DEPARTMENT OF ENGINEERING MANAGEMENT

The integrated on-demand bus routing problem

Lissa Melis, Michell Queiroz & Kenneth Sörensen

UNIVERSITY OF ANTWERP
Faculty of Business and Economics
City Campus
Prinsstraat 13, B.226
B-2000 Antwerp
Tel. +32 (0)3 265 40 32
www.uantwerpen.be

FACULTY OF BUSINESS AND ECONOMICS

DEPARTMENT OF ENGINEERING MANAGEMENT

The integrated on-demand bus routing problem

Lissa Melis, Michell Queiroz & Kenneth Sörensen

RESEARCH PAPER 2021-004
AUGUST 2021

University of Antwerp, City Campus, Prinsstraat 13, B-2000 Antwerp, Belgium
Research Administration – room B.226
phone: (32) 3 265 40 32
e-mail: joeri.nys@uantwerpen.be

D/2021/1169/004

The integrated on-demand bus routing problem

Lissa Melis, Michell Queiroz, Kenneth Sörensen

August 2021

Abstract

In this work we analyse the performance of integrating a large-scale on-demand bus system with a fixed line public transport network in an urban context. Given are a high-speed metro network, a set of real-time requests, a set of bus station locations and a fleet of fixed capacity minibuses. Requests have a set of possible departure/arrival[1] bus stations within walking distance of the actual departure/arrival location and have to be served within a certain time window. The aim is to simultaneously (1) decide on the trip type for each passenger (only bus, metro or mixed), (2) route the on-demand buses, (3) assign each passenger to a departure and arrival bus station (bus station assignment), and (4) in the case of a metro-leg in the trip, decide the assigned transfer station(s) and used metro lines (transfer station assignment). We call this problem the integrated on-demand bus routing problem. After presenting a mathematical model, we propose a quick and scalable insertion-based heuristic to solve the problem.

The results found by the heuristic are further used to compare the performance of an integrated system, to a system that only uses on-demand buses. It is concluded that the integrated system always performs better regarding the service rate or number of served requests. Depending on the speed and layout of the metro network, also the average user ride time per passenger improves by the integration.

1 Introduction

Considerable high travel times and accessibility issues, are some of the reasons for discontent with the current operation of traditional public transportation. Consequently, people are increasingly drawn to use private cars, which provide more convenience and flexibility. The rise in number of vehicles in the streets inevitably leads to many negative effects, such as traffic congestion and increased emissions. Integrated transit systems, which include traditional, fixed route public transportation (FPT) and demand responsive transit (DRT), hold the potential to address these problems (Mourad et al., 2019). Consisting of a set of predefined fixed routes and schedules, FPT trips usually are performed by (trolley)buses, trams, trains and subways. In contrast, DRT adapts its routes and schedules according to bookings, i.e., requests are sent by users and are typically served by cars or minibuses.

DRT services, especially services with zero or low ride-sharing levels, e.g., Uber (Uber, 2019), have higher operating costs, take excessive road space, and increase emissions. Envisioning to diminish these negative impacts, *the on-demand bus service* emerged. Bus routes, instead of being pre-specified, would operate under the passengers' demands. An on-demand bus service aggregates similar travel requests in space and time by allowing ride-sharing on a larger scale. And, as more people decide to use the system instead of their private cars, traffic congestion is likely to be reduced. If provided in large scale, such system can also lead to enhance the level of service provided while decreasing costs to the passenger (Archetti et al., 2018a).

The optimization problem related to an on-demand bus service is firstly introduced by (Melis and Sörensen, 2020, 2021), and called the *on-demand bus routing problem (ODBRP)*. Given are a fleet of fixed capacity buses, a network with a set of bus stations[2] and travel times between them, along with a collection of transportation requests. In order for these requests to be fulfilled, each passenger must be assigned to a departure and arrival bus station within walking distance, while the routes of the buses must respect each request's time window, delineated by an earliest departure and latest arrival time. By assigning stations to requests (called *bus station assignment*), instead of directly guiding

[1] We will use the terms origin and departure, as well as the terms destination and arrival interchangeably.

[2] The term "bus station" is referred to as the physical location where a bus stops and the term "bus stop" as the buses' activity of stopping. A bus station is not necessarily a permanent physical shelter, but can also be simple pile or a (virtually) assigned place on the streets.

passengers to the nearest station, passenger demand can be pooled even more and the flexibility to do the routing of the on-demand buses rises. The objective is to minimize the total user ride time (URT). Beyond introducing the problem, the authors also present a large neighborhood search heuristic with embedded local search to solve diverse problem instances. In addition, the ODBRP is compared with the use of FPT and it is found that URT's can be decreased considerably by using the on-demand buses. In later work, the authors investigate the cost of allowing real-time requests by introducing the dynamic ODBRP (Melis and Sörensen, 2021).

The ODBRP is intended to be conceived in an urban, high-demand, environment. Urban mobility is the context where most of transportation challenges are rising due to broad and dispersed demand and unpredictable movement patterns. Cities are experiencing a phenomenon of growth in overall population, even though the population density is decreasing (Meijers and Burger, 2010). This in turn motivates the design of new systems to serve this diffused population and enhance urban mobility, such as the one presented in this paper. Accordingly, in this paper we study the operational and performance aspects of the implementation of a system that consists of integrating an on-demand bus (ODB) service with a fixed route public transport system (FPT). The FPT network is presumed to be a high-frequency network, e.g. an underground metro network. The ODB service is operated by a set of fixed capacity minibuses. We refer to the studied problem as the *Integrated On-Demand Bus Routing Problem (I-ODBRP)*.

Users send a transportation request informing their departure and arrival locations. A time window consisting of an earliest departure time and a latest arrival time is also specified. The system responds to the user with an itinerary as a proposal that fits the time window constraints. Both in the ODBRP and the I-ODBRP, requests can be static or dynamic. In the former, all requests are known beforehand, while in the latter, requests are continuously presented during the course of the day, and vehicle routes are adjusted in real-time. In this paper, requests are dynamic and presumed to be sent just-in-time, as this resembles reality the most. We solve the problem deterministically, and leave the incorporation of future demand in the scheduling process for future research.

In the I-ODBRP on-demand buses (ODB) are combined with a fixed, high-frequency metro network (FPT). Consequently we assume five possible *trip types* which are explained below.

(a) ODB: The trip is composed of a single leg carried out by an on-demand bus. Passengers walk towards the assigned departure station where they board the on-demand bus and travel to the assigned arrival station. From here they walk to their destination location.

(b) ODB + FPT: The trip is composed of two legs. The first is carried out by an on-demand bus, the second by metro. Passengers walk towards the assigned departure station where they board the on-demand bus. They get of the bus at the assigned transfer station[3] and board a vehicle that follows the designated fixed route. They travel to the assigned arrival bus station. From here they walk to their destination location.

(c) FPT + ODB: The trip is composed of two legs. The first is carried out by metro, the second and final by an on-demand bus. Passengers walk towards the assigned departure metro station, where they board a vehicle that follows the designated fixed route. Passengers travel to the assigned transfer station and board the on-demand bus to travel to the assigned arrival metro station. From here they walk to their destination location.

(d) ODB + FPT + ODB: The trip is composed of three legs. The first is carried out by an on-demand bus, the second by metro, and the third by another on-demand bus. Passengers walk towards the assigned departure station where they board the on-demand bus. Once at the assigned transfer station, passengers board a vehicle that follows the designated fixed route. Once at the second transfer station, passengers board another on-demand bus and travel to the assigned arrival station. From here they walk to their destination location. Throughout the paper we will refer to the two ODB-legs as leg 1 and leg 3, leg 2 is the FPT-leg.

(e) FPT: The trip is composed of a single leg carried out by fixed route public transport. Passengers walk towards the assigned departure metro station, where they board a vehicle that follows the designated fixed route and travels to the assigned arrival metro station. From here they walk to their destination location.

Note that when the itinerary has a FPT-leg, it is possible for passengers to have to switch lines within the metro network to reach their assigned transfer or destination station. These transfers *within the metro network* are presumed to be highly efficient and unlimited in number.

[3] The term "transfer station" refers to stations where passengers switch vehicle type: from ODB to metro, or vice versa.

The algorithm has four decisions to make: (i) Decide on the trip type for each passenger, (ii) Routing of the on-demand buses , (iii) Assigning each passenger to a departure and arrival bus station within walking distance of their actual departure and arrival location (bus station assignment), (iv) In the case of a FPT leg in the trip, decide the assigned transfer station(s) and used metro lines (transfer station assignment).

The objective of the I-ODBRP is the same as for the ODBRP, we minimize the total user ride time. However, instead of only considering the time passengers spend on the buses, the total user ride time now consists of the time between the first pick up at an assigned pick up station and the last drop-off at an assigned drop-off station. The first pick up and last drop-off can be by bus or by metro depending on the trip type. Nevertheless, other objective functions are possible to measure performance, such as profit-related (e.g. minimize the total distance traveled by the buses) and completeness-related ones (e.g. maximize the number of served passengers). We opt to minimize the total URT aiming to improve the level of service. This is a rational option to attract more users to the system, which can have a meaningful effect on its profitability.

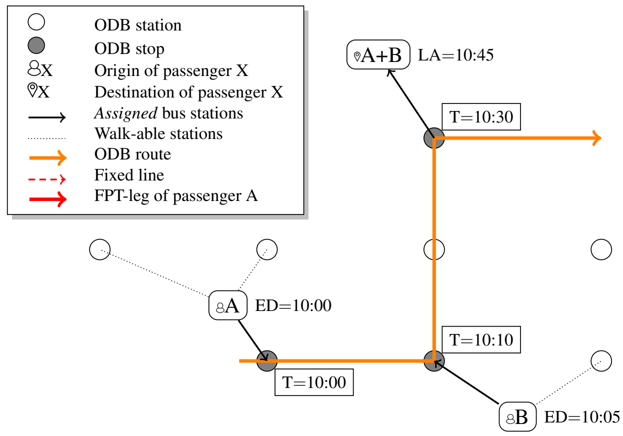
Decisions (ii) and (iii) are introduced in Melis and Sörensen (2020, 2021) as part of the ODBRP, while (i) and (iv) are specific for the I-ODBRP. Decision (i) refers to the different possible trip types explained before. Decision (iv) is an extension of the bus station assignment procedure of decision (iii). Bus station assignment is still supported by the fact that in cities, because of safety reasons, a bus cannot stop anywhere. Buses have higher capacities than vehicles utilized for most ride-sourcing systems. Therefore, buses should stop only at signaled bus stations. Melis and Sörensen (2020, 2021) proof that bus station assignment decreases the total URT considerably and makes it possible to serve more requests with a given fleet size. Therefore, in the I-ODBRP, bus station assignment is expanded to the fixed transport legs as well and called *transfer station assignment*. This involves that the operator will also assign the transfer stations for the fixed leg journey, if there is any. This adds even more flexibility to the scheduling of the requests, but also makes the problem characteristics more complex compared to the non-integrated ODBRP.

Figure 1 illustrates the increase in solution possibilities when going from the ODBRP to the I-ODBRP. Figure 1a shows two passenger requests (A and B) scheduled according to the ODBRP, by only using one on-demand bus. Figure 1b shows the same passenger requests, however, the on-demand bus can be integrated with the fixed metro line shown in the figure. Passenger A is consequently assigned a multi-leg trip consisting of a FPT (indicated by the red arrow) and ODB leg. The itinerary of passenger B remains the same in the example, however because the work load of the on-demand buses decreases, the time schedule could be adapted. By considering multi-leg trips, instead of only considering direct rides by on-demand bus, the total user ride time can be further decreased.

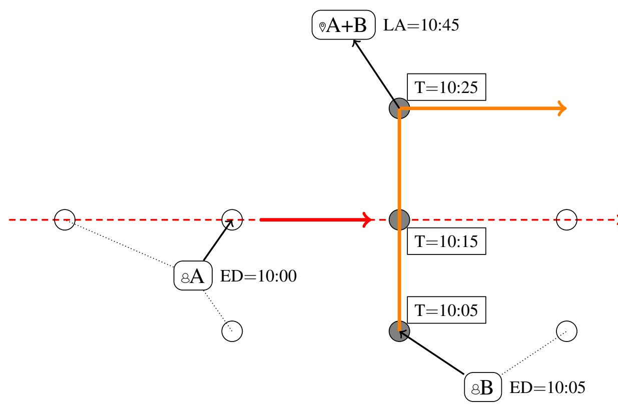
The remainder of this paper is organized as follows. Section 2 presents a literature review on the topic of combining demand responsive transit and fixed routes. The I-ODBRP is formally introduced in Section 3, followed by a mixed integer programming (MIP) formulation. To the best of our knowledge, this is the first time this problem is tackled in the literature. In Section 4, we propose a heuristic approach based on greedy insertion and efficient on-demand vehicle and fixed leg assignment to obtain solutions of real size problem instances. Section 5 analyses different aspects of the proposed algorithm. Section 6 investigates the performance of an integrated on-demand bus system by comparing such system to a system that uses only on-demand buses. Lastly, we present some conclusions and future research in Section 7.

2 Literature review and research gap

From a modeling perspective the I-ODBRP proposed in this paper is an extension of the ODBRP, which is a combination of the dial-a-ride problem (DARP), the School Bus Routing Problem (SBRP) and the Pick-up and Delivery Problem with Time Windows (PDPTW). The DARP is most closely related to the ODBRP, but differs in terms of the extra constraint on the maximum ride time, the two pairs of time windows (both on the pick-up and the delivery), and mostly the objective. In addition, it is a door-to-door service that does not use bus stations or passenger pooling through bus station assignment and it generally used in rural, low-demand areas. The static DARP that combines DAR-vehicles with fixed public transport is called the Bimodal or Integrated DARP and is solved by Liaw et al. (1996), Aldaihani and Dessouky (2003), Häll et al. (2009), Posada et al. (2017), Posada and Häll (2020) and Molenbruch et al. (2020). Liaw et al. (1996) show an increase of 10% of requests that can be served along with a decrease of 10% in the number of vehicles required compared to when no fixed route buses are considered for instances up to 85 requests. Aldaihani and Dessouky (2003) have shown that such integration reduces the vehicles' traveled distances, meanwhile



(a) Non-integrated ODBRP - Example solution



(b) Integrated ODBRP - Example solution

Figure 1: ODBRP versus I-ODBRP: more routing opportunities

customers riding with combined trips experience an increase of travel time of about 5%. Both authors use a simple heuristic algorithm. Häll et al. (2009) and Posada et al. (2017) present a mathematical model for the problem. While the first considers the frequency of the FPT to be high, the second takes into account a timetable of the fixed lines. Posada and Häll (2020) propose a meta-heuristic based on Adaptive Large Neighborhood Search (ALNS), and a case study with real-world data of a rural area that shows a reduction of 16% of distance driven by the DAR-vehicles. The solution approach of Molenbruch et al. (2020) consists of an exact method incorporated into a LNS to synchronize the routing of the DAR with FPT. Experiments display operational savings for the DAR providers when integration with FPT is considered. Another ALNS-based heuristic for the DARP with Transfers is proposed in Masson et al. (2014). In this problem passengers can transfer from one DAR-vehicle to another. According to the computational experiments carried by the authors on real life instances, the introduction of transfers lead to savings around 8%.

In contrast to the DARP, adopting a door-to-door service, the SBRP uses stations, and in this problem bus station assignment was initially introduced (Schittekat et al., 2013). However, it adopts a many-to-one instead of many-to-many setting. Bögl et al. (2015) study the SBRP with Transfers. It differs from the traditional SBRP in the sense that students might change buses during transportation from their home to school. With the objective of minimizing total operating costs, the authors present a mathematical model and develop a heuristic algorithm. Results show that allowing transfers significantly reduces total operating costs while user ride times are equivalent to solutions without transfers.

The PDPTW, is also similar to the ODBRP, but handles freight requests. Transporting goods instead of people brings less complexities regarding the quality of service, e.g., the time spend on the vehicle. Ghilas et al. (2016) propose an ALNS heuristic for the PDPTW and Scheduled Lines (PDPTW-SL). For the PDPTW-SL, the goal is to serve freight requests by scheduling a set of vehicles with the possibility of combining part of the journey with a fixed scheduled public transportation line. The authors compare solutions with the regular PDPTW, and conclude that the PDPTW-SL leads to significant cost savings and fewer CO_2-emissions.

The problem of an integrated public[4] transport system, i.e., a combination of ride-sourcing and fixed routes was firstly introduced by Wilson et al. (1976) and Potter (1976). More recently an investigation of integrated transit service in a low demand area is performed in Hickman and Blume (2001). The authors use limited geographical circles around the origin and arrival of a request to determine whether or not a fixed transport leg is a possibility. The authors take into account the passenger's level of service constraints, such as maximum travel time and number of transfers and handle requests in a dynamic way. The integrated transit system saves around 15% of operating costs. Edwards et al. (2011) also integrate traditional transit with demand-responsive vehicles by introducing the Network-Inspired Transportation System (NITS). The authors route passengers analogically to routing packets in a telecommunications network. The performance of the NITS is evaluated with simulation studies in Edwards et al. (2012), which concludes that the system produce better results in low density urban areas.

