stochastic requests) after each time bucket. The benefits of this recourse action are explored by contrasting our algorithm with and without its implementation. Additionally, a hybrid approach combining dynamic insertion is also tested. Finally, we conduct a quantitative assessment of the proposed strategy's effectiveness across various forecast accuracy levels, time bucket lengths, and realization probabilities. Specifically, we determine the optimal time bucket length as a function of forecast accuracy and probability.

We generate and use artificial yet realistic instances to evaluate the algorithm's performance and conduct a sensitivity analysis to ascertain the impact of different algorithmic settings on the objective function value. A series of large-scale experiments demonstrate the superiority of prepositioning over the pure DODBRP and its robustness across a range of parameter values (instance sizes, stochastic request probabilities, time bucket lengths). Lastly, our algorithms' performance is validated under inaccurate estimation of stochastic requests, and potential solutions to further augment solution quality are assessed.

In summary, this paper's main *contributions*, ranked in order of importance, are as follows.

1. This study represents the first application of the concept of *prepositioning* to the *on-demand bus routing problem*.
2. We assess the advantages of *bus station assignment*, a unique characteristic of the ODBRP, within the context of prepositioning.
3. Unlike previous research, we model dynamic stochastic requests as occurring within a *time interval* rather than at a fixed point in time. The realization time of these requests is represented by a normal distribution. Additionally, we investigate the impact of the interval size, which has not been explored in previous literature.
4. We propose a *recourse action* that involves removing empty stations after each time bucket. Furthermore, we examine a hybrid solution combining prepositioning, removal of empty stations, and dynamic insertion (referred to as the DODBRP). Such a recourse action has not been suggested in the existing literature within a similar context.

5. We *quantitatively evaluate the effectiveness of the proposed approach* through numerical testing under various levels of forecast accuracy, time bucket durations, and probabilities of request realization. Specifically, we examine the optimal time bucket length in conjunction with forecast accuracy and probability. In contrast, the literature lacks quantitative studies on the sensitivity analyses of these variables.

The remainder of this paper is structured as follows. Section 2 delivers a literature review. Subsequently, Sections 3 and 4 outline the problem under study and the solution method, respectively. Section 5 then presents the computational experiments and their results. Finally, Section 6 concludes the paper and discusses potential avenues for future research.

2. Literature review

The dynamic on-demand bus routing problem bears a significant relationship to the dial-a-ride problem, a well-known variant of the vehicle routing problem (VRP). The DARP, in general, is a model applied to establish vehicle routes and schedules that effectively transport passengers from their respective starting points to their desired destinations. Each involved vehicle begins and ends its journey at a depot. The DODBRP deviates from the dynamic DARP, notably in the provision that passengers are allocated to one among several possible bus stations for both boarding and alighting purposes.

In the mathematical formulation of the DARP, the most frequent objective function aims to minimize operational costs such as total travel distance or duration, while being subjected to certain constraints. These constraints are related to passengers' time windows, maximum duration or detour, vehicle capacity, coupling, and precedence. The last two constraints enforce that each passenger boards and alights from the same bus, and that alighting occurs after boarding. The (D)ODBRP presents a subtle distinction from the DARP in that the objective is typically to minimize the *total user ride time*. This essentially implies that any trips that do not involve passengers (for instance, trips to and from the depot) are disregarded and thus are not modeled.

The DARP can be categorized as static or dynamic and as deterministic or stochastic, depending on whether the information is known and the certainty of the information when computing a solution. When all requests are known beforehand, the problem is labeled as *static*. However, in the *dynamic* case, some requests are revealed as execution progresses, necessitating real-time adjustments of solutions. If request information is known with certainty, the problem is *deterministic*; otherwise, it is *stochastic*. Dynamic and/or stochastic variants of the DARP have been less examined in comparison to static or deterministic variants. For comprehensive reviews of all variants of the DARP, we refer to Cordeau and Laporte (2007); Molenbruch et al. (2017); Ho et al. (2018).

2.1. Dynamic DARP

The dynamic DARP presents a situation where requests arrive in real time as execution continues. Consequently, an updated solution is required to accommodate these new requests. Periodic re-optimization is a standard approach used to convert a dynamic DARP to its static version (Pillac et al., 2013). This allows the dynamic DARP to be broken down into a series of static DARPs. The dynamic DARP, depending on the trigger for the update, progresses along the time axis either by event (upon arrival of a new request or a vehicle reaching a station) (Wong et al., 2014; Hanne et al., 2009; Berbeglia et al., 2012; Marković et al., 2015; Santos and Xavier, 2015; Melis and Sörensen, 2022a), or by a predetermined duration. The latter frequently employs the use of a rolling horizon

(Yang et al., 1999; Luo and Schonfeld, 2011; Gaul et al., 2021). The dynamic DARP necessitates the development of techniques for managing real-time requests with short response times while simultaneously providing high-quality solutions. Consequently, the strategy of employing an insertion and an improvement heuristic is standard, as demonstrated by Attanasio et al. (2004); Coslovich et al. (2006); Beaudry et al. (2010); Lois and Ziliaskopoulos (2017). Nonetheless, exact methods for the static DARP (Cordeau, 2006; Gaul et al., 2022), extended to the dynamic case, have also been suggested.

2.2. Stochastic DARP

A variant of the DARP, known as the stochastic DARP, models stochastic information to anticipate uncertain events and generate solutions with superior performance. The literature reflects the exploration of different types of stochastic information, such as vehicle failures, accidents, traffic jams, stochastic delays at pickup locations, drop-off times, stochastic travel speeds, and cancellation of requests (Issaoui et al., 2013; Heilporn et al., 2011; Maalouf et al., 2014; Schilde et al., 2014; Hyytiä et al., 2010; Xiang et al., 2008; Paquay et al., 2020; Lu et al., 2022). To solve these variants, different methods have been proposed, such as two-stage stochastic programming, fuzzy logic control, and local search (LS) heuristics.

Regarding stochastic *requests*, various models and solution methods have been proposed.

Ichoua et al. (2006) leverage knowledge about future demands to demonstrate that a vehicle waiting at the position of its last request can be beneficial if new nearby requests have a high probability of realization. They develop a parallel tabu search heuristic to solve this problem. Ho and Haugland (2011) examine a probabilistic DARP where each user requires service only with a given probability and the goal is to minimize the expected travel cost. They propose a time-efficient local search and a tabu search procedure to solve this problem. In a DARP with stochastic return transports, Schilde et al. (2011) assess the benefit of using stochastic information to evaluate candidate solutions. They compare different heuristics and conclude that integrating stochastic information about return transports can positively impact solution quality under specific conditions.

Recently, Lowalekar et al. (2018) address a large-scale online spatiotemporal matching problem of taxis and customers. They propose a multi-stage stochastic optimization formulation and demonstrate the superiority of incorporating future demand. Moreover, Li et al. (2019) propose an algorithm to dispatch vehicles to potential requests under stochastic traffic conditions. They use scenarios to represent different samples of future requests and traffic conditions. Experiments with real-world data indicate that prepositioning can enhance average profit and decrease waiting time. As an alternate method for dealing with stochastic requests, Tafreshian et al. (2021) generate a large pool of routes that might be valuable under different demand realizations in a large-scale DARP system. In the online phase, shuttles are proactively routed based on the results from the offline phase. Moreover, the online phase enables shuttles to switch between different routes in the pool, allowing them to react to stochastic changes in travel patterns, where the patterns are determined by a probability density function learned from historical data.

2.3. Solving the DARP with variable neighborhood search (VNS)

The variable neighborhood search (VNS) metaheuristic framework is among the most prominent methods applied to solve the DARP. This subsection provides a brief overview of the literature on this topic.

One of the first approaches is proposed by Parragh et al. (2010), who use three types of neighborhoods to solve the DARP: the first neighborhood utilizes simple swap operations tailored for the DARP; the second neighborhood is based on an ejection chain operator; and the third neighborhood employs the idea of zero split. The approach leads to new best results for 16 out of 20 benchmark instances.

To solve a large-scale DARP, Muelas et al. (2015) propose a distributed VNS algorithm that partitions requests and merges routes. Recently, Johnsen and Meisel (2022) use adaptive VNS to solve a rural DARP with interrelated trips and autonomous vehicles. Specifically, they use roulette wheel selection with adaptive weight adjustment for the selection of removal and insertion heuristics.

Hybrid VNS algorithms have also been proposed, such as the evolutionary variable neighborhood search algorithms for solving the DARP with time windows (Belhaiza, 2017) and the DARP with electric vehicles, battery swapping stations, and a realistic energy consumption function (Masmoudi et al., 2018). Additionally, the DARP has been solved using a combined VNS and constraint programming algorithm (Vamsi Krishna Munjuluri et al., 2023).

The problem examined in this paper, the ODBRP with prepositioning, bears similarities to the dynamic stochastic DARP in Schilde et al. (2011) and the DARP with prepositioning proposed by Li et al. (2019). However, several crucial differences exist. Generally, bus station assignment is a significant aspect of the (D)ODBRP, which has been shown to significantly improve the solution compared to door-to-door transport in the DARP (Melis and Sörensen, 2022b). In Schilde et al. (2011), prepositioning is not considered; instead, the authors only model stochastic requests to evaluate solution quality without inserting corresponding stations. In comparison to Li et al. (2019), our study explicitly investigates whether to adopt the recourse action (removal of empty stations). Additionally, in our problem setting, the realization time of stochastic requests is modeled as either fixed or following a normal distribution. Moreover, we examine a potential combination of prepositioning, the recourse action, and the dynamic solution. We also numerically model forecast accuracy and test its corresponding impact. Finally, we investigate whether the key property of the ODBRP, i.e., BSA, still contributes to the solution quality when prepositioning is implemented.

3. Problem description

This paper examines the on-demand bus routing problem (ODBRP) with prepositioning, a dynamic and stochastic passenger transportation problem tackled in two steps. In the first step, the static version of the ODBRP is tackled, encompassing bus routing and scheduling, passenger-bus assignment, and bus station assignment. The aim of this step is to transport passengers with the minimum total user ride time.

In the second step, solutions are adjusted in real time to accommodate dynamic and stochastic requests. Specifically, stations catering to stochastic requests (modeled in scenarios) can be incorporated into bus routes, and the corresponding bus schedules can be adjusted accordingly. In the following, we explain the static ODBRP in Section 3.1, and the prepositioning approach in Section 3.2.

All the abbreviations, acronyms, notations, symbols and variables used in this paper are listed in Table 1.

3.1. The first step: the static ODBRP

The initial step of our approach utilizes the static ODBRP as modeled in Melis and Sörensen (2022b), and relies on the mathematical formulation in Appendix A. The objective function aims

Table 1: Acronyms, variables and symbols used in this paper

Acronyms			
BSA	bus station assignment	ODBRP	on-demand bus routing problem
DARP	dial-a-ride problem	SMC	simple matching coefficient
DODBRP	dynamic on-demand bus routing problem	URT	user ride time
LNS	large neighborhood search	VNS	variable neighborhood search
LS	local search	VRP	vehicle routing problem
Indices			
s	node, i.e., station	t	time
k	rank number of station in bus route	n	scenario
p	passenger	τ	time bucket
b	bus		
Decision variables / Variables			
x_{skb}	1 if the k-th station of bus b is bus station s and 0 otherwise		
y^u_{pkb}	1 if passenger p is picked up at the k-th station of bus b and 0 otherwise		
y^o_{pkb}	1 if passenger p is dropped off at the k-th station of bus b and 0 otherwise		
q_{kb}	net number of passengers picked up (or dropped off) at the k-th station of bus b		
t^a_{kb}	arrival time of bus b at its k-th station		
t^d_{kb}	departure time of bus b at its k-th station		
T_p	user ride time of passenger p		
obj_n	objective value of scenario n		
$\overline{obj}_n(\tau)$	average objective value among N scenarios		
Sets			
B	fleet of buses, $\|B\|$ denotes the number of buses		
P	set of transportation requests, $\|P\|$ denotes the number of requests		
S	set of bus stations, $\|S\|$ denotes the number of stations		
A	set of arcs, i.e., paths between stations		
R_{sta}	set of static requests		
$R_d(t)$	set of dynamic requests revealed at time t		
$S(t)$	set of scenarios		
Parameters			
Q	capacity of bus		
e_p	earliest pick-up time for passenger p		
l_p	latest drop-off time for passenger p		
a^u_{ps}	1 if passenger p can be assigned to station s for pick-up and 0 otherwise		
a^o_{ps}	1 if passenger p can be assigned to station s for drop-off and 0 otherwise		
f	parameter that controls the length of time windows		
$TT_{dir,p}$	direct travel time of passenger p from the origin to the destination		
H	duration of time buckets		
$prob$	realization probability of stochastic requests		
N	number of scenarios		
σ	number of solutions in the LNS algorithm		
$TT_{ss'}$	travel time between station s and station s'		
M	a sufficiently large constant		

to minimize the total URT, constrained by time windows, bus capacity, precedence, and coupling. This subsection offers a textual explanation of the problem to facilitate comprehension.

