The real-time dynamic online feeder service with a maximum headway at mandatory stops

Bryan David Galarza Montenegro*
Department of Engineering Management (ENM), University of Antwerp
bryan.galarzamontenegro@uantwerpen.be

Kenneth Sörensen
Department of Engineering Management (ENM), University of Antwerp
kenneth.sorensen@uantwerpen.be

Pieter Vansteenwegen
KU Leuven Institute for Mobility - CIB, KU Leuven
pieter.vansteenwegen@kuleuven.be

Data availability statement
The authors confirm that the data supporting the findings of this study are available within the article and its supplementary materials.

Funding statement
This project was supported by the FWO (Research Foundation Flanders) project G.0759.19N.

The real-time dynamic online feeder service with a maximum headway at mandatory stops

Bryan David Galarza Montenegro[a] , Kenneth Sörensen[a] and Pieter Vansteenwegen[b]

[a] Department of Engineering Management (ENM) University of Antwerp, Antwerp, Belgium;
[b] KU Leuven Institute for Mobility - CIB, KU Leuven, Leuven, Belgium

ARTICLE HISTORY
Compiled June 23, 2023

ABSTRACT
On the one hand, fully flexible demand-responsive feeder services efficiently tailor their service to passengers' needs. Traditional services, on the other hand, offer predictability and easier cost control. This paper considers a semi-flexible feeder service that combines positive characteristics of both traditional and fully flexible services. There are two types of bus stops in this service. Mandatory bus stops have a maximum allowable headway for bus departures. Optional stops are only visited when there is demand for transportation nearby. When new passenger requests arrive, the performance of this feeder service is optimised in real-time. A metaheuristic with two phases is developed to optimise the service. The dynamic optimization method is compared to a model that optimises the service when all requests are known beforehand. The results show that the dynamic method has an average gap of 6.5% with respect to the static model, and an average acceptance rate of 95.1%. A case study in the city of Antwerp shows that, when compared to existing transit options in the region, this feeder service can increase the service quality by 31.6% when enough resources are available.

KEYWORDS
transportation ; public bus transport ; feeder service ; demand-responsive transportation ; meta-heuristics ; real-time optimization

1. Introduction

Traditional public bus services are services that have a fixed route and timetable, which are determined before the service is implemented, and never or rarely change. These transportation services take demand for transportation into account by determining both routes and timetables based on historical demand data. Historical data in a conventional bus service provides a long-term and collective relationship between demand and supply (Iliopoulou et al., 2019; Schöbel, 2012). However, individual user requests for transportation made in a short-term scale are generally not taken into account in the planning of either the routes or the timetables of the buses in such a service. This can lead to inefficiencies in the operation of such services when demand for transportation is either low or has a large variance (Li and Quadrifoglio, 2010).

Consequently, Demand-Responsive Transportation Services (DRTS) emerged to deal

CONTACT Bryan David Galarza Montenegro. Email: bryan.galarzamontenegro@uantwerpen.be

with the shortcoming of traditional bus services. DRTS are bus services that do not operate completely fixed routes and timetables. This flexible approach to public bus planning takes into account the individual demand for transportation, but it requires additional information about potential passengers. Information such as the origin, destination and preferred time of arrival and departure of individual passengers is important for the planning and operation of such services. This information can, for example, be obtained by asking passengers to make an explicit request for transportation using a mobile device or website. Although the collection of this data can be time-consuming and/or costly, the benefits are that DRTS take individual and short-term demand into account and adapt bus routes and/or timetables in much shorter time frames. This, in turn, leads to higher service quality. However, having a service where nothing is fixed, i.e., fully flexible DRTS, makes the service unpredictable, not easy to use for most people and less cost-efficient. The individual demand for transportation that is not explicitly requested or predicted cannot be met in fully flexible DRTS. Furthermore, recent findings suggest that high frequency bus services with fixed bus stops, i.e., conventional transportation services, can be used to guide urban planning towards more public-transit-friendly city designs (Knowles et al., 2020; Ibraeva et al., 2020). For these reasons, semi-flexible DRTS, i.e., DRTS where the planning is partially fixed, can offer a middle-ground between fully flexible DRTS and conventional services.

In this paper we work with feeder services. In these services, all passengers have the same destination but different origins. Passengers from typically sparsely populated areas are transported to areas with a high demand for transportation or to transportation hubs, where they can continue their journey. Feeder services can be an answer to the problem of overfull parking lots at transportation hubs and congestion in the surrounding area. These services can greatly benefit from both the flexibility that DRTS provides and the predictability and cost efficiency of conventional services. Therefore, the feeder service discussed in this paper is a semi-flexible DRTS.

More specifically, the semi-flexible DRTS studied in this paper is a variant of the Demand-Responsive Feeder System (DRFS), which has been previously introduced by Galarza Montenegro et al. (2021). This variant is called the Feeder Service with Mandatory Stops (FSMS). It provides service to two types of bus stops: mandatory stops and clustered optional stops. The mandatory stops are visited by each bus in the line. In the variant considered in this paper, these mandatory stops also have a maximum allowable headway, which means that a bus visits these stops within a specific time interval. A maximum headway of 20 minutes, for example, means that passengers will never have to wait more than 20 minutes at a mandatory stop. The maximum headway, or the length of this time interval, is a parameter chosen by the service provider. A bus only visits the optional stops when a potential passenger within walking distance requests transportation. Optional bus stops are introduced because for some passengers, such as children, disabled, or senior passengers, it is inconvenient to reach one of the limited number of bus stops that are typically present in a conventional service (Mistretta et al., 2009). Potential passengers make a transportation request to the transportation hub by stating their location and the time they wish to depart or arrive at the hub. In practice, this can be done through a website or a phone application. Passengers are then assigned to a departure bus stop, to which they must walk to. This means that the bus routes and schedules are not completely fixed and can be changed based on demand. The mandatory stops provide some predictability

and act as a safety net for 'passengers without reservations', as these passengers can still board a bus at the mandatory stops. As previously mentioned, the use of fixed transit stops is a driving factor in transit-oriented urban development (Knowles et al., 2020; Ibraeva et al., 2020), which makes the use of the mandatory stops more attractive as well. Furthermore, passengers at mandatory stops do not have to wait more than a certain amount of time because at least one bus departs from these stops within a certain time interval. Buses that arrive at the hub are expected to return to be reused for subsequent trips, which means that fewer buses are required to meet demand. It is found that attributes such as comfort and arrival time at the destination, but most importantly the reliability of a service, are highly valued by passengers (Beirão and Sarsfield Cabral, 2007). These attributes are present in the FSMS.

A static variant of the FSMS has been previously introduced in Galarza Montenegro et al. (2022b), in which the FSMS was optimised under the assumption that all requests are known beforehand. The main contribution of this paper is the design of an efficient dynamic optimization heuristic that optimises the FSMS, in a real-time manner, whenever new requests are received. We denote this variant of the FSMS as the dynamic FSMS (DFSMS). We assume that passenger requests are partially or completely unknown before the start of operation. Therefore, the service needs to adapt its planning whenever new requests are received, while still respecting several constraints of the service. The dynamic nature of the DFSMS makes optimization much more complex and thus requires more research. In order to optimise the performance of the service, a heuristic with an insertion phase and an improvement phase is developed. The insertion phase consists of an algorithm that aims to insert as many passenger requests into the solution as possible. The improvement phase consists of a greedy randomised algorithm that constructs different solutions based on randomised construction parameters. When compared to a static model of the problem, results suggest that the heuristic performs well in most cases, reaching a service quality similar to the static system, for which the performance is easier to optimise.

In the next section, a literature review on public transport feeder services is presented. In Section 3, we present a detailed description of the DFSMS. Section 4 explains the heuristic that is developed to optimise the performance of the DFSMS. Section 5 is an explanation of the experimental set-up. In Section 6, the results for several experiments are presented and discussed. The last section concludes the paper and discusses plans for future research.

2. Literature Review

The Feeder Service with Mandatory Stops (FSMS) has been previously presented in Galarza Montenegro et al. (2022b). Unlike this paper, Galarza Montenegro et al. (2022b) optimise the service with the assumption that all requests are known beforehand. This allows the static optimization algorithm to consider more feasible options and essentially find solutions with a better objective function value compared to a real-time optimization model. The static FSMS is an extension of the Demand-Responsive Feeder Service (DRFS) presented in Galarza Montenegro et al. (2021). In the DRFS, a fleet of buses transports passengers from a suburban area to an area with high demand for transportation, such as a train station or a city. The DRFS accepts

passenger requests until a certain deadline before dispatching the first bus. Each request specifies the passenger's current location as well as the desired arrival time at the destination. There are two different bus stops. Each bus always visits mandatory stops, while optional stops are visited if there is a demand for transportation nearby. The DRFS is optimised with the use of a Large Neighbourhood Search (LNS) heuristic in Galarza Montenegro et al. (2021). With the exact optimization methods provided by Galarza Montenegro et al. (2022a), optimal solutions for fourteen benchmark instances were obtained. It was then concluded that the LNS heuristic developed in Galarza Montenegro et al. (2021) yields high quality solutions with low optimization gaps for these benchmark instances. The following are the main differences between the FSMS (Galarza Montenegro et al., 2022b) and the DRFS (Galarza Montenegro et al., 2021): the service now needs to guarantee that a bus departs from each mandatory stop within a certain time interval, the return trip of the buses is now explicitly considered, and passengers can now state a desired departure time as well. These additions significantly complicate service optimization while also improving service quality.

The DFSMS differs from the FSMS (Galarza Montenegro et al., 2022b) by optimizing the service in real-time. Whenever new requests are received, the planning of the service is updated, taking the new information into account. Consequently, the service must be optimised several times throughout the planning horizon. This makes the optimization much more complex, since there are additional restrictions, related to the dynamic nature of the problem, that need to be respected. In the DFSMS, passengers are notified when their request is accepted or rejected. The time before passengers receive a response must not be too long, which imposes an additional constraint on the model. Furthermore, when requests are accepted, passengers are notified about a promised pickup time window and a promised departure stop. This imposes two additional constraints on the model: these passengers must be picked up at certain stop and within a certain time window.

Vansteenwegen et al. (2022) present a recent and extensive survey and categorise Demand-Responsive Transportation Services (DRTS) into different groups. The authors classify DRTS with three different dimensions in mind: level of flexibility, level of responsiveness and a distinction between many-to-many and many-to-one services. The latter refers to the number of origins or destinations passengers can have. Many-to-many services serve passengers that have distinct origins and destinations. Many-to-one services or feeder services serve passengers that have a common departure or destination. An example of a many-to-many service is the service introduced by Melis and Sörensen (2021), i.e., the On-Demand Bus Routing Problem (ODBRP). The ODBRP starts with a given fleet of buses with fixed capacity, a set of bus stops and a set of requests. Each request specifies a time frame for the transportation and a list of nearby bus stops for both departure and arrival. The algorithm determines the routing of the buses as well as the assignment of stops to requests. The authors optimise total user ride time, which is the time passengers spend on the bus, using an LNS algorithm. Furthermore, Melis and Sörensen (2021) show that bus stop assignment increases routing flexibility and efficiency because passenger pickup and drop-off can be grouped at bus stops to avoid extra stops along the route. An example of a many-to-one service is the service studied in Dou and Meng (2019). The authors study a stop-based feeder bus service that transports passengers from a low demand area to a transportation hub, namely to a bus and/or train terminal. The goal of this service is to bring passengers, who have different origin

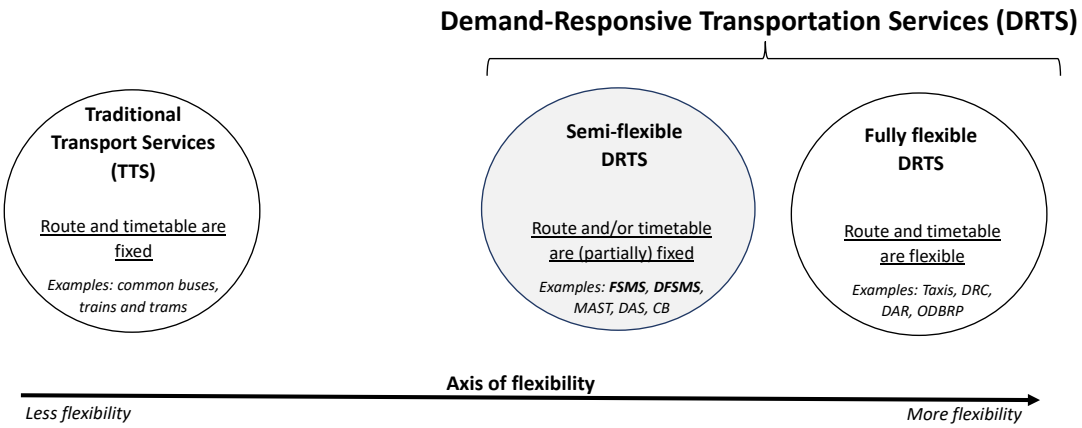


Figure 1.: Flexibility within DRTS

locations, to a single destination where they can continue their journey. The paper presents a heuristic algorithm to optimise the performance of the service. Specifically, it focuses on transfers between the feeder buses and the vehicles at the terminal.