Mahéo et al. (2019) presents the BusPlus, a project that consists of combining high-frequency bus routes between key hubs with shuttles that transports passengers from their origin to the closest hub and also take them from the last bus stop to their destination. The authors focus on designing this multi-modal network in off-peak hours, by starting from a MIP formulation, then present a Benders decomposition approach. According to the results, such model may have an impact in decreasing the transit time by a factor of two. Extensions of the study presented in Mahéo et al. (2019) are performed by Auad and Van Hentenryck (2021); Basciftci and Van Hentenryck (2021); Auad et al. (2021), however solution methods remain based on the MIP, without using heuristics. The authors intoduce a new classification: On-Demand Multi-modal Transit System (ODMTS). The routes of the shuttles involve three types: direct non-ride-sharing type, drop-off or pick-up type. In the last a shuttle picks up several passengers before dropping them off at a common transport hub, while in the second, a shuttle drops off several passengers after picking them up at a common transport hub. Contributions of these papers consist of the inclusion of latent demand, i.e., new riders adopting the system, ride-sharing in the shuttle rides, and the examination of the system's resiliency during a pandemic scenario. Auad and Van Hentenryck (2021) demonstrates that ride-sharing for ODMTS has the potential of reducing costs by about 26% with less than 5% of increase in travel times. They also run the model for a 12-hour period, which included both peak and off-peak hours. Meanwhile, results presented by Basciftci and Van Hentenryck (2021) show high adoption rates and shorter trip duration compared to the existing transit system. Moreover, Auad et al. (2021) show the flexibility of ODMTS, as it responds well in terms of cost, convenience, and accessibility for the ample range of scenarios during a pandemic.

Steiner and Irnich (2020) combine fixed bus lines with demand-responsive transport for first/last leg journeys. The authors decide on the passenger routes, the existence of line segments for the fixed route network, on which areas of the city the demand-responsive system should cover, and how the two modes interact via transfer points. The authors test their model on instances generated with real-world data in a medium-sized city. They solve the problem using a branch-and-price algorithm and an advanced enumeration based approach. Narayan et al. (2020) develop a multi-modal route choice and assignment model to study the combination of fixed route public transport and on-demand services. The authors show that demand-responsive system covers less than 30% of the trip length, and is mostly used to serve as access to the FPT. The combined use of transport modes has a significant impact on reducing the waiting time of passengers in comparison when the modes are used separately. Also Zhu et al. (2020) propose a network model in which demand-responsive transport works as both feeder and competitive service to public transit, but they focus more on the fare ratios between modes.

Both Shen et al. (2018) and Salazar et al. (2018) perform a simulation study of a system that integrates Automated Vehicles (AV) and public transportation. The first replace low demand bus routes with shared AV. The second focuses on a pricing and tolling scheme that allows to achieve a social optimum under the assumption of a perfect market. Both authors report improved service quality, less emissions and costs, but they also mention the need for a real-time

[4]In this section we focus on public transport services that operate with professional drivers or fully autonomous. Ride-sharing literature using private vehicles, where typically a driver is matched with one or two riders who have a similar journeys, is not included in this review. Examples of this type of ride-sharing integrated with public transport can be found in Stiglic et al. (2018) and Kumar and Khani (2021).

operational algorithm to optimize efficiency.

To the best of our knowledge, this is the first time the ODBRP is integrated with fixed route public transport, making it the first time the I-OBDRP is studied in the literature. The main feature that differentiates the I-ODBRP is bus station assignment, which we will adopt both on the on-demand and fixed transport legs (then called transfer station assignment), making it possible to solve the problem in a high-demand urban environment. Finally, contrasting several previous integrated services that measured their performance based on operational costs, the I-ODBRP assesses the efficiency of the system regarding level of service, i.e., travel times. Especially when using autonomous electrical vehicles, the operating costs decrease immensely. The main goal is to alter the travel behaviour of people that use their private cars to public transport, which is more sustainable.

3 Problem formulation and mathematical model

Although a heuristic approach is used to perform the computational study and obtain solutions to realistic instances, we introduce a mixed integer programming (MIP) formulation to formally define the problem. The MIP formulation for the I-ODBRP is an extension of the model introduced in Melis and Sörensen (2020) for the ODBRP. In this section the problem is assumed to be static, meaning that all requests are known in advance. Further, the problem is solved and investigated in dynamic setting. All used notation is summarized in Table 1.

Let P be the set of transportation requests announced by passengers to the system. A transportation request $p \in P$ is characterized by an origin (o_p), a destination (d_p), as well as a time window, consisting of an earliest departure time (e_p^u) and a latest arrival time (l_p^o). The set of origins and destinations of the users are denoted as $O = \bigcup_{p \in P} o_p$ and $D = \bigcup_{p \in P} d_p$, respectively. To be feasible, a trip for request $p \in P$ must start after e_p^u and be completed before l_p^o. The maximum walking time (u_p) represents the passenger's willingness to walk, but is not included in the before-mentioned time window.

The problem is formulated over a network $G = (V, A)$ with a set V of nodes and a set A of arcs. V consists of four sets: bus stations S; stations used by fixed route public transport F; origins O; and destinations D. Accordingly, $V = S \cup F \cup O \cup D$. The network G is then partitioned in three subnetworks. First, consider a network $G_s = (S, A_s)$ with a set of bus stations $S \subset V$ and set of arcs $A_s \subset A$ connecting them. Associated with each arc $(i, j) \in A_s$ is an estimated travel time τ_{ij}^b by bus. Furthermore, consider the fixed route public transport network $G_f = (F, A_f)$ with a set $F \subset V$ of stations, along with a set of arcs $A_f \subset A$ connecting them. Each arc $(i, j) \in A_f$ holds the opportunity to travel within the fixed route network between stations i and j with estimated travel time τ_{ij}^f. Finally, $G_w = (V, A_w)$ contemplates a network safe for the movement of passengers with a set V of nodes linked by arcs $A_w \subset A$ indicating walkable paths. An estimated travel time by walk τ_{ij}^w is associated with each arc $(i, j) \in A_w$. Table 1 summarizes the used notation.

The system integrates two services: an on-demand bus service (ODB) and fixed route public transport service (FPT) or metro network. The different types of itineraries are described in Section 1. The ODB is composed by a fleet B of homogeneous buses with fixed capacity C. The FPT has unlimited capacity and is operated in a high-frequency manner, so we assume waiting times to be negligible. Transfers between ODB and FPT are assumed to occur at the same metro station, i.e., walking time is zero[5]. However, in the model presented in this section, they are indexed as different stations for generalization. If in future research walking between transfer stations would be included, the same model can be applied. Transfers within the metro network are unlimited. We remark that the unlimited number of transfers within the fixed route network is modeled according to the way variable z_{pij} is designed. Simply put, the variable indicates the possibility to travel between stations i and j, independently if they are served by same line within the fixed route network.

Let the route of bus $b \in B$ consist of a set $N_b = \{n_1, n_2, ..., n_{|N_b|}\}$ where each position n_i is filled with a bus station $s \in S$ representing its i-th stop. Bus routes are accordingly defined as a consecutive sequence of bus stops. Each bus $b \in B$ has a maximum capacity C that should not be exceeded. This is modeled by the variable q_{nb}, which indicates the net number of passengers that are picked up or dropped off at the $n-$th stop of bus b. An example bus route is shown in Fig. 2.

Let continuous variables t_{nb}^a and t_{nb}^d represent, respectively, the arrival and departure time of bus b at its $n-$th stop.

[5]If the metro station is large, a walking time of zero is not realistic. However, an average walking time within the metro station can be included in the travel time matrix if this station is the starting or ending-point of the metro-trip of a passenger.

Table 1: Summary of used notation.

Notation	Definition
	Parameters
B	The fleet of buses
C	The capacity of a bus
P	The set of requests
S	The set of bus stations
F	The set of metro stations
O	The set of all origin nodes
D	The set of all destination nodes
V	$S \cup F \cup O \cup D$
o_p	The origin node of passenger p
d_p	The destination node of passenger p
A	The set of all arcs connecting nodes in V
A_s	The set of arcs connecting nodes in S
A_f	The set of arcs connecting nodes in F
A_w	The set of arcs that represent walkable paths between nodes in V
e^u_p	The earliest departure time for passenger p
l^o_p	The latest arrival time for passenger p
τ^b_{ij}	The travel time from node i to j by bus
τ^f_{ij}	The travel time from node i to j by fixed route
τ^w_{ij}	The walking time from node i to j
u_p	The maximum walking time for passenger p
N_b	Positions in the representation of a bus route with $\|N_b\|$ the number of positions available in bus b
	Discrete decision variables
x_{snb}	1 if the n-th stop of bus b is bus station s, 0 otherwise
y^u_{pnb}	1 if passenger p is picked up at the n-th stop of bus b, 0 otherwise
y^o_{pnb}	1 if passenger p is dropped off at the n-th stop of bus b, 0 otherwise
v_{pbij}	1 if passenger p travels with bus b from station i to j, 0 otherwise
z_{pij}	1 if passenger p uses fixed route from station i to station j, 0 otherwise
w_{pij}	1 if passenger p walks/transfers from node/station i to j, 0 otherwise
q_{nb}	Net number of passengers picked up (or dropped off) at the n-th stop of bus b
	Continuous decision variables
t^a_{nb}	The arrival time of bus b at its n-th stop
t^d_{nb}	The departure time of bus b at its n-th stop
fi_{pi}	The time at which passenger p is picked up or dropped off at station i by a fixed vehicle
de_p	The time at which passenger p leaves the assigned departure station
ar_p	The time at which passenger p arrives at the assigned arrival station

Moreover, fi_{pi} express the time at which passenger p is picked up or dropped off at station i by a vehicle following a fixed route. Finally, to ensure time window consistency for each request, variables de_p and ar_p are introduced, which represent the times at which passenger p leaves from the assigned departure station and arrives at the assigned arrival station, respectively.

Finally, consider the discrete decision variables shown in Table 1 which represent bus route and passenger trip solutions.

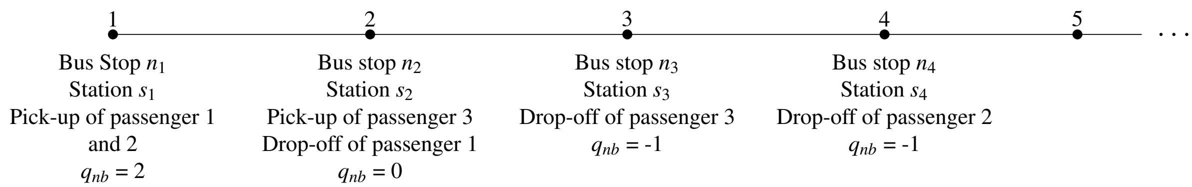


Figure 2: Example bus route in the ODBRP

The I-ODBRP can be summarized as an iterative interaction between passenger requests and the transit agency responsible for the system, where optimization procedures are executed to appropriately accommodate those requests. In a real-world scenario, it would remain the choice for the users to decide whether to accept the trip proposal. However, this is not modeled in the present study.

The I-ODBRP with the objective of minimizing the total user ride time can be formulated as follows:

$$URT = \min \sum_{p \in P} ar_p - de_p \tag{1}$$

$$\sum_{s \in S} x_{snb} \leq 1, \qquad \forall\, b \in B, n \in N_b \tag{2}$$

$$\sum_{s \in S} x_{snb} - \sum_{s' \in S} x_{s'(n+1)b} \geq 0, \qquad \forall\, b \in B, n \in N_b \tag{3}$$

$$\sum_{n \in N_b} (n \cdot y^u_{pnb} - n \cdot y^o_{pnb}) \leq 0, \qquad \forall\, p \in P, b \in B \tag{4}$$

$$\sum_{n \in N_b} (y^u_{pnb} - y^o_{pnb}) = 0, \qquad \forall\, p \in P, b \in B \tag{5}$$

$$\sum_{n \in N_b} y^u_{pnb} \leq 1, \qquad \forall\, p \in P, b \in B \tag{6}$$

$$\sum_{n \in N_b} y^o_{pnb} \leq 1, \qquad \forall\, p \in P, b \in B \tag{7}$$

$$M \sum_{s \in S} x_{snb} - \sum_{p \in P} (y^u_{pnb} + y^o_{pnb}) \geq 0, \qquad \forall\, b \in B, n \in N_b \tag{8}$$

$$\sum_{s \in S} x_{snb} - \sum_{p \in P} (y^u_{pnb} + y^o_{pnb}) \leq 0, \qquad \forall\, b \in B, n \in N_b \tag{9}$$

$$\sum_{(i,j) \in A_s} v_{pbij} - \sum_{n \in N_b} y^u_{pnb} = 0, \qquad \forall\, p \in P, b \in B \tag{10}$$

$$\sum_{(i,j) \in A_s} v_{pbij} - \sum_{n \in N_b} y^o_{pnb} = 0, \qquad \forall\, p \in P, b \in B \tag{11}$$

$$\sum_{(i,j) \in A_s} v_{pbij} + y^u_{pnb} - x_{inb} \leq 1, \qquad \forall\, p \in P, i \in S, b \in B, n \in N_b \tag{12}$$

$$\sum_{(i,j) \in A_s} v_{pbij} + y^o_{pnb} - x_{jnb} \leq 1, \qquad \forall\, p \in P, j \in S, b \in B, n \in N_b \tag{13}$$

$$x_{snb} + y^u_{pnb} - \sum_{l \in F \cup o_p} w_{pls} \leq 1, \quad \forall\, p \in P, s \in S, b \in B, n \in N_b \tag{14}$$

$$x_{snb} + y^o_{pnb} - \sum_{l \in F \cup d_p} w_{psl} \leq 1, \quad \forall\, p \in P, s \in S, b \in B, n \in N_b \tag{15}$$

$$\sum_{n \in N_b} \sum_{b \in B} y^u_{pnb} - \sum_{s \in S} \sum_{l \in F \cup o_p} w_{pls} = 0, \quad \forall\, p \in P \tag{16}$$

$$\sum_{n \in N_b} \sum_{b \in B} y^o_{pnb} - \sum_{s \in S} \sum_{l \in F \cup d_p} w_{psl} = 0, \quad \forall\, p \in P \tag{17}$$

$$\sum_{b \in B} \sum_{(i,j) \in A_s} v_{pbij} - \sum_{(l,i) \in A_w | l \in F \cup o_p} w_{pli} = 0, \quad \forall\, p \in P, i \in S \tag{18}$$

$$\sum_{b \in B} \sum_{(i,j) \in A_s} v_{pbij} - \sum_{(j,l) \in A_w | l \in F \cup d_p} w_{pjl} = 0, \quad \forall\, p \in P, j \in S \tag{19}$$

$$\sum_{(i,j) \in A_f} z_{pij} - \sum_{(h,i) \in A_w | h \in S \cup o_p} w_{phi} = 0, \quad \forall\, p \in P, i \in F \tag{20}$$

$$\sum_{(i,j) \in A_f} z_{pij} - \sum_{(j,h) \in A_w | h \in S \cup d_p} w_{pjh} = 0, \quad \forall\, p \in P, j \in F \tag{21}$$

$$\sum_{j \in S \cup F} w_{po_pj} = 1, \quad \forall\, p \in P \tag{22}$$

$$\sum_{j \in S \cup F} w_{pjd_p} = 1, \quad \forall\, p \in P \tag{23}$$

$$\sum_{j \in S \cup F} w_{po_pj} \cdot \tau^w_{o_pj} \leq u_p, \quad \forall\, p \in P \tag{24}$$

$$\sum_{j \in S \cup F} w_{pjd_p} \cdot \tau^w_{jd_p} \leq u_p, \quad \forall\, p \in P \tag{25}$$

$$\tau^w_{ij} - M(1 - w_{ij}) \leq 0, \quad \forall\, p \in P, i \in S, j \in F \tag{26}$$

$$\tau^w_{ij} - M(1 - w_{ij}) \leq 0, \quad \forall\, p \in P, i \in F, j \in S \tag{27}$$

$$\sum_{(i,j) \in A_w} w_{pij} \leq 4, \quad \forall\, p \in P \tag{28}$$

$$w_{pij} + w_{pjk} \leq 1, \quad \forall\, p \in P, (i,j), (j,k) \in A_w \tag{29}$$

$$t^a_{(n+1)b} - t^d_{nb} - \tau^b_{ss'} + (x_{snb} + x_{s'(n+1)b} - 2)(-M) \geq 0, \quad \forall (s,s') \in A_s, b \in B, n \in N_b \tag{30}$$

$$fi_{pj} \geq fi_{pi} + \tau^f_{ij} - M(1 - z_{pij}), \quad \forall\, p \in P, (i,j) \in A_f \tag{31}$$

$$fi_{pf} \geq t^a_{nb} - M(1 - w_{psf}), \quad \forall\, p \in P, s \in S, f \in F, b \in B, n \in N_b \tag{32}$$

$$t^d_{nb} \geq fi_{pf} - M(1 - w_{pfs}), \quad \forall\, p \in P, s \in S, f \in F, b \in B, n \in N_b \tag{33}$$

$$t^d_{nb} \geq de_p - M(1 - w_{po_ps}), \quad \forall\, p \in P, s \in S, b \in B, n \in N_b \tag{34}$$

$$ar_p \geq t^a_{nb} - M(1 - w_{psd_p}), \quad \forall\, p \in P, s \in S, b \in B, n \in N_b \tag{35}$$

$$fi_{pf} \geq de_p - M(1 - w_{po_pf}), \quad \forall\, p \in P, f \in F \tag{36}$$

$$ar_p \geq fi_{pf} - M(1 - w_{pfd_p}), \quad \forall\, p \in P, f \in F \tag{37}$$