In the ODBRP, passengers submit travel requests through a mobile application or website. These requests can be made in advance—e.g., one day prior to their trip—or in real time. The former category of passengers (or requests) are termed static, while the latter are called dynamic. The service of static passengers is guaranteed, whereas dynamic passengers may be declined service if either the time windows or bus capacity constraints are violated.

The problem is formally described as follows.

Consider $G = (S, A)$, a directed graph where S represents the set of nodes and A is the set of arcs. Each node in set S corresponds to a predefined station s. Contrary to most transportation problems, we do not model a depot node explicitly. Instead, each bus commences its journey from its first passenger's pickup station and concludes at its last passenger's drop-off station, ignoring any route from or to a depot. Each node $s \in S$ can be visited multiple times (even by the same bus) or not at all. For each arc $(s, s') \in A$, the travel time $TT_{ss'}$ remains constant over time.

A homogeneous fleet of buses, each with a finite capacity Q, is available for the operation.

Passengers are distributed geographically within a service area, each with a specific origin and destination. This allows us to assign to each request all stations within walking distance. It should be noted that the locations of origins and destinations do not belong to S, as the ODBRP does not deal with door-to-door transportation. In our problem setting, each passenger is guaranteed at least one station to board and alight. Let R_{sta} represent the set of static requests known in advance, and $R_d(t)$ the set of dynamic and stochastic requests revealed over time. Specifically, a dynamic request $p \in R_d(t)$ is revealed at time e_p. Given that each dynamic request simply requires a ride as soon as possible, e_p becomes the earliest permitted pickup time, although the requests do not need to be picked up precisely at e_p. However, in practice, dynamic requests can also require a ride later than e_p, and the impact of the gap between the received time and the earliest permitted pickup time warrants future investigation. The latest arrival time l_p is then calculated as described in the next paragraph. For simplicity, each request corresponds to one passenger. The binary indicators a^u_{ps} and a^o_{ps} respectively denote whether passenger p can be assigned to station s for pickup and drop-off.

In the literature, for instance in Cordeau and Laporte (2003), a general formulation of time windows distinguishes inbound or outbound requests, as well as defines separate time windows for pickup and drop-off, alongside a constraint for the maximum riding time. However, in the proposed model, each request has a single hard time window of earliest departure and latest arrival $[e_p, l_p]$. The duration of each time window is determined by $f \cdot TT_{dir,p}$, where $f \geq 1$ is a constant, and $TT_{dir,p}$ is the direct travel time from origin to destination. The value of e_p is randomly generated and l_p is subsequently calculated as $l_p = e_p + f \cdot TT_{dir,p}$. Given that any alternative station is within walking distance of the passengers' origins and destinations, the difference in travel time between various combinations is deemed negligible. The parameter f controls the strictness of the time windows. A small value leads to a stringent time window and a large value to a broad time window.

The arrival and departure time of each bus b at the k-th station are respectively denoted as t^a_{kb} and t^d_{kb}.

The sequence of stations that each bus b must visit is a decision variable. Each station in the sequence is visited only when necessary, i.e., when at least one passenger needs to board or alight. The assignment of a bus to each passenger, i.e., which bus will serve each passenger, is another decision that the ODBRP has to make.

3.2. The second step: prepositioning with dynamic and stochastic requests

As described in the previous subsection, the initial solution with static requests is first constructed. Then, in real time, the ODBRP with dynamic and stochastic requests, along with prepositioning, is solved. The concept of prepositioning in passenger transportation was previously introduced by Li et al. (2019). Later in this subsection, we will clarify and contrast this approach with our own.

To differentiate prepositioning from the DODBRP, we begin with graphic illustrations of both cases. In the DODBRP (Figure 2), the bus initially follows the planned route (Figure 2a). When a dynamic request emerges, the algorithm is triggered to find a feasible routing and scheduling solution to serve it. Immediate alteration of the bus route from its current position is not permitted (as shown in Figure 2b), but rather only after visiting its next station (Figure 2c). The reason for keeping the next station in the solution unchanged is the assumption that the driver, being human, cannot be expected to constantly monitor the next stop the bus is supposed to drive to. The rule that keeps the next station unchanged applies throughout the second step in this paper, namely, it holds for prepositioning and the recourse action (which will be explained later in this subsection) as well. In the case of autonomous buses, the route could potentially be changed immediately, but this complicates the model (as buses do not only drive between a finite set of stops but also between an infinite number of positions in between stops) and is left for future research.

In the prepositioning case (Figure 3), the algorithm is triggered the moment a potential request is detected (Figure 3a). As shown in Figure 3b, a bus can already be en route to pick up the corresponding passenger when the request is actually realized, resulting in a more efficient bus route (Figure 3c).

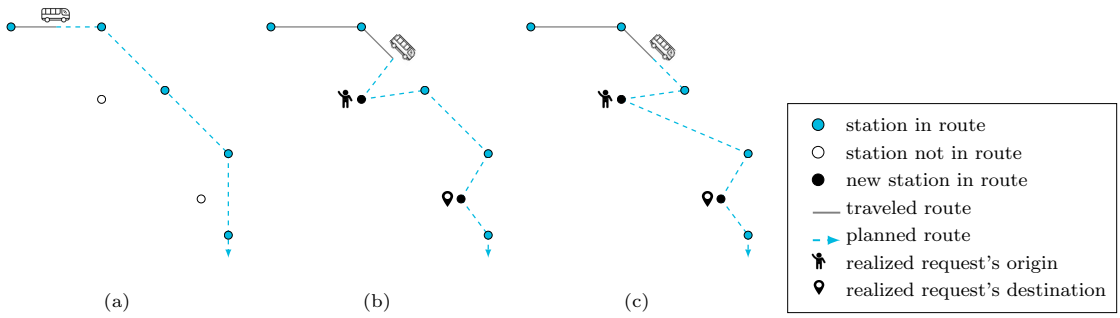

Figure 2: Procedure of the DODBRP

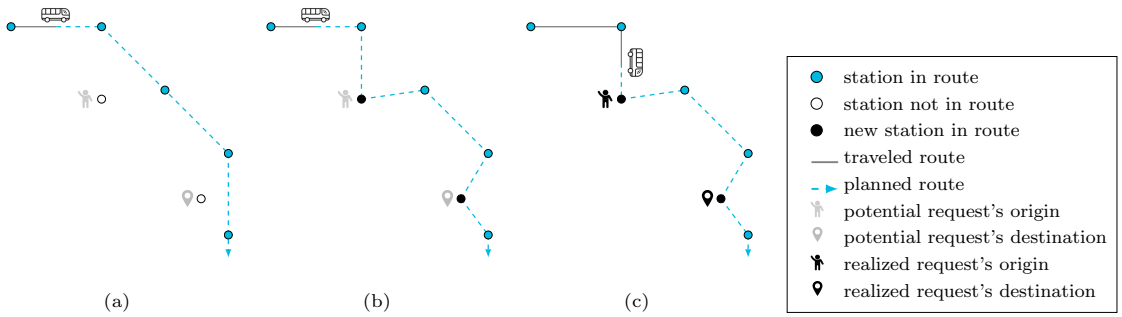

Figure 3: Procedure of prepositioning

To implement prepositioning, we first discretize the time horizon into an ordered set $\tau = \{0, 1, 2, ..., T\}$ of time buckets, each with a constant duration H, for example, 10 minutes. Scenarios are used to represent samples of stochastic requests. $S(t) = \{S_1(t), S_2(t), ..., S_N(t)\}$ is the set of scenarios used at time t_τ, each containing the dynamic requests that appear in the time bucket $[t_\tau, t_{\tau+1})$. Additionally, a constant value N denotes the number of scenarios, which is fixed for all time buckets. Each dynamic and stochastic request has a probability of appearing in each scenario.

Scenarios can be generated based on historical data or a request forecasting model in practice. In this study, each stochastic request is assigned a probability $prob \in (0, 1)$, and the probability for a request to appear in the set of scenarios $S(t)$ is independently and identically distributed (i.i.d.). In other words, the expected number of scenarios containing this request is $prob \cdot N$. Unlike Li et al. (2019), where the scenario probabilities are unspecified, we numerically model the probabilities and conduct a sensitivity analysis on these probabilities in Section 5.2.

In addition to modeling the probability of the scenarios, we propose two extensions to existing models: the modeling of realization time and the recourse action.

The realization time of stochastic requests is modeled according to two distinct models. In the first model, the appearance of a stochastic request is stochastic, but the time at which it will be realized (if at all) is deterministic. In other words, if a request is not sent at a specific time point, it will not be sent later on either. Therefore, for all the scenarios containing this request, the information remains the same, including its time window, a^u_{ps}, and a^o_{ps}. Given this assumption, a reasonable recourse action after each time bucket is to remove the stations that will not be used (because no requests for these stations have been realized) and adjust the schedule accordingly.

The second model considers that for each stochastic request, both its appearance and realization time are stochastic. The time at which this request will be realized follows a Gaussian distribution with an expected value of e_p. Thus, the realization time varies across scenarios containing this request. In this case, investigating the impact of the recourse action is still valuable.

A conceptual illustration of the recourse action is shown in Figure 4: (4a) First, get-on and get-off stations are inserted for a potential request. (4b) If the request is not realized, the corresponding stations are removed if and only if they have not been visited yet and will not be visited as the next station. In contrast, if the recourse action is not applied, the route and the schedule remain the same as shown in Figure (4c). We investigate the performance of the algorithm with or without this recourse action.

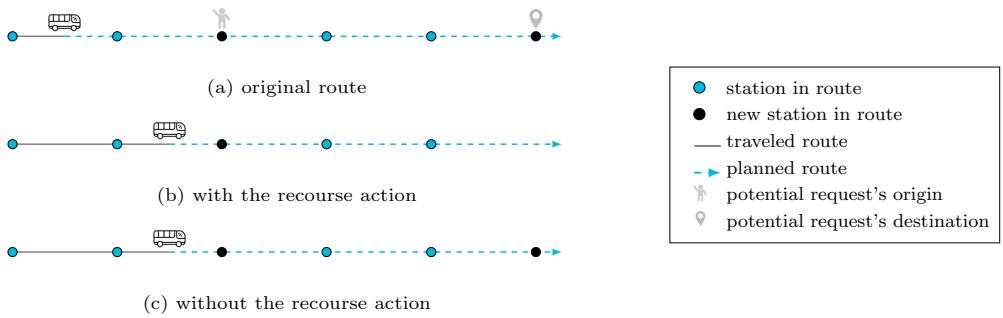

Figure 4: Conceptual illustration of the recourse action in prepositioning

The objective function differs between the two steps. In the first step, all requests must be served, and the objective function aims to minimize the total URT. In the second step, it is unrealistic to expect buses to serve all dynamic requests due to time window constraints and limited bus capacity. Therefore, the objective function in the second step is to maximize the expected number of served passengers. To summarize, the goal of this research is to design an online algorithm that proactively routes buses to achieve the objective in the second step. However, considering that dynamic requests are revealed over time, solely maximizing the expected number of served passengers may lead to the algorithm getting stuck in local optima. Therefore, we also minimize the total URT of all accepted requests in each time bucket to indirectly increase the possibility of serving more passengers later on.

4. Solution method

This section explains the solution method, which consists of two steps executed sequentially. In the first step, only static and deterministic requests are handled, and the solution is constructed using the greedy insertion heuristic described in Section 4.1.1. Afterward, VNS is applied to improve the solution's quality, as explained in Section 4.1.2. In the second step, dynamic and stochastic requests appear. In each scenario, the problem to be solved is a dynamic and deterministic ODBRP, and modified versions of the greedy insertion and VNS algorithms are applied. The similarities and differences compared to the first step are explained in Section 4.2.1. Finally, the prepositioning procedure is illustrated in Section 4.2.2. Each subsection explicitly specifies the comparisons to the literature.

4.1. The first step: static and deterministic ODBRP

In the first step, a greedy insertion heuristic is implemented to construct the initial solution, as explained in Section 4.1.1. Then, the VNS algorithm with three local search (LS) operators is applied to improve the solution's quality, as detailed in Section 4.1.2.