The level of flexibility refers to how much the bus service is allowed to change its planning to satisfy the needs of the passengers. Figure 1 gives an overview of the different levels of flexibility within DRTS. 'Fully flexible' DRTS are the services with the most flexibility. These services typically don't have fixed routes or timetables, and take into consideration the individual demand for transportation. Consequently, these services are better able to deal with sparse and ever-changing demand compared to conventional services (Alonso-González et al., 2018). Some well-known fully flexible DRTS are the Dial-A-Ride (DAR) problem and the Demand-Responsive Connector (DRC). The DAR service improves transit accessibility by providing door-to-door service to passengers who make a request with their desired pickup and drop-off locations (Molenbruch et al., 2017). Sun et al. (2018) present a mixed-integer linear programming model for a DAR-like demand-responsive feeder service. A heuristic algorithm is used to solve the model, and the service is used in a case study in Nanjing City, China. In the DRC, buses transport passengers from their starting point to transfer hubs within a predefined service area (Li, 2009; Ceder, 2013). Both Quadrifoglio and Li (2009) and Li and Quadrifoglio (2010) present analytical and simulation models to assist service providers in deciding between a fixed feeder service and a DRC based on operational circumstances. Passengers can notify the service provider of their arrival using a smart-phone app or an Internet booking service. Waiting customers are scheduled right before the start of each trip. Li et al. (2023) present a model and solution method for the optimisation of a DRC fed by shared bikes. The DRC is fed by bikes in order to decrease the number of excessive detours which lead to diseconomies of scale for the service area. The shared bikes serve as access/egress model for certain request points. The authors propose a heuristic solution algorithm, which combines simulated annealing with a branch-and-bound algorithm. Experiments on numerical cases demonstrate that the introduction of shared bikes can reduce the DRC tour length and operational costs.

'Semi-flexible' DRTS have a partially predetermined planning that can be modified in order to meet the needs of the passengers, and therefore have less flexibility compared to fully flexible DRTS. Some examples of semi-flexible DRTS are the

Mobility Allowance Shuttle Transit (MAST) service (Quadrifoglio et al., 2008), the Customised Bus (CB) (Liu and Ceder, 2015) and the Demand-Adaptive System (DAS) (Crainic et al., 2005). Vehicles in a MAST service have a fixed set of bus stops that they must always visit, i.e., a fixed path, and these stops also have fixed timetables. The vehicles, however, may deviate from the predetermined path. Customers served outside of the fixed path are served at their preferred location, which must be within a certain radius of the fixed path in a so-called 'zone'. This service combines the high flexibility of door-to-door services with a fixed main route (Quadrifoglio et al., 2008). This concept has also been applied to feeder services. To optimise such a problem, as well as a bus assignment sub-problem, Lu et al. (2015) create a three-stage heuristic algorithm. Furthermore, Qiu et al. (2015) investigate a feeder service similar to a MAST service that was implemented in Salt Lake City, USA. Tang et al. (2023) study a similar transportation problem to MAST. However, in this service, Modular Autonomous Vehicles (MAV) are used. These MAVs can decouple from the main fleet to serve door-to-door requests. The authors develop an optimisation model to determine the deviation of the routes, which MVAs should decouple and the corresponding timetable. The goal is to minimise the total passenger and operator costs. A case study is conducted using real data from Dalian (China). The results show that, compared with a conventional service, passenger walking and waiting times are significantly reduced.

The CB is a bus service that connects an origin area to a destination area with a fixed express service and mostly dedicated lanes. However, the routes inside both areas are flexible (Liu and Ceder, 2015). Guo et al. (2019) develop a CB service and optimise the planning with an exact optimization model, similar to a DAR optimization model with time windows. The solution includes intermediate stops where passengers can change lines or services. The model is implemented for a case study in Beijing, China. The results of the branch-and-cut algorithm are compared to those of a Genetic Algorithm (GA) and a Tabu Search (TS) algorithm. Dou et al. (2021) propose a CB problem which considers uncertain travel demands. The authors propose a Mixed Integer Linear Programming (MILP) model to optimize the bus routing, timetabling and bus deployment. To capture the uncertainty of travel demands, a random variable describing the likelihood that the offered bus services are rejected by potential passengers and two associated control parameters are embedded in the MILP model.

In DAS, flexible routes are created for passengers who request transportation, while buses continue to serve mandatory stops on fixed schedules. Initially, a master scheduling defines only a portion of the routes and timetables. Later, each bus' actual schedule is built to include optional stops. Requests may be denied if they render the tour impossible or unprofitable. These bus services were introduced and studied by Crainic et al. (2005, 2012).

At times, a DRTS can be optimised together with a Traditional Transportation Service (TTS), i.e., a service with fixed routes and timetables that do not change based on the current demand for transportation. Zhao et al. (2021) present a model to optimise a DRTS and a TTS simultaneously. The TTS serves all stops on a transit line segment in both directions. However, the bus lines in the TTS cannot be extended, nor can a new line be introduced. The DRTS serves all demand for transportation within a certain demand area. The service is optimised with a two-step heuristic.

Another example can be found in Leich and Bischoff (2019). The authors optimise a service where a DRTS can be combined with a TTS. The results of this optimization are compared with the results with a base case where only the TTS is available. It is concluded that adding the DRTS lowers the waiting time and overall travel time.

The transportation services can be divided based on the degree of responsiveness as well. Services are designated 'Static' when the planning is completed prior to a specific deadline and no changes are possible afterwards. In 'Dynamic' services, the planning can be modified when new incoming requests are received. The service in Lee and Savelsbergh (2017) is an example of static planning. The study considers a DRC to reduce operator costs while taking into account the time window for pickups and drop-offs. The authors take into account the train frequency at the station and use these time windows as operation parameters. The authors optimise the service before the start of operation, using both a heuristic and an exact model, after which the planning is fixed. The service presented in Fu and Liu (2003) is an example of dynamic planning. The authors develop a real-time scheduling model with dynamic stop skipping. The model optimises the vehicle schedule just before they leave the depot. This allows for changes to the planning to be made before the vehicles are dispatched but not after. This responsiveness is referred to as 'Dynamic Offline' by Vansteenwegen et al. (2022). More responsiveness can be found in the service presented by Pratelli et al. (2018). The service starts from a standard route and deviates from it when a request is made, regardless of the position of each vehicle. Each route has a minimum and maximum number of deviating stops. Requests are received in real-time and each bus can change its planning at any time, even after the start of its operation. This type of responsiveness is denoted as 'Dynamic Online'. Wong et al. (2014) study the influence of dynamism on the optimization of DRTS. The authors consider different degrees of dynamism, i.e., the ratio of dynamic requests vs static requests, for a pickup and delivery problem resembling a DAR problem. It is concluded that a system with an intermediate level of dynamism incurs higher operational costs and less requests are accepted when compared to fully static or fully dynamic systems.

The optimization model of the DFSMS is also similar to the Clustered Vehicle Routing Problem with Time Windows (CluVRP), which was first introduced by Sevaux and Sörensen (2008). In the (standard) Vehicle Routing Problem (VRP), a set of customers are served by a set of vehicles that often have a limited capacity and are dispatched from a central depot. The goal is to find the optimal routes to serve all customers with the available vehicles (Clarke and Wright, 1964). The CluVRP adds a constraint to the VRP, which forces the vehicles to visit all customers in a certain cluster before moving on to the next cluster or returning to the depot. Defryn and Sörensen (2017) propose a two-level VNS heuristic for solving CluVRP. It was possible to obtain high-quality solutions for large scale benchmark instances in a limited amount of computational time. Hintsch and Irnich (2018) present a Large Neighbourhood Search (LNS) that uses multiple destruction and repair operators along with a local improvement procedure. This solution managed to improve upon the best-known solutions for seven benchmark instances. Freitas et al. (2023) propose a branch-and-cut algorithm to solve the CluVRP with exact methods. Their algorithm was able to solve instances of larger scale, which were previously not possible to be solved with exact methods.

The DFSMS can be classified as a semi-flexible, dynamic online, many-to-one bus

service. The DFSMS is most similar to the feeder service variant of the MAST service and the DAS. Our feeder service, like MAST and DAS services, has a fixed route from which it can deviate. MAST services provide door-to-door service to some customers within a certain radius, whereas our service assigns and groups passengers at a limited number of bus stops. As previously mentioned, the use of fixed bus stops can benefit transit-oriented urban development (Knowles et al., 2020; Ibraeva et al., 2020). DAS work with bus stops, but there is no bus stop assignment and the passengers' walking times are not taken into account. However, bus stop assignment improves the efficiency of bus assignment and routing (Melis and Sörensen, 2021). The fixed route's timetable is also fixed and cannot be changed in MAST. DAS requires buses to depart from fixed route stops within specific time windows. This limits the amount of time DAS or MAST buses can spend deviating from the main route. Our service is more adaptable, while still ensuring that at least one bus departs from the mandatory bus stops within a certain time frame.

Semi-flexible services can offer opportunities to improve service quality. These services can incorporate positive characteristics of both TTS (predictability, ease of use, cost efficiency) and DRTS (adaptiveness to changing demand). Furthermore, demand-responsive services can greatly benefit from dynamic online optimisation since this makes these services more attractive to be implemented by offering more flexibility in the day-to-day operation of the service. However, Vansteenwegen et al. (2022) show that there is a limited number of articles that study dynamic online many-to-one services. More specifically, only Pratelli et al. (2018) study a dynamic online semi-flexible many-to-one DRTS. There is thus a clear gap in the literature for such DRTS. Moreover, there is a need for semi-flexible many-to-one DRTS that can handle a larger number of on-demand requests while still taking into account the needs of passengers without a reservation. Moreover, most services similar to the FSMS and DSFMS that are studied in the literature are optimized on either small scale instances, optimized with simple insertion heuristics or are based on a problem that is divided into subproblems to reduce the complexity of the optimization model, which can lead to solutions of lesser quality. Very few papers aim to solve these services as an integrated problem and in a real-time manner, as we will do in this paper.

3. Problem Description

In this section, the dynamic Feeder Service with Mandatory Stops (DFSMS) is described in more detail. The notation of the different parameters, sets and variables can be found in Table 1. Note that the number of trips $|J|$ that each bus needs to make is not fixed. The size of J depends on the number of transportation requests that are made within a certain time-frame. If a certain bus needs to make an additional trip because more transportation requests are received, the set is updated. The maximum headway at the mandatory stops can influence the number of trips as well. A small headway means that buses must depart more frequently at the mandatory stops, which can lead to each bus making a larger number of trips.