$$de_p - e^u_p \geq 0, \quad \forall\, p \in P \tag{38}$$

$$ar_p - l^o_p \leq 0, \quad \forall\, p \in P \tag{39}$$

$$ar_p - de_p > 0, \quad \forall\, p \in P \tag{40}$$

$$x_{snb} + x_{s(n+1)b} \leq 1, \quad \forall s \in S, b \in B, n \in N_b \tag{41}$$

$$\sum_{p \in P} (y^u_{pnb} - y^o_{pnb}) - q_{nb} = 0, \quad \forall\, b \in B, n \in N_b \tag{42}$$

$$\sum_{n \geq n' \in N_b} q_{n'b} \leq C, \qquad \forall\, b \in B, n \in N_b \tag{43}$$

$$\sum_{b \in B} \sum_{n \in N_b} y^u_{pnb} + \sum_{(i,j) \in A_f} z_{pij} \geq 1, \qquad \forall p \in P \tag{44}$$

$$x_{snb} \in \{0,1\}, \qquad \forall\, s \in S, b \in B, n \in N_b \tag{45}$$

$$y^u_{pnb}, y^o_{pnb} \in \{0,1\}, \qquad \forall\, p \in P, b \in B, n \in N_b \tag{46}$$

$$v_{pbij} \in \{0,1\}, \qquad \forall\, p \in P, b \in B, (i,j) \in A_s \tag{47}$$

$$z_{pij} \in \{0,1\}, \qquad \forall\, (i,j) \in A_f \tag{48}$$

$$w_{pij} \in \{0,1\}, \qquad \forall\, p \in P, (i,j) \in A_w \tag{49}$$

$$q_{nb}, t^a_{nb}, t^d_{nb} \in \mathbb{Z}_+, \qquad \forall\, b \in B, n \in N_b \tag{50}$$

$$fi_{pi} \in \mathbb{Z}_+, \qquad \forall\, p \in P, i \in F \tag{51}$$

$$de_p, ar_p \in \mathbb{Z}_+, \qquad \forall\, p \in P \tag{52}$$

The objective function (1) minimizes the total user ride time. Constraints (2) ensures that a bus can only stop at one station at the same time. Constraints (3) make sure that the positions used in the bus route are used consecutively and start at the first position. Constraints (4) ensure that a passenger is always dropped off after he/she is picked up. Constraints (5) establish that both pick-up and drop-off events have to be carried out by the same bus. Constraints (6) and (7) limit the number of times a bus can serve a passenger to one. Constraints (8) impose that no passenger can get on or off a bus at position n, when a bus does not make a stop at position n, meanwhile constraints (9) ensure at least one passenger gets on or off a bus when a stop is made. Constraints (10)-(11) make sure if a passenger travels with a bus between two stations, he/she is picked up and dropped off exactly by that bus. Constraints (12)-(13) force that when a passenger travels between i and j with bus b, he/she is picked up (dropped off) at the same position that b stops at station i (j). Constraints (14)-(21) guarantees connectivity in the route of the passenger by ensuring that when they are dropped off they must transfer from such drop-off station to another node, and when they are picked up they must transfer from a node towards the pick-up station. Constraints (22) and (23) establish that each passenger transfers once from their origin to a station and from a station to their destination. Constraints (24)-(25) determine that walking from origin to departure station and from arrival station to the destination can not exceed more than a pre-defined threshold. Constraints (26)-(27) ensure that transfers occur at the same station, i.e., walking times are zero. Constraints (29) prohibits the passenger to walk consecutively until reach its destination. Constraints (28) limit the number of transfers between modes. Constraints (30) incorporate the time aspect of the ODBRP, in which if s is the n-th stop of the bus and s' is the $(n+1)$-th, then the time of arrival at station s' should be greater or equal to the departure time at s plus the travel time between stations s and s'. Constraints (31) ensure that if a passenger is picked up at FPT station i that the drop-off time at station j is greater than or equal to the sum of pick-up time at i plus the travel time by FPT between i and j. Constraints (32)-(37) guarantee travel time consistency between transfers. Constraints (38) ensure that the route of the passenger will start after their earliest departure time, meanwhile constraints (39) establish that the passenger reaches the arrival station before their latest arrival time. Constraints (41) forbid that the same bus station is visited twice in a row. Constraints (42) and (43) are capacity constraints for the buses. Constraints (44) state that every request is served. Constraints (45)-(52) ensure the non-negativity and integrality requirements of the variables.

Note that we deliberately removed walking times between transfer stations from the model, however, this could be easily contemplated by removing constraints (26) and (27), as well as modifying constraints (32) and (33), after which they become:

$$fi_{pf} \geq t^a_{nb} + \tau^w_{sf} - M(1 - w_{psf}), \qquad \forall\, p \in P, s \in S, f \in F, b \in B, n \in N_b \tag{53}$$

$$t^d_{nb} \geq fi_{pf} + \tau^w_{fs} - M(1 - w_{pfs}), \qquad \forall\, p \in P, s \in S, f \in F, b \in B, n \in N_b \tag{54}$$

Furthermore, we introduce the following variables, constants, and constraints to incorporate penalties for transferring within the fixed route network. Consider variable l_p be a positive integer reporting the number of line transfers of passenger p within the fixed route network. Let η_{ij} be a constant expressing the number of line transfers between stations i and j, which is precomputed by running a shortest path algorithm on a graph where edges represent a fixed line running between two stations (nodes). Each transfer within the fixed route network is then penalized with a constant time denoted by θ. The new objective function and constraints are:

$$\min \sum_{p \in P} ar_p - de_p + l_p \cdot \theta \tag{55}$$

$$M(1 - z_{pij}) + l_p \geq \eta_{ij}, \qquad \forall p \in P, (i,j) \in A_f \tag{56}$$

$$l_p \in \mathbb{Z}_+ \qquad \forall\, p \in P \tag{57}$$

The objective function (55) now minimizes total user ride time and penalties. Constraints (56) ensure that if a passenger p utilizes the fixed route network, variable l_p correctly informs how many transfers were performed between i and j. Otherwise, l_p is automatically set to 0 if the passenger does not ride with FPT, as the objective is a minimization function. Nonnegativity and integrality are ensured by constraints (57). The rest of the model would remain the same.

4 Efficient insertion-based heuristic for the I-ODBRP

As the (I-)ODBRP is NP-hard[6], exact methods will not suffice to solve large-scale instances. In Section 3 we formulated the mathematical model for the problem assuming all requests are known in advance. However in real-life, requests will be coming in last-minute or just-in-time. Especially in this dynamic context, a fast system response is necessary. The dynamic, non-integrated ODBRP is already introduced in Melis and Sörensen (2021). In their work, requests are handled in real-time by using a LNS framework with embedded local search. However, in the I-ODBRP even more solution possibilities are present (choice of trip-type, fixed line, transfer stations, on-demand buses, etc.). To ensure fast response times, we focus in this section on a greedy insertion-based heuristic with efficient on-demand vehicle and fixed line assignment procedures. By using insertion heuristics in a dynamic setting, the initially assigned stations, on-demand bus and fixed line of a request cannot change after insertion, but the time predictions on departure and arrival can still be delayed by adding new requests to the solution. Because heuristics for integrated on-demand public transport are rather scarce (see Section 2), we looked into recent literature on non-integrated many-to-many dynamic on-demand public transport, and found that both greedy insertion and greedy insertion combined with efficient vehicle assignment are commonly used methods to schedule requests (see Table 2). Efficient vehicle assignment procedures do not check every vehicle for request insertion, but rather select a smaller set of vehicles to try for insertion based on a criterion, e.g., distance between vehicle and the origin location of a request. This method is efficient, because vehicles that are most likely not able to serve a request within the given time constraints, are not tried for insertion. Consequently the computation time is lower, compared to trying all vehicles.

Greedy insertion
Ronald et al. (2013), Ronald et al. (2015), Atasoy et al. (2015), Ikeda et al. (2015), Archetti et al. (2018b), Bischoff et al. (2017), Narayan et al. (2017), Navidi et al. (2018), Viergutz and Schmidt (2019)
Greedy insertion with efficient vehicle assigment
Tsubouchi et al. (2009) , Bertelle et al. (2009), Hyland and Mahmassani (2020), Jäger et al. (2018), Ma et al. (2013), Van Engelen et al. (2018), Winter et al. (2018)

Table 2: Literature on many-to-many dynamic/online on-demand public transport using insertion based heuristics (with efficient vehicle assignment)

In the I-ODBRP, when a new request needs to be scheduled, the request is first tried to be assigned a trip containing at least one FPT-leg. The rationale behind this is that by directing passengers to use the metro network, there is less pressure on the ODB service so that more passengers can be served. When both the origin and arrival location of

[6] This is because the I-ODBRP is an extension of the ODBRP, which is a combination of both the DARP, PDPTW and SBRP, which are all NP-hard.

the request are not in the proximity of a metro station, the trip type considered first would be ODB + FPT + ODB. On one hand, the algorithm could check *all* bus stations for departure and arrival, *all* buses for insertion of the first and last leg, *all* transfer stations for transferring from ODB to FPT and vice versa and *all* ways to go from the first transfer station to the next by using the metro network. However checking all these insertion possibilities would be too computationally expensive. In contrast, the algorithm could only check *the closest* bus stations for departure and arrival, *the closest* buses and transfer stations. In the latter an ODB brings the passenger from the closest origin station within walking distance to the nearest metro station, from which the passenger takes the metro to the metro station closest to the destination. Here another ODB brings the passenger from the metro station to the closest arrival station within walking distance of the actual arrival location of the passenger. This option will result in low computation times, however, one limits the flexibility of the routing of the on-demand buses. The total URT will increase or the service rate[7] will decline. To merge the benefits of these two extreme possibilities, we introduce two procedures for algorithm speed-up and two procedures for maintaining flexibility, depicted in Table 3, and further explained in the following subsections. Because bus station assignment is already thoroughly explained and investigated in Melis and Sörensen (2020, 2021), we will not elaborate on this measure in this section. After explaining the speed-up and flexibility procedures Section 4.1 to Section 4.3, the overall algorithm is presented in Algorithm 1 in Section 4.4.

	FPT-level	ODB-level
Speed-up	Efficient metro segment assignment	Efficient on-demand bus assignment
Flexibility	Transfer station assignment	Bus station assignment

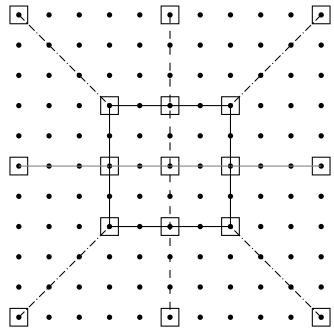
Table 3: Algorithm speed-up and flexibility measures

4.1 Speed-up - Metro segment assignment

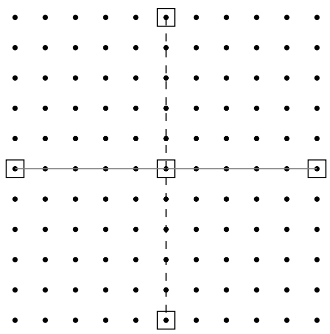
The speed-up measure on the FPT-level is called efficient metro segment assignment. To explain our procedure we first introduce the concept of a metro segment by using the star-shaped network of Fig. 3a. *A metro segment* is every piece of the metro network starting and ending in a key station. *The metro segments set L* contains all possible metro segments of the network. Sometimes multiple paths are possible to go from one key station to another. Considering all possible paths would be unnecessary, as some paths will be sufficiently longer compared to the shortest possible path. This is shown in Fig. 4a. The path shown in the figure, connecting the black colored key stations, will not be included in the metro segments set of the network, because it is considerably longer than the shortest path. The cut-off value used to determine which metro segments are too long is set at λ times the shortest metro segment for the star shaped network. λ can of course be adapted to fit the network[8]. Two paths connecting the same two key stations, which are almost equal in distance, are both kept in the metro segments set. An example is illustrated in Fig. 4b. As both paths include different metro stations, both should be considered, because depending on the demand for transportation, one or the other might be the better choice to minimize the total URT. Further, for metro segments connecting two key stations which are both begin- or end-points, only the actual shortest path is added to the metro segments set. This can be done because the other possible paths, which will include different metro stations, will be included in the metro segments connecting the key stations in between. Lastly, a metro segment should have a minimum length. In this paper the minimum length is set at 4 metro stations. When generating all metro segments using the above mentioned criteria, in both directions, there are 289 possible metro segments for the star-shaped network. This indicates a strong need for metro segment assignment to schedule each request.

As every request is first tried to be assigned a trip containing at least one FPT-leg, we need to generate for each request p a set of possible metro segments, $L_p \subset L$. Each metro segment $l \in L_p$ will consequently be considered to be used for the FPT leg (line 2 in Algorithm 1). A metro segment is added to L_p when it meets three criteria. These three criteria are based on direction, distance (or time) and accessibility. The criteria will justify the use of metro segments instead of entire lines, next to the fact that by using metro segments unnecessary pieces of the metro network will never be considered for insertion.

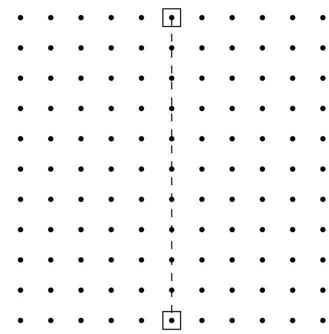
[7]The service rate is the number of served requests divided by the total number of requests.
[8]All additional notation introduced in Section 4 is summarized in Appendix A Table A1.



(a) Star-shaped network: 7 direct lines (a circular line around city centre, an horizontal and vertical line and 4 lines coming from the outer city to the corners of the city centre)

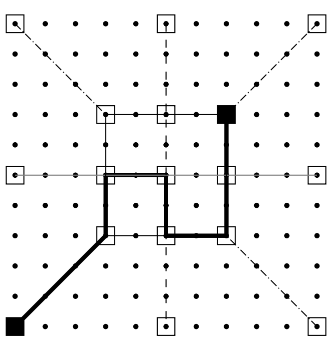


(b) Cross shaped network: 2 direct lines (one horizontal and one vertical line)

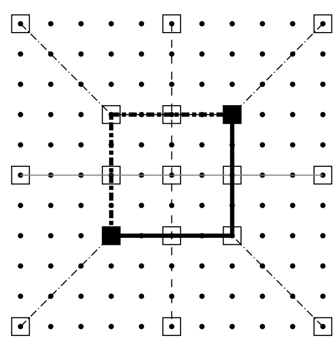


(c) Single line network: 1 direct line (vertically)

Figure 3: Star-shaped, cross-shaped and single line network (Key stations are indicated as squares)



(a) Example of a metro segment that is eliminated from the metro segments set



(b) Example of keeping two metro segments connecting the same two key stations

Figure 4: Generating the metro segments set of the star-shaped network

Direction The corner between the direction of the request and the direction of the metro segment is calculated by using Cosines rule. This is also done by Tsubouchi et al. (2009) to assign requests to on-demand vehicles. When x_{p1} and y_{p1} (x_{p2} and y_{p2}) are the x and y coordinates of the origin (arrival) of the request and x_{l1} and y_{l1} (x_{l2} and y_{l2}) are the x and y coordinates of the first (last) metro station of the metro segment, the formula to compute the corner (in degrees) between request and metro segments is shown in Eq. (58). All metro segments where $Corner_{rl} > \varepsilon_d$, with ε_d the *the cut off corner* (in degrees), are eliminated from L_p.

$$Corner_{pl} = \arccos\left(\frac{(x_{p2}-x_{p1})\times(x_{l2}-x_{l1})+(y_{p2}-y_{p1})\times(y_{l2}-y_{l1})}{\sqrt{(x_{p2}-x_{p1})^2+(y_{p2}-y_{p1})^2}\times\sqrt{(x_{l2}-x_{l1})^2+(y_{l2}-y_{l1})^2}}\right)\times\frac{180}{\pi} \tag{58}$$

Distance/time Even though the direction of the metro segment is the same as the direction of the request, the metro segment can be situated in an entirely different area of the city. Therefore, more metro segments will be eliminated from consideration based on a distance/time metric[9]. All metro segments where $(\tau^b_{o_p s^o_{pl}} + \tau^b_{a_p s^a_{pl}}) > (\tau^f_{s^o_{pl} s^a_{pl}} \times \varepsilon_t)$, are eliminated from L_p, with s^o_{pl} the closest metro station on metro segment l to the origin location of passenger p, s^a_{pl} the closest metro station on metro segment l to the arrival location of passenger p, and ε_t the threshold value which is dependent on the instance. The direct driving time by bus from the origin location to s^o_{pl} is calculated and added to the direct driving time by bus from the arrival location of the request to s^a_{pl}. If this sum is larger than the time a passenger would spend on the metro segment from s^o_{pl} to s^a_{pl} times a threshold value, the metro segment is eliminated from consideration. This way, the allowed travel time by bus from the metro segment is dependent on the amount of time the passenger would spend on the metro line. If the passenger requests a long trip, a larger distance from the metro segment is allowed compared to when the passengers demands a short trip.

Accessibility The last criterion is based on whether or not the metro segment is accessible for the passenger by walking. If a passenger can walk to/from one (or some) metro station(s) for his origin and/or arrival, only the metro segments containing at least one of these stations remain in L_p.

By applying these three criteria, only a few metro segments remain per request out of the entire set of 289 metro segments for the star-shaped network. L_p is pre-processed for each origin-destination pair to reduce computation time. One could argue that not all x and y coordinates located in a city can be pre-processed. This can be solved by partitioning the city in blocks or areas and pre-process for these. However, for research purposes, we pre-process based on the actual x and y coordinates of requests.