4.1.1. Greedy insertion

The greedy insertion heuristic constructs a solution by sequentially inserting one request at a time. The requests are initially ranked based on their earliest allowed pickup time e_p, with the request having the smallest e_p ranked first. The procedure then selects the first request p and iterates over all buses, attempting to insert the request at every position along each bus's trip. The

order of the buses is determined by their IDs, and each position is examined sequentially to insert the get-on and get-off stations. If multiple feasible solutions exist, the one with the minimum total URT is chosen. If no feasible solutions are found, request p is served by dispatching a new bus. Compared to the literature that uses the greedy insertion approach to solve the ODBRP (Melis and Sörensen, 2022b,a; Lian et al., 2022b), a new bus is dispatched only when no feasible solution has been found yet, allowing for a potentially smaller fleet size. After completing the insertion procedure for every request, the VNS algorithm is triggered to further reduce the total URT.

4.1.2. The VNS framework

Following the constructive heuristic, a VNS algorithm is applied to improve the solution's quality. This VNS algorithm draws inspiration from the VNS developed in Lian et al. (2022b), which successfully solved a dynamic ODBRP. The neighborhood structure consists of three local search (LS) operators: *alternative get-off stations*, *swap*, and *reinsert*. The concept of each LS operator is as follows:

- The *alternative get-off stations* operator replaces a passenger's get-off station with its substitute if the passenger has at least two options. If the station is exclusively used by this passenger, it is directly replaced. Otherwise, the original station is retained, and the substitute is inserted right in front.

- The *swap* operator exchanges a passenger's get-off station with one in an earlier position. If the passenger's get-off station is also used by other passengers, the swap is only allowed if the precedence constraint still holds for them.

- The *reinsert* operator removes a passenger from one route and reinserts them into another route if it results in a reduced total URT.

The three LS operators are applied in the same sequence as described. Each operator terminates if the solution is no longer improved after a round of examination for each passenger.

Additionally, a large neighborhood search (LNS) algorithm (Lian et al., 2022a) is utilized for the shaking phase. The LNS randomly removes multiple requests at a time and reinserts them at positions that yield the smallest increase in the total URT. The number of requests to be removed at a time is set to a value greater than 1 (but not equal to 1) based on experimental results (Lian et al., 2022a), and in this study, it is set to 2. The LNS is repeated σ times, generating a maximum of σ feasible solutions. The solution with the lowest objective value among the candidates generated by the repair operator is considered the most promising. Setting $\sigma > 1$ allows for investing time in more promising current solutions, which has been shown to yield better solutions than $\sigma = 1$ in previous experiments (Lian et al., 2022a). Therefore, in this study, σ is set to 10.

The VNS procedure is outlined in Algorithm 1. The algorithm terminates when a predetermined runtime is reached.

Algorithm 1: VNS: Post-optimization after greedy insertion

Result: Improve the initial solution with VNS
Shaking: LNS.
for *each passenger p* **do**
    Apply alternative get-off stations operator;
    Choose the solution with the minimum total URT;
for *each passenger p* **do**
    Apply swap operator;
    Choose the solution with the minimum total URT;
for *each passenger p* **do**
    Apply reinsert operator;
    Choose the solution with the minimum total URT;

4.2. The second step: dynamic and stochastic ODBRP

The second step deals with dynamic and stochastic requests. Similar to the first step, the greedy insertion and VNS are also applied upon and after inserting new requests. However, there are differences with the first step, both with and without prepositioning. The details are explained in Section 4.2.1. The prepositioning procedure is then explained in Section 4.2.2.

4.2.1. Algorithms in two steps: similarities and differences

The greedy insertion and VNS are applied in both steps. In the second step, there are two cases depending on whether prepositioning is applied. The case without prepositioning is simply the DODBRP, where each dynamic request is inserted in the current solution after it is actually received. The case with prepositioning allows for proactive insertion of each stochastic request in the current solution, as described in Section 3.2 and explained in Section 4.2.2.

The greedy insertion procedure described in Section 4.1.1 is almost the same for the three cases: static requests, dynamic requests with prepositioning, and dynamic requests without prepositioning. All requests are sorted in ascending order of their earliest allowed departure, and every combination of the positions in each bus is tested to insert a request. The solution with the minimum total URT is selected. If no feasible solution is found yet, the request is either assigned to a newly dispatched bus in the first step, or rejected in the second step.

However, there are major differences in terms of the request set and allowed positions. First, the set of passengers is different. For the initial solution in the first step, the set is R_{sta}. In the DODBRP, each dynamic request p is part of the set $R_d(t)$, and for the ODBRP with prepositioning, the request set is $S_n(\tau)$, which corresponds to the scenario in use at time t_τ.

The second difference concerns the allowed range of modification of the bus routes and schedules, both for the greedy insertion and VNS. In the second step, when the current time is t ($t \geq 0$), the range that can be modified is limited. The routes that have already been traveled or the stations that will be visited next cannot be changed, as explained in Section 3.2. In contrast, in the first step, no such limitations exist. The value of t changes as the simulation clock progresses. For the DODBRP, $t := e_p$ each time a dynamic request p appears at e_p. For the case with prepositioning, t is equal to t_τ, as τ is set to $0, 1, 2, ..., T$ sequentially in Section 3.2.

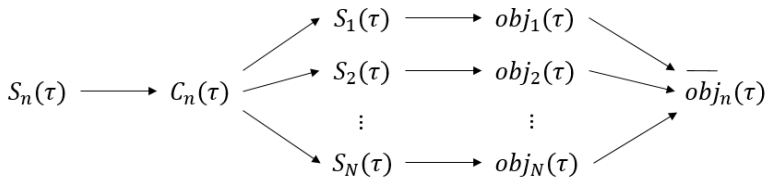

Figure 5: Illustration of the scenario-based method

4.2.2. Method of prepositioning

In this subsection, the prepositioning procedure with scenarios is explained in detail. The scenario-based prepositioning framework was previously proposed by Li et al. (2019), while the formulation of stochastic requests is original within the scenarios. We introduce the recourse action to the method of prepositioning.

As explained in Section 3.2, requests are deterministic in each scenario, and the problem to be solved is the deterministic ODBRP. Consequently, in each scenario, the routes and schedules of the buses after t_τ are adapted to maximize the number of accepted new requests in the time bucket while minimizing the total URT. The corresponding solution of buses' routing and scheduling is called a *candidate* solution $C_n(\tau)$. Among all the candidate solutions generated for each scenario, the one that fits all scenarios best is selected.

To determine which solution is best, each candidate solution is tested on each scenario $S_1(\tau)$, $S_2(\tau)$, ..., $S_N(\tau)$ to calculate the number of served passengers and the total URT, denoted as $obj_n(\tau)$ (both values are stored in $obj_1(\tau)$, $obj_2(\tau)$, ..., $obj_N(\tau)$). In other words, the requests from each scenario are inserted in sequence into each candidate solution, and a request can be served if no violations occur. Each accepted request is inserted at the position with the least increase in the total URT. Then, for each candidate solution $C_n(\tau)$, the average number of served requests among all scenarios is calculated, along with the average total URT. These average values form $\overline{obj}_n(\tau)$.

The motivation for this procedure is to maximize the expected value of served passengers since, at time t_τ, it is unknown which stochastic dynamic request will actually be realized. Among all the candidate solutions, the one with the largest expected number of served passengers will be implemented in the next time bucket. If two or more candidate solutions result in the largest expected number of served passengers, the one with the smallest total URT will be implemented. This criterion corresponds to the objective function described in Section 3.2. The procedure is also shown in Figure 5, where $C_n(\tau)$ is generated given $S_n(\tau)$, and each $C_n(\tau)$ is evaluated among all scenarios $S(\tau)$ using the average objective value.

When at time t_τ, a specific scenario $S_n(\tau - 1)$ is realized, each stochastic request in the time interval $[t_{\tau-1}, t_\tau)$ either has appeared or not. As described in Section 3.2, we propose to model the stochastic requests in two ways in this work.

For the case where requests only appear at a fixed time, the problem becomes deterministic. The stochastic requests contained in $S_n(\tau - 1)$ have appeared, while those that have not appeared are considered as false information.

In contrast, for the case where the realization time is also stochastic, at time t_τ, the requests between $[t_{\tau-1}, t_\tau)$ that have not yet been realized might still possibly be realized later on. Never-

theless, the same procedure, described below, is applied.

The realized scenario does not necessarily correspond to the one generating the best candidate solution, so there can be empty stations in routes that were intended for the no-show requests. According to our assumption, this can cause detours and increase the total URT, potentially preventing the buses from serving upcoming requests (unless any of them happen to use an empty station in the route around the time the bus passes by that station). However, our model aims for on-demand transportation and tries to avoid instances in which a bus passes by a station with no one getting on or off. Hence, in terms of the recourse action at t_τ, we can define two strategies. One strategy is to remove empty stations, while the other strategy is to keep the empty stations in the route. We refer to removing the empty stations as the *recourse* action. In the recourse strategy, the empty stations that have already been visited before t_τ or the empty station being visited as the next station cannot be modified. Instead, only the remaining route from the next station to the end can be subjected to the recourse action. Algorithm 2 outlines the procedure of the ODBRP with prepositioning. In this algorithm, the *if*-block (lines 3 - 6) represents the recourse action for the former strategy ($\tau > 0$ such that there is one realized scenario), while it does not exist for the latter strategy. For easier comparison, the procedure to insert dynamic requests in the DODBRP is also outlined in Algorithm 3.

Algorithm 2: Procedure of ODBRP with prepositioning

```
1  for realized_scenario = 1, 2, ..., N do
2      for τ = 0, 1, ..., T do
3          if τ > 0 then
4              for bus b in all buses do
5                  for empty stations do
6                      remove empty station i if t^a_bi > t_τ and i is not the station b is heading
                        towards;
7          for n = 1, 2, ..., N do
8              for each request p ∈ S_n(τ) do
9                  insert p at the position with the minimum increase in the total URT;
10                 VNS, Algorithm 1;
11             Evaluate C_n(t);
12         Implement the best C_n(t);
13 Calculate the average objective value of N scenarios.
```

Algorithm 3: Procedure to insert dynamic requests, without prepositioning

```
1  for realized_scenario = 1, 2, ..., N do
2      for each request p ∈ R_dn do
3          insert p at the position with the minimum increase in the total URT;
4          VNS, Algorithm 1;
5  Calculate the average objective value of N scenarios.
```

5. Computational experiments

In this section, we evaluate the performance of our algorithms through various experiments. We begin by describing the generated instances for testing purposes in Section 5.1. Subsequently, we conduct different experiments to assess the effectiveness and robustness of the algorithms, providing practical guidance. The results of the algorithm for different parameter values are presented in Section 5.2.

All algorithms have been implemented in C++ and executed on a Windows 10 computer system equipped with an Intel® Core^{TM} i7-8850H processor operating at 2.60 GHz and 16 GB RAM.

Regarding the runtime of the VNS algorithm (Algorithm 1), in the first step, it is set to $0.1 \cdot sta$ seconds, where *sta* represents the number of static requests specified in Table 2. In the second step, the runtime is set to 0.01 seconds times the instance size, namely the number in the first column in Table 2, each time the VNS is invoked.

5.1. Instance generation

The instances were generated based on the map of Antwerp, Belgium, using a custom instance generator developed specifically for on-demand passenger transportation problems (Queiroz et al., 2022). The city covers an area of 204.51 km^2 and consists of 10 randomly located population clusters, each with a maximum area of 4 km^2 (2 km $\times$ 2 km). Figure 6 illustrates the city map and the road network with the clusters marked in red. To enhance realism, the actual stations within the clusters are used as boarding and alighting points for passengers. There are a total of 215 stations, with an average distance of approximately 500 meters between neighboring stations. Furthermore, passengers' trips can be either within a cluster or between different clusters.

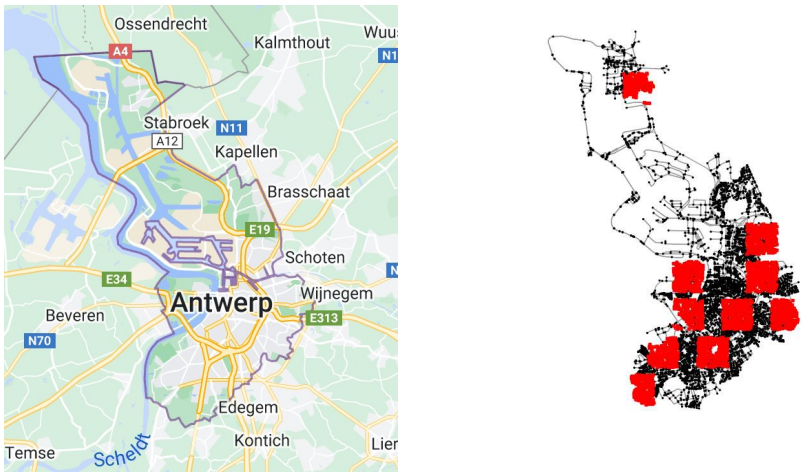

Figure 6: Map and clusters' locations

The number of passengers in the instances varies from small to medium to large, specifically ranging from 100 to 900 with increments of 100, and then 1000, 1500, 2000, and 3000. Therefore, there is a total of 13 instances, with one instance for each size. It is important to note that the

topology of the clusters and stations, as well as the generation principle of passengers, remain consistent across all instances.