3.1. *Description of the DFSMS*

Bus lines of the DFSMS are designated shuttle buses that transport residents of a low-demand area to a transportation hub or a nearby city centre, i.e., all passengers

	Sets
B	Set of buses
J	Set of all possible trips of a bus
O	Set of optional bus stops
F	Set of mandatory bus stops
S	Set of all bus stops: $S = F \cup O$
P_1	Set of passengers with a desired arrival time
P_2	Set of passengers with a desired departure time
P_b	Set of passengers that have been already assigned to a bus in a previous optimization iteration
P_a	Set of new incoming passengers that requested the service
P	Set of all passengers using the service
	Parameters
R	Total number of passenger requests throughout the planning horizon
K_k	Number of optional bus stops in cluster k
M	Number of clusters with optional stops
T^{t}_{ij}	Travel time from bus stop $i \in S$ to bus stop $j \in S$
T^{w}_{pi}	Walking time of passenger $p \in P$ to departure bus stop $i \in S$
T^{arr}_{p}	Desired arrival time of passenger $p \in P_1$ at the destination bus stop $m_{\lvert F\rvert-1}$
T^{dep}_{p}	Desired departure time of passenger $p \in P_2$ at their assigned departure stop
T^{sr}	Amount of time needed to travel from m_0 to $m_{\lvert F\rvert-1}$
T^{0}_{p}	Original departure time communicated to passenger $p \in P_b$ at their assigned departure stop
D^{f}	Maximum headway at the mandatory stops
D^{w}	Maximum value for individual walking time
D^{la}	Maximum value for arriving later than the desired arrival time T^{arr}_{p}
D^{ea}	Maximum value for arriving earlier than the desired arrival time T^{arr}_{p}
D^{ld}	Maximum value for departing later than the desired departure time T^{dep}_{p}
D^{ed}	Maximum value for departing earlier than the desired departure time T^{dep}_{p}
D^{tl}	Maximum value for departing later than the original departure time T^{0}_{p}
D^{te}	Maximum value for departing earlier than the original departure time T^{0}_{p}
D^{rt}	Maximum waiting time for a transportation request notification
C	Capacity of the buses
C_b	Available number of seats of bus $b \in B$ for a given solution
W_i	Relative weight given to objective function component i
H^{t}_{p}	Time at which the request of passenger $p \in P_a$ is received
	Decision Variables
x_{btij}	0-1 variables determining if bus $b \in B$, on his t^{th} trip, visits bus stop $j \in S$ immediately after visiting bus stop $i \in S$
y_{pbti}	0-1 assignment variables which assume value 1 if passenger $p \in P$ is assigned to bus $b \in B$, on his t^{th} trip, and to departure bus stop $i \in S$
a_p	Arrival time of passenger $p \in P$ at destination bus stop $m_{\lvert F\rvert-1}$
d_p	Departure time of passenger $p \in P$ at their assigned departure stop
a^{late}_{p}	$a_p - T^{\mathrm{arr}}_{p}$ when passenger $p \in P_1$ is late
a^{early}_{p}	$T^{\mathrm{arr}}_{p} - a_p$ when passenger $p \in P_1$ is early
d^{late}_{p}	$d_p - T^{\mathrm{dep}}_{p}$ when passenger $p \in P_2$ is late
d^{early}_{p}	$T^{\mathrm{dep}}_{p} - d_p$ when passenger $p \in P_2$ is early
d^{s}_{bti}	Departure time of bus $b \in B$, on its t^{th} trip, at stop $i \in S$
t^{bd}_{bt}	Time at which bus $b \in B$, on its t^{th} trip, is available for departure at stop m_0

Table 1.: Notation for the optimization problem after new requests are received

arrive at the same location. We assume that each bus line has its own area to cover, including all possible walking areas, and that no bus stops are shared with other lines. Therefore, all bus lines in the DFSMS are operated and scheduled independently. A bus line is operated by a fleet of $|B|$ buses, i.e., there are $|B|$ buses with different schedules that travel along the same line. The buses serve $|S|$ bus stops, which can be divided into two types of bus stops: $|F|$ mandatory stops and $|O|$ optional stops. Each bus visits all mandatory stops and these stops have a maximum allowable headway of D^{f} seconds, i.e., at least one bus must depart from each mandatory stop within D^{f} seconds after the previous bus. This means that passengers, who did not make a formal request for transportation, waiting at mandatory stops do not need to wait longer than D^{f} seconds for a bus. The optional stops are only visited when demand is assigned to them. Passengers are assigned a departure stop, within walking distance, to which they must walk to. A stop is considered within walking distance for a passenger if the walking time to the bus stop does not exceed a maximum walking time of D^{w} seconds. The mandatory stops can, for example, be placed along a main road. The optional stops are grouped into different clusters. The optional stops in a cluster are typically close together and scattered across a small town or neighbourhood near the main road where the mandatory stops are located.

Note that the maximum headway at the mandatory stops does not necessarily need to be smaller than the travel time between mandatory stops. In the case a bus would arrive too early at a mandatory stop, i.e., too early to meet the headway constraints, the bus is allowed to wait at the mandatory stop to still satisfy the headway constraints. In most cases, if there are enough buses available, this would not be an issue since a bus can be dispatched afterwards to meet the maximum headway constraints. The bus that is dispatched afterwards does not have to be assigned to a passenger request and is dispatched from the depot as late as possible, while still respecting the headway constraints.

The buses always start at the first mandatory stop m_0 and end at the last mandatory stop $m_{|F|-1}$, this is denoted as a 'trip'. A bus route can deviate from the main route and visit some optional stops in a cluster. The bus can travel from one cluster to the next mandatory stop, an optional stop in the same cluster, or an optional stop in a different, nearby cluster. After a bus has reached the last mandatory stop $m_{|F|-1}$, it goes back to the first mandatory stop m_0 following the shortest path without serving any stops. Afterwards, this bus can be reused for the next trip. The DFSMS also considers a limited capacity C for each bus. The available number of seats C_b of bus $b \in B$ at the time of re-optimization can differ for each bus, depending on how many passengers are onboard bus b. Although the service can still overcrowd if there are too many passengers without reservations, the capacity constraints help to control the crowding of passengers with reservations, which can alleviate crowding in general.

A passenger requests a ride online, in which they specify their starting location and desired arrival time T_p^{arr} or departure time T_p^{dep}. Arriving or departing respectively D^{ea} or D^{ed} too early, and arriving or departing respectively D^{la} or D^{ld} too late is not permitted. Furthermore, arriving or departing sooner or later than the desired arrival time is penalised in the objective function. Passengers' arrival and departure times are thus part of both the constraints and the objective function. The timetable of the buses at the mandatory and optional stops is know and updated publicly every

time the planning is re-optimised. Potential passengers who did not make a request but are aware of the timetable information can board a bus at either the mandatory or optional stops. Potential passengers who are unaware of the service request or timetable information can still catch a bus at the mandatory stops with a guaranteed (low) maximum waiting time.

3.2. *Objective function*

The objective is to optimise the service quality by minimizing four factors. First, all passengers' in-vehicle time. Second, the passengers' walking time from their starting point to the designated bus stop. Finally, for each passenger, the time difference between the desired arrival time and the actual arrival time or the time difference between the desired departure time and the actual departure time. These metrics are then weighted and added to form an objective function. The weights of this sum can be determined based on the situation and the service provider's preferences. The objective function is given below.

Min

$$z = W_1 \left[\sum_{p \in P} (a_p - d_p) \right] \tag{1}$$

$$+ W_2 \left[\sum_{b \in B} \sum_{t \in J} \sum_{i \in S} \sum_{p \in P} T^{\mathrm{w}}_{pi} y_{pbti} \right] \tag{2}$$

$$+ \sum_{p \in P_1} \left(W_3 a_p^{\mathrm{late}} + W_4 a_p^{\mathrm{early}} + \right) + \sum_{p \in P_2} \left(W_5 d_p^{\mathrm{late}} + W_5 d_p^{\mathrm{early}} \right) \tag{3}$$

The objective function above is only used for accepted passenger requests. When a passenger is rejected, a penalty must be added to the objective function. This penalty can be composed of the objective function's maximum values for each component. In particular, a penalty z^{p} is given by:

$$z^{\mathrm{p}} = W_1 T^{\mathrm{short}} + W_2 T^{\mathrm{w}} + W_3 \max \left(D^{\mathrm{ed}}, D^{\mathrm{ld}}, D^{\mathrm{ea}}, D^{\mathrm{la}} \right) \tag{4}$$

In this equation, T^{short} is the travel time from m_0 to $m_{|F|-1}$ if two random optional stops per cluster are visited. This is expected to be a relatively long route for an average bus trip and can thus serve as a realistic upper bound for the first component of the objective function.

3.3. *Constraints of the optimization model*

The first constraints deal with the routing of the buses. Constraints (5) ensure that, for the mandatory stops, exactly one arc enters or leaves. Constraints (6) ensure that, for all other bus stops, at most one arc enters or leaves any stop. If an arc enters the stop, there must be an arc leaving the stop and vice versa (7). The only exceptions are m_0, where exactly one arc leaves and none enter, and $m_{|F|-1}$, where exactly one arc enters and none leave. Constraints (8) and constraints (9) ensure that no bus ever has stop m_0 as a successor or stop $m_{|F|-1}$ as a predecessor.

$$\sum_{j \in S} x_{btij} = 1 \qquad \forall\ i \in F,\ b \in B,\ t \in J \quad (5)$$

$$\sum_{j \in S} x_{btij} \leq 1 \qquad \forall\ i \in O\ b \in B,\ t \in J \quad (6)$$

$$\sum_{l \in S} x_{btil} = \sum_{l \in S} x_{btli} \qquad \forall\ i \in S_{0,N-1},\ b \in B,\ t \in J \quad (7)$$

$$\sum_{i \in S} x_{bti0} = 0 \qquad \forall\ b \in B,\ t \in J \quad (8)$$

$$\sum_{i \in S} x_{btN-1i} = 0 \qquad \forall\ b \in B,\ t \in J \quad (9)$$

A second group of constraints deals with capacities or threshold values. Constraints (10) ensure that no passenger needs to walk for a longer time than a predefined maximum value D^{w}, this is important for the passenger-stop assignment. Similarly, constraints (11) ensure that any optional stop that is farther away than a mandatory stop to a passenger, is not considered as a possible departure stop for that passenger. It needs to be noted that both sets of constraints can be dealt with as an input, i.e., if $T^{\mathrm{w}}_{pi} > \min\left(D^{\mathrm{w}}, \min_{k \in F} T^{\mathrm{w}}_{pk}\right)$ then $y_{pbti} = 0$, $\forall\ p \in P,\ i \in S,\ b \in B$ and $t \in J$. Constraints (12) regulate the number of passengers on each bus, so that buses cannot transport more passengers than the current number of available seats C_b on bus b. Constraints (13) to (14) ensure that all passengers arrive and depart within the required time window. It must be noted that parameters $D^{\mathrm{ld}}, D^{\mathrm{ed}}, D^{\mathrm{la}}$ and D^{ea} can be chosen by the service provider. Setting one of these parameters to a value of zero transforms the desired arrival/departure time of the passengers into the earliest or latest arrival/departure time.

$$T^{\mathrm{w}}_{pi} y_{pbti} \leq D^{\mathrm{w}} \qquad \forall\ p \in P,\ i \in S,\ b \in B,\ t \in J \quad (10)$$

$$T^{\mathrm{w}}_{pi} y_{pbti} \leq \min_{k \in F} T^{\mathrm{w}}_{pk} \qquad \forall\ p \in P,\ i \in O,\ b \in B,\ t \in J \quad (11)$$

$$\sum_{p \in P} \sum_{i \in S} y_{pbti} \leq C_b \qquad \forall\ b \in B,\ t \in J \quad (12)$$

$$a^{\mathrm{late}}_p \leq D^{\mathrm{la}},\ a^{\mathrm{early}}_p \leq D^{\mathrm{ea}} \qquad \forall\ p \in P_1 \quad (13)$$

$$d^{\mathrm{late}}_p \leq D^{\mathrm{ld}},\ d^{\mathrm{early}}_p \leq D^{\mathrm{ed}} \qquad \forall\ p \in P_2 \quad (14)$$

Constraints (15) and (16) define the decision variables a^{early}_p, a^{late}_p, d^{early}_p and d^{late}_p as positive deviations between the actual arrival or departure time and the desired arrival or departure time of the passengers. Given the objective function, in each set of constraints, one of the two variables will be zero for each passenger p in the optimal solution.

$$T^{\mathrm{arr}}_p - a_p + a^{\mathrm{late}}_p - a^{\mathrm{early}}_p = 0 \qquad \forall\ p \in P_1 \quad (15)$$

$$T^{\mathrm{dep}}_p - d_p + d^{\mathrm{late}}_p - d^{\mathrm{early}}_p = 0 \qquad \forall\ p \in P_2 \quad (16)$$

Constraints (17) and (18) define the variables d_p and a_p, the departure time and the arrival time of a passenger p, respectively.

$$\text{If } y_{pbti} = 1 \text{ then } d_p = d^{\mathrm{s}}_{bti} \qquad \forall\ i \in S,\ b \in B,\ t \in J,\ p \in P_2 \quad (17)$$

$$\text{If } \sum_{i \in S} y_{pbti} = 1 \text{ then } a_p = d^{\mathrm{s}}_{bt|F|-1} \qquad \forall\ b \in B,\ t \in J,\ p \in P_1 \quad (18)$$

Constraints (19) to (21) define the variables d^{s}_{bti} and t^{d}_{bt}. Furthermore, buses are not allowed to depart from the first mandatory stop, on any trip t, before their available departure time t^{d}_{bt}. These constraints ensure that a bus stop is not served later in time than a following bus stop in the route. This makes subtour elimination constraints unnecessary.