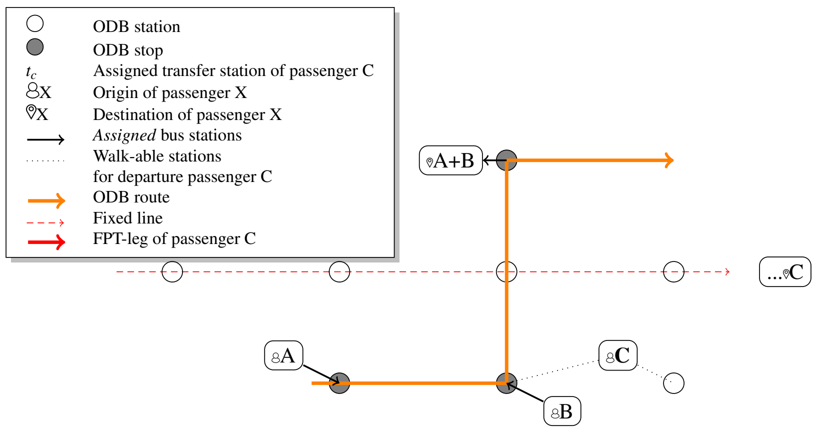
4.2 Flexibility - Transfer station assignment

As mentioned above, when adding a request to the solution, it is first checked if it is possible to assign the passenger a trip containing a FPT-leg. Therefore, each $l \in L_p$ is checked to be used in the FPT-leg. However, not all stations on the segment need to be considered. In the ODBRP, we already introduced bus station assignment to increase the ODB routing flexibility. In this case, not necessarily the closest bus stations for departure and arrival are chosen, bus stations within walking distance are assigned. As mentioned in Section 1, this flexibility can be extended to the transfer stations as well. Not necessarily the transfer stations which are closest to the origin and arrival location of the passenger, are the best for the overall URT. The final choice of transfer stations will depend on the other requests in the solution. The more passengers can be pooled, the less the total URT will increase.

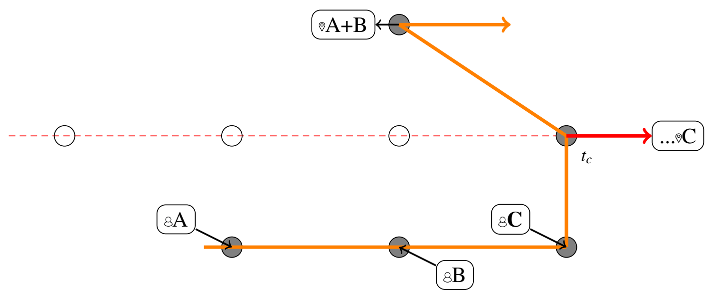
Figure 5a shows an example of the I-ODBRP with two already scheduled passenger requests (both with a ODB itinerary), one fixed line and one on-demand bus. The time aspect of the problem is not illustrated in the figure for simplicity, but of course all requests need to be scheduled within their time windows. Passenger A gets on the bus at the first stop and gets off at the third[10]. Passenger B gets on at the second bus stop and gets off together with passenger A. Now passenger C needs to be scheduled. Passenger C can walk to two neighbouring bus stations from his origin

[9] Distance is always expressed in time units

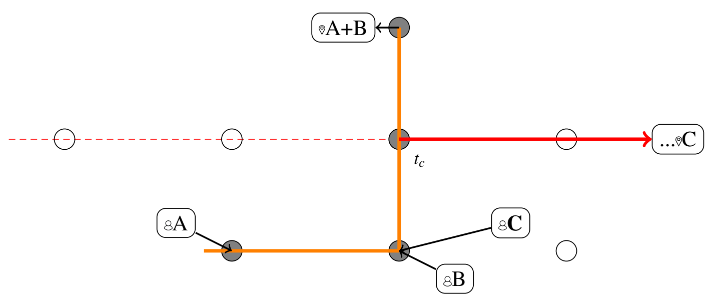
[10] Note that the bus does not initially stop at the metro station it passes, as there is no demand present at this station.



(a) Passenger C needs to be inserted in the solution



(b) Inserting passenger C into the solution when only considering the closest transfer station



(c) Inserting passenger C into the solution with flexible transfer station assignment

Figure 5: Example I-ODBRP with two already scheduled requests (A and B), when the request of passenger C is issued

location. Because of bus station assignment, the algorithm has to assign one of these two as the origin station for passenger C. His destination is in another area of the city, consequently his URT will be shorter if he travels part of his journey in the high-speed metro, instead of a direct ODB. The closest transfer station to the origin of passenger C is the last depicted station on the fixed line. If passenger C would be directed to this metro station, the resulting solution is the one shown in Fig. 5b. The on-demand bus makes a detour to pick up passenger C and drops him off at the assigned transfer station t_c. Both the URT's of passengers A and B will increase. However, if we allow some flexibility concerning the transfer station assignment, passengers can be pooled, which results in a lower total URT as detours and extra stops can be avoided. This is illustrated in Fig. 5c. The on-demand bus passes a metro station already, so adding an extra stop here is easy. This station would consequently be assigned as the transfer station of passenger C. The URT's of passengers A and B do not increase by inserting passenger C and even though the trip of passenger C starts to go in the opposite direction of his destination, his URT will not differ extremely from his URT as depicted in Fig. 5b, making this the better option.

To determine which metro stations are considered as transfer stations on a certain metro segment (necessary for the loops on lines 4 and 10 of Algorithm 1), first the index of the closest station on the segment to the origin location of the request is determined and called I_{olp}. The same is done for the arrival (I_{alp}). In Fig. 6 this is the metro station with index 2 for the origin and 6 for the arrival. However, as is illustrated in Fig. 5, we should consider multiple stations, preferably in the neighbourhood of the closest metro station. Therefore another parameter is introduced: the index range or I_r. This parameter indicates how many metro stations we will differ from the closest one on the segment. The more dense the network is, with lots of metro stations in a relatively small area, the larger this index range needs to be. Fig. 6 shows the metro stations considered on the metro segment for both origin (indexes $[1,3]$) and arrival (indexes $[5,7]$) in dotted lines, when an index range of 1 would be adopted. The indexes for origin and arrival cannot overlap and the indexes have to exist in the metro segment. I_{endl} denotes the index of the last station in the metro segment. $i \in [Index_{olp\text{-}min}, Index_{olp\text{-}max}]$ is the origin index under consideration, when determining the minimum and maximum index for the arrival. The minimum and maximum index for origin and arrival for a specific metro segment l are calculated as follows:

$$Index_{olp\text{-}min} = \max\left(0, I_{olp} - I_r\right) \tag{59}$$
$$Index_{olp\text{-}max} = \min\left(I_{olp} + I_r, I_{endl}\right) \tag{60}$$
$$Index_{alp\text{-}min} = \max\left(i+1, I_{alp} - I_r\right) \tag{61}$$
$$Index_{alp\text{-}max} = \min\left(\max\left(Index_{alp\text{-}min}, I_{alp} + I_r\right), I_{endl}\right) \tag{62}$$

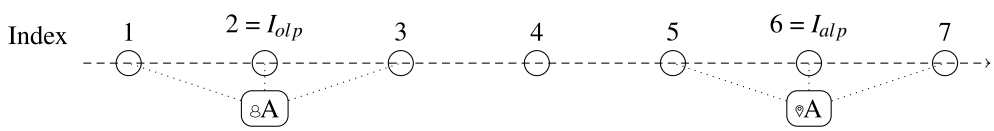


Figure 6: Example transfer stations of a metro segment under consideration when index range amounts 1

4.3 Speed-up - On-demand bus assignment

In Melis and Sörensen (2020, 2021), the authors first fill every empty available bus with a request before adding multiple requests in one bus. When there are no empty buses left, we consider insertion of a request in *all* available buses of the fleet size. In contrast to the ODBRP, where the algorithm checks all buses of B *once* when trying to insert a request, the I-ODBRP tries to insert one request $((2 \times I_r + 1) + (2 \times I_r + 1)^2)$ times when the trip is of the ODB+FPT+ODB type[11] This equals, worst case, 12 times when the index range is 1, 30 times when the index range is 2, and this for all metro segments in L_p. Because of the importance of the computation time, not all buses from B

[11]This amount is deductable from the nested for-loops in Algorithm 1 on lines 4 and 10, given that the transfer stations under consideration as shown in Fig. 6 do not overlap.

are checked every time the algorithm tries to insert a request. An efficient on-demand bus assignment procedure is implemented that reduces the number of buses checked for the first leg to $B_{plleg1} \in B$ (generated in line 3 in Algorithm 1 and for the second leg $B_{plleg3} \in B$, generated in line 11 in Algorithm 1). We test three different on-demand bus assignment methods, explained in this section, and compare them regarding their influence on the objective function value and the induced decrease in computation time in Section 5.3.

4.3.1 ODB assignment based on the earliest departure time and origin location

The first method is based on the distance between the location of the bus at the earliest departure time of a request and the origin location of the request. This method is similar to procedures adopted by Ma et al. (2013) and Bertelle et al. (2009). Buses located at a total distance smaller than a threshold value ε_a, are included in $B_{plleg1/leg3}$. If none of the buses are within a this radius, all buses will be tested for insertion.

An example on ODB assignment for a first leg or ODB-only trip can be found in Appendix B. A similar reasoning applies for checking a second leg ODB insertion. However it should be noted that the earliest departure time of a second ODB leg depends on the chosen first ODB leg (both chosen on-demand bus and metro station under consideration) and on the metro station under consideration for leg 3. The earliest departure time for leg 3 will change, as the passenger will spend a shorter/longer amount of time on the metro line. And depending on the on-demand bus chosen for leg 1, the passenger might be dropped off earlier or later at a metro station of the metro network. Note that the second leg is always chosen given the best first leg found. Alternatively, one could consider all combinations of leg 1 and 2 legs together as the first has major influence on the second, concerning the time aspect. It is possible that, for example. the second best first leg combined with its best second leg results in a better overall URT compared to the best first leg combined with its best second leg. However checking all combinations would be too computationally expensive.

4.3.2 ODB assignment based on the earliest departure time and origin location, and (latest arrival time and) the arrival location

Ma et al. (2013) test, next to an assignment procedure based on the earliest departure time and origin location, also an assignment procedure based on the above and the distance between the bus's location at the latest arrival time and arrival location of a request. However, in the problem studied there are two time windows, one for the departure and one for the arrival and they minimize the distance traveled. In the (I-)ODBRP there is only one, wide time window, and the user ride time is minimized. When selecting buses based on both the origin location and earliest departure time, and the destination and latest arrival time, the transportation of requests is pushed to take as long as the width of the entire time window, which counteracts our objective. Therefore, we initiated an ODB assignment procedure based on the average of the distance between the origin location of the request and the approximate location of the bus at the earliest departure time of a request, and the distance between the destination location of the request and the station that is closest to this destination that is already scheduled further in the bus route but before the latest arrival time of a request. If the average is smaller than a certain threshold value ε_a, the bus is tested for insertion. The first distance, based on the origin and earliest departure time, is checked by using the data structure from Appendix B Fig. B1. If the data structure returns a -1 for the earliest departure time of a request, the bus is included in the potential bus assignments set, as the bus is not busy during the time window of the request.

4.3.3 ODB assignment based on driving direction between earliest departure and latest arrival time

Tsubouchi et al. (2009) adopt a vehicle choosing procedure based on the direction that the vehicle is driving in during the time window of a certain request, and the direction the request needs to go in. We tested the same method in our algorithm. The difference in directions between the request and a bus is expressed as the corner between them, calculated by the formula in Eq. (58). If this corner is smaller than a certain threshold value ε_a, the bus is tested for insertion.

Instead of using the coordinates of the first and last station of a metro segment, x_{l1} and y_{l1} are the x- and y coordinates of the station where the checked bus is at the earliest departure time of the request and x_{l2} and y_{l2} are the x- and y coordinates of the station where the bus is located at the latest arrival time of the request. The stations are once again found by using the data structure explained in Appendix B Fig. B1b. If the data structure indicates a -1 for both the earliest departure time and the latest arrival time of a request, the bus is added to the list of potential bus assignments, as this bus is not servicing other requests at this time. If the data structure indicates a -1 only for the latest arrival time, the coordinates of the last scheduled station of this bus are taken for x_{l2} and y_{l2}.

The earliest departure and latest arrival time of a request depend on the chosen trip type. When wanting the generate a potential buses set for a direct trip, the time window is taken as is. However, when generating a set of buses for a first leg trip, the latest arrival time needs to be adapted. In this case the latest arrival time equals the earliest departure time plus 1.5 times the distance between the origin location and the transfer station which is currently tested in the algorithm. On the other hand, for a second leg trip, it is the earliest departure time that is adapted and equaled to earliest departure time plus the time needed to travel to the transfer station via a FPT or ODB+FPT trip.

4.4 Algorithm outline

Algorithm 1 shows the overall algorithm to insert a request p in the solution, given the fact that there are no empty buses available anymore. When there are still empty buses lines 3, 5 and 11-12 are superfluous as a request can then be assigned to an empty bus in the way a request would be assigned to a private taxi. The same logic is applied in Melis and Sörensen (2020, 2021): first all empty buses are filled with one request before adding multiple requests to one bus. Generating L_p and ranges of the loops on lines 4 and 10 are described in Section 4.1 and Section 4.2. For every metro segment l and metro station index i, the best first leg for passenger p is found. If a feasible first leg is obtained, the best second leg is found. The second leg needs to be served by a different bus then the one serving the first leg, otherwise the point of taking the metro network instead of using a direct ODB-service is lost. In the end, the best combination of both legs is saved and executed. When no feasible leg 1 and 2 combination is found with L_p, it is checked whether or not a feasible direct ODB-service is possible. If so, this trip type is assigned to the request.

One additional computational speed-up measure is taken by saving the best first leg insertions for given metro stations. This is explained in Appendix C.

4.4.1 Finding the best insertion

Finding the best leg 1 and 3 insertion is summarized in lines 5 and 12. An important note concerning these lines is the following: if the metro station under consideration (with index i/j on metro segment l) is within walking distance of the origin/arrival of the passenger, the ODB-leg becomes empty and is actually just a short walk to/from the metro network[12]. If the metro station under consideration is not within walking distance, the same constructive heuristic of Melis and Sörensen (2021) is used to insert a passenger in one of the on-demand buses. Basically the request is added in a greedy way, so that the total URT, of all inflicted passengers, increases the least. Because in this work, we solve the I-ODBRP instead of the ODBRP, there are some technical adjustments that need to be made regarding the insertion criterion and the feasibility check. These adjustments are explained in this section. Note that the request still needs to be served within the given time window, and capacity constraints of the vehicle cannot be exceeded.

Insertion criterion Similar to the default constructive heuristic in Melis and Sörensen (2020, 2021), requests in the I-ODBRP are always inserted where the total URT increases the least. This includes (1) the URT of the, to be inserted, passenger (URT_p) and (2) the increase in URT of the already scheduled passengers (ΔURT). The insertion criterion is calculated on lines 8 (leg 1 insertion) and 13 (leg 3 insertion) of Algorithm 1.