For simplicity, a constant travel speed of 30 km/h is assumed throughout the instances. The operation time is set to two hours, and passengers' time windows are evenly distributed within this time frame. Each passenger's time window is determined as 1.5 times the direct ride, where the factor of 1.5 strikes a balance between strict and loose time windows, aligning with the concept of on-demand bus service that combines aspects of public transport and direct rides.

In these instances, all static requests are deterministic, while all dynamic requests are stochastic. For each instance, every dynamic and stochastic request has the same probability of realization, which varies among three levels: 0.1, 0.25, and 0.5.

Additionally, the percentage of static requests out of the total number of requests is set at 20%. For example, if an instance has a total of 100 requests, it will consist of 20 static requests and 80 dynamic stochastic requests. Considering a realization probability of 0.1, the expected number of dynamic and total requests would be 8 and 28, respectively. Conversely, an instance with a total of 3000 requests would include 600 static requests and 2400 dynamic stochastic requests. With a realization probability of 0.5, the expected number of dynamic and total requests would be 1200 and 1800, respectively.

The buses are homogeneous and equipped with 8 seats. The number of buses for each instance is set to the minimum required to accommodate all static passengers, and no additional buses are dispatched for dynamic requests. This setup reflects the practical scenario where public buses are pre-scheduled while passengers arrive dynamically and stochastically. Furthermore, the static formulation of the ODBRP neglects the presence of depots, assuming that each bus starts its route from the first passenger's get-on station and ends at the last passenger's get-off station. This is because the positions of depots do not affect passengers' URT. The issue of bus prepositioning from depots is left for future research.

Table 2 provides the specific values for each instance. The following notations are used in the table: sta represents the number of static requests, $|D|$ denotes the number of dynamic requests, $|P|$ indicates the number of total requests, $\mathrm{E}(|D|)$ represents the expected number of dynamic requests, $\mathrm{E}(|P|)$ corresponds to the expected number of total requests, $|B|$ signifies the number of buses, and $sta/|B|$ denotes the average number of static passengers per bus.

Table 2: Number of requests per instance class

Instance	***sta***	$\|D\|$	$\mathbf{E}(\|D\|)$			$\mathbf{E}(\|P\|)$			$\|B\|$	***sta*/$\|B\|$**
			0.1	**0.25**	**0.5**	**0.1**	**0.25**	**0.5**		
100	20	80	8	20	40	28	40	60	5	4.0
200	40	160	16	40	80	56	80	120	9	4.4
300	60	240	24	60	120	84	120	180	10	6.0
400	80	320	32	80	160	112	160	240	12	6.7
500	100	400	40	100	200	140	200	300	16	6.3
600	120	480	48	120	240	168	240	360	16	7.5
700	140	560	56	140	280	196	280	420	19	7.4
800	160	640	64	160	320	224	320	480	19	8.4
900	180	720	72	180	360	252	360	540	26	6.9
1000	200	800	80	200	400	280	400	600	27	7.4
1500	300	1200	120	300	600	420	600	900	33	9.1
2000	400	1600	160	400	800	560	800	1200	44	9.1
3000	600	2400	240	600	1200	840	1200	1800	60	10.0
average	185	738	74	185	369	258	369	554	23	7.2

The earliest allowed time to start operating is 0, and the operation ends after 120 minutes. The earliest departure times are evenly distributed in time and can be deterministic or follow a normal distribution. In the latter case, the expected value is equal to the fixed value in the deterministic case, and the standard deviation is set to 4 minutes. It is important to note that the time at which the request appears (i.e., the realization time) is equal to the earliest departure time. The standard deviation value is chosen to ensure that passengers appear within a reasonable range, neither too large nor too small.

Each request can have up to two stations for boarding and two stations for alighting. In our data sets, 90% of the requests have two stations for boarding and/or alighting, while the remaining 10% have only one station each. This assumption is based on the fact that alternative stations may not be available for everyone in real life.

The number of scenarios is fixed at 9 for all instances.

5.2. Results

After setting up the experiments, we proceed to test the effectiveness of prepositioning. We first evaluate it under the ideal case, which serves as an upper bound for the approach in this problem setting. Then, we strategically vary the scenarios to assess the efficacy of the approach when the forecast information is inaccurate.

5.2.1. Efficacy with perfect forecast accuracy

We begin by testing the efficacy of prepositioning under the ideal case, where the forecast of dynamic and stochastic passengers is 100% accurate. At the same time, we analyze the impact of factor H, which represents the length of a time bucket to look ahead. A smaller H indicates a relatively short-term plan, while a larger H extends the plan further into the future. We set H equal to 10, 20, 40, and 120 minutes, and study the solution quality for all instances. When H is set to 120 minutes, all dynamic and stochastic requests are considered before the operation starts. On the other hand, 10, 20, and 40 correspond to 12, 6, and 3 time buckets, respectively.

We conduct experiments for the pure DODBRP case first and record the results per instance, including the number of served passengers, the average URT, the average travel distance of each bus, as well as the standard deviation among the scenarios. The average URT and average travel distance are excluded from the objective function but provided for reference. Detailed values can be found in Table B.4 - B.6 in the Appendix.

Next, we present the average relative results, primarily measured by the number of served passengers, compared to the pure DODBRP. These results are shown in Figure 7, and the average values are calculated over all 13 instances, corresponding to the first column in Table 2. The x-axis labels indicate the probability of realization (first number) and whether the requests appear within an interval (1) or not (0) (second number). Detailed values, along with the average URT and average traveled distance, can be found in Table B.7 - B.12 in the Appendix.

The results demonstrate that when the forecast information is perfect, the algorithm *with* prepositioning serves significantly more passengers than the DODBRP. Furthermore, the efficacy of prepositioning increases monotonically as H increases. This suggests that considering stochastic requests as far in advance as possible is most beneficial, as a larger H provides more possibilities to find a good solution. Even though a larger H corresponds to fewer rounds of algorithm executions and VNS, which can possibly contribute to the solution.

A similar trend is found by Li et al. (2019), who also study prepositioning in the context of DARP. They observe that increasing H from 5 to 20 minutes leads to higher profit, lower waiting time, but larger detours. However, they did not investigate the optimal length of the time bucket based on forecast accuracy, which we explore in this subsection and Subsection 5.2.3. In our experiments, the relationship between the URT and traveled distance on one hand and the number of served passengers on the other hand (Table B.7 - B.12 in the Appendix) is not obvious. In other words, serving more passengers can result in either larger or smaller average URT and average traveled distance.

Another observation is that as the probability of requests being realized increases, the gap between prepositioning and the DODBRP also increases. One possible reason for this is that larger probabilities correspond to larger instances (i.e., larger E($|P|$) in Table 2). However, when comparing different instance sizes from 100 to 3000 (Table B.7 - B.12 in the Appendix), there is no increasing trend (but a slight decrease) as the instance size increases. This could be attributed to the fact that the buses' average travel distance approaches the upper bound (60 km, given the operation time and constant speed) as explained in Section 5.1. Consequently, the buses' routes and schedules become dense, resulting in a smaller solution space for new requests. This limitation may restrict the advantage of prepositioning.

Comparing the cases where requests appear at a fixed time point or within a range, neither the DODBRP nor prepositioning with perfect forecast accuracy shows significant benefits.

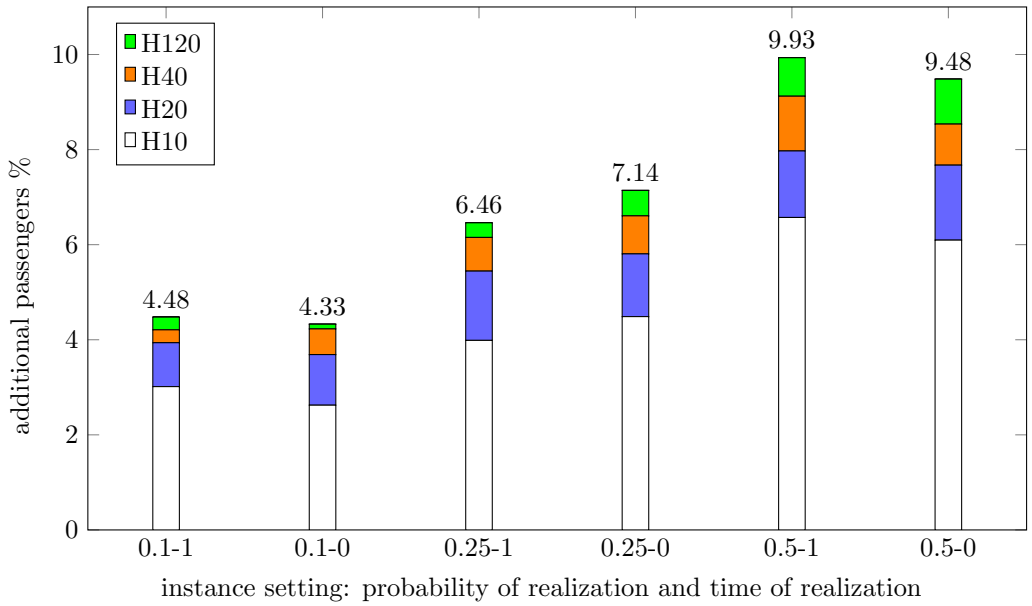

Figure 7: Additional passengers served (%) with various lengths of time bucket and perfect forecast

5.2.2. Comparison with static ODBRP

To further evaluate the disparity between prepositioning and the DODBRP, we consider an assumption where all requests are static and deterministic within each scenario. This allows us to solve the static ODBRP and use its solutions as an "upper bound" for these instances. However, the optimal solution for each instance remains unknown. The gaps between the static ODBRP and prepositioning, as well as between the static ODBRP and the DODBRP, are illustrated in Figure 8. In the figure, "Dy" represents the DODBRP, "Pre" represents prepositioning, and "int 1" (or "int 0") indicates stochastic (or deterministic) realization times. From the figure, it is evident that solution gaps exist between the static and dynamic ODBRP. The DODBRP exhibits an average gap of approximately 7% to 14% in terms of the number of served passengers across these experimental instances. In contrast, the prepositioning approach can reduce the gap to around 4% to 7%, roughly halving it.

5.2.3. Inaccurate forecast

As the ideal case is not feasible in practice, we aim to assess the performance of prepositioning compared to the DODBRP and explore potential enhancements when faced with inaccurate forecast information. Unless otherwise specified, the instances used in this subsection involve requests with stochastic realization times.

To represent different levels of accuracy, we calculate the similarity between scenarios and vary the scenario sets used to generate solutions. The simple matching coefficient (SMC) is employed as a similarity measure, where each scenario is transformed into a binary vector indicating the presence (1) or absence (0) of each dynamic and stochastic request. The SMC is calculated by comparing each realized scenario with the other scenarios, summing the matching values, and dividing by the total number of dynamic and stochastic requests. The resulting SMC ranges from 0 to 1. However,

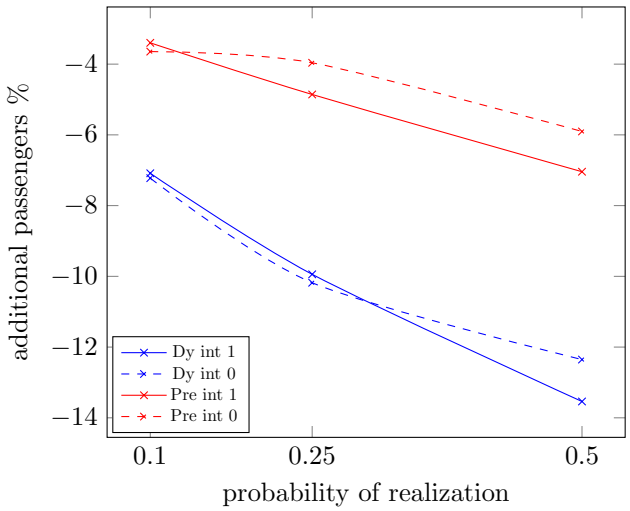

Figure 8: Additional passengers served (%) with various lengths of time bucket and perfect forecast

the similarity measure only considers request presence/absence and does not account for differences in realization times among scenarios.

Next, for each given scenario, we vary the scenario set for candidate solutions at different accuracy levels: 0.8, 0.6, and 0.4. For example, with an accuracy of 0.8, a proportion of $0.2 \cdot \mathrm{E}(|D|)$ dynamic and stochastic requests are randomly selected to flip their binary values. In the case where requests appear within an interval, if a 0 is flipped to 1, a random realization time is generated from its aforementioned normal distribution. Importantly, every scenario in the set differs from the actual scenario, meaning the actual scenario is not included in the set.