$$d^{\mathrm{s}}_{bt0} \geq t^{\mathrm{d}}_{bt} \qquad \forall\ b \in B,\ t \in J_0 \quad (19)$$

$$t^{\mathrm{d}}_{bt} = d^{\mathrm{s}}_{bt-1N-1} + T^{\mathrm{sr}} \qquad \forall\ b \in B,\ t \in J_0 \quad (20)$$

$$\text{If } x_{btij} = 1 \text{ then } d^{\mathrm{s}}_{btj} = d^{\mathrm{s}}_{bti} + T^{\mathrm{t}}_{ij} \qquad \forall\ i \in S_{|F|-1},\ j \in S_0,\ b \in B,\ t \in J \quad (21)$$

Constraints (22) to (23) ensure the amount of time between two consecutive buses departing from a mandatory stop does not exceed D^{f}. These constraints model the maximum allowable headway at the mandatory stops. We denote them as the headway constraints from now on.

$$\delta_{bti} \leq d^{\mathrm{s}}_{ldi} - d^{\mathrm{s}}_{bti} \qquad \forall\ l \neq b \in B,\ d \neq t \in J,\ b \in B_{|B|},\ t \in J_{|J|},\ i \in F \quad (22)$$

$$\delta_{bti} \leq D^{\mathrm{f}} \qquad \forall\ b \in B_{|B|},\ t \in J_{|J|},\ i \in F \quad (23)$$

Lastly, constraints (24) ensure that every passenger is assigned to at most one bus on a trip and one departure bus stop.

$$\sum_{b \in B} \sum_{t \in J} \sum_{i \in S} y_{pbti} \leq 1 \qquad \forall\ p \in P \quad (24)$$

Constraints (25) ensure that optional stops are visited when there is at least one passenger assigned to the optional stop.

$$\text{If } \sum_{p \in P} y_{pbti} > 0 \text{ then } \sum_{l \in S} x_{bitl} = 1 \qquad \forall\ i \in O,\ b \in B,\ t \in T \quad (25)$$

The remaining constraints determine the domains of the variables.

$$y_{pbti} \in \{0, 1\} \qquad \forall\ b \in B,\ t \in J,\ b \in B,\ p \in P \quad (26)$$

$$x_{btij} \in \{0, 1\} \qquad \forall\ b \in B,\ t \in J,\ i \in S,\ j \in S \quad (27)$$

$$d^{\mathrm{s}}_{bti},\ \delta_{bti} \in \Re^+ \qquad \forall\ b \in B,\ t \in J,\ i \in S \quad (28)$$

$$t^{\mathrm{d}}_{bt} \in \Re^+ \qquad \forall\ b \in B,,\ t \in J \quad (29)$$

$$a_p,\ a^{\mathrm{late}}_p,\ a^{\mathrm{early}}_p \in \Re^+ \qquad \forall\ p \in P_1 \quad (30)$$

$$d_p,\ d^{\mathrm{late}}_p,\ d^{\mathrm{early}}_p \in \Re^+ \qquad \forall\ p \in P_2 \quad (31)$$

3.4. *Explanatory example*

Figure 2 shows a small example of the DFSMS. In this example, a single bus is used for two trips, bus trips A and B. During the planning horizon, the service receives thirteen passenger requests. Each passenger must arrive at their destination or depart from a bus stop within a specific time window. Bus trip A accepts five passenger requests, while bus trip B accepts six. Two of the thirteen requests cannot be served and are rejected. Accepted passengers are green coloured triangles, while rejected passengers are white triangles. The passengers are identified with numbers in the figure as well.

Six mandatory stops are labelled m_0 through m_5. The first mandatory stop is at the start of each bus trip's route, and the last mandatory stop is at the destination. There is a cluster of six optional stops between two mandatory stops. The clusters are named c_1 through c_5. A dashed line represents the main route, which is the shortest route from the start to the destination that only visits the mandatory stops. To pick up passengers, the bus deviates from the main route. If a bus stops at an optional stop to pick up a passenger, the optional stop is coloured black.

Passengers make a request online through a mobile app, at any time before or during the planning horizon. Passenger requests can be accepted or rejected, an answer is provided within minutes. Accepted passengers are assigned to both a bus trip and a bus stop. This assignment is represented by a dotted line connecting a passenger and a bus stop visited by a bus during a trip. Two passengers are assigned to the same optional stop on bus trip B in cluster c_4, despite the fact that that bus stop is not the closest stop to them both. This is done to reduce the in-vehicle time of the passengers onboard the bus on trip B, thereby further reducing the objective function and/or making the solution feasible. After all, if a passenger does not arrive at their destination within the specified time window, a solution may become infeasible. This can happen, for example, if their bus makes too many stops and takes too long to arrive at their destination.

Table 2 gives an overview of how the pick up times of the passengers in the explanatory example are determined. The time at which each passenger made a request is shown in the second column and the rows are sorted in ascending order of this value. When a passenger request is accepted, the service providers provide a tentative pick up time to the passenger. Often, this pick up time would change over time, i.e., if the actual pick up time (in column five) is different from the tentative time (column four), when new requests are received. However, the actual pick up time is always within the promised time window, which is given in the third column of Table 2. Passengers two and 13 are rejected. For example, this can be the due to the fact that their desired arrival or departure times were too far ahead in the future to still accommodate both the already accepted passenger requests and the new incoming request.

When the bus on trip A arrives at its destination, it is returned to the starting point to be reused for the next trip, i.e., bus trip B. Figure 2 depicts the bus schedule at the mandatory stops as well. This schedule is not fixed and is determined by the demand for transportation. The bus on trip A, for example, takes longer to travel from m_1 to m_2 than Bus trip B because it must pick up more passengers in cluster c_2. When the departure times of both bus trips are compared, it is clear that a bus departs

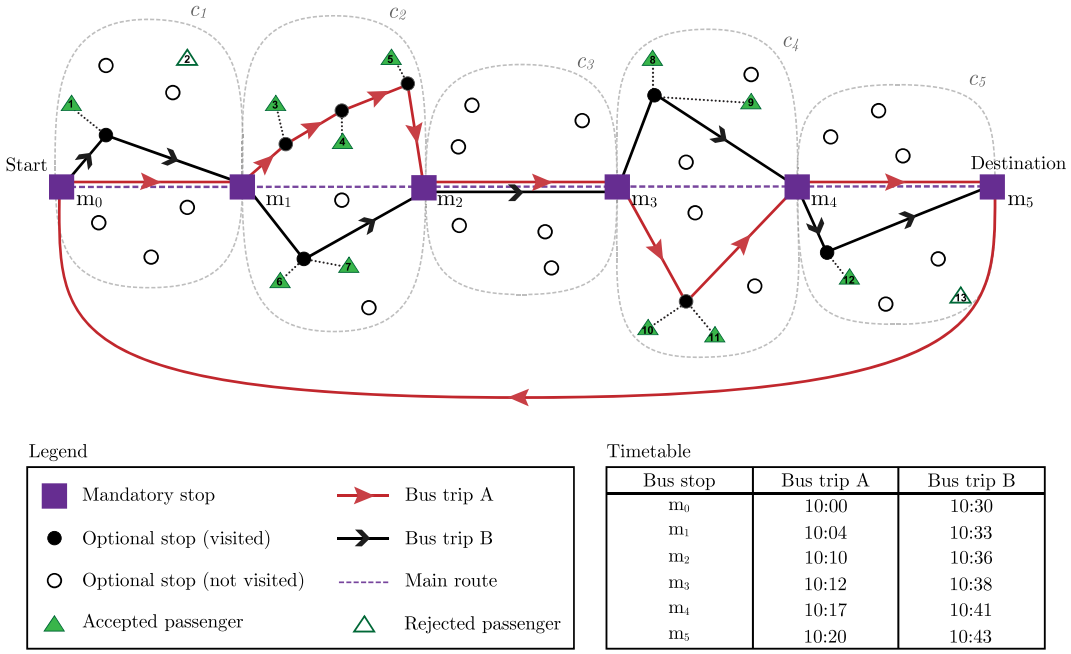
Passenger	Time at which a request is made	Promised pick up time window	Tentative pick up time	Actual pick up time	Bus trip
3	9:40	[10:00, 10:10]	10:05	10:05	A
5	9:50	[10:04, 10:14]	10:09	10:09	A
4	9:58	[10:02, 10:12]	10:07	10:07	A
10	10:01	[10:10, 10:20]	10:15	10:14	A
11	10:07	[10:05, 10:15]	10:10	10:14	A
2	10:10		*Rejected*		
7	10:10	[10:30, 10:40]	10:35	10:34	B
9	10:20	[10:38, 10:48]	10:43	10:39	B
1	10:23	[10:26, 10:36]	10:31	10:31	B
6	10:28	[10:29, 10:39]	10:34	10:34	B
8	10:32	[10:30, 10:40]	10:35	10:39	B
12	10:35	[10:38, 10:48]	10:43	10:42	B
13	10:45		*Rejected*		

Table 2.: Determination of passenger pick up times for the explanatory example of the dynamic Feeder Service with Mandatory Stops

from each mandatory stop in less than 30 minutes. In this example $D^{\mathrm{f}} = 30$ minutes, making this solution feasible. Furthermore, once the bus arrives at the last mandatory stop on trip A, it takes 10 minutes to reach the first mandatory stop to begin trip B. To obtain a feasible solution, the time it takes for the bus to return to be reused for the next trip must be considered.

3.5. *Aspects of the dynamic online optimization*

The main differences between the FSMS and the DFSMS is that the latter is optimised in real-time, which brings some additional restriction for the optimization model. Passengers receive a response whether or not their request has been scheduled. To ensure passengers are notified about the response to their requests, there is a restriction on the maximum waiting time D^{rt} for a request notification. If it is not possible to serve a passenger within the given restrictions, their request is rejected. Passengers that receive a confirmation for their request are given a departure stop and a tentative pickup time T^0. Furthermore, passengers are guaranteed to be picked up at their assigned departure stop within a promised pickup time window $\Delta^{\mathrm{d}} = \left[T^0 - D^{\mathrm{te}}, T^0 + D^{\mathrm{tl}}\right]$, with D^{tl} and D^{te} parameters of the service that determine the length of the pickup time window. The earliest pickup time is communicated to the passengers, this allows them to begin walking towards their departure stop on time. Therefore, we assume the waiting time for a bus for passengers who made a request is zero. Waiting times can still occur when there are delays due to externalities such as congestion. However, we work with deterministic travel times in our model, which cannot change over time. This, in turn, means that the model cannot account for real-time delays. Requests are received by the service throughout the planning horizon, which is typically a few hours. Whenever a new request is received, the route of each bus is (re-)optimised. New passengers are notified about their departure stop and tentative time of departure, while the other passengers are notified about their updated departure time, which is within their promised time window Δ^{d}. Moreover, we will assume that passengers do not need to wait more than a certain amount



Timetable

Bus stop	Bus trip A	Bus trip B
m_0	10:00	10:30
m_1	10:04	10:33
m_2	10:10	10:36
m_3	10:12	10:38
m_4	10:17	10:41
m_5	10:20	10:43

Figure 2.: Example of the DFSMS

of time D^{rt}, after sending a request, for a notification about their journey or a rejection.

To simplify the problem to some extent, the following assumptions are made. The first assumption concerns the re-optimization, leading to some additional constraints for each bus. There are three types of situations in which the planning is fixed and cannot be changed. First, bus stops and timetables that were already scheduled to be visited in the past, with respect to the time of re-optimization, cannot be changed. Second, if the bus is moving between bus stops during the re-optimization process, the next bus stop that was supposed to be visited must be visited in accordance with the previous timetable. Third, if a departure bus stop has already been assigned to a passenger, it must be visited as well. Furthermore, passengers that have already made a requests need to be picked-up within the promised time window Δ_p^{d}.

We also assume that $H_p^{\mathrm{t}} < T_p^{\mathrm{dep}} - D^{\mathrm{ed}}$ or $H_p^{\mathrm{t}} < T_p^{\mathrm{arr}} - T_{s(|F|-1)}^{\mathrm{t}} - D^{\mathrm{ea}}$ with s the closest bus stop to p and H_p^{t} the time at which p made a request. This is required to ensure that all new passenger-bus assignments are at least theoretically feasible.

Furthermore, it needs to be noted that the routes of the buses are partially fixed. The locations of the mandatory stops are sorted by the distance to the destination $m_{|F|-1}$ and are visited in that order. We assume that the buses are not allowed to revisit a mandatory stop or optional stops in a cluster that has been visited already. For example, if a passenger makes a request at 8:30 to be picked up at mandatory stop m_1 and bus b visited m_1 at 8:25 already, then bus b is not allowed to visit m_1 again

to pick up this passenger. The passenger is then either picked up by a different bus or the request is rejected if there are no other buses available at that time.