For the ODBRP the first part is straightforward and shown in Eq. (63), with o and a the origin and arrival stop of the passenger. In contrast, for the I-ODBRP the URT_p is calculated depending on the trip type. This is illustrated in Fig. 7. The current time is located before the trip schedules as passengers need to be inserted somewhere in the future time. The URT_p is calculated as shown in Eq. (64), where A2 is the final drop-off stop and O1 the first pick-up stop. This can be both by bus or metro depending on the trip type. For singular trips of type ODB or FTP this is straightforward (Fig. 7d and Fig. 7e). For dual trips of type FPT+ODB or ODB+FPT (Fig. 7c and Fig. 7b), when calculating the URT_p-part of the insertion criterion, we always assume that both legs are consecutive. When actually inserting the

[12]This would result in a journey ODB+FPT or FPT+ODB or FPT. In the latter both the considered origin metro station and the considered arrival metro station are within walking distance

Algorithm 1: Insert request p (Min. URT)

1 Best insertion criterion found = ∞ ;
2 **for** *Every* $l \in L_p$ **do**
3   Generate set of possible on-demand buses for leg 1 B_{lpleg1};
4   **for** *Every* $i \in [Index_{o-min} Index_{o-max}]$ **do**
5   Find best ODB leg 1 from one of the origin stations of p to metro station on l with index i when considering buses from set B_{lpleg1} ;
6   **if** *Feasible leg 1 found* **then**
7   Let $b_{oli} \in B_{lpleg1}$ be the bus where insertion of the first leg fits best (given l and i) ;
8   Insertion criterion leg 1 = $URT_{pleg1} + \Delta_{URT_{leg1}}$;
9   Best leg 2-3 insertion criterion = ∞ ;
10   **for** *Every* $j \in [Index_{a-min}, Index_{a-max}]$ **do**
11   Generate set of possible on-demand buses for leg 3 B_{lpleg3} ($b_{oli} \notin B_{lpleg3}$) ;
12   Find best ODB leg 3 from metro station on l with index j to one of the walk-able arrival stations of p when considering buses from set B_{plleg3} ;
13   Insertion criterion leg 2-3 = $URT_{pleg3} + \Delta_{URT_{leg3}}$;
14   **if** *Feasible leg 2-3 found and insertion criterion* $<$ *Best leg 2-3 insertion criterion* **then**
15   Save leg 2-3 ;
16   Best leg 2-3 insertion criterion = Insertion criterion leg 2-3 ;
17   Insertion criterion = Insertion criterion leg 1 + Best insertion criterion leg 2-3 ;
18   **if** *Insertion criterion* $<$ *Best insertion criterion found* **then**
19   Save this insertion (leg 1 and saved leg 2-3 on line 15) ;
20   Best insertion criterion found = Insertion criterion ;
21 Find direct ODB ride ;
22 **if** *Feasible insertion found* **then**
23   Perform best insertion ;

passenger into the solution, both legs will be scheduled consecutively, however because of dynamic requests coming in last minute, this might change later on. For trips of type ODB+FPT+ODB (Fig. 7a), we assume and initially schedule the first two legs consecutively, but in between the second and the third leg, waiting time can occur. This waiting time is included in the URT_p.

$$URT_p = t^a_{ab} - t^d_{ob} \qquad ODBRP \qquad (63)$$

$$URT_p = t^a_{A2b} - t^d_{O1b} \qquad I-ODBRP \qquad (64)$$

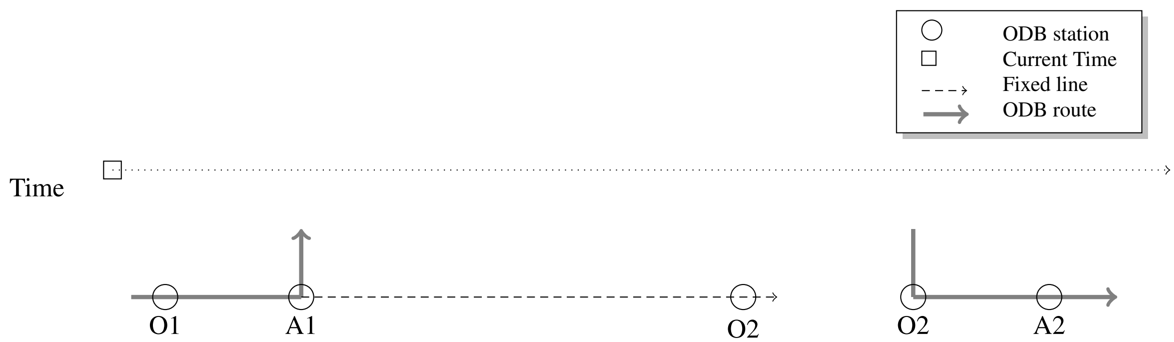
$$\Delta URT = \delta_{op'} \times q_{bn} + \delta_{ap'} \times q_{bn'} \qquad ODBRP \qquad (65)$$

$$\Delta URT = \delta_{op'} \times q_{bn} + \delta_{ap'} \times q_{bn'} + \Delta_{FPT+ODB} + \Delta_{ODB+FPT+ODB} + \Delta_{timeshift} \qquad I-ODBRP \qquad (66)$$

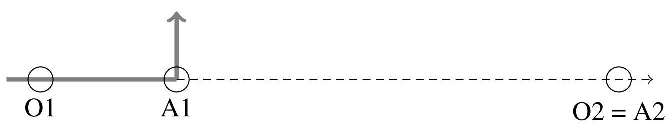
In the ODBRP, inserting an extra passenger p' in bus b can only have consequences for passengers scheduled on bus b. The increase in URT of the already scheduled passengers is shown in Eq. (65), with $\delta_{op'}$ the detour made by b to pick up the extra passenger p' at stop n, and $\delta_{ap'}$ the detour made by b to drop off the extra passenger p' at stop n'. In contrast, inserting a passenger in b in the I-ODBRP, might have consequences for passengers scheduled to be picked up by this bus at a later moment in time (at a bus stop $n'' > n'$). These passengers are not part of the load count q_b at stop n or n'. This is due to the fact that some passengers have multiple legs which are connected. Besides the ΔURT calculated for the ODBRP based on detour times load, there are three extra situations that cause the ΔURT to increase in the I-ODBRP, shown in Eq. (66). The first two, $\Delta_{FPT+ODB}$ and $\Delta_{ODB+FPT+ODB}$, are depicted in Fig. 8. $\Delta_{timeshift}$ is explained in Fig. 9 in the next paragraph.

In Fig. 8a a pick-up is delayed of a passenger p'' who is currently traveling on the metro network and is scheduled in bus b_1 afterwards. The square indicates the current time. The delay is caused by passenger p' because his pickup and/or drop-off, occur before the pickup of passenger p''. If both the origin and arrival of p' are located before O2 of passenger p'', the delay amounts $\Delta_{p'} = \delta_{op'} + \delta_{ap'}$. Passenger p'' will experience an increase in URT, as he will have to wait at the end of his FPT-leg until the on-demand bus arrives. The FPT-leg cannot be delayed anymore because p'' is already on the move. If p'' would still be at his origin location, he can just take a later metro to the location of $O2$ as we assume the metro network to be of high-frequency. Similarly, if passenger p'' has a three-leg trip type, delaying his third leg, or second ODB-leg by inserting the origin and/or arrival of p' in the same bus b_2 before the pick-up of p'' in his final leg, will cause the URT to increase. This is illustrated in Fig. 8b. In contrast to the previous situation, this increase in URT should be taken into account regardless whether or not passenger p'' has already left his origin location. Even if p'' takes a later metro, there will still be an extra time gap after arriving at the location of $A1$ with the first scheduled on-demand bus.

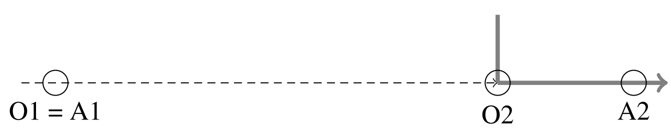
Feasibility check and time shift Besides the adjustments that need to be made regarding the insertion criterion, there are also some adaptations necessary for checking feasibility. In the ODBRP, when inserting a passenger in bus b_1, this can only have consequences for passengers scheduled in b_1. However, in the I-ODBRP, this might also influence passengers that are *not* scheduled in b_1, in particular when passengers have the ODB-FPT-ODB trip type. The same feasibility check of Melis and Sörensen (2020, 2021) using the buffer data structure is kept for all passengers in b_1. In addition, an extra feasibility check is introduced for passengers with the three-leg trip type. This is illustrated in Fig. 9. When inserting passenger p' in bus b_1, the detours necessary to pick up and drop off this passenger can generate a delay for the existing passengers in the route. For passenger p'', this delay amounts $\Delta_{p'}$. However passenger p'' has an ODB+FPT+ODB trip type and the leg scheduled in b_1 is his first leg. The delay will consequently cause delays for the other two trips as well. In the case of p'', the passenger will take a later metro to the location of $O2$ and arrive late for his pickup by on-demand bus b_2 scheduled to depart from this place (see the overlap in time between the two $O2$-points). As a solution, the algorithm will try to introduce a linear time shift for the entire schedule of bus b_2 starting from the pickup of passenger p''. This way, the solution is once again feasible. However, before doing the linear time shift, feasibility is checked for all passengers conflicted. Note that a chain reaction is possible. If there are passengers in b_2 who are in there first ODB-leg and have a three-leg trip type, more linear time shifts in other buses might need to happen. If one of the time shifts will result in unfeasibility, the insertion of passenger p' in b_1 will be declared unfeasible and will not be performed. The linear time shift and possibly resulting chain reaction of time shifts, can



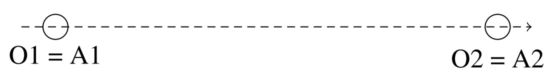
(a) Passenger with ODB-FPT-ODB as a trip type is inserted



(b) Passenger with ODB-FPT as a trip type is inserted



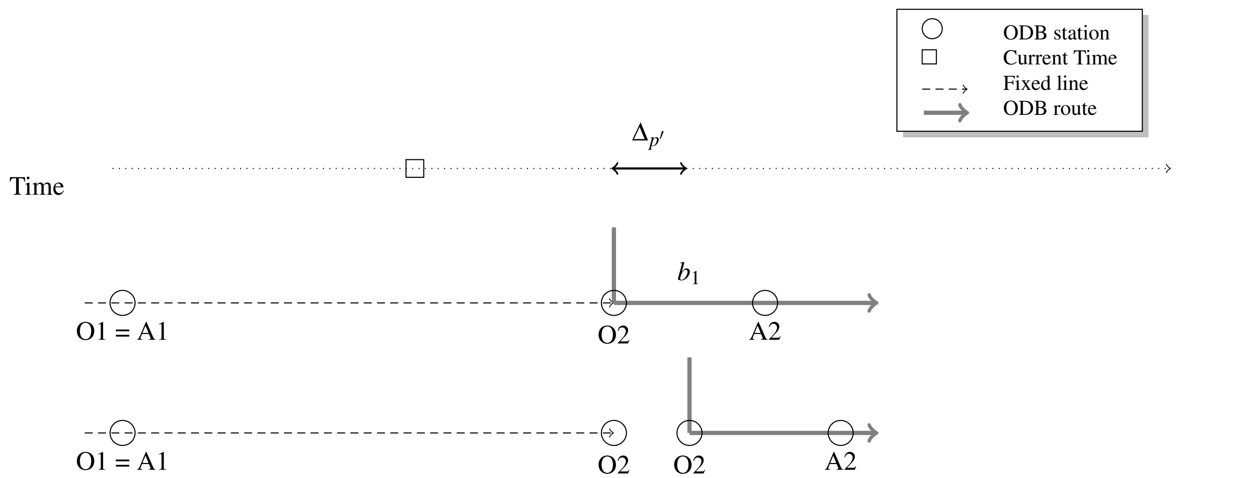
(c) Passenger with FPT-ODB as a trip type is inserted



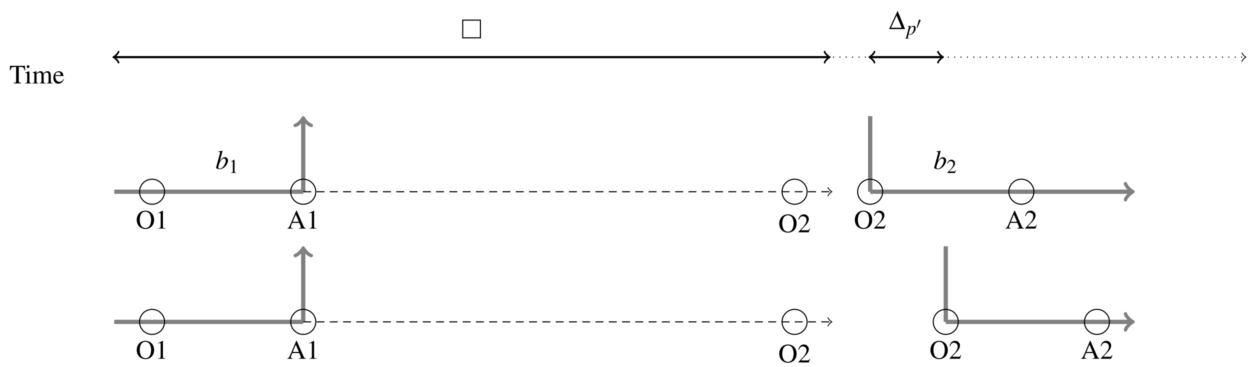
(d) Passenger with FPT as a trip type is inserted

(e) Passenger with ODB as a trip type is inserted

Figure 7: Increase in URT_P depending on trip type



(a) $\Delta_{FPT+ODB}$ - Indicated is the FPT+ODB trip of passenger p''. The last ODB-leg of this passenger is delayed by inserting pickup and/or drop-off of p' last minute in the same bus b_1 before the pickup of p''



(b) $\Delta_{ODB+FPT+ODB}$ - Indicated is the ODB-FPT-ODB trip of passenger p''. The last ODB-leg of this passenger is delayed by inserting pickup and/or drop-off of p' in the same bus b_2 as the last ODB-leg of p'', before the pickup of p''

Figure 8: Increase in ΔURT

cause the ΔURT to increase even more for the situations explained in Fig. 8. Instead of a delay caused by a passenger that is inserted in the same bus, the delay would now be caused by the linear time shift.

A toy example to illustrate a single linear time shift (without chain reaction) can be found in Fig. 10. Information on requests and a distance matrix can be found in Appendix B Table B1. The initial schedule consists of two buses and two requests. The first bus picks up passenger 6 at his first stop and drops him off at his second stop. From here passenger 6 will take a fixed line to station 10. Note that the fixed line distance between station 1 and station 10 amounts only 7, instead of the 35 minutes by on-demand bus depicted in Table B1. From station 10, passenger 6 will be picked up by a second bus, together with passenger 7. The latter also came from a fixed line leg without a preceding on-demand bus. He has a FPT+ODB trip type, meaning that his origin location is within walking distance of the fixed line. Passenger 6, on the other hand, has a ODB+FPT+ODB trip type. The initial schedule is illustrated in Fig. 10a. In Fig. 10b a possible insertion for passenger 8 is checked for feasibility. The current time is now 9, right before the departure of the first bus. For the rules regarding the insertion of real-time passengers, we refer to Melis and Sörensen (2021). The pickup tested is located right after the first bus stop in the sequence of bus 1 and the drop-off would be together with passenger 6 at the previous second bus stop at station 1. The insertion that is tested here is a direct ODB trip type. The insertion criterion amounts $URT_{p8} + \Delta URT$. The first amounts $57 - 42 = 15$, which is the difference between the arrival time of passenger 8 at his assigned arrival bus station minus the departure time of passenger 8 at his assigned origin bus station. ΔURT for the first bus amounts the detour times load caused by the origin and arrival respectively. For the origin this is $36 \times 1 = 36$, for the arrival there is no detour[13]. However because passengers 6 and 7 have multiple leg trip types, there might be a ΔURT in other buses as well, in this case only the second bus in the figure is linked to the first, so we only have to check for possible delays in this bus. In Fig. 10b an overlap in time occurs like the one illustrated in Fig. 9. Passenger 6 will not make it in time to be picked up by the second bus after his fixed line leg. Therefore a time shift, with the size of $\Delta URT = 36$ from the first bus, is tested for feasibility. Both passengers 6 and 7 would still arrive on time, however the insertion criterion would further increase by 36, caused by the increase in URT of passenger 7, as the latter would already be on the metro when the insertion occurs. He will consequently arrive at station 10 at the previously scheduled time of 29, and will have to wait until time 65 to board the second bus. The total insertion criterion amounts $(57 - 42) + 36 \times 1 + 36 = 87$. If this is the lowest (feasible) insertion criterion found, the insertion will be performed, as is shown in Fig. 10c.

The inter-dependencies between legs in the I-ODBRP are a second reason for the choice of an efficient insertion based heuristic. Changing the itinerary of one passenger, which would be necessary when using Local Search operators, would have too many consequences on the solution as a whole. Repairing and recalculating the solution would cause the computation time to increase substantially which is disadvantageous for a dynamic environment.

4.5 Instance generation

In the remainder of this paper requests are randomly generated in a 100×100 Euclidean plane. There are 121 bus stations equally distributed over the plane in a grid-shape (see the dots in Fig. 3) . The driving time by bus between two adjacent stations is 10 minutes, but on-demand buses can drive in all directions. The driving time by metro between two adjacent stations is 5, unless stated otherwise. In addition a within-metro transfer time of 1 minute is adopted, which is equal to the dwell time of the on-demand buses. This time is necessary for passengers to be able to get off one vehicle and on to the next one. All buses have an identical capacity of 8 people, which represents the size of a typical minibus, the metro has unlimited capacity. 10 instances of 2000 randomly generated requests are used, of which the earliest departure times are all between time 10 and 70 of the time horizon. λ, the multiplier used to determine which metro segments longer then the shortest paths, are included in L, is set at 1.01. The other parameters regarding the metro segment (ε_d and ε_t), transfer station (I_r) and ODB (ε_a) assignment will be determined in the next section. A summary of the chosen parameter values can be found in Appendix A Table A2.

[13]The passenger to be inserted, is not counted in the load.

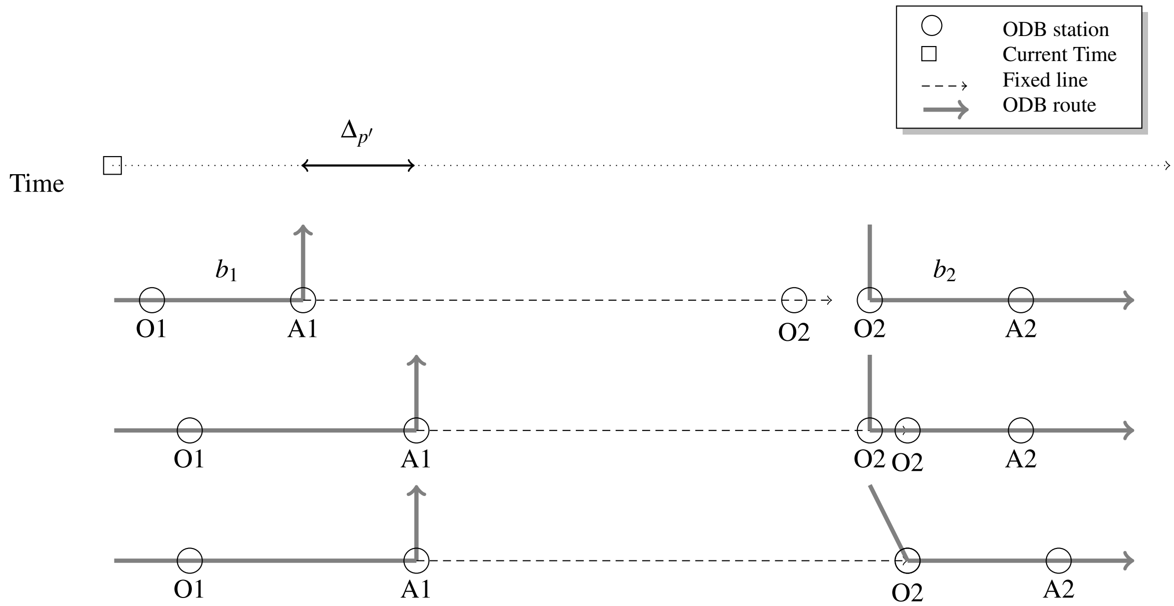


Figure 9: Indicated is the ODB+FPT+ODB trip of passenger p''. A linear time shift of the second ODB-leg (bus b_2) of passenger p'' is caused by a delay in first ODB-leg (bus b_1) resulting from insertion of passenger p'

5 Algorithm analysis

5.1 Speed-up - Metro segment assignment

In this section, instances are solved without using transfer station assignment or on-demand bus assignment procedures[14]. This means that when a metro segment from L_p, is tried for insertion, all pairs of metro stations on the metro segment will be checked for both the first and second ODB-leg of the route. In addition, when trying to insert (part of) a request in an on-demand bus, the entire fleet B is checked for insertion. There are two parameters deciding on the metro segment assignment: the *cut of corner (direction)* or ε_d and the *threshold value (distance/time)* or ε_t. The third criterion of the metro segment assignment procedure (accessibility) is already included. The cut off corner, threshold value and accessibility criterion are explained in Section 4.1.