For the different accuracy levels, the instances are tested using the aforementioned values of H: 10, 20, 40, and 120.

Additionally, the recourse action (removing empty stations as described in Section 3.2) is evaluated. In our problem setting, the recourse action is only applicable when the forecast accuracy is strictly less than 100% and when H is smaller than the operation duration, i.e., when there are at least two time buckets. The experiments are conducted for instances with varying H. Specifically, each probability of realization (0.1, 0.25, and 0.5) is combined with three accuracy levels (0.8, 0.6, and 0.4), resulting in a total of 9 distinct cases. Furthermore, the cases are implemented with three time bucket lengths (10, 20, and 40 minutes). Subsequently, the algorithm is tested with and without the recourse action for each time bucket length. Finally, for each case, the 13 instances with different total numbers of passengers (ranging from 100 to 3000) are solved, and the results are represented by the average number of served passengers among the 13 instances. To compare the results with the DODBRP, the additional percentage of passengers served compared to the DODBRP is computed for each case, grouped by the probability of realization (in the order 0.1, 0.25, and 0.5). These results are illustrated in Figures 9, 10, and 11. The average number of served passengers for the DODBRP is 199.7, 215, and 237, respectively, indicated by a vertical line at 0. In the legends, the time bucket lengths are shown as numbers and differentiated by color, while the recourse action is abbreviated as “r” and represented by “$\diamond$”.

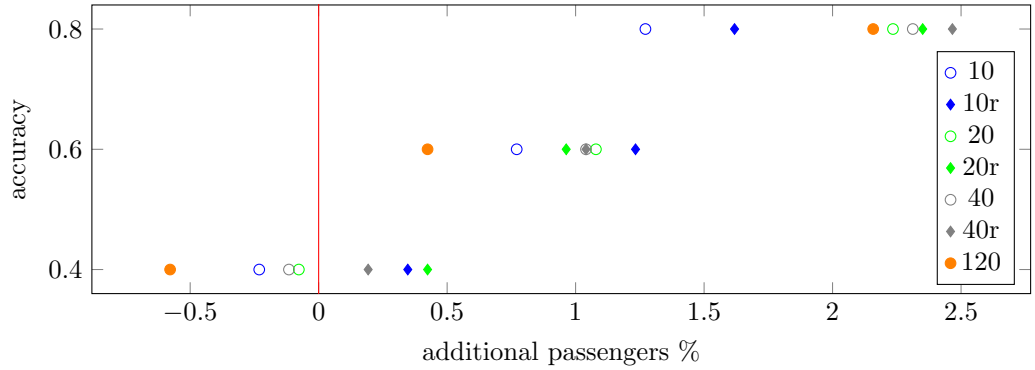

Figure 9: Probability 0.1, additional passengers (%) with various lengths of time bucket and accuracy

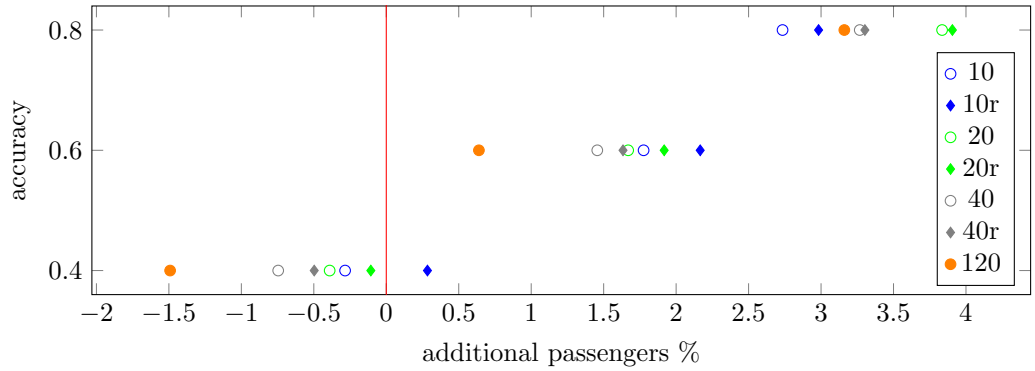

Figure 10: Probability 0.25, additional passengers (%) with various lengths of time bucket and accuracy

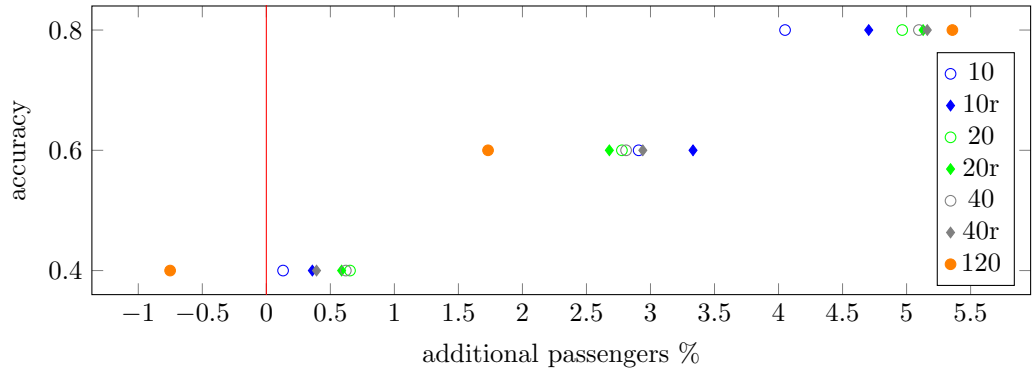

Figure 11: Probability 0.5, additional passengers (%) with various lengths of time bucket and accuracy

First, we analyze the optimal values of H for different probabilities and accuracy levels. In Figure 11, when the accuracy is high (0.8), even using the entire operation duration as H can

outperform other values for a probability of 0.5, maintaining the decreasing trend from 120 minutes to 10 minutes observed in the ideal case. However, a smaller value of H is generally preferable. As the accuracy decreases, there is a trend towards shorter favorable lengths of H. Specifically, as the accuracy drops from 0.8 to 0.6, the optimal H decreases from 40 minutes to 10 minutes in Figure 9, from 20 minutes to 10 minutes in Figure 10, and from 120 minutes to 10 minutes in Figure 11. This indicates that considering requests in the near future and planning more frequently can generally lead to better solutions when the accuracy is moderate or good. Conversely, a one-off plan (H = 120 minutes) consistently performs the worst in the tested cases when the accuracy is 0.6.

When the accuracy is poor (0.4), prepositioning has little advantage, with a difference of less than 0.5% or even worse solutions compared to the DODBRP, regardless of the value of H. However, setting H to 120 minutes only worsens the solution quality in the tested cases when the accuracy is 0.4. To summarize the experiments with varying accuracy levels, the superiority of prepositioning over the DODBRP holds only when the accuracy is above a certain level, which is approximately 40% in this problem setting. Naturally, the more accurate the forecast, the more advantageous prepositioning becomes.

In addition to the time bucket, we analyze the recourse action. Overall, the recourse action leads to more served passengers compared to the algorithm without the recourse action, for all probabilities, especially when the accuracy is acceptable (0.8 and 0.6). Even when the accuracy is poor (0.4), the recourse action still tends to improve the solution, although the advantage over the DODBRP is negligible. However, in cases where the accuracy is very low, such as an accuracy of 0.4 in Figure 10, the recourse action is hardly worth the effort.

Numerically, compared to the average values achieved without the recourse action, the recourse action results in an additional 29.7%, 6.6%, and 5.7% of passengers served with H equal to 10, 20, and 40 minutes, respectively, considering the average of the three accuracy levels. Thus, more frequent checks of empty stations are preferable, while when H is large, it can be too late to remove the empty stations. When comparing the performance among the accuracy levels and considering the average of the time bucket lengths, the recourse action achieves an additional 6.2% and 10.0% of served passengers when the accuracy is 0.8 or 0.6, respectively. When the accuracy is 0.4, the recourse action improves the average increase in the number of served passengers from -0.05% to 0.22%. Therefore, it is more important to adopt the recourse action when the forecast is inaccurate. In summary, the recourse action generally leads to better results compared to its counterpart under different accuracy and time bucket levels, and it is especially effective when the mismatch between the forecast and realization is small to moderate.

Finally, comparing the probabilities of realization, the results are similar to the ideal case: the larger the probability of realization, the greater the beneficial effect of applying prepositioning compared to the DODBRP.

The importance of forecast accuracy and the length of the time bucket in achieving good solution quality is demonstrated by the above results. Additionally, we investigate whether increasing or decreasing the number of sampled scenarios can improve or reduce the solution quality, respectively. In this regard, we reduce the size of the scenario set from 9 to 2, while keeping the other settings constant (including the application of the recourse action). The computational results are presented in Table 3. In the header of the table, for example, the notation "80 H10" represents the case with an accuracy of 0.8 and a time bucket length of 10 minutes. The abbreviation "ave" corresponds to "average". The values in the table represent the percentage of the nine-scenario case relative to the two-scenario case, averaged over 13 instances.

Naturally, having more scenarios aids the algorithm in evaluating stochastic requests and mak-

ing more precise routing decisions when compared to the two-scenario case. This advantage is particularly beneficial when the forecast is less accurate, the probability of realization is higher, and the length of the time bucket is shorter.

Table 3: Additional passengers served (%) with 9 scenarios compared to 2

	80 H10	**80 H20**	**80 H40**	**60 H10**	**60 H20**	**60 H40**	**Row ave**
0.1	0.80	1.07	0.99	1.59	1.12	0.96	1.09
0.25	1.01	1.32	0.87	2.53	1.77	1.31	1.47
0.5	1.84	1.71	1.16	3.33	1.68	2.04	1.96
Column ave	1.22	1.36	1.01	2.48	1.52	1.44	
80/60 ave		1.20			1.82		

5.2.4. Effect of bus station assignment

The experiments conducted in the previous subsections demonstrate the effectiveness of repositioning. It is also worth investigating whether bus station assignment, as introduced in Section 1, contributes to the overall solution quality. Bus station assignment is a key characteristic that distinguishes the ODBRP from the DARP. To evaluate its impact, we test instances with different probabilities of realization using the optimal length of time buckets (with one exception, as explained below), as presented in Figures 9 to 11 in Subsection 5.2.3. Specifically, the case with a probability of 0.1 and 80% accuracy is tested with H equal to 40, while the case with a probability of 0.25 and 80% accuracy is tested with H equal to 20. The exception is the case with a probability of 0.5 and 80% accuracy, which is tested with a suboptimal H value of 40, even though the best solution is achieved with H equal to the entire operation time. The recourse action is applied in all cases. Consistent with previous experiments, all 13 instances for each probability listed in Table 2 are used, and the average value is chosen as the reference. The instances used for the bus station assignment evaluation have stochastic realization times.

To compare the cases with and without BSA, passengers with two alternative stations are simply assigned to the station that is closest to their origin or destination. In other words, each passenger has only one station for pickup and one station for drop-off. The percentage of additional served passengers over the 13 instances with BSA relative to the cases without BSA is calculated, and the results are illustrated in Figure 12.

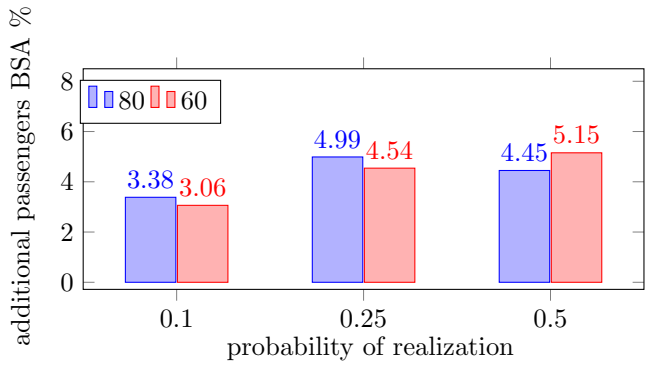

Figure 12: Additional passengers served (%) with bus station assignment

The results indicate that BSA contributes to the improved performance of prepositioning in serving more passengers. When comparing different probabilities, a slightly higher gap is achieved by BSA as the probability of passenger realization increases. Specifically, the average additional passengers achieved by BSA for probabilities 0.1, 0.25, and 0.5 are 3.2%, 4.8%, and 4.8%, respectively. Comparing the accuracy levels, the differences are even smaller, with BSA achieving approximately 4.3% additional passengers for both 80% and 60% accuracy levels. In summary, the effectiveness of BSA remains robust across various combinations of probabilities and accuracy levels.