4. Solution approach

In this section, the heuristic that is developed to optimise the DFSMS is explained in more detail. First, the steps need to be taken before the start of the algorithm are explained. Afterwards the insertion phase is explained, followed by the improvement phase.

4.1. *Preprocessing*

Before the start of the planning horizon, an algorithm to determine route BR is executed once. The inputs for this algorithm are the travel times of a bus between all bus stops. The output is route BR, which is a route that is constructed with the aim of minimizing the total travel time of a bus that visits all bus stops. This route is constructed in order to insert additional optional stops in the best possible position in existing routes later on in algorithm `Update_Route`. The exact manner of how BR is utilised in `Update_Route` is explained in Section 4.3.

The algorithm to determine BR is as follows. First, an initial feasible route is constructed and afterwards it is improved with a 2-opt procedure. The initial route is constructed by adding all the mandatory stops first and then adding optional stops of a certain cluster between the two corresponding mandatory stops. The 2-opt algorithm implemented here is a first-improvement algorithm, and selects two edges of the existing route and swaps them if and only if the objective function value is lowered by this swap. The algorithm randomly chooses the first edge in the existing route. The second edge is chosen from either two optional stops in a nearby cluster or from edges connecting mandatory stops nearby. This is done because most or all inefficiencies are localised in or around the clusters of optional stops, due to the structure of the bus stops.

4.2. *Prior to the start of operation*

In this section, the initial planning of the service is determined. This is the planning before the service starts to operate and it concerns the routing and timetabling of all buses and bus trips. The heuristic starts by optimizing the planning of the service with the requests that are received prior to the start of operation. This is done with the algorithm presented in Section 4.4, here denoted as `Improvement(`$p \in P_b^0$`)`, with P_b^0 the set of passengers that requested the service before the start of the planning horizon, i.e., the passenger requests that were known in advance. In Section 4.4, this algorithm is subject to additional restrictions related to the dynamic nature of the DFSMS, such as the promised pickup time windows and previously assigned departure bus stops. However, for the construction of the initial static solution this is not the case.

The outputs of `Improvement(`$p \in P_b^0$`)`, as well as the output of any re-optimization, are the current free number of seats C_b of each bus $b \in B$, the promised pickup time windows $\Delta_p^{\mathrm{d}} = \left[T_p^0 - D^{\mathrm{te}}, T_p^0 + D^{\mathrm{tl}}\right]$ for each passenger $p \in P_b$, as well as the route

and timetable of each bus trip. If no requests are received prior to the start of the service, all buses are dispatched to meet the maximum headway constraints at the mandatory stops, i.e., buses are dispatched every D^{f} seconds.

4.3. *Insertion phase*

Whenever a new request is received, the heuristic must re-optimise the service with the `Insertion` algorithm. The algorithm begins by iteratively processing each passenger request received at the time. Requests received first will be processed first, i.e., on a first-come-first-serve basis.

For each new passenger request p, the algorithm will go over each possible bus b on trip t that can be assigned to passenger p. To further improve the runtime of the `Insertion` algorithm, certain buses on certain trips are not considered. Only buses b on their t^{th} trip that satisfy the following conditions are considered:

$$\left| d^{\mathrm{0s}}_{bt(|F|-1)} - T^{\mathrm{arr}}_{p} \right| \leq \delta^1 \qquad \forall\ p \in P_a \cap P_1 \tag{32}$$

$$\left| \hat{d^{\mathrm{0s}}_{btp}} - T^{\mathrm{dep}}_{p} \right| \leq \delta^2 \qquad \forall\ p \in P_a \cap P_2 \tag{33}$$

$$T^{\mathrm{w}}_{ps} + H^{\mathrm{t}}_{p} - \hat{d^{\mathrm{0s}}_{btp}} \leq \delta^3 \qquad \forall\ p \in P_a \tag{34}$$

With d^{0s}_{bti} the departure time of bus b on trip t departing from bus stop i, which was determined in the previous optimization. $\hat{d^{\mathrm{0s}}_{btp}}$ is an estimate departure time of bus b on trip t from the closest bus stop s from passenger p. If s is already part of the route, then $\hat{d^{\mathrm{0s}}_{btp}} = d^{\mathrm{0s}}_{bts}$. If s is not part of the route already, then $\hat{d^{\mathrm{0s}}_{btp}}$ is determined by adding s to the current route and calculating the departure time at s. δ^1, δ^2 and δ^3 are positive threshold values and are considered to be parameters of the `Insertion` algorithm. These parameters determine the number of bus trips that are considered and thus influence the runtime as well as the objective value. The set of bus trips t that satisfy these conditions is denoted as T_f. Inequality 32 and 33 make sure that the departure time of bus b at departure stop s is not too distant in time from, respectively, the desired arrival time or the desired departure time of the incoming passenger request p. This ensures that buses that are scheduled too early or too late, and therefore cannot be feasible candidates, are not considered. Inequality 34 makes sure that the departure time of bus b at stop s is later (or not too much earlier) than the time at which the request is received plus the time it takes to walk to bus stop s. This ensures that passengers have enough time to walk to their assigned departure stop to catch the bus.

Hereafter, we first determine whether or not there are currently enough available seats in the bus, i.e., if $C_b < C$. If and only if this is the case, the algorithm must determine whether it is feasible to assign passenger p to bus b on trip t and departure stop s. We denote the set of feasible bus stops, i.e., the bus stops that are within walking distance, as S_c. First, the route of trip t is temporarily updated with `Update_Route(`t, s, p`)` in case stop s is not part of the route already. This is done with the help of the route BR, the highly efficient route that visits all bus stops, determined during preprocessing. The departure bus stop s is inserted in the current route depending on where s is placed in BR. For example, if BR is $[m_0, o_0, o_1, m_1, o_2, o_3, m_2]$, the current route is $[m_0, m_1, m_2]$ and we wish to insert bus stop o_2 in the current route, then it should be

inserted between m_1 and m_2 because that is the most similar placement of o_2 in BR. By updating the route in this manner, we can assure that the route remains efficient, without the need to perform heavy time-consuming improvements on it. Since part of the route is fixed, this method of updating the route yields good results.

Afterwards, we temporarily update the timetable with `Update_Timetable`(t, s, p) as well, in order to make the assignment feasible. When a new passenger p is assigned to bus trip t, the timetable of trip t needs to be modified in order to make the assignment feasible. On the one hand, if $p \in P_1$, the bus needs to arrive at the destination within time window $\left[T_p^{\text{arr}} - D^{\text{ea}}, T_p^{\text{arr}} + D^{\text{la}}\right]$. On the other hand, if $p \in P_2$, the bus needs to depart from s within time window $\left[T_p^{\text{dep}} - D^{\text{ed}}, T_p^{\text{dep}} + D^{\text{ld}}\right]$. The minimal shift that is needed to accommodate the new passenger p is denoted as T_{ps}^{s}. The maximum shift that the timetable is allowed to shift, while still making the assignment of p possible is denoted as T_{ps}^{m}. Both T_{ps}^{s} and T_{ps}^{m} can be negative or positive, however, $\left|T_{ps}^{\text{s}}\right| \leq \left|T_{ps}^{\text{m}}\right|$ always holds true. It needs to be noted that it is possible that the existing timetable is already feasible for the insertion of passenger p, in that case $T_{ps}^{\text{s}} = 0$.

The next step is to calculate the maximum allowable time that a bus trip's timetable can shift forwards or backwards before constraints are violated. The desired arrival or departure times of the passengers onboard, combined with the headway constraints, create feasible time windows for the bus to depart at each bus stop. These feasible time windows are determined by the desired arrival times T_p^{arr} or the desired departure times T_p^{dept}, as well as parameters D^{tl} and D^{te} of passengers p onboard bus b on trip t who have not been picked up yet. Furthermore, the headway constraints of the mandatory stops that have not been visited yet must also be considered. For passengers onboard bus b on trip t that were already given a tentative pickup time window, the Δ_p^{d} constraints also need to be respected. Finally, the availability of the buses must be considered as well. If the bus has not yet left m_0, then t_{bt}^{bd} can be used to determine whether or not the bus can shift its timetable backwards. In the case the bus has already left m_0, the part of the route and timetable that is in the past with respect to H_p^{t} cannot be changed. The remainder of the timetable can then only be shifted forwards. We can then generate two lists L^{l} and L^{u}, containing the differences in time between the current departure time $d_{bti}^{0\text{s}}$ of bus b on trip t and respectively the lower and upper bounds of the feasible time window of each visited stop s. The parameters D^{sp} and D^{sf} are the minimum values of respectively L^{l} and L^{u}.

The passenger assignment is only feasible if $T_{ps}^{\text{s}} \leq D^{\text{sp}}$ in case T_{ps}^{s} is negative and $T_p^{\text{s}} \leq D^{\text{sf}}$ in case T_{ps}^{s} is positive. If a bus is driving at the time of re-optimization, the shift in timetable can only be in the future and this can involve idle waiting time in order to satisfy the desired departure time constraints. For the `Insertion` algorithm, we opt to modify the timetable with the smallest possible shift, i.e., with T_{ps}^{s}. Later on, the planning of the operation will be improved further with the function `Improvement`, where we will not always modify the timetable with the smallest shift in time, as explained in the next section. To further decrease the runtime of the algorithm, we will only consider bus stops that are within walking distance from passenger p, i.e., $\forall s \in S | T_{ps}^{\text{w}} \leq D^{\text{w}}$.

The additional cost of each passenger-bus stop assignment is calculated for feasible

bus trips. The additional cost ΔC_{ps} of assigning passenger $p \in P_a$ to bus stop s is calculated as follows:

$$\Delta C_{ps} = |P_t|\,\Delta T + T^{\mathrm{w}}_{ps} + \Delta W \tag{35}$$

P_t denotes the set of passengers onboard bus b during trip t at the time of the re-optimization. ΔT is the additional in-vehicle time incurred by visiting bus stop s. The difference between the third component of the objective function in the previous optimization and the current optimization is denoted by ΔW and is calculated as follows:

$$\Delta W = \sum_{p \in P_t \cap P_1} \left(\left(a_p^{\mathrm{late}} - a_p^{0\mathrm{late}} \right) + \left(a_p^{\mathrm{late}} - a_p^{0\mathrm{early}} \right) \right) + \sum_{p \in P_t \cap P_2} \left(\left(d_p^{\mathrm{early}} - d_p^{0\mathrm{late}} \right) + \left(d_p^{\mathrm{early}} - d_p^{0\mathrm{early}} \right) \right) \tag{36}$$

It should be noted that ΔT and T^{w}_{ps} are always positive, whereas ΔW can become negative if enough of its components are negative. However, a negative ΔW is unlikely to be a common occurrence.

After the additional cost ΔC_{ps} of all the possible passenger-bus and passenger-bus-stop assignments are calculated, the assignment that inquires the lowest additional cost $\Delta C_p^{\min}$ is chosen. Passenger requests without a feasible assignment are rejected. The pseudo-code of the insertion heuristic is shown in Algorithm 1.

4.4. *Improvement phase*

To further optimise the operation of the DFSMS, part of the service is improved with the use of algorithm `Improvement`. First, we make a distinction between passengers $p \in P_s$ that are assigned to a bus that has already left m_0 and passengers $p \in P_d$ that have been assigned to a bus that has not left m_0 yet. Hereafter, we optimise the service taking only passengers $p \in P_d$ into consideration and assume that the service planning of passengers $p \in P_s$ cannot be changed anymore. In essence, algorithm `Improvement` can be viewed as an algorithm that improves the solution of a static FSMS, with additional constraints related to the dynamic nature of the DFSMS. The reason to make a distinction between P_s and P_d is because algorithm `Improvement` is expected to run for a relatively longer time compared to the `Insertion` algorithm . This runtime time might be too large to notify a passenger $p \in P_s$ on time, who is assigned to a bus that is already in operation, about the planning. The inputs for this algorithm are the timetables and routes that are in the past and cannot be changed anymore, as well as the current trip t each bus $b \in B$ is at and the updated available times t^{bd}_{bt} for departure from the depot m_0. It has to be noted that from now on, the available time for departure of each bus is referred to as t^{bd}_b and is updated as new trips are generated.