Figure 11 shows the results for both the average URT per passenger[15] and the computation time for all three proposed networks. It is clear that for the star-shaped network, which is a very dense network, depicted in Fig. 11a, the direction parameter is of major importance when looking at the URT per passenger, while for the less dense cross-shaped and single line network (shown in Fig. 11c and Fig. 11e) the distance/time parameter is of makes the biggest differences.

For the star-shaped network, for each of the tested fleet sizes, the URT per passenger is at his absolute lowest when $\varepsilon_d = 180$. This means metro segments going in all directions are tested to function as the fixed leg. However, URT's per passenger do not show extreme differences for cut off corners between 15 and 180. For all tested fleet sizes, the URT's per passenger for $\varepsilon_d = 30$, are closest to the best found URT per passenger, independent of ε_t. Moreover, when $\varepsilon_d = 30$, no significant differences in URT per passenger can be found when changing ε_t. In contrast, the computation time, shown in Fig. 11b, does show significant differences when altering ε_t. Therefore, $\varepsilon_t = 1$ and $\varepsilon_d = 30$ are the selected values for the star-shaped network[16].

[14]Bus station assignment is already included in the algorithm, because the procedure's good performance is already thoroughly confirmed in Melis and Sörensen (2020, 2021). Further, in Section 5.4 the influence of bus station assignment for the I-ODBRP will be investigated.

[15]The average URT per passenger is the total URT per passenger divided by the number of served passengers. Throughout this paper also called the URT per passenger.

[16]An ε_t lower than 1, was also tested but not included in the graphs because the URT's per passenger showed substantial increases.

P7

Bus stop	1	2	1	2	3
Bus station	4	1	10	11	3
Arrival time	10	21	29	35	66
Time	11	22	30	36	67
P on	6		6,7		
P off		6		6	7
q_{nb}	1	0	1	1	0

P6

(a) Initial schedule (current time is 0): Passenger 6 has a ODB+FPT+ODB schedule and passenger 7 has a FPT+ODB schedule

P7

Bus stop	1	*	2	1	2	3
Bus station	4	5	1	10	11	3
Arrival time	10	41	57	29	35	66
Time	11	42	58	30	36	67
P on	6	8		6,7		
P off			6,8		6	7
q_{nb}	1	2	0	1	1	0

P6

(b) Passenger 8 is checked for insertion in first bus. The current time is 9.

P7

Bus stop	1	2	3	1	2	3
Bus station	4	5	1	10	11	3
Arrival time	10	41	57	65	71	102
Time	11	42	58	66	72	103
P on	6	8		6,7		
P off			6,8		6	7
q_{nb}	1	2	0	1	1	0

P6

(c) Insertion of passenger 8 is performed, with a linear time shift in the second bus as a consequence

Figure 10: Toy example - Insertion criterion, feasibility check and linear time shift for the I-ODBRP (dotted lines represent fixed line connections taken by the passenger indicated)

For the cross-shaped and single line network, it is clear that $\varepsilon_d = 30$ and $\varepsilon_t = 1$ are the most optimal parameters regarding the URT per passenger. Regarding the computation time, $\varepsilon_t = 0.5$ yields slightly better results, but considering the increase in URT per passenger, $\varepsilon_t = 1$ seems the better choice.

5.2 Flexibility - Transfer station assignment

In this section we will investigate the influence of transfer station assignment, which adds flexibility to the routing part of the algorithm for passengers who can be served with a multi-leg trip. This consequently has a positive impact on the URT per passenger. The parameter defining the induced flexibility is called the index range and is explained in Section 4.2.

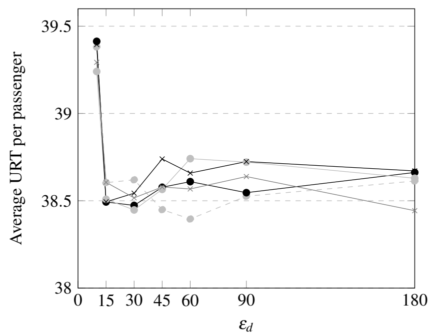
Results are shown in Fig. 12. The more dense the network, the more beneficial it is to add flexibility through transfer station assignment. For a fleet size of 400, there is a 1.3% decrease in URT per passenger for the star-shaped network, a 1% decrease for the cross-shaped network and a 0.3% decrease for the single line network, when going from zero flexibility to an index range of 1. Index ranges larger than 1 do not significantly contribute to the solution quality, while the computation time rises. Note that an index range of 0, is not equal to the problem where only the single closest transfer stations for departure and arrival are tested. When the index range is 0, the closest transfer stations for departure and arrival are tested for *every* metro segment $l \in L_p$, which still results in a somewhat flexible approach. For a clean comparison of the I-ODBRP with flexible metro segments and transfer station assignment and the I-ODBRP which only considers the single closest stations, we refer to Fig. 13 depicting the results for the star-shaped network. Case 1 depicts the results for checking all metro segments and transfer stations for insertion, while case 3 indicates the results obtained with our metro segment and transfer station assignment procedures. When going from case 1 to 3, the computation time decreases with 94% while the URT per passenger also slightly decreases with 0.5%. Case 5 illustrates the results when only the closest metro segment and closest transfer stations on this segment are checked for a multi-leg insertion. When going from case 3 to 5, the computation time decreases 54% more, but the URT per passenger increases with 35% and the service rate drops from 100 to 98%.

5.3 Speed-up - On-demand bus assignment

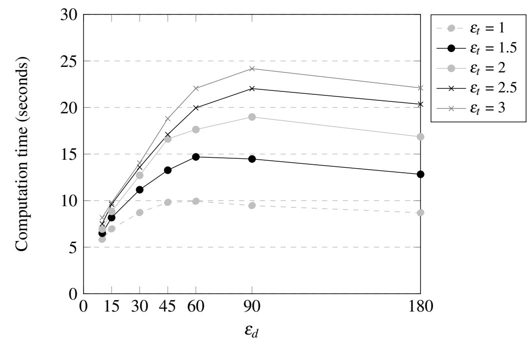
In Section 4.3 three efficient assignment procedures to quickly assign a request to a possible set of buses are introduced. In this section, we will analyse these procedures, and compare them to a regular insertion based heuristic that tries every available bus for insertion of the request. The latter is called the baseline. The baseline results are obtained by including the metro segment and transfer station assignment procedures with the decided-upon parameters of Section 5.1 and Section 5.2. Results of the URT per passenger, computation time and service rate are shown in Fig. 14 for all three proposed metro networks, solved with a fleet size of 400. The 'origin' and 'origin and arrival'-based procedure are explained in Section 4.3.1 and Section 4.3.2, respectively, the 'direction'-based procedure in Section 4.3.3. The threshold values, ε_a, which decide whether a bus is included in the possible bus insertion set of a request is depicted on the x-axis and expressed in minutes for the first two procedures (lower x-axis), and in degrees for the last (upper x-axis).

It is clear that on-demand bus assignment especially has a positive impact on the more extended, star-shaped network, with URT's per passenger which are close or even better than the baseline for both the 'origin'- and 'origin and arrival'-based procedure. For the cross-shaped and single line network, when lowering ε_a, the URT per passenger increases first and afterwards drops again. This can be explained by the fact that when ε_a is too small, and no buses are found that qualify the criteria for possible insertion, all buses are tested and results become equal to a high ε_a.

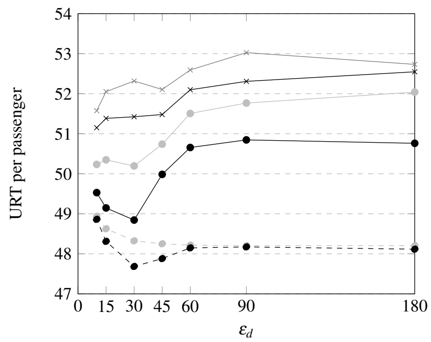
In general, for all networks, the ODB assignment procedure based on both the earliest departure time and origin location, and (latest arrival time and) the arrival location (origin + arrival in the figure), has the best performance when looking at the URT per passenger and the total computation time. However, one should also take into account the influence on the service rate. This is the percentage of passengers that can be served while meeting all constraints on time windows, capacity of the buses, etc. There is always a trade-off between computation time on one hand and the service quality (URT per passenger and service rate) on the other. Deciding upon the used method for ODB assignment and the accompanying ε_a is therefore no exact science. In Fig. 14, one can see that the higher ε_a, the more the computation rise rises, even higher than the baseline. The reason for this difference is that in the baseline, without ODB assignment, no calculations need to be made to determine which buses will be tested for insertion, while



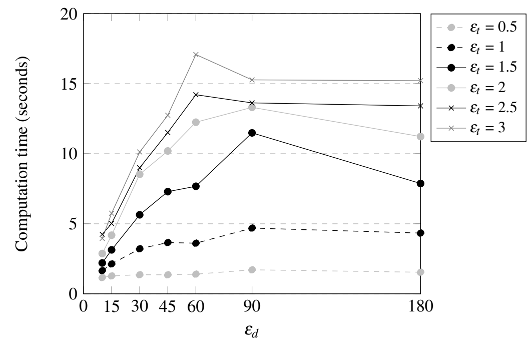
(a) Average URT per passenger - star-shaped network



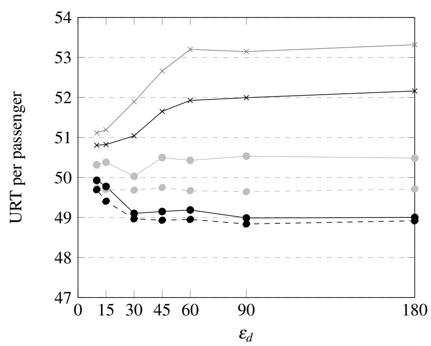
(b) Total computation time - star-shaped network



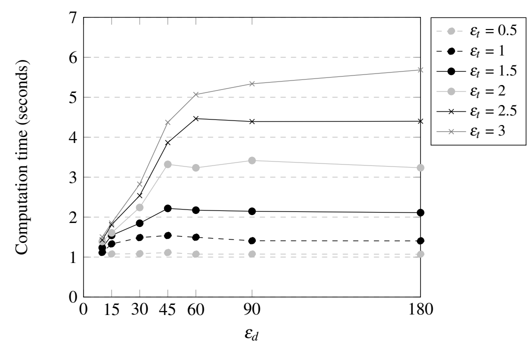
(c) URT per passenger - cross-shaped network



(d) Total computation time - cross-shaped network



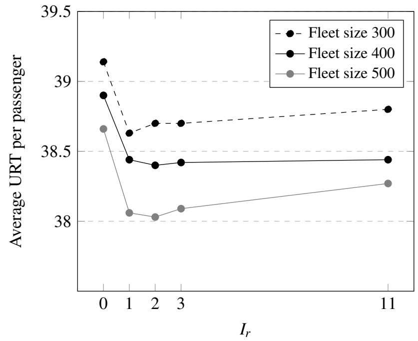
(e) URT per passenger - single-line network



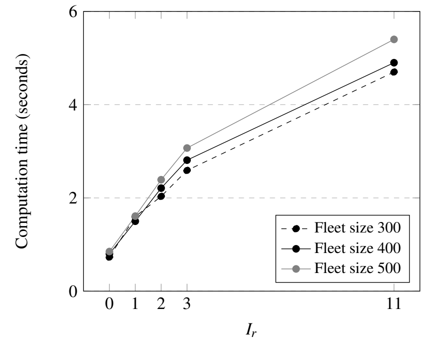
(f) Total computation time - single-line network

Instance: fleet size 400, number of requests 2000

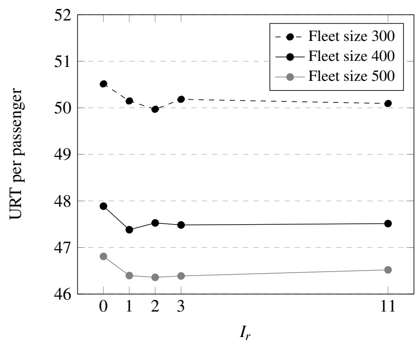
Figure 11: Parameter setting metro segment assignment for star-shaped, cross-shaped and single line network



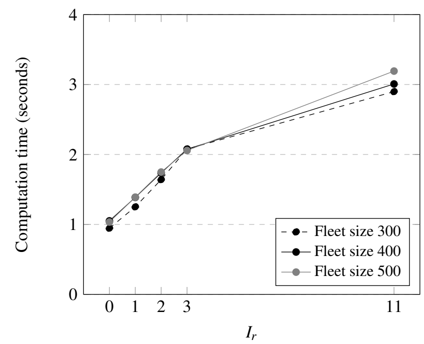
(a) Average URT per passenger - star-shaped network



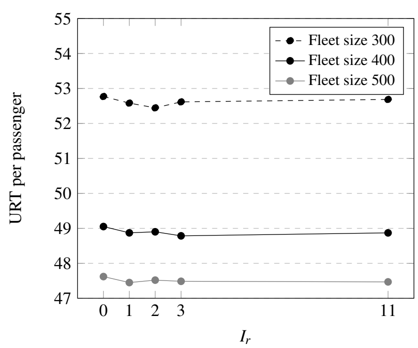
(b) Total computation time - star-shaped network



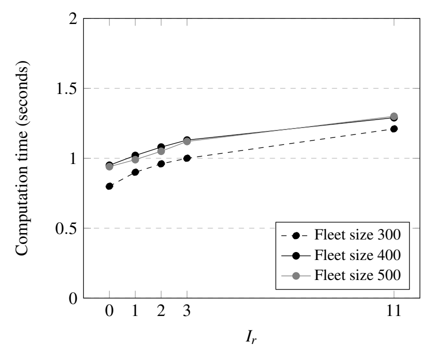
(c) URT per passenger - cross-shaped network



(d) Total computation time - cross-shaped network



(e) URT per passenger - single-line network

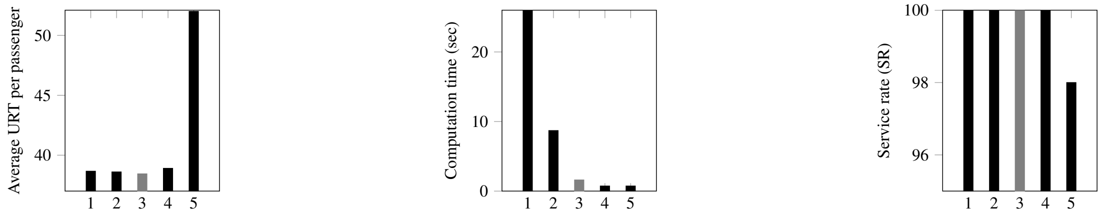


(f) Total computation time - single-line network

Instance: number of requests 2000

Figure 12: The influence of transfer station assignment for different fleet sizes

	Considered for insertion:				
	1	2	3	4	5
Metro segments	all	efficient	efficient	efficient	closest
Transfer stations	all	all	efficient	closest	closest



Instance: fleet size 400, number of requests 2000

Figure 13: Comparison checking all metro segments and transfer stations for insertion with checking only the closest ones for the star-shaped network

using an ODB assignment procedure always needs extra calculations. However, when the number of buses tried for insertion is lower, the additional computation time for the assignment procedures is counteracted with the decrease in computation time by smartly assigning passengers to buses.

We chose $\varepsilon_a = 10$ (in minutes) for the cross-shaped and single line network, because in this case, the average URT per passenger rises slightly, and the service rate decreases slightly, while there are mayor benefits regarding the computation time. One could choose a threshold value of 15 or 20 minutes to decrease the computation time even more. However, in this case the service quality is at its lowest point. For the star-shaped network, we chose $\varepsilon_a = 15$. The average URT per passenger is even slightly lower compared to the baseline. The percentage differences obtained by introducing the ODB assignment procedure with the ”origin + arrival” method in Table 4. The ODB assignment procedures can also be applied to the ODBRP. For completeness, this analysis is included in Appendix D. For a quick comparison, we included the results for the ”origin + arrival” procedure with a threshold value of 10 minutes in Table 4 as well. The average URT per passenger for the ODBRP is less influenced by the ”origin + arrival” ODB assignment method, compared to the I-ODBRP-variants, while the service rate is more influenced. Differences between the I-ODBRP and the ODBRP will be further investigated in Section 6.1.