5.2.5. Combining prepositioning and the DODBRP

In this subsection, we assess another potential solution when the forecast is inaccurate, which involves combining the DODBRP with prepositioning and the removal of empty stations. The procedure for prepositioning and empty station removal remains the same, with an additional step: when a dynamic request is realized, we attempt to insert it into the current solution following the principles of the DODBRP, but only if it has not already been planned for by prepositioning. If prepositioning has already included the request in the routing solution, no further actions are taken.

The same cases tested in Subsection 5.2.4 are selected to evaluate whether this combination can further improve solution quality. As before, the instances used have stochastic realization times. The percentage of additional served passengers among the 13 instances achieved by this combination, relative to the case with only prepositioning and empty station removal, is calculated. The results are presented in Figure 13.

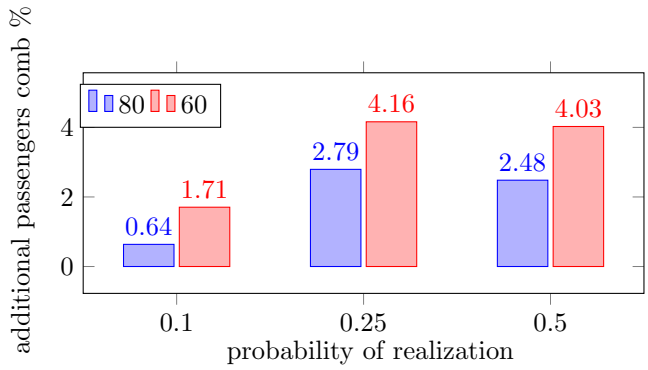

Figure 13: Additional passengers served (%) with the combination of prepositioning, removal of empty stations and dynamic insertion of requests (i.e., the DODBRP)

As expected, the combined approach results in an increase in the number of served passengers. When the probability of realization increases, the effectiveness of the combination first increases and then slightly decreases. The increasing trend from 0.1 to 0.25 is consistent with the previous findings, as the expected number of passengers is larger. However, the slight drop from 0.25 to 0.5 could be due to the increased difficulty of finding feasible solutions through simple insertions, given that the solution space becomes smaller when there are already more passengers in the routing solution, especially when the number of buses is set to the minimum value required to serve all static requests.

As expected, the approach is more effective when the prediction accuracy is lower, as there is more room for improvement. In summary, the combined approach is effective when the accuracy is

not perfect.

5.2.6. Overall observations

In this subsection, we summarize the key findings from the experiments.

First, the efficacy of prepositioning is evaluated under the assumption of 100% forecast accuracy. The results show that prepositioning outperforms the DODBRP, and longer time buckets lead to serving more passengers. The nature of the realization time (deterministic or stochastic) does not significantly impact the solution quality for both the DODBRP and the approach with prepositioning.

Second, the experiments vary the accuracy levels and assess the impact of the recourse action and different time bucket lengths on instances with different probabilities of realization. The removal of empty stations generally improves the solution quality of prepositioning, and this performance remains robust across different accuracy levels, probability of realization, and time bucket sizes. There is a trend that as the accuracy decreases, shorter time buckets (from 120 to 40 or 20, and from 40 or 20 to 20 or 10) are preferred.

Third, increasing the size of the scenario set from which candidate solutions are generated can also improve solution quality, in addition to shortening the time bucket. By attempting to insert dynamic requests that have appeared but have not yet been included in the current solution, along with prepositioning and the removal of empty stations, more passengers can be served during the buses' operation time.

Finally, incorporating bus station assignment when applying prepositioning helps to serve more passengers.

To summarize, the results of our experiments are promising. Despite the limitations of the modeling, the satisfactory solution quality of prepositioning provides confidence that it can improve solution quality in a dynamic stochastic version of the on-demand bus routing problem.

6. Conclusion

In this paper, we have investigated the effectiveness of prepositioning, specifically in the context of a stochastic and dynamic variant of the on-demand bus routing problem. We have proposed heuristic algorithms with and without the recourse action to determine whether considering future requests while routing and scheduling buses can improve solution quality. Through an extensive computational experiment on realistic problem instances, we have demonstrated that prepositioning is indeed effective across a wide range of parameter values.

Specifically, the prepositioning strategy outperforms the dynamic solution without prepositioning when stochastic requests have a high probability of realization or when the forecast accuracy of the stochastic requests is high.

We have also identified important factors that influence the performance of the prepositioning strategy, such as the length of the time bucket (H). A shorter H is preferred when the forecast accuracy is low, while a longer H is preferred when the accuracy is high.

Furthermore, we have found that the recourse action of removing empty stations after each realized time bucket generally improves solution quality. Additionally, a larger scenario set contributes to better performance. Finally, combining prepositioning with the dynamic solution algorithm and the recourse action leads to high-quality solutions.

In future work, we plan to explore more realistic formulations of stochastic requests and investigate other heuristic algorithms for solving this type of problem.

Acknowledgements

The authors would like to thank the Bijzonder Onderzoeksfonds of the University of Antwerp for funding the main author of this research.

Credit Authorship Contribution Statement

Ying Lian: Conceptualization, Methodology, Software, Formal Analysis, Investigation, Writing - Original Draft, Writing - Review and Editing, Visualization. **Flavien Lucas**: Validation, Writing - Review and Editing, Formal Analysis, Visualization, Supervision. **Kenneth Sörensen**: Conceptualization, Validation, Formal Analysis, Resources, Writing - Review and Editing, Project Administration, Funding Acquisition, Supervision.

References

Attanasio, A., Cordeau, J. ., Ghiani, G., and Laporte, G. (2004). Parallel tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem. *Parallel Computing*, 30(3):377–387.

Banerjee, N., Morton, A., and Akartunalı, K. (2020). Passenger demand forecasting in scheduled transportation. *European Journal of Operational Research*, 286(3):797–810.

Beaudry, A., Laporte, G., Melo, T., and Nickel, S. (2010). Dynamic transportation of patients in hospitals. *OR spectrum*, 32(1):77–107.

Belhaiza, S. (2017). A data driven hybrid heuristic for the dial-a-ride problem with time windows. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8. IEEE.

Berbeglia, G., Cordeau, J.-F., and Laporte, G. (2012). A hybrid tabu search and constraint programming algorithm for the dynamic dial-a-ride problem. *INFORMS Journal on Computing*, 24(3):343–355.

Cordeau, J.-F. (2006). A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54(3):573–586.

Cordeau, J.-F. and Laporte, G. (2003). A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6):579–594.

Cordeau, J.-F. and Laporte, G. (2007). The dial-a-ride problem: models and algorithms. *Annals of operations research*, 153(1):29–46.

Coslovich, L., Pesenti, R., and Ukovich, W. (2006). A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem. *European Journal of Operational Research*, 175(3):1605–1615.

Gaul, D., Klamroth, K., and Stiglmayr, M. (2021). Solving the dynamic dial-a-ride problem using a rolling-horizon event-based graph. In *21st Symposium on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.

Gaul, D., Klamroth, K., and Stiglmayr, M. (2022). Event-based milp models for ridepooling applications. *European Journal of Operational Research*, 301(3):1048–1063.

Hanne, T., Melo, T., and Nickel, S. (2009). Bringing robustness to patient flow management through optimized patient transports in hospitals. *INFORMS Journal on Applied Analytics*, 39(3):241–255.

Heilporn, G., Cordeau, J.-F., and Laporte, G. (2011). An integer l-shaped algorithm for the dial-a-ride problem with stochastic customer delays. *Discrete Applied Mathematics*, 159(9):883–895.

Ho, S. C. and Haugland, D. (2011). Local search heuristics for the probabilistic dial-a-ride problem. *OR Spectrum*, 33(4):961–988.

Ho, S. C., Szeto, W., Kuo, Y.-H., Leung, J. M., Petering, M., and Tou, T. W. (2018). A survey of dial-a-ride problems: Literature review and recent developments. *Transportation Research Part B: Methodological*, 111:395–421.

Hyytiä, E., Aalto, S., Penttinen, A., and Sulonen, R. (2010). A stochastic model for a vehicle in a dial-a-ride system. *Operations Research Letters*, 38(5):432–435.

Ichoua, S., Gendreau, M., and Potvin, J.-Y. (2006). Exploiting knowledge about future demands for real-time vehicle dispatching. *Transportation Science*, 40(2):211–225.

Issaoui, B., Zidi, I., Zidi, K., and Ghedira, K. (2013). A distributed approach for the resolution of a stochastic dial a ride problem. In *2013 International Conference on Advanced Logistics and Transport*, pages 205–210.

Johnsen, L. C. and Meisel, F. (2022). Interrelated trips in the rural dial-a-ride problem with autonomous vehicles. *European Journal of Operational Research*, 303(1):201–219.

Li, D., Antoniou, C., Jiang, H., Xie, Q., Shen, W., and Han, W. (2019). The value of prepositioning in smartphone-based vanpool services under stochastic requests and time-dependent travel times. *Transportation Research Record*, 2673(2):26–37.

Lian, Y., Lucas, F., and Sörensen, K. (2022a). The electric on-demand bus routing problem with partial charging and nonlinear function. Working paper, University of Antwerp.

Lian, Y., Lucas, F., and Sörensen, K. (2022b). The on-demand bus routing problem with real-time traffic information. Working paper, University of Antwerp.

Liyanage, S., Abduljabbar, R., Dia, H., and Tsai, P.-W. (2022). Ai-based neural network models for bus passenger demand forecasting using smart card data. *Journal of Urban Management*, 11(3):365–380.

Lois, A. and Ziliaskopoulos, A. (2017). Online algorithm for dynamic dial a ride problem and its metrics. In *Transportation Research Procedia*, volume 24, pages 377–384.

Lowalekar, M., Varakantham, P., and Jaillet, P. (2018). Online spatio-temporal matching in stochastic and dynamic domains. *Artificial Intelligence*, 261:71–112.

Lu, C., Wu, Y., and Yu, S. (2022). A sample average approximation approach for the stochastic dial-a-ride problem on a multigraph with user satisfaction. *European Journal of Operational Research*, 302(3):1031–1044.

Luo, Y. and Schonfeld, P. (2011). Online rejected-reinsertion heuristics for dynamic multivehicle dial-a-ride problem. *Transportation Research Record*, 2218(1):59–67.

Maalouf, M., MacKenzie, C. A., Radakrishnan, S., and Court, M. (2014). A new fuzzy logic approach to capacitated dynamic dial-a-ride problem. *Fuzzy Sets and Systems*, 255:30–40. Theme: Decision and Optimisation.

Mariñas-Collado, I., Sipols, A. E., Santos-Martín, M. T., and Frutos-Bernal, E. (2022). Clustering and forecasting urban bus passenger demand with a combination of time series models. *Mathematics*, 10(15).

Marković, N., Nair, R., Schonfeld, P., Miller-Hooks, E., and Mohebbi, M. (2015). Optimizing dial-a-ride services in maryland: Benefits of computerized routing and scheduling. *Transportation Research Part C: Emerging Technologies*, 55:156–165.

Masmoudi, M. A., Hosny, M., Demir, E., Genikomsakis, K. N., and Cheikhrouhou, N. (2018). The dial-a-ride problem with electric vehicles and battery swapping stations. *Transportation Research Part E: Logistics and Transportation Review*, 118:392–420.

Melis, L. and Sörensen, K. (2022a). The real-time on-demand bus routing problem: The cost of dynamic requests. *Computers & Operations Research*, 147:105941.

Melis, L. and Sörensen, K. (2022b). The static on-demand bus routing problem: large neighborhood search for a dial-a-ride problem with bus station assignment. *International Transactions in Operational Research*, 29(3):1417–1453.

Molenbruch, Y., Braekers, K., and Caris, A. (2017). Typology and literature review for dial-a-ride problems. *Annals of Operations Research*, 259(1):295–325.

Muelas, S., LaTorre, A., and Peña, J.-M. (2015). A distributed vns algorithm for optimizing dial-a-ride problems in large-scale scenarios. *Transportation Research Part C: Emerging Technologies*, 54:110–130.

Paquay, C., Crama, Y., and Pironet, T. (2020). Recovery management for a dial-a-ride system with real-time disruptions. *European Journal of Operational Research*, 280(3):953–969.

Parragh, S. N., Doerner, K. F., and Hartl, R. F. (2010). Variable neighborhood search for the dial-a-ride problem. *Computers & Operations Research*, 37(6):1129–1138.

Pillac, V., Gendreau, M., Guéret, C., and Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11.

Queiroz, M., Lucas, F., and Sörensen, K. (2022). Instance generation tool for on-demand transportation problems. Working paper, University of Antwerp. arXiv:2208.12225.

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

Schilde, M., Doerner, K., and Hartl, R. (2011). Metaheuristics for the dynamic stochastic dial-a-ride problem with expected return transports. *Computers and Operations Research*, 38(12):1719–1730.