4.4.1. Outline of the algorithm

`Improvement(`$p \in P_d$`)` is an iterative greedy constructive heuristic. The outline of the `Improvement` algorithm is shown in Algorithm 2. The `Improvement` algorithm constructs a complete solution in each of its iterations i, which we denote as construction iterations. A complete solution consists of a set of buses, that make a number of trips. A bus trip corresponds to the journey of a bus from the starting point to the destination. The construction of a complete solution in each iteration i is denoted as `Construct_Solution` in Algorithm 2. `Construct_Solution` contains,

Algorithm 1: Main outline of the Insertion algorithm for the re-optimization

```
Sort passengers p ∈ P_a according to H_p^t
for p ∈ P_a do
    ΔC_p^min = ∞
    s_m = −1
    t_m = −1
    Determine the set of favorable buses T_f
    for t ∈ T_f do
        if C_b < C then
            Determine D^sp and D^sf
            S_c = {∀s ∈ S | T_ps^w ≤ D^w}
            for s ∈ S_c do
                Temporarily modify t with Update_Route(t, s, p) and
                 Update_Timetable(t, s, p)
                Determine T_ps^s
                if T_ps^s ≤ D^sp ≤ 0 or 0 ≤ T_ps^m ≤ D^sf then
                    Calculate ΔC_ps
                    if ΔC_p^min > ΔC_ps then
                        ΔC_p^min = ΔC_ps
                        s_m = s
                        t_m = t
                    end
                end
            end
        end
    end
    if s_m ≠ −1 then
        Assign passenger p to bus trip t_m and bus stop s_m with additional cost
         ΔC_p^min
        Update_Route(t_m, s_m, p)
        Update_Timetable(t_m, s_m, p)
    end
end
if P_d ≠ ∅ then
    Improvement(p ∈ P_d)
end
if No feasible assignment possible for p then
    Reject passenger p
else
    Add p to P_b
end
```

among others, the generation of many bus trips. The generation of a single bus trip consists of assigning passengers to a bus b on a certain trip t and constructing its route and timetable. The generation of a bus trip is denoted as a trip generation. When no more bus trips can be scheduled within a certain planning horizon, the solution is complete and `Construct_Solution` ends.

The construction algorithm `Construct_Solution`, in any iteration i, is greedy because it aims to construct the best routes, timetables and assignments for one bus trip at a time regardless of the next bus trips. This means that there are instances where the resulting solution is infeasible because not all passengers are assigned to a bus. Due to the tight constraints of the optimization model, infeasible solutions can often not be restored to feasibility without constructing the solution from scratch. For this reason, these infeasible solutions are discarded. The construction algorithm `Construct_Solution` is explained in more detail in Section 4.4.2.

To bring more variance into the construction of solutions, construction parameters are introduced. These parameters guide `Construct_Solution` and bring a balance between greediness and feasibility of the construction algorithm by determining whether or not a certain passenger is assigned to a bus trip. This allows `Construct_Solution` to find more solutions with a better objective value or to find more feasible solutions for instances with strict constraints. The mechanisms of these parameters are discussed in Section 4.4.3.

The values of the construction parameters are randomly sampled and a new complete solution is generated and evaluated for a certain number of times N^{stop} or after a certain amount of computation time T^{stop} has passed.

Algorithm 2: Main outline of the Improvement algorithm of the re-optimization of the dynamic Feeder Service with Mandatory Stops

```
1  Sample feasible incumbent construction parameters r^b1, ..., r^bm
2  Incumbent solution = Construct_Solution(r^b1, ..., r^bm)
3  i = 1
4  τ = 0
5  while i ≤ N^stop and τ ≤ T^stop do
6  |  Randomly sample candidate construction parameters r^c1, ..., r^cm ; // See 4.4.3
7  |  Candidate solution = Construct_Solution(r^c1, ..., r^cm) ;  // See 4.4.2
8  |  if new solution is feasible then
9  |  |  ΔE = objective function value candidate solution - objective function
   |  |    value incumbent solution
10 |  |  if ΔE < 0 then
11 |  |  |  Incumbent solution ← Candidate solution
12 |  |  |  r^b1, ..., r^bm ← r^c1, ..., r^cm
13 |  |  i++
14 |  |  τ = current time
15 |  end
16 end
```

4.4.2. Construction algorithm of one construction iteration

The pseudo-code of the `Construct_Solution` algorithm is shown in Algorithm 3. In the first step of the construction algorithm, all passengers $p \in P_d$ are placed in a queue Q_p. The passengers are then sorted according to their T_p^{dep} or estimated departure time $\hat{T_p^{\text{dep}}}$. The latter is determined as following:

$$\hat{T_p^{\text{dep}}} = T_p^{\text{arr}} - T_p^{\text{tr}} \tag{37}$$

Here, T_p^{tr} is the shortest travel time to the destination. The passengers are sorted in order to consider the passenger assignments that are most likely to be feasible first.

The `Construct_Solution` algorithm then enters a loop in which the planning for each single bus trip is determined at a time. Each iteration of this loop is denoted as a trip generation. The loop ends when all passengers $p \in P_d$ are assigned to a bus trip. In each trip generation, the first step is to select the earliest available bus b on trip t, i.e., the bus trip with the lowest t_b^{bd}. Hereafter, we construct a timetable and route visiting only the mandatory stops. The bus leaves the depot m_0 after its available time t_b^{bd}, while still respecting the maximum headway constraints. The exact time the bus leaves, is determined by construction parameter r^{hw}. The exact mechanism of the construction parameters is explained in the next section.

Afterwards, we enter another loop in which we attempt to assign as many passengers $p \in Q_p$, that are not assigned to any bus trip yet, as possible to trip t. It needs to be noted that the number of available seats C_b must not exceed the maximum capacity C. We denote each iteration of this loop as an assignment iteration. In each assignment iteration, a passenger $p \in P_d$ is considered to be assigned to trip t. In case $p \in P_a$, i.e., p has not yet been notified of his or her current planning, all the bus stops within walking time are considered as potential departure stops. Otherwise, if $p \in P_b$, only the promised departure stop s_p of passenger p is considered and the promised pickup time windows Δ_p^{d} need to be respected. The timetable and route of trip t are temporarily updated with `Update_Route` and `Update_Timetable`. Similarly as in the `Insertion` algorithm, the additional cost of assigning a passenger to a bus trip with a certain departure stop is calculated. If the assignment is feasible, the assignment with the lowest additional cost is chosen. The `Update_Timetable` algorithm computes the minimum time-shift T_{ps}^{s} that is needed to make an assignment feasible, as well as the maximum time-shift T_{fs}^{m} that still makes the assignment possible. In case that the assignment is possible without the need for a shift, T_{ps}^{s} and T_{ps}^{m} are respectively the maximum time that the timetable can shift backwards or forwards without becoming infeasible. In the `Insertion` algorithm, the smallest possible shift, i.e., T_{ps}^{s}, is used. However, unlike in the `Insertion` algorithm, the function `Update_Timetable` in the `Improvement` algorithms will use a time-shift shift that lies between T_{ps}^{s} and T_{ps}^{m}. The exact time-shift is determined by construction parameter r^{tt}. Whether or not a feasible assignment is chosen, depends on construction parameter r^{acc}. The mechanisms of the construction parameters are explained in Section 4.4.3.

4.4.3. Construction parameters and randomness

Since the `Construct_Solution` algorithm is greedy, infeasible solutions are possible. Therefore there is a need to balance the search for a feasible solution and the

Algorithm 3: Construction algorithm of one solution

Calculate $C^{\min}$; `// A construction iteration`
Add all passengers $p \in P_d$ to Q_p
Sort passengers $p \in Q_p$ according to $\hat{T_p^{\text{dep}}}$
while $Q_p \neq \emptyset$ **do**
  Determine ealiest available bus (lowest t_b^{bd}) ; `// A trip generation`
  Initialise route visiting all mandatory stops
  Initialise timetable with departure time t_b^0 from m_0
  Add route and timetable to trip t
  for $p \in Q_p$ **do**
    $\Delta C_p^{\min} = \infty$; `// An assignment iteration`
    $s_m = -1$
    $t_m = -1$
    Determine D^{sp} and D^{sf}
    if $p \in P_a$ **then**
      $S_c = \left\{\forall s \in S | T_{ps}^{\text{w}} \leq D^{\text{w}}\right\}$
    else
      $S_c = \{s_p\}$
    end
    for $s \in S_c$ **do**
      Temporarily modify trip t with `Update_Route`(t, s, p, r^{tt}) and `Update_Timetable`(t, s, p, r^{tt})
      Determine T_{ps}^{s}
      if $T_{ps}^{\text{s}} \leq D^{\text{sp}} \leq 0$ **or** $0 \leq T_{ps}^{\text{s}} \leq D^{\text{sf}}$ **then**
        Calculate ΔC_{ps}
        if $\Delta C_p^{\min} > \Delta C_{ps}$ **then**
          $\Delta C_p^{\min} = \Delta C_{ps}$
          $s_m = s$
          $t_m = t$
        end
      end
    end
    if $s_m \neq -1$ **and** $\frac{\Delta C_{ps} - C^{\min}}{\Delta C_{ps}} \leq r^{\text{acc}}$ **then**
      Assign passenger p to bus trip t_m and bus stop s_m
      `Update_Route`$(t_m, s_m, p, r^{\text{tt}})$
      `Update_Timetable`$(t_m, s_m, p, r^{\text{tt}})$
      remove p from Q_p
    end
  end
  Update t_b^{bd}
  Add trip t to the solution
end

greediness of the construction. With this goal in mind, three types of construction parameters are introduced. The value of these parameters is randomly sampled, which creates variability in the construction process and, in turn, allows the algorithm to find more feasible solutions as well as solutions with a lower objective function value.

The first construction parameter is denoted as r^{hw}. This parameter determines at what time a bus leaves the depot m_0 when the timetable is first initialised in a trip generation. This is especially important if there is a bus trip, that is scheduled within the planning horizon, which does not pick up passengers that made a formal request. Buses cannot leave before their available time of departure t_b^{bd} but also need to leave early enough to still satisfy the maximum headway constraints at the mandatory stops. If we denote the latest feasible departure time of bus b as t_b^{md} and the earliest departure time as t_b^{bd}, then the actual departure time t_b^0 is determined as follows:

$$t_b^0 = \left(r^{\text{hw}} - 1\right) t_b^{\text{bd}} + r^{\text{hw}} t_b^{\text{md}} \tag{38}$$

Parameter r^{hw} can take on values between zero and one. This means that the departure time t_b^0 will always depart within the feasible time window $\left[t_b^{\text{bd}}, t_b^{\text{md}}\right]$.

The next parameter is denoted as r^{tt}. This parameter is used in each assignment iteration when the timetable of a trip t is modified to accommodate the assignment of passenger p with function `Update_Timetable`. In this function, the timetable often needs to be shifted a certain amount of time, either forwards in time or backwards, in order to make the passenger assignment possible. The minimum and maximum amount of time that the timetable can be shifted, backward or forward, to accommodate passenger p with departure stop s are T_{ps}^{s} and T_{ps}^{m} respectively. The actual shift $T_{ps}^{0\text{n}}$ is then given by:

$$T_{ps}^{0\text{n}} = \left(r^{\text{tt}} - 1\right) T_{ps}^{\text{s}} + r^{\text{tt}} T_{ps}^{\text{m}} \tag{39}$$

Parameter r^{tt} also takes on values between zero and one, which means that the actual shift is $T_{ps}^{0\text{n}}$ is always within time window $\left[T_{ps}^{\text{s}}, T_{ps}^{\text{m}}\right]$.

The last parameter is r^{acc}, the acceptance probability. After a feasible assignment is determined, there is also a choice whether or not to accept this assignment. It is possible that accepting a feasible assignment can lead to a worse overall solution or even an infeasible solution. An assignment is accepted if the following is true:

$$\frac{\Delta C_{ps} - C^{\min}}{\Delta C_{ps}} \leq r^{\text{acc}} \tag{40}$$

With ΔC_{ps} the additional cost of assigning passenger p with departure stop s to trip t and $C^{\min}$ the theoretical minimum additional cost of assigning any passenger to any bus trip and departure stop. The latter is estimated as one fourth of the maximum walking time D^{w} plus half of the minimum travel time from depot m_0 to the destination $m_{|F|-1}$. The parameter r^{acc} takes on values between zero and one as well. This means that a very good assignment with a lower additional cost than $C^{\min}$ will always be accepted and assignments with double or more the additional cost compared to $C^{\min}$

are never accepted. Furthermore, the lower the additional cost ΔC_{ps} is, the more likely it is to be accepted.