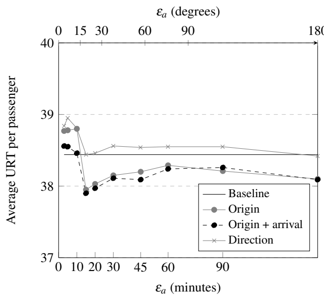
Table 4: Percentage differences between baseline and ODB assignment ”origin + arrival”

Network	ε_a	Average URT per passenger	CT	Service rate
Star-shaped	15	-1.4%	-33.7%	-0.03%
Cross-shaped	10	+1.7%	-28.8%	-5%
Single line	10	+7.3%	-17.9%	-8.7%
Non-integrated ODBRP	10	+0.6%	-21.9%	-10.5%

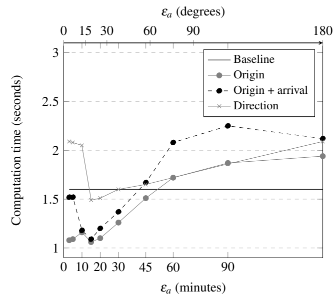
Instance: fleet size 400, number of requests 2000

5.4 Flexibility - Bus station assignment

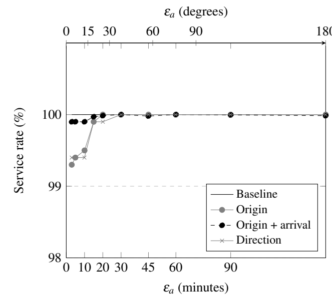
In this paper, all conducted experiments include bus station assignment. To once again proof the performance increases induced by this feature, in this final section of the algorithm analysis, we excluded bus station assignment from the algorithm. This means that passengers choose their own bus stations for departure and arrival, typically the closest ones to their origin and arrival location. The metro segment, transfer station and ODB assignment procedures are kept in place. Table 5 shows the percentage differences for the average URT per passenger and the service rate with and without bus station assignment. Regarding the average URT per passenger, bus station assignment is of even bigger importance when the ODBRP is integrated with a fixed line network. In contrast, the service rate suffers less from the



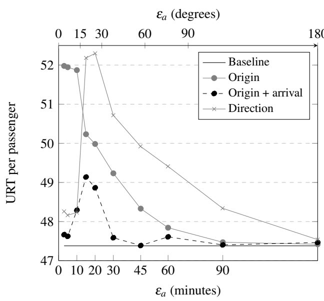
(a) Average URT per passenger - star-shaped network



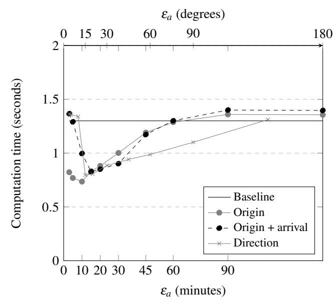
(b) Total computation time - star-shaped network



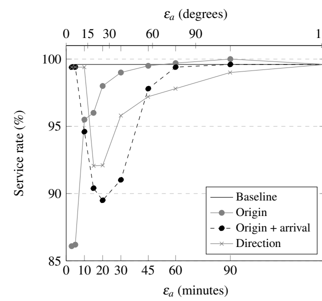
(c) Service rate - star-shaped network



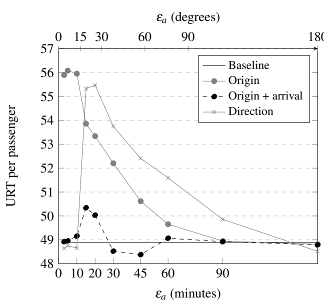
(d) URT per passenger - cross-shaped network



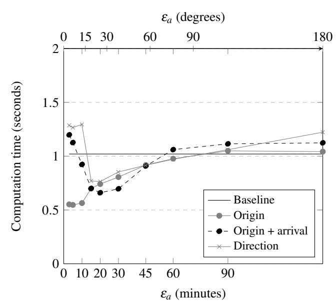
(e) Total computation time - cross-shaped network



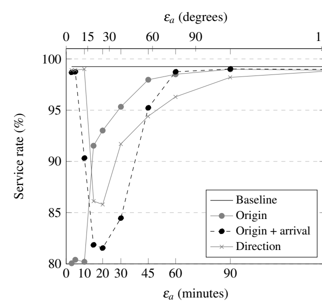
(f) Service rate - cross-shaped network



(g) URT per passenger - line-shaped network



(h) Total computation time - line-shaped network



(i) Service rate - line-shaped network

Instance: fleet size 400, number of requests 2000

Figure 14: On-demand bus assignment analysis for the star-shaped, cross-shaped and single line network

exclusion of bus station assignment when the ODBRP is integrated with a dense fixed network, like the star-shaped one. In general, similar to the non-integrated ODBRP, bus station assignment results in a significant better solution quality for the I-ODBRP.

Table 5: Percentage differences with and without bus station assignment

Network	Average URT per passenger	Service rate
Star-shaped	+39.9%	-4.8%
Cross-shaped	+29.7%	-10.9%
Single line	+28.7%	-11.0%
Non-integrated ODBRP	+28.3%	-11.5%

Instance: fleet size 400, number of requests 2000

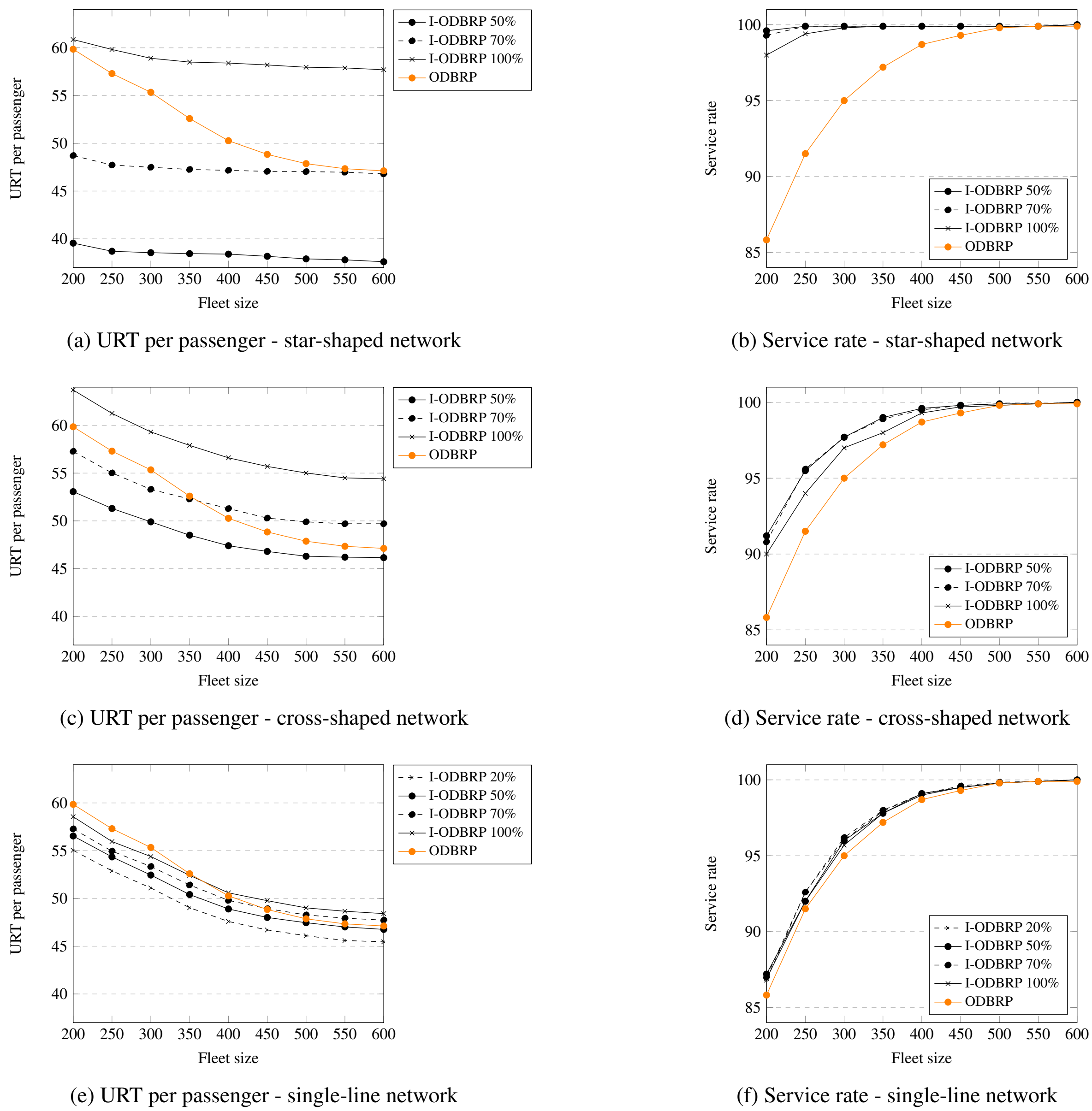
6 Results analysis

6.1 Is it beneficial to integrate on-demand buses with a fixed metro network?

In this section we will investigate whether or not better performance is reached when integrating on-demand buses with a fixed metro network. We learned from Section 5.3 that using on-demand bus assignment procedures as a computational speedup is beneficial for the computation time, but might have a negative impact on the solution quality, expressed in the average URT per passenger and the service rate. For a clean comparison we therefore check the entire fleet of on-demand buses for insertion for both the I-ODBRP as the ODBRP, as different ODB assignment procedures and threshold value have different impacts on both problem variants. Once again, the URT per passenger and the service rate are included as a measure of performance in the comparison. Instances are solved with different fleet sizes and metro-line speeds. Results can be found in Fig. 15. The metro speed is depicted as a percentage of the on-demand bus speed. A percentage of 100 corresponds to an equal speed of the metro network and the on-demand buses, a percentage of 50 means that the metro drives 2 times as fast as the on-demand buses.

Average URT per passenger Of course the more dense the network, the more the metro speed influences the URT per passenger, and this for all fleet sizes of on-demand buses. On the other hand, the graphs clearly show that that the available fleet size is of minor importance regarding the URT per passenger when the metro network is more dense. For the star-shaped network the I-ODBRP has equal or better performance when the metro speed is 50% or 70% of the on-demand bus speed. When the metro speed is equal to the ODB speed, integrating ODB with FPT worsens the URT per passenger. For the less dense cross-shaped and single line network, the fleet size of on-demand buses becomes more important. When the fleet size is relatively small, integration performs better, but when the fleet size is sufficiently large only using on-demand buses works best. The metro speed at which this trade-off happens is higher for the less dense single line network than for the cross-shaped network. This means that in a single line network, when there is enough fleet size, it is better to use only on-demand buses. Only when the metro speed is 20% of the on-demand bus speed, integration is better for all available fleet sizes. In conclusion, regarding the URT per passenger it is better to integrate with a metro network during peak hours, when demand is high compared to the available fleet size, but during off-peak hours passengers can better be transported using only on-demand buses. When the network is really dense like the star-shaped network, this switch in operating mode is not necessary, integration is always recommended.

Service rate When looking at the service rate, the ODBRP is severely influenced by the fleet size, while the star-shaped network used in the I-ODBRP, has overall high service rates, independent of the fleet size or metro speed. For the cross-shaped and single line network, service rates are also overall higher compared to the ODBRP, but the difference is smaller the less dense the network. In conclusion, integrating on-demand buses with a metro network yields better service rates compared to only using on-demand buses.



(a) URT per passenger - star-shaped network

(b) Service rate - star-shaped network

(c) URT per passenger - cross-shaped network

(d) Service rate - cross-shaped network

(e) URT per passenger - single-line network

(f) Service rate - single-line network

Instance: number of requests 2000

Figure 15: Comparison results I-ODBRP with ODBRP for different fleet sizes and, metro-line speeds and networks

7 Conclusion and future research

In this paper the integrated on-demand bus routing problem or I-ODBRP is introduced to integrate a large-scale on-demand bus system with a high-frequency metro network. In the I-ODBRP passengers can have multi-leg trip types consisting of both metro and bus trips. To solve the I-ODBRP we used the constructive heuristic for the ODBRP of Melis and Sörensen (2021) to insert passengers in on-demand buses, however we adopted the insertion criterion and feasibility checks to fit the I-ODBRP. To decide upon the trip of each passenger, two extreme insertion strategies exist: one could check all buses, metro stations and lines for insertion, or one could only check the closest ones. Our algorithm finds middle ground between these two extremes and introduces two procedures for computational speed-up and two for flexibility. On the ODB-level, bus stop assignment, already introduced and proven to yield better performance in Melis and Sörensen (2021), is also included in this algorithm to keep the flexibility of the routing of the on-demand buses high. We found that also in the I-ODBRP bus station assignment remains important, especially regarding the average URT per passenger. In addition, because the I-ODBRP has more routing opportunities compared to the ODBRP, an on-demand bus assignment procedure is introduced to improve the computation time. With regard to the choosing of metro lines and transfer stations, we included a metro segment and transfer station assignment procedure for speed-up and flexibility, respectively. We used our algorithm to investigate whether or not it is beneficial to integrate on-demand buses with a high-frequency metro network, using three different types of metro networks. Regarding the URT per passenger it is mostly better to integrate with a metro network during peak hours, when demand is high compared to the available fleet size, but during off-peak hours passengers can better be transported using only on-demand buses. Only when the network is really dense, integration is always recommended. In addition, for all network types the service rate improves and a lower fleet size is needed, which results in lower operating costs.

In this work we assumed that all metro lines have a high-frequency service, so that transfer times within the metro network are negligible. In reality many fixed line public transport systems have fixed timetables. Future research is needed to include these timetables in the algorithm. In addition, instances based on real cities can be used and real-time traffic information could be included in the algorithm. Moreover there is opportunity for a more sophisticated heuristic that includes local search operators to improve the solution.

References

Aldaihani, M. and Dessouky, M. M. (2003). Hybrid scheduling methods for paratransit operations. *Computers & Industrial Engineering*, 45(1):75–96.

Archetti, C., Speranza, M. G., and Weyland, D. (2018a). On-demand public transportation. *Int. Trans. Oper. Res*.

Archetti, C., Speranza, M. G., and Weyland, D. (2018b). A simulation study of an on-demand transportation system. *International Transactions in Operational Research*, 25(4):1137–1161.

Atasoy, B., Ikeda, T., and Ben-Akiva, M. E. (2015). Optimizing a flexible mobility on demand system. *Transportation Research Record*, 2563(1):76–85.

Auad, R., Dalmeijer, K., Riley, C., Santanam, T., Trasatti, A., Van Hentenryck, P., and Zhang, H. (2021). Resiliency of on-demand multimodal transit systems during a pandemic. *arXiv preprint arXiv:2101.05395*.

Auad, R. and Van Hentenryck, P. (2021). Ridesharing and fleet sizing for on-demand multimodal transit systems. *arXiv preprint arXiv:2101.10981*.

Basciftci, B. and Van Hentenryck, P. (2021). Capturing travel mode adoption in designing on-demand multimodal transit systems. *arXiv preprint arXiv:2101.01056*.

Bertelle, C., Nabaa, M., Olivier, D., and Tranouez, P. (2009). A decentralised approach for the transportation on demand problem. In *From System Complexity to Emergent Properties*, pages 281–289. Springer.

Bischoff, J., Maciejewski, M., and Nagel, K. (2017). City-wide shared taxis: A simulation study in berlin. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 275–280. IEEE.

Bögl, M., Doerner, K. F., and Parragh, S. N. (2015). The school bus routing and scheduling problem with transfers. *Networks*, 65(2):180–203.

Edwards, D., Elangovan, A. K., and Watkins, K. (2012). Reaching low-density urban areas with the network-inspired transportation system. In *2012 15th International IEEE Conference on Intelligent Transportation Systems*, pages 826–831. IEEE.

Edwards, D., Trivedi, A., Elangovan, A. K., and Dickerson, S. (2011). The network-inspired transportation system: A hierarchical approach to integrated transit. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1507–1512. IEEE.

Ghilas, V., Demir, E., and Van Woensel, T. (2016). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows and scheduled lines. *Computers & Operations Research*, 72:12–30.

Häll, C. H., Andersson, H., Lundgren, J. T., and Värbrand, P. (2009). The integrated dial-a-ride problem. *Public Transport*, 1(1):39–54.

Hickman, M. D. and Blume, K. L. (2001). An investigation of integrated transit service. Technical report, Southwest University Transportation Center, Texas Transportation Institute ....

Hyland, M. and Mahmassani, H. S. (2020). Operational benefits and challenges of shared-ride automated mobility-on-demand services. *Transportation Research Part A: Policy and Practice*, 134:251–270.

Ikeda, T., Fujita, T., and Ben-Akiva, M. E. (2015). Mobility on demand for improving business profits and user satisfaction. *FUJITSU Sci. Tech. J*, 51(4):21–26.

Jäger, B., Brickwedde, C., and Lienkamp, M. (2018). Multi-agent simulation of a demand-responsive transit system operated by autonomous vehicles. *Transportation Research Record*, 2672(8):764–774.

Kumar, P. and Khani, A. (2021). An algorithm for integrating peer-to-peer ridesharing and schedule-based transit system for first mile/last mile access. *Transportation Research Part C: Emerging Technologies*, 122:102891.