Schilde, M., Doerner, K., and Hartl, R. (2014). Integrating stochastic time-dependent travel speed in solution methods for the dynamic dial-a-ride problem. *European Journal of Operational Research*, 238(1):18–30.

Tafreshian, A., Abdolmaleki, M., Masoud, N., and Wang, H. (2021). Proactive shuttle dispatching in large-scale dynamic dial-a-ride systems. *Transportation Research Part B: Methodological*, 150:227–259.

Vamsi Krishna Munjuluri, V. S., Shankar, M. M., Vikshit, K. S., and Gutjahr, G. (2023). Combining variable neighborhood search and constraint programming for solving the dial-a-ride problem. In Choudrie, J., Mahalle, P., Perumal, T., and Joshi, A., editors, *IOT with Smart Systems*, pages 209–216, Singapore. Springer Nature Singapore.

Wong, K., Han, A., and Yuen, C. (2014). On dynamic demand responsive transport services with degree of dynamism. *Transportmetrica A: Transport Science*, 10(1):55–73.

Xiang, Z., Chu, C., and Chen, H. (2008). The study of a dynamic dial-a-ride problem under time-dependent and stochastic environments. *European Journal of Operational Research*, 185(2):534–551.

Yang, J., Jaillet, P., and Mahmassani, H. S. (1999). On-line algorithms for truck fleet assignment and scheduling under real-time information. *Transportation Research Record*, 1667(1):107–113.

Appendix A. Mathematical model

The objective function is to minimize the total URT. Constraints (A.2) enforce the fact that a bus can only stop at one station at the same time. Constraints (A.3) make sure the positions used in the bus route are used consecutively and start at the first position. Constraints (A.4) and (A.5) respectively enforce a bus to stop at one station if and only if at least one passenger uses it either to board or alight. Constraints (A.6) and (A.7) respectively impose that a station is designated for a passenger to board/alight only if the station belongs to the passenger, i.e. it is within the predefined walking distance. Constraints (A.8) impose for any two consecutive stations, the arrival time at the latter station is (larger than or) equal to the departure time at the previous one plus the travel time. Constraints (A.9) guarantee that the departure time at a passenger's pickup station is greater than or equal to the earliest allowed value. Correspondingly, constraints (A.10) guarantee the arrival time at a passenger's drop-off station is smaller or equal to the latest allowed value. Constraints (A.11) impose that the pickup station precedes the corresponding drop-off one for any passengers. Constraints (A.12) enforce that each passenger gets on and gets off the same bus. Constraints (A.13) make sure that each passenger is served at most once. Together with constraints (A.18), every request is served once and only once. Constraints (A.14) forbid two consecutive stations from being the same. Constraints (A.15) calculate the net capacity at each station, which is equal to the number of passengers getting on minus the one of getting off. Consequently, constraints (A.16) forbid the violation of bus capacity. Constraints (A.17) calculate the URT of each passenger. Constraints (A.19 - A.22) define the range for each variable.

$$\min \text{URT} = \sum\nolimits_p T_p \tag{A.1}$$

s.t.

$$\sum\nolimits_s x_{skb} \leq 1 \qquad \forall k \in K, b \in B \tag{A.2}$$

$$\sum\nolimits_s (x_{skb} - x_{s(k+1)b}) \geq 0 \qquad \forall k \in K, b \in B \tag{A.3}$$

$$M \sum\nolimits_s x_{skb} - \sum\nolimits_p (y^u_{pkb} + y^o_{pkb}) \geq 0 \qquad \forall k \in K, b \in B \tag{A.4}$$

$$\sum\nolimits_s x_{skb} - \sum\nolimits_p (y^u_{pkb} + y^o_{pkb}) \leq 0 \qquad \forall k \in K, b \in B \tag{A.5}$$

$$x_{skb} + y^u_{pkb} - a^u_{ps} \leq 1 \qquad \forall s \in S, k \in K, p \in P, b \in B \tag{A.6}$$

$$x_{skb} + y^o_{pkb} - a^o_{ps} \leq 1 \qquad \forall s \in S, k \in K, p \in P, b \in B \tag{A.7}$$

$$t^a_{(k+1)b} - t^d_{kb} - TT_{ss'} + (x_{skb} + x_{s'(k+1)b} - 2)(-M) \geq 0 \qquad \forall s, s' \in S \mid s \neq s', k \in K, b \in B \tag{A.8}$$

$$t^d_{kb} - e_p + (y^u_{pkb} - 1)(-M) \geq 0 \qquad \forall p \in P, k \in K, b \in B \tag{A.9}$$

$$t^a_{kb} - l_p + (y^o_{pkb} - 1)M \leq 0 \qquad \forall p \in P, k \in K, b \in B \tag{A.10}$$

$$\sum\nolimits_k (k y^u_{pkb} - k y^o_{pkb}) \leq 0 \qquad \forall p \in P, b \in B \tag{A.11}$$

$$\sum\nolimits_k (y^u_{pkb} - y^o_{pkb}) = 0 \qquad \forall p \in P, b \in B \tag{A.12}$$

$$\sum\nolimits_b \sum\nolimits_k y^u_{pkb} \leq 1 \qquad \forall p \in P \tag{A.13}$$

$$x_{skb} + x_{s(k+1)b} \leq 1 \qquad \forall s, k, b \tag{A.14}$$

$$\sum\nolimits_p (y^u_{pkb} - y^o_{pkb}) - q_{kb} = 0 \qquad \forall k \in K, b \in B \tag{A.15}$$

$$\sum\nolimits_{k' \leq k} q_{k'b} \leq Q \qquad \forall k, k' \in K \mid k \geq k', b \in B \tag{A.16}$$

$$T_p + (2 - y^o_{pk'b} - y^u_{pkb})M - t^a_{k'b} + t^d_{kb} \geq 0 \qquad \forall k, k' \in K \mid k' > k, p \in P, b \in B \tag{A.17}$$

$$\sum\nolimits_p \sum\nolimits_b \sum\nolimits_k y^u_{pkb} = |P| \tag{A.18}$$

$$x_{skb} \in \{0, 1\} \qquad \forall s \in S, k \in K, b \in B \tag{A.19}$$

$$y^u_{pkb} \in \{0, 1\} \qquad \forall p \in P, k \in K, b \in B \tag{A.20}$$

$$y^o_{pkb} \in \{0, 1\} \qquad \forall p \in P, k \in K, b \in B \tag{A.21}$$

$$q_{kb} \in \mathbb{Z} \qquad \forall k \in K, b \in B \tag{A.22}$$

Appendix B. Computational results

Tables B.4 - B.6 present the results of the DODBRP. In terms of the notations, we note $|P|$ as the number of served passengers, and we denote the average URT per passenger as U, then "dis" and "std" is respectively the abbreviation of "traveled distance" and "standard deviation". The results in each row are of the corresponding instance in Table 2. For example, the first row presents the results of the instance "100", while the second row "200", etc. Finally, each value in the last row is the average of the above.

Table B.4: DODBRP $prob = 0.1$, number of served passengers, average URT per passenger, and traveled distance per bus

***prob* = 0.1, int = 0**						***prob* = 0.1, int = 1**					
$\|P\|$	**std ($\|P\|$)**	**U in min**	**std(U)**	**dis in km**	**std(dis)**	$\|P\|$	**std ($\|P\|$)**	**U in min**	**std(U)**	**dis in km**	**std(dis)**
23.2	4.7	18.6	1.4	30.5	8.5	22.9	2.6	19.0	1.6	30.7	12.3
46.8	6.6	17.3	0.9	36.9	10.8	46.8	5.4	17.3	0.9	36.9	9.1
63.5	5.0	22.5	1.2	48.4	10.7	63.7	9.2	22.5	1.5	48.1	13.7
88.0	6.3	17.7	0.5	45.5	4.2	87.1	7.4	17.7	0.7	45.7	4.9
109.7	8.5	19.4	0.9	44.4	6.9	109.1	6.2	19.3	1.1	44.5	7.5
132.7	7.6	18.9	0.5	49.1	3.7	131.5	7.3	18.7	0.6	49.1	4.4
148.7	4.9	23.4	0.7	50.7	4.5	150.5	7.4	23.4	0.7	51.2	4.7
165.1	7.7	19.7	0.4	52.9	2.9	166.1	6.4	19.9	0.5	53.0	2.0
193.8	9.0	22.8	0.4	45.4	3.7	193.4	8.4	23.0	0.6	45.5	3.2
224.9	12.6	20.7	0.8	46.0	4.5	223.7	10.6	20.7	0.9	46.1	4.4
315.4	5.9	20.5	0.6	44.8	2.1	315.5	4.7	20.4	0.4	45.0	3.5
429.3	14.8	20.5	0.5	49.0	3.0	428.3	11.8	20.4	0.5	49.0	2.2
655.0	22.5	21.2	0.4	53.5	2.2	657.9	21.8	21.3	0.3	53.6	2.8
199.7		20.2		45.9		199.7		20.3		46.0	

Table B.5: DODBRP $prob = 0.25$, number of served passengers, average URT per passenger, and traveled distance per bus

***prob* = 0.25, int = 0**						***prob* = 0.25, int = 1**					
$\|P\|$	**std ($\|P\|$)**	**U in min**	**std(U)**	**dis in km**	**std(dis)**	$\|P\|$	**std ($\|P\|$**	**U in min**	**std(U)**	**dis in km**	**std(dis)**
27.0	4.2	19.0	1.8	35.2	17.5	26.2	4.7	19.9	1.4	36.2	13.6
54.2	6.0	18.4	1.3	42.1	7.2	52.3	7.6	18.6	1.3	42.2	10.0
68.0	5.3	23.6	1.2	50.0	4.5	69.3	5.3	23.5	1.0	50.4	6.8
96.8	7.7	18.8	1.2	47.6	7.6	94.6	9.1	18.7	1.2	48.1	8.5
117.3	10.4	20.0	0.7	46.0	3.9	116.7	9.1	20.1	0.9	46.1	5.7
140.3	10.6	19.7	0.7	51.3	5.7	140.8	12.6	19.6	0.5	51.4	7.0
158.1	13.6	24.1	0.9	52.0	5.0	158.0	7.4	24.0	0.8	52.1	4.2
178.9	9.0	20.7	0.5	54.6	5.6	177.1	10.8	21.0	0.6	54.6	5.2
207.4	9.7	23.9	0.5	47.2	4.3	209.1	9.4	24.0	0.6	47.8	4.7
236.5	14.0	21.6	0.6	47.4	3.1	239.4	11.3	21.7	0.4	47.7	6.0
336.2	9.9	21.6	0.3	46.7	3.8	336.3	10.1	21.6	0.5	46.8	3.2
468.2	14.4	21.3	0.7	50.3	3.2	461.9	9.8	21.4	0.7	50.5	2.2
714.3	21.8	22.5	0.3	54.9	2.0	713.5	26.7	22.5	0.4	55.3	3.2
215.6		21.2		48.1		215.0		21.3		48.4	

Table B.6: DODBRP $prob = 0.5$, number of served passengers, average URT per passenger, and traveled distance per bus

***prob* = 0.5, int = 0**						***prob* = 0.5, int = 1**					
$\|P\|$	**std ($\|P\|$)**	**U in min**	**std(U)**	**dis in km**	**std(dis)**	$\|P\|$	**std ($\|P\|$)**	**U in min**	**std(U)**	**dis in km**	**std(dis)**
29.5	4.3	19.2	1.1	38.2	9.3	29.5	6.0	19.2	2.2	38.6	10.8
57.6	7.4	19.6	1.8	44.8	11.1	56.5	6.7	20.3	2.2	45.5	13.5
73.6	8.9	24.6	1.0	52.7	8.5	75.1	9.3	24.5	1.6	52.9	10.5
104.3	13.2	19.7	1.1	49.7	7.2	102.3	11.5	19.8	1.3	50.6	8.4
123.9	7.7	21.5	0.6	48.1	5.0	123.7	9.6	21.2	0.8	47.9	4.9
157.2	13.6	20.8	0.9	53.2	9.0	154.0	12.8	20.6	0.9	53.5	9.7
168.9	11.1	25.2	0.7	53.4	3.7	171.3	12.5	25.4	0.8	54.1	4.4
194.8	8.8	21.5	0.4	56.0	3.0	193.2	11.6	22.0	0.9	56.5	4.8
230.5	11.4	25.2	0.6	49.9	4.3	230.6	10.9	25.1	0.7	50.2	5.1
260.9	9.5	23.3	0.6	49.7	6.2	259.7	18.1	23.1	0.5	49.9	6.1
361.0	12.6	22.8	0.5	48.1	4.4	360.3	14.2	22.5	0.7	48.2	4.5
521.5	14.5	22.6	0.4	51.9	2.8	517.8	25.4	22.6	0.3	52.5	2.2
803.1	35.8	24.1	0.3	56.9	3.2	806.7	33.8	24.0	0.5	57.2	3.8
237.4		22.3		50.2		237.0		22.3		50.6	

Next, Tables B.7 - B.12 present the results of prepositioning relative to the DODBRP. The corresponding values are thus in percentage. $|P|$ —the number of served passengers, U —URT per passenger, dis —traveled distance per bus.