5. Experimental set-up

The instances that are used to evaluate the heuristic's performance are discussed in this section. The instances are used for simulations of a real-time operation of the feeder service during a planning horizon of approximately two hours. A computer with a Windows 10 Enterprise operating system, an Intel Core$^{\text{TM}}$ i7-8850H, 2.60Ghz CPU and 16 GB of RAM is used for these experiments. All instances that are used for the experiments are listed in Table 3. Instances I_1 to I_{34} are randomly generated instances. The mandatory stop positions are chosen at random and are equidistant. The optional stops are scattered around a location between the mandatory stops. Passengers' positions are chosen at random within a certain radius of the bus stops. Within a two-hour planning horizon, desired arrival or departure times are randomly sampled in alternating time intervals. The time at which the passengers make a request is randomly sampled to be 10 to 30 minutes before they wish to depart. The instances have different attributes, such as the number of buses $|B|$, requests $|P|$ and bus stops $|S|$. There is one cluster between two mandatory stops, so there are $M = |F|-1$ clusters. Furthermore, without loss of generality, each cluster has the same number of bus stops K. The first half of the $|P|$ passenger requests have desired arrival times and the other half have desired departure times. To give each component of the objective function equal importance, all objective weights W_i are given equal weight. A detailed study of the effect of the weight values on the solutions is beyond the scope of this paper. We further assume that the maximum values D^{tl} and D^{te} for respectively departing later or earlier than the originally communicated departure time T_p^0 to a certain passenger p are equal to each other. We denote this value D^{t} from now on.

Furthermore, after some trial-and-error, it was determined that the parameters δ^1, δ^2 and δ^3 of the Insertion heuristic will be set to 1000s, 1000s and 750s respectively. These values give the best results in the shortest runtime. The maximum number of iterations in the Improvement heuristic is set to 30000, in most cases a larger number of iterations does not improve the solution significantly, if at all. The maximum amount of time passengers have to wait to receive a notification about their request is $D^{\text{rt}} = 300s$. The details of the parameters of the instances, as well as the solutions discussed in this paper are available in detail online: `https://www.mech.kuleuven.be/en/cib/drbp/mainpage#section-18`.

6. Performance of the algorithm

In this section, the results for all the instances that are presented in Section 5 are discussed. First, the heuristic is run on all the instances in order to assess the quality of the solutions by comparing them with the solutions obtained by the algorithm discussed in Section 4.2, which is a similar algorithm to the one used in Galarza Montenegro et al. (2022b) to optimise a static FSMS. Second, the influence of the different instance parameters is studied. Lastly, an analysis on the degrees of freedom

($|B|$ = Number of buses, $|F|$ = Number of mandatory bus stops,
K = Number of optional stops per cluster, $|S|$ = Number of bus stops,
R = Number of passenger requests,
D^{f} = Maximum headway at the mandatory stops, C = Bus capacity)

Instance	**|B|**	**|F|**	**K**	**|S|**	**R**	**$\mathbf{D^f}$ (s)**	**C**
I_1	6	6	5	31	30	1200	40
I_2	6	6	5	31	70	1200	40
I_3	6	6	5	31	140	1200	40
I_4	6	6	5	31	200	1200	40
I_5	6	6	5	31	380	1200	40
I_6	6	6	3	31	30	1200	40
I_7	6	6	8	31	30	1200	40
I_8	6	6	10	31	30	1200	40
I_9	6	6	5	31	30	600	40
I_{10}	6	6	5	31	30	1800	40
I_{11}	6	6	5	31	30	2400	40
I_{12}	6	6	5	31	30	1200	10
I_{13}	6	6	5	31	30	1200	20
I_{14}	6	6	5	31	30	1200	30
I_{15}	3	6	5	31	30	1200	40
I_{16}	10	6	5	31	30	1200	40
I_{17}	15	6	5	31	30	1200	40
I_{18}	10	6	8	46	140	1200	20
I_{19}	10	6	8	46	30	1200	20
I_{20}	10	6	8	46	70	1200	20
I_{21}	10	6	8	46	200	1200	20
I_{22}	10	6	8	46	380	1200	20
I_{23}	10	6	3	46	140	1200	20
I_{24}	10	6	5	46	140	1200	20
I_{25}	10	6	10	46	140	1200	20
I_{26}	10	6	8	46	140	600	20
I_{27}	10	6	8	46	140	1800	20
I_{28}	10	6	8	46	140	2400	20
I_{29}	10	6	8	46	140	1200	10
I_{30}	10	6	8	46	140	1200	40
I_{31}	10	6	8	46	140	1200	60
I_{32}	3	6	8	46	140	1200	20
I_{33}	6	6	8	46	140	1200	20
I_{34}	15	6	8	46	140	1200	20

Table 3.: List of test instances

of the dynamic model is presented.

6.1. *Experimental results*

The heuristic is used to solve all of the instances that are described in Section 5. The results are summarised in Table 4. The instances are sorted to identify the instances that have a single parameter that varies, such as an increasing number of passenger requests. For example, the first five instances have the same parameters except for the number of passenger requests, which increases from instance to instance. There are two sets of instances that have different varying parameters. Each set has one base instance, namely instances I_1 and I_{18}, which is used to create the other instances where one parameter changes at the time.

From now on we refer to the objective function value that also includes the penalties as the global objective value. The objective function value that only includes the accepted passenger requests is referred to as the accepted objective value. The first column (Inst.) of the table shows the name of the instance, while the second column (Param.) shows the parameter that changes, along with the values it takes. In the third column (Global Obj.) of the table, the mean global objective function values per passenger of 5 simulation runs are given. The solutions of the instances solved with a static model, i.e., with the assumption that all requests are known beforehand, are calculated as well. The gap between the static objective function value and the mean objective function value and the best observed objective function value are presented in fourth and fifth columns (Mean Gap and Best Gap) respectively. The sixth column (Acc. Obj.) shows the accepted objective function value per passenger. The seventh column (IVT) shows the average ratio between the in-vehicle time per passenger obtained by the algorithm and the best possible in-vehicle time. The latter is determined by transporting every passenger directly to the destination, with the assumption that buses need to go to one of the mandatory stops first in order to reach the main road. In the eight column (WT), the ratio between the average walking time per passenger obtained by the algorithm and the best possible walking time, i.e., the walking time to the closest available bus stop, is presented. The columns Δ^{arr} and Δ^{dep} hereafter show the average difference between the desired and actual arrival time per passenger and the average difference between the desired departure time and actual departure time per passenger respectively. These last four columns solely consider the accepted passengers. The last column shows the average acceptance rate of passenger requests.

6.1.1. Service quality

It can be seen that the dynamic model heuristic performs quite well for most instances, having low gaps with respect to the static model. Figure 3 shows an overview of the average gap of the performance metrics, with respect to the mean values of the five runs. On average the gap between the mean global objective value and the static model is 6.5%. The gap between the best observed value and the static model is 4.4%. The gap of the accepted objective value is lower at 0.9%. We must note that this comparison is not entirely fair towards the dynamic model. In the dynamic model, there are additional constraints that emerge due to the real-time aspect of the optimization, such as the promised pickup time windows and previously assigned departure bus stops. In the static model, all information is known in advance and

(R = Number of passenger requests, K = Number of optional stops per cluster, D^{f} = Maximum headway at the mandatory stops, C = Bus capacity, $|B|$ = Number of buses)

Inst.	Param.		Global Obj. (s)	Mean Gap	Best Gap	Acc. Obj. (s)	IVT	WT	Δ^{arr} (s)	Δ^{dep} (s)	Acc. rate
I_1		30	454	3.1%	2.4%	449	1.07	1.05	63	85	99.3%
I_2		70	526	7.7%	3.2%	513	1.12	1.19	138	121	98.3%
I_3	R	140	616	11%	4.8%	567	1.20	1.37	167	126	93.3%
I_4		200	638	10%	8.3%	584	1.27	1.34	168	127	92.3%
I_5		380	684	6.1%	3.0%	597	1.34	1.54	153	136	87.5%
I_6		3	469	4.0%	0.0%	463	1.08	1.01	94	91	99.3%
I_1	K	5	454	3.1%	2.4%	449	1.07	1.05	63	85	99.3%
I_7		8	471	7.8%	6.3%	425	1.06	1.01	44	76	94.7%
I_8		10	471	8.4%	5.5%	419	1.05	1.02	46	60	94.0%
I_9		600	645	27%	27%	443	1.11	1.10	118	61	76.0%
I_1	D^{f}	1200	454	3.1%	2.4%	449	1.07	1.05	63	85	99.3%
I_{10}	(s)	1800	447	1.1%	1.0%	447	1.07	1.03	69	88	100%
I_{11}		2400	480	8.6%	7.6%	452	1.14	1.02	54	86	96.7%
I_{12}		10	530	16%	16%	414	1.08	1.04	79	36	86.7%
I_{13}	C	20	466	5.8%	0.8%	438	1.08	1.04	60	65	96.7%
I_{14}		30	468	5.2%	5.1%	440	1.05	1.01	71	89	96.7%
I_1		40	454	3.1%	2.4%	449	1.07	1.05	63	85	99.3%
I_{15}		3	806	26%	24%	494	1.02	1.40	212	91	60.7%
I_1	|B|	6	454	3.1%	2.4%	449	1.07	1.05	63	85	99.3%
I_{16}		10	420	3.4%	0.0%	408	1.03	1.01	38	51	98.7%
I_{17}		15	408	3.7%	0.0%	396	1.02	1.02	29	33	98.7%
I_{19}		30	405	1.9%	0.2%	405	1.05	1.01	27	53	100%
I_{20}		70	450	2.4%	0.6%	450	1.08	1.13	75	97	100%
I_{18}	R	140	508	4.2%	2.3%	507	1.11	1.18	128	118	99.9%
I_{21}		200	541	6.1%	2.5%	532	1.18	1.24	147	115	98.9%
I_{22}		380	668	9.6%	4.7%	570	1.27	1.61	158	120	86.4%
I_{23}		3	522	2.8%	0.7%	516	1.13	1.14	134	120	99.3%
I_{24}	K	5	507	2.3%	0.7%	507	1.13	1.20	111	102	100%
I_{18}		8	508	4.2%	2.3%	507	1.11	1.18	128	118	99.9%
I_{25}		10	508	3.8%	1.0%	503	1.11	1.24	115	102	99.4%
I_{26}		600	513	4.1%	0.1%	507	1.14	1.20	121	103	99.3%
I_{18}	D^{f}	1200	508	4.2%	2.3%	507	1.11	1.18	128	118	99.9%
I_{27}	(s)	1800	510	5.7%	4.7%	500	1.12	1.12	130	113	98.7%
I_{28}		2400	515	9.0%	7.7%	504	1.14	1.20	113	112	98.6%
I_{29}		10	525	5.3%	0.0%	513	1.15	1.24	125	100	98.4%
I_{18}	C	20	508	4.2%	2.3%	507	1.11	1.18	128	118	99.9%
I_{30}		40	525	6.6%	5.6%	513	1.14	1.23	118	134	98.4%
I_{31}		60	513	4.8%	3.6%	510	1.12	1.25	128	116	99.6%
I_{32}		3	888	13%	12%	568	1.15	1.68	113	156	55.4%
I_{33}	|B|	6	598	3.4%	1.9%	553	1.21	1.41	132	133	93.9%
I_{18}		10	508	4.2%	2.3%	507	1.11	1.18	128	118	99.9%
I_{34}		15	479	4.8%	4.4%	479	1.12	1.15	97	86	100%
$\mu_{I_1\hat{}I_{34}}$				6.5%	4.4%		1.12	1.18	103	98	95.1%

Table 4.: Results of all instances

the planning can be optimised accordingly, while in the dynamic model a previously determined solution needs to be updated when new requests are received. This means that the static model can only be viewed as a lower bound, and thus not necessarily as a feasible solution, for the dynamic model.

The average acceptance rate is 95.1%. This is an indication that, on average, a passenger request has a very high chance of being accepted. However, the static model has an acceptance rate of nearly 100%, which is why the global objective is higher than the accepted objective value. Instances with a relatively low number of buses and instances with low maximum headway D^{f} have larger gaps. These are instances that have more strict constraints, which leads to a higher number of passengers being rejected, which increases the global objective value by adding more penalties.

Forecasting the demand of passenger requests could be beneficial for increasing service quality because it can lead to higher acceptance rates and more foresight to accommodate the passengers' needs. For example, we could forecast that in the near future a certain number of passengers will make a request from a certain area. The algorithm could then ensure that there are enough buses available by that time in that area to pick up these passengers. This could be done by assigning certain already known passenger requests to different buses or bus stops, in order to free some capacity and/or to shorten the travel time of buses that are scheduled in the future. This, in turn, ensures that fewer future requests are rejected due to a lack of availability of buses. However, the low gaps with respect to the static model imply that forecasting the passenger request demand would not be very beneficial for most cases. A low gap with the static model implies that the dynamic model is performing almost as well as the static model. Since the static model is essentially a model with perfect prediction accuracy, this means any prediction in the dynamic model would not increase the service quality significantly. However, on certain instances the forecasting of the demand could prove to be important for increasing the acceptance rate and the service quality.