Liaw, C.-F., White, C. C., and Bander, J. (1996). A decision support system for the bimodal dial-a-ride problem. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 26(5):552–565.

Ma, S., Zheng, Y., and Wolfson, O. (2013). T-share: A large-scale dynamic taxi ridesharing service. In *2013 IEEE 29th International Conference on Data Engineering (ICDE)*, pages 410–421. IEEE.

Mahéo, A., Kilby, P., and Van Hentenryck, P. (2019). Benders decomposition for the design of a hub and shuttle public transit system. *Transportation Science*, 53(1):77–88.

Masson, R., Lehuédé, F., and Péton, O. (2014). The dial-a-ride problem with transfers. *Computers & Operations Research*, 41:12–23.

Meijers, E. J. and Burger, M. J. (2010). Spatial structure and productivity in us metropolitan areas. *Environment and planning A*, 42(6):1383–1402.

Melis, L. and Sörensen, K. (2020). The on-demand bus routing problem: A large neighborhood search heuristic for a dial-a-ride problem with bus station assignment. *Working papers, University of Antwerp, Faculty of Business and Economics*, Article ID 2020005.

Melis, L. and Sörensen, K. (2021). The dynamic on-demand bus routing problem: What is the cost of real-time requests? *Working papers, University of Antwerp, Faculty of Business and Economics*, Article ID 2021003.

Molenbruch, Y., Braekers, K., Hirsch, P., and Oberscheider, M. (2020). Analyzing the benefits of an integrated mobility system using a matheuristic routing algorithm. *European Journal of Operational Research*.

Mourad, A., Puchinger, J., and Chu, C. (2019). A survey of models and algorithms for optimizing shared mobility. *Transportation Research Part B: Methodological*, 123:323–346.

Narayan, J., Cats, O., van Oort, N., and Hoogendoorn, S. (2017). Performance assessment of fixed and flexible public transport in a multi agent simulation framework. *Transportation Research Procedia*, 27:109–116.

Narayan, J., Cats, O., van Oort, N., and Hoogendoorn, S. (2020). Integrated route choice and assignment model for fixed and flexible public transport systems. *Transportation Research Part C: Emerging Technologies*, 115:102631.

Navidi, Z., Ronald, N., and Winter, S. (2018). Comparison between ad-hoc demand responsive and conventional transit: a simulation study. *Public Transport*, 10(1):147–167.

Posada, M., Andersson, H., and Häll, C. H. (2017). The integrated dial-a-ride problem with timetabled fixed route service. *Public Transport*, 9(1-2):217–241.

Posada, M. and Häll, C. H. (2020). A metaheuristic for evaluation of an integrated special transport service. *International Journal of Urban Sciences*, pages 1–23.

Potter, B. (1976). Ann arbor's dispatching system. *Transportation Research Record*, (608).

Ronald, N., Thompson, R., Haasz, J., and Winter, S. (2013). Determining the viability of a demand-responsive transport system under varying demand scenarios. In *Proceedings of the Sixth ACM SIGSPATIAL International Workshop on Computational Transportation Science*, pages 7–12.

Ronald, N., Thompson, R. G., and Winter, S. (2015). Comparison of constrained and ad hoc demand-responsive transportation systems. *Transportation Research Record*, 2563(1):44–51.

Salazar, M., Rossi, F., Schiffer, M., Onder, C. H., and Pavone, M. (2018). On the interaction between autonomous mobility-on-demand and public transportation systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2262–2269. IEEE.

Schittekat, P., Kinable, J., Sörensen, K., Sevaux, M., Spieksma, F., and Springael, J. (2013). A metaheuristic for the school bus routing problem with bus stop selection. *European Journal of Operational Research*, 229(2):518–528.

Shen, Y., Zhang, H., and Zhao, J. (2018). Integrating shared autonomous vehicle in public transportation system: A supply-side simulation of the first-mile service in singapore. *Transportation Research Part A: Policy and Practice*, 113:125–136.

Steiner, K. and Irnich, S. (2020). Strategic planning for integrated mobility-on-demand and urban public bus networks. *Transportation Science*, 54(6):1616–1639.

Stiglic, M., Agatz, N., Savelsbergh, M., and Gradisar, M. (2018). Enhancing urban mobility: Integrating ride-sharing and public transit. *Computers & Operations Research*, 90:12–21.

Tsubouchi, K., Hiekata, K., and Yamato, H. (2009). Scheduling algorithm for on-demand bus system. In *2009 Sixth International Conference on Information Technology: New Generations*, pages 189–194. IEEE.

Uber (2019). How it works. https://www.uber.com/be/nl/ride/how-it-works/. Online; accessed 12 February 2019.

Van Engelen, M., Cats, O., Post, H., and Aardal, K. (2018). Enhancing flexible transport services with demand-anticipatory insertion heuristics. *Transportation Research Part E: Logistics and Transportation Review*, 110:110–121.

Viergutz, K. and Schmidt, C. (2019). Demand responsive-vs. conventional public transportation: A matsim study about the rural town of colditz, germany. *Procedia Computer Science*, 151:69–76.

Wilson, N. H., Weissberg, R. W., and Hauser, J. (1976). Advanced dial-a-ride algorithms research project. Technical report.

Winter, K., Cats, O., Correia, G., and van Arem, B. (2018). Performance analysis and fleet requirements of automated demand-responsive transport systems as an urban public transport service. *International journal of transportation science and technology*, 7(2):151–167.

Zhu, Z., Qin, X., Ke, J., Zheng, Z., and Yang, H. (2020). Analysis of multi-modal commute behavior with feeding and competing ridesplitting services. *Transportation Research Part A: Policy and Practice*, 132:713–727.

Appendix

A Additional notation - heuristic

Notation	Definition
	Metro segment assignment - Section 4.1
λ	Multiplier used to include multiple metro segments from/to the same key stations when they have nearly same travel distances by metro
L	Overall set of metro segments for the given network
L_p	Possible metro segments set of passenger p
ε_d	Cut off corner (in degrees) for direction criterion of metro segment assignment procedure
s^o_{pl}	The closest metro station on metro segment l to the origin location of passenger p
s^a_{pl}	The closest metro station on metro segment l to the arrival location of passenger p
ε_t	Threshold value (in time) for distance/time criterion of metro segment assignment procedure
	Transfer station assignment - Section 4.2
$Index_{olp-min/max}$	Minimum/Maximum index for origin of passenger p on metro segment l
$Index_{alp-min/max}$	Minimum/Maximum index for arrival of passenger p on metro segment l
I_{olp}	Index of the closest metro station on metro segment l to the origin location of passenger p
I_{alp}	Index of the closest metro station on metro segment l to the arrival location of passenger p
I_{endl}	Index of the last metro station on metro segment l
I_r	Index range
	ODB assignment - Section 4.3
B_{lpleg1}	Number of buses checked for insertion for the first leg of passenger p on metro segment l
B_{lpleg3}	Number of buses checked for insertion for the second leg of passenger p on metro segment l
ε_a	Threshold value (in degrees or time, depending on chosen criterion) for ODB assignment procedure
	Algorithm outline - Section 4.4
ΔURT	The increase in URT of the already scheduled passengers in the solution
δ_{up}	The detour made to pick up passenger p
δ_{op}	The detour made to drop off passenger p
Δ_p	The total detour made to pick up and drop off passenger p
URT_p	The URT of the, to be inserted, passenger p
A2	The final drop off stop of a passenger trip (metro or bus stop, depending on trip type)
O1	The first pick up stop of a passenger trip (metro or bus stop, depending on trip type)

Table A1: Additional notation - heuristic

Parameter	Value
λ	1.01 (multiplier)
ε_d	30 (degrees)
ε_t	1 (multiplier)
I_r	1 (station)
ε_a	10 (minutes) (15 (minutes) for star-shaped network)

Table A2: Summary of chosen parameter values

B Toy example - ODB assignment

First the ODB assignment procedure loops over all buses and it checks their location at the earliest departure time of the passenger. The approximate locations of these buses for different time intervals are saved in a separate data structure. This way, the location of the buses at a certain time can be checked in linear time ($O(1)$). An example bus route and corresponding data structure to check the location of the bus can be found in Fig. B1. The distance matrix and total request list of the toy example in this figure can be found in Table B1. To simplify, all passengers in the toy example have direct ODB trips, but this does not have to be the case. When the bus is not driving to or standing near a bus station, the data structure indicates -1, which means that during these time intervals the bus has no requests assigned to it. If a fifth request is tried to be inserted with an earliest departure time larger than 155 (the data structure will indicate -1), the distance is expressed as the distance from the last served station in the route to the origin location of the passenger. If the earliest departure time is smaller than 155 (the data structure will indicate the bus stop s), the distance is expressed as the distance from the station scheduled as the s'th bus stop in the route to the origin location of the passenger. For passenger 5 with an earliest departure time of 90, the distance amounts 45 (the distance from station 10 to station 8 or 9). If , for example, a threshold of 15 minutes would be adopted, this results in the fact that this bus will not be considered for the insertion of passenger 5, unless the resulting set of possible buses for insertion remains empty. In this case all buses will be tested for insertion.

Table B1: Toy example - Requests and distance matrix

p	e_p^u	l_p^o	Departure stations	Arrival stations
1	0	50	4	5 or 6
2	10	90	4	10 or 11
3	70	160	10 or 11	5
4	120	170	7	5 or 6
5	90	180	8 or 9	3
6	10	75	4	11
7	5	110	N/S	3
8	20	80	5	1 or 2 or 3

Distance Matrix	1	2	3	4	5	6	7	8	9	10	11
1	0	5	5	20	15	13	20	25	20	35	35
2		0	5	20	15	14	15	15	20	35	35
3			0	20	15	10	12	14	14	25	30
4				0	30	35	40	40	40	60	60
5					0	5	30	28	32	40	41
6						0	30	28	32	40	38
7							0	5	5	40	45
8								0	5	45	50
9									0	45	50
10										0	5
11											0

P = Passenger, e_p^u = Earliest departure time of P, l_p^o = Latest arrival time of P

Bus stop	1	2	3	4	5
Bus station	4	5	10	7	5
Arrival time	10	41	82	123	154
Time	11	42	83	124	155
P on	1,2		3	4	
P off		1	2		3,4
q_{nb}	2	-1	0	1	-2

(a) Example bus route

Time	0	...	10	11	12	...	40	41	42	43	...	81	82	83	84	...
Bus stop (s)	0	...	0	0	0	...	0	2	2	2	...	2	3	3	3	...

Time	122	123	124	125	...	153	154	155	156	157	...	...
Bus stop (s)	3	4	4	4	...	4	5	5	-1	-1	...	-1

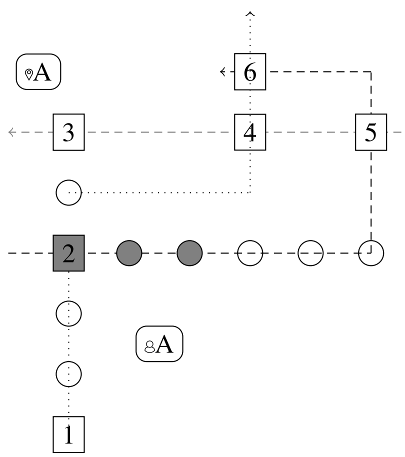
(b) Data structure indicating approximate bus location at a certain time

Figure B1: Toy example - Approximate bus location at a certain time

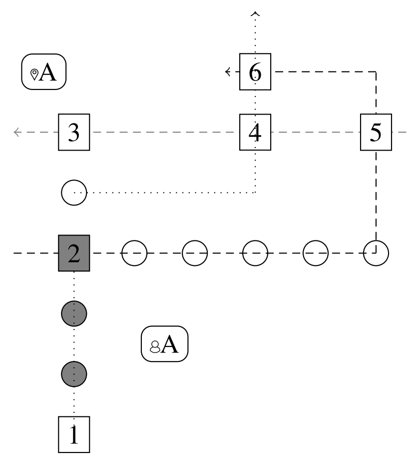
C Computational speed-up: Saving first leg insertions

In Algorithm 1, line 5 represents the possible insertion of a first leg ODB-ride. In contrast to the second leg, the earliest departure time does not change depending on the choice of past legs. Therefore computational speed-up is possible by saving promising insertions. Figure C1 shows the origin and arrival location of passenger A and a metro network consisting of three direct lines in both directions. The algorithm will look for a ODB+FPT+ODB trip. The on-demand bus stations that are not part of the metro network are not shown in the figure. There are two possible metro segments in L_p: (1) 2-5-6, (2) 1-2-4-3. The metro segments are described by their consecutive key stations (the squares in the figure). The algorithm will loop over these two metro segments and try to use a part of them as a FPT-leg for the trip of passenger A. Every sub-figure in Fig. C1 shows the considered metro stations as an arrival station of the first ODB-leg for a certain metro segment, colored in gray. The index range, I_r, is set to 1, so the stations include the closest station to the origin location of A, plus and minus one station on the metro segment. The algorithm will find the best ODB-leg from one of the possible (non-shown) origin stations of A to each of these metro stations colored in gray. However it is clearly visible that there is overlap in the considered stations for each metro segment. First the best first leg ODB-ride is found for each of the colored metro stations in Fig. C1a and all the insertion information for this best first ODB-leg is saved in a separate data structure. The algorithm continues after finding the best ODB-leg to each metro station by finding second ODB-legs just like in Algorithm 1. When considering the colored metro stations of the second metro segment (Fig. C1b), the best first leg ODB-ride to key station 2, does not need to be re-calculated. It is already saved. The best ODB-rides to the other two stations still need to be found, and are once again saved in case they are needed for other metro segments.

Saving second ODB-legs is not possible as the earliest departure time for the second ODB-leg in a three-leg trip will always depend on the first ODB and FPT trip that is chosen.



(a) Considered metro stations colored in black for arrival of leg 1 for metro segment (1) 2-5-6



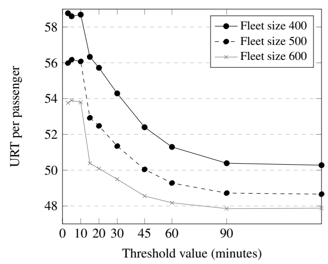
(b) Considered metro stations colored in black for arrival of leg 1 for metro segment (2) 1-2-4-3

Let (1) 2-5-6 and (2) 1-2-4-3 be two metro segments of L_A. In this case, there is overlap in considered metro stations when checking all metro segments L_A. (I_r = 1)

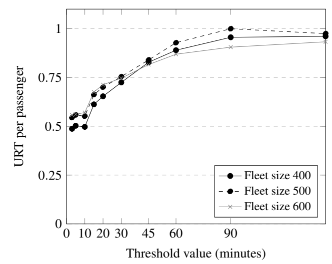
Figure C1: Saving first leg insertions

D On-demand bus assignment for the ODBRP

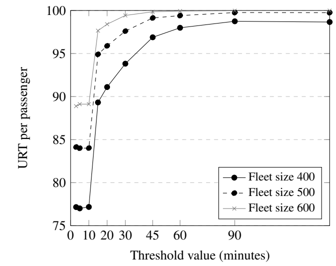
Fig. D1 shows the results for the three proposed ODB assignment procedures performed om the ODBRP.



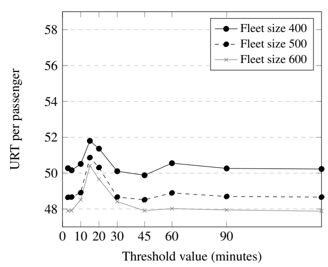
(a) URT per passenger - origin



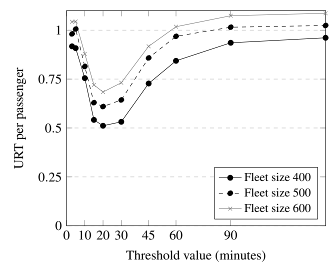
(b) Total computation time - origin



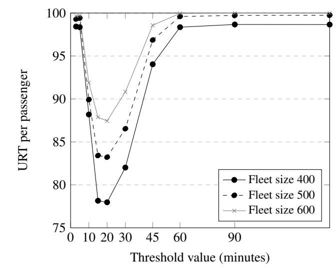
(c) Service rate - origin



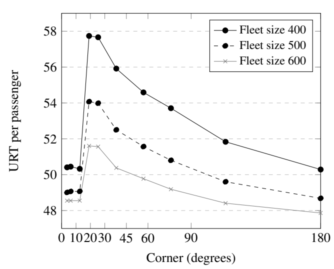
(d) URT per passenger - origin + arrival



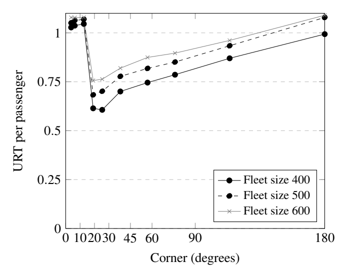
(e) Total computation time - origin + arrival



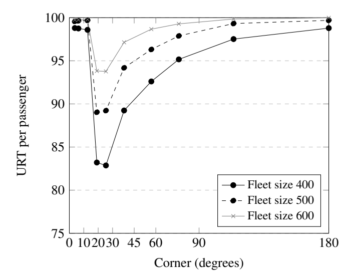
(f) Service rate - origin + arrival



(g) URT per passenger - direction



(h) Total computation time - direction



(i) Service rate - direction

Instance: number of requests 2000

Figure D1: Effect of the fleet size and different on-demand bus assignment procedures for the ODBRP