Table B.7: Prepositioning under various time buckets with ideal forecast, $prob = 0.1$, int = 0

Instance	$\|P\|$				**U**				**dis**			
	H10	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**
100	4.5	9.1	9.1	9.1	2.9	0.9	0.9	0.9	0.0	1.6	1.6	1.6
200	4.3	4.3	4.3	4.3	-0.4	-0.4	-0.4	-0.4	0.0	0.0	0.0	0.0
300	1.6	3.2	3.2	3.2	-1.6	-0.6	-0.6	-0.6	0.0	0.0	0.0	0.0
400	3.3	3.3	3.3	3.3	0.5	0.5	0.5	0.5	1.1	1.1	1.1	1.1
500	1.9	1.9	1.9	1.9	-1.2	-1.2	-1.2	-0.9	0.0	0.0	0.0	0.0
600	2.3	3.8	5.3	4.6	0.4	1.4	-0.7	-0.9	-2.1	1.0	0.0	-1.0
700	2.6	2.6	3.9	3.9	2.1	2.1	3.1	3.1	-1.9	-1.9	0.0	0.0
800	2.4	4.8	6.0	6.0	-1.2	-2.5	-2.0	-2.0	0.0	1.0	1.9	1.9
900	1.5	3.1	5.1	5.1	1.7	2.1	2.0	2.0	-4.3	-3.3	-2.2	-2.2
1000	2.3	2.7	2.7	4.5	1.3	1.1	1.1	0.6	-1.1	-2.2	-2.2	-1.1
1500	2.2	2.5	2.5	2.5	1.0	0.8	0.8	0.8	-2.2	-2.2	-2.2	-2.2
2000	3.3	4.0	5.0	5.2	0.4	0.1	0.4	0.4	-2.0	-2.0	-1.0	-1.0
3000	1.8	2.6	2.6	2.6	0.8	0.2	0.2	0.2	-0.9	0.0	-0.9	-0.9
average	2.6	3.7	4.2	4.3	0.5	0.3	0.3	0.3	-1.0	-0.5	-0.3	-0.3

Table B.8: Prepositioning under various time buckets with ideal forecast, $prob = 0.1$, int = 1

Instance	$\|P\|$				**U**				**dis**			
	H10	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**
100	0.0	4.2	4.2	4.2	1.3	-0.6	-0.6	-0.6	-1.6	1.6	1.6	1.6
200	2.1	2.1	2.1	2.1	-2.1	-2.1	-2.1	-2.1	-1.4	-1.4	-1.4	-1.4
300	3.2	4.8	4.8	4.8	1.9	1.9	1.9	1.9	1.0	1.0	1.0	1.0
400	5.7	6.8	6.8	6.8	-1.1	-1.9	-1.8	-1.9	-1.1	0.0	0.0	0.0
500	0.9	0.9	0.9	0.9	0.3	0.3	0.3	1.6	0.0	0.0	0.0	-1.2
600	4.7	6.3	7.8	9.4	2.8	1.2	2.8	3.4	-2.1	2.1	0.0	-2.1
700	3.2	2.6	2.6	3.2	3.2	3.6	3.6	3.3	1.0	1.0	1.0	1.0
800	7.3	9.1	9.1	9.1	-2.4	-3.2	-3.2	-3.2	0.0	0.9	0.9	0.9
900	3.6	3.6	4.6	5.2	2.6	2.6	3.0	3.3	-4.3	-4.3	-3.2	-3.2
1000	2.3	3.2	3.7	4.1	0.1	-0.5	-0.3	-0.3	-1.1	-1.1	-1.1	-1.1
1500	1.3	1.3	1.3	1.3	-0.1	-0.1	-0.1	-0.1	-2.1	-2.1	-2.1	-2.1
2000	2.4	3.1	3.5	3.5	0.3	-0.1	0.4	0.4	0.0	0.0	0.0	0.0
3000	2.6	3.3	3.3	3.6	-0.5	-0.7	-1.2	-1.3	-0.9	-0.9	-0.9	-0.9
average	3.0	3.9	4.2	4.5	0.5	0.0	0.2	0.3	-1.0	-0.2	-0.3	-0.6

Table B.9: Prepositioning under various time buckets with ideal forecast, $prob = 0.25$, int = 0

Instance	$\|P\|$				U				dis			
	H10	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**
100	3.4	3.4	3.6	3.6	-0.2	-0.2	-0.2	3.7	4.5	4.5	4.5	7.5
200	1.9	3.8	5.8	5.8	-1.5	-2.7	-0.3	-0.3	-2.4	-2.4	-1.2	-1.2
300	7.1	10.0	12.9	12.9	-4.4	-5.6	-7.1	-7.1	2.0	2.0	2.0	2.0
400	2.0	4.0	4.0	5.0	-4.1	-5.1	-5.0	-5.3	0.0	2.1	2.1	1.0
500	10.7	8.0	12.5	12.5	-1.9	-1.7	-1.6	-1.6	2.2	0.0	2.2	2.2
600	1.3	4.7	2.7	6.0	0.7	-0.3	0.6	0.5	-1.9	0.9	-1.9	-0.9
700	6.7	7.9	6.7	6.7	-0.6	-1.9	-0.8	-0.8	0.0	0.0	0.0	0.0
800	3.4	5.6	5.6	5.6	0.2	1.5	1.0	1.0	0.9	2.8	1.9	1.9
900	2.4	5.2	6.2	6.6	1.7	2.1	1.8	1.4	-3.0	-3.0	-2.0	-2.0
1000	6.2	7.1	7.5	9.3	0.3	0.0	-0.1	-0.6	0.0	1.1	1.1	1.1
1500	3.9	4.2	4.8	4.8	0.9	0.6	0.5	0.5	-2.1	-1.1	-1.1	-1.1
2000	5.1	5.8	7.5	7.7	-1.3	-1.7	-0.8	-0.9	-1.0	-1.0	-1.0	-1.0
3000	4.1	5.6	6.2	6.3	-1.8	-2.5	-2.8	-2.8	-0.9	-0.9	-0.9	-0.9
average	4.5	5.8	6.6	7.1	-0.9	-1.4	-1.1	-0.9	-0.1	0.4	0.4	0.7

Table B.10: Prepositioning under various time buckets with ideal forecast, $prob = 0.25$, int = 1

Instance	$\|P\|$				U				dis			
	H10	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**
100	3.8	7.7	7.7	7.7	7.1	8.0	8.0	8.1	7.6	7.6	7.6	7.6
200	1.9	5.8	7.7	7.7	-0.5	-1.8	0.5	0.5	0.0	-2.4	-1.2	-1.2
300	7.0	7.0	8.5	8.5	0.1	-0.2	0.2	0.4	0.0	0.0	1.0	1.9
400	9.8	10.9	10.9	10.9	-8.0	-8.6	-8.6	-8.3	-3.0	-3.0	-3.0	-4.0
500	2.6	1.7	2.6	3.4	5.8	4.4	6.3	6.3	-2.2	-1.1	-2.2	-1.1
600	2.7	5.4	6.8	7.5	1.0	2.9	3.9	3.6	-4.8	-1.9	-3.8	-3.8
700	3.0	4.9	4.9	6.7	4.3	3.0	3.0	2.4	-0.9	-0.9	-0.9	-0.9
800	3.4	5.7	6.3	6.3	0.6	0.9	0.8	0.7	0.0	0.9	0.9	0.9
900	1.9	3.2	4.6	4.6	1.8	0.8	0.7	0.7	-4.0	-5.0	-4.0	-4.0
1000	3.8	3.8	4.7	4.7	-0.8	-0.8	-1.1	-1.1	-3.1	-3.1	-3.1	-3.1
1500	3.9	5.1	5.1	5.1	0.4	-0.8	-0.8	-0.8	-3.2	-3.2	-3.2	-3.2
2000	5.6	6.2	6.7	6.9	-2.0	-2.3	-2.4	-2.4	-1.0	-1.0	-1.0	-1.0
3000	2.4	3.3	3.8	4.2	-0.3	-0.2	-0.4	-0.4	-0.9	-0.9	-0.9	-0.9
average	4.0	5.4	6.2	6.5	0.7	0.4	0.8	0.7	-1.2	-1.1	-1.1	-1.0

Table B.11: Prepositioning under various time buckets with ideal forecast, $prob = 0.5$, int = 0

Instance	$\|P\|$				U				dis			
	H10	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**
100	7.1	7.1	7.1	10.7	-10.1	-10.1	-5.5	-4.9	-1.4	-1.4	-1.4	0.0
200	7.1	10.7	10.7	10.7	-1.0	-3.1	-3.1	-3.1	0.0	1.1	1.1	1.1
300	11.8	14.5	15.8	17.1	-0.6	-0.1	0.2	-3.6	1.9	1.9	1.9	1.9
400	1.9	2.8	3.7	5.6	-0.4	-0.3	0.8	-0.7	3.0	4.0	4.0	2.0
500	9.6	12.8	16.8	16.8	-0.1	-3.0	-2.0	-2.0	0.0	0.0	0.0	0.0
600	6.3	10.1	10.8	12.0	0.0	-0.6	-1.0	-1.3	-0.9	1.9	0.0	0.0
700	6.6	8.4	7.2	7.2	3.5	1.9	3.6	3.6	1.9	2.9	2.9	2.9
800	4.1	5.7	7.2	7.2	-1.6	-1.7	-2.3	-2.3	0.0	1.8	0.0	0.0
900	5.5	6.4	8.1	10.2	1.2	1.3	1.0	1.4	-2.9	-2.9	-1.9	-1.9
1000	8.8	9.6	11.1	12.3	1.8	1.2	0.7	0.7	0.0	0.0	0.0	0.0
1500	3.9	4.2	3.6	4.2	0.0	0.0	0.3	0.8	-1.1	-1.1	-1.1	-1.1
2000	3.6	3.6	4.2	4.3	-0.2	-0.2	0.9	1.9	-0.9	-0.9	-1.9	-1.9
3000	2.8	3.9	4.7	4.9	-1.5	-1.7	-1.1	-1.0	0.0	0.0	0.0	0.0
average	6.1	7.7	8.5	9.5	-0.7	-1.3	-0.6	-0.8	0.0	0.6	0.3	0.2

Table B.12: Prepositioning under various time buckets with ideal forecast, $prob = 0.5$, int = 1

Instance	$\|P\|$				U				dis			
	H10	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**	**H10**	**H20**	**H40**	**H120**
100	11.5	15.4	19.2	19.2	-0.7	-1.6	-2.0	-5.7	-2.7	-2.7	-4.0	-8.0
200	7.5	13.2	13.2	17.0	-5.7	-9.9	-9.9	-9.9	1.1	2.3	2.3	-1.1
300	9.3	10.7	13.3	14.7	-3.9	-4.6	-2.8	-4.7	0.0	1.0	1.0	1.0
400	1.9	1.9	2.9	4.8	-2.8	-2.8	-3.1	-3.5	0.0	0.0	0.0	0.0
500	14.9	12.4	17.4	17.4	2.0	3.2	2.2	2.2	0.0	-1.1	0.0	0.0
600	5.8	9.1	8.4	8.4	-1.8	-3.2	-4.3	-3.7	-6.4	-2.8	-5.5	-5.5
700	1.7	2.9	2.9	2.9	0.2	-0.3	-0.3	-0.4	-1.9	-1.9	-1.9	-1.9
800	4.7	6.3	6.3	6.3	-0.8	-1.6	-1.6	-1.6	-1.8	-1.8	-1.8	-1.8
900	7.5	7.9	9.2	10.1	0.7	0.4	1.1	1.4	-1.9	-1.9	-1.9	-1.9
1000	6.3	7.5	8.6	10.4	-2.4	-1.6	-1.4	-2.3	0.0	-1.0	-1.0	0.0
1500	5.1	5.9	5.9	5.9	3.2	3.0	3.3	3.3	-3.1	-3.1	-2.1	-2.1
2000	5.2	6.1	6.5	6.9	0.6	0.2	0.5	0.5	-1.9	-1.9	-0.9	-0.9
3000	3.9	4.4	4.8	5.2	0.6	0.2	0.2	0.1	-0.9	-0.9	-0.9	-0.9
average	6.6	8.0	9.1	9.9	-0.8	-1.4	-1.4	-1.9	-1.5	-1.2	-1.3	-1.8