The objective function value of the accepted passengers and the global objective appear to be relatively consistent. The coefficient of variation, which is the standard deviation divided by the mean, of the global and accepted value are 0.0196 and 0.0144 respectively. On average, the in-vehicle time per passenger is 12% higher than the lowest possible value, the average walking time is 18% above the lowest possible value and the average difference between the arrival and departure times is 97s and 93s respectively. On average, a passenger would have a total journey time of around 20 minutes, which is 15% above the lowest possible journey time of seventeen minutes, and would depart or arrive approximately one minute and 35 seconds later or earlier than their desired time. The biggest gaps with the static model are found in the differences of departure and arrival times. As a trade-off, the in-vehicle times and walking times are on average slightly better in the dynamic model.

6.1.2. Runtime and response lead time

The average runtime of a single re-optimization varies slightly from instance to instance. On average, the runtime of a re-optimization is 26.4s, which is not a long runtime. The experiments are run on a normal work laptop, as mentioned above, and the algorithm is not being parallelised. In practice, the algorithm can be run on a much

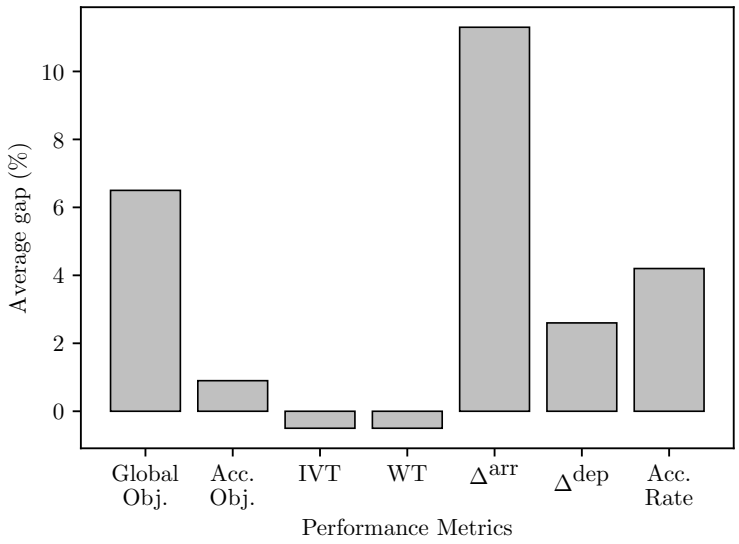
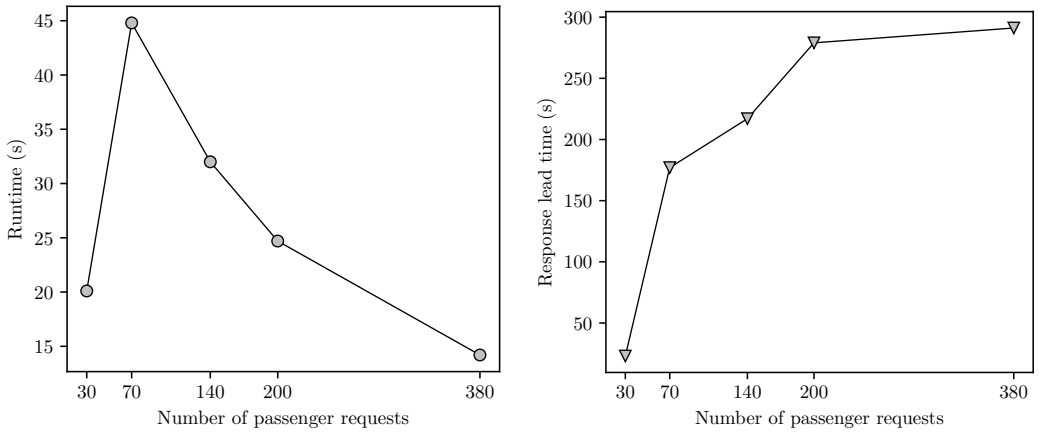


Figure 3.: Average gap of the heuristic with the improvement phase compared to the static model

more powerful device and the code can be parallelised to further increase the speed. Furthermore, the iterations for the search in Phase 2 can be reduced in order to obtain smaller runtimes if needed. In most cases the runtime remains relatively consistent. However, as can be seen in Figure 4(a), there is a clear correlation with the number of passenger requests and the runtime. The runtime of a re-optimization increases with the number of passenger requests until a certain point. When more passenger requests are obtained, the runtime starts to decrease with an increasing number of requests. When there is a small number of requests, the algorithm can accommodate the requests very easily and rapidly because there are many options for inserting the requests in the solution. When there is a large number of requests, the algorithm does not have many options for inserting the requests in the solution. This means less options are explored and the runtime is reduced. The most difficult re-optimizations, and thus the longest runtimes, are somewhere in between. Furthermore, there is a maximum amount of time D^{rt} that a passenger can wait for an answer after a request is made, i.e., the maximum response lead time. When there are too many passenger requests, these requests get backlogged until the previous requests are processed. Each request is still processed one by one, which means that the re-optimization is called whenever a new request arrives. This limits the amount of time the algorithm can devote to improving the current solution in each re-optimization and thus reduces the average runtime of a re-optimization. As can be seen in Figure 4(b), the average waiting time for a request response increases monotonically with the number of passenger requests. For example, instances with 30 passengers, i.e., most of the instances in the first set of experiments, have an average runtime of 20.1s and an average response lead time of 23s. Instances with 140 passengers, i.e., most of the instances in the second set of experiments, have an average runtime of 32s and an average response lead time of 217s. Instances with more than 140 passengers have an average runtime of 19.5s and an average lead time of 279.2s.

Figure 5 shows the improvement that is provided by the improvement phase. These results are obtained by running the algorithm on the instances without the improvement



(a) Average runtime (b) Average response leadtime

Figure 4.: Average runtime and response lead time w.r.t. the number of requests

phase and comparing them with the mean values of the results presented in Table 4. On average the global objective value improves by 21.2% and the accepted objective value by 14.4%. The improvement phase also increases the acceptance rate with 5.3%. These improvements show that the additional runtime of the improvement phase is justified.

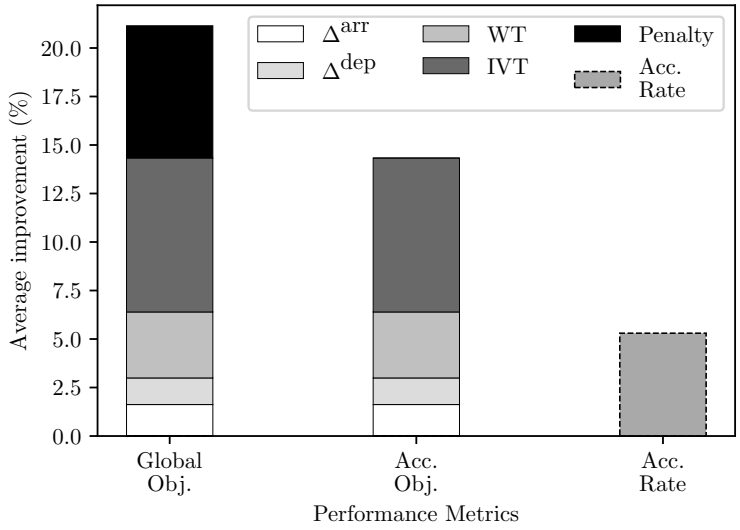


Figure 5.: Average improvement of Phase 2

6.2. *Influence of instance parameters*

As previously stated, the results presented in Table 4 are sorted in order to identify the instances that have a single varying parameter. Two sets of instances are created from two base instances, namely instance I_1 and instance I_{18}. Instance I_1 has a low number of passengers and a relative low number of buses, while instance I_{18} is a larger instance with more passenger requests and a higher number of buses. These two base instances

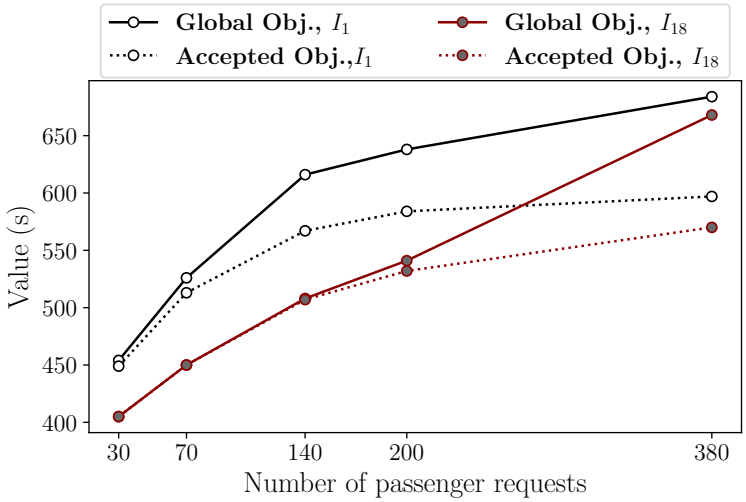
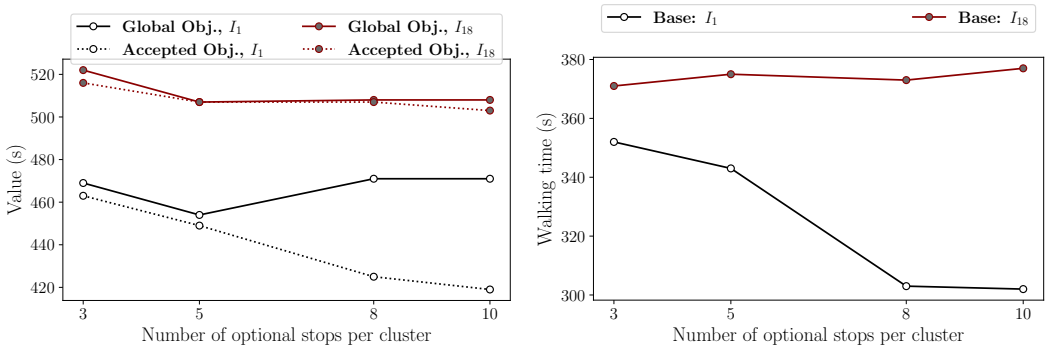


Figure 6.: Influence of the number of requests on the service quality

are used to create other instances, where a single parameter takes on different values while the other parameter values remain unchanged. The instance parameters that are discussed are the following: the number of passengers R making a request during the planning horizon, the number of optional stops K per cluster, the maximum headway D^{f}, the bus capacity C and the number of buses $|B|$. The number of passenger requests R is an external parameter, while the other parameters are resource parameters that can be controlled by the service provider.

6.2.1. Number of passenger requests

Figure 6 shows the influence of the passenger requests on the global and accepted objective value. It is clear that the number of passenger requests has a big influence on the global objective value. The more passenger requests there are, the larger the global and the accepted objective values are. This is to be expected, the presence of more passengers implies that the service cannot be tailored as well to each individual passenger. The global objective value also worsens with the number of passengers due to a worsening acceptance rate. The more requests there are, the more likely it is for a request to be rejected. The gap between the static model and the dynamic model seems to worsen as well. However, for the set of instances with fewer buses, the gap improves when more passenger requests are present. This is likely due to the fact that a very high number of passengers also makes it more difficult for the static model to optimally plan the service when the number of buses is relatively low. Meanwhile, the dynamic model's solution quality remains consistent. This, in turn, makes the gap improve. As expected, the difference between departure and arrival times worsens the most. When there are more requests and the vehicle fleet remains the same, it becomes more difficult to provide a customised service to each passenger where passengers are picked up and dropped off at their desired times. More requests consequently also result in longer travel and walking times because buses must travel longer routes. However, as more passengers are served, this worsening diminishes. After a certain point, passengers can be serviced more easily in groups, making the service relatively more efficient.



(a) Influence on the service quality

(b) Influence on the walking time

Figure 7.: Influence of the number of optional stops

6.2.2. Number of optional stops per cluster

In Figure 7 (a), the global and accepted function are plotted against the number of optional stops per cluster. It can be seen that a low number of optional stops worsens the global objective value. On the one hand, a larger number of optional stops seems to have a diminishing effect on the global objective value, with seemingly no improvement at all or even a worsening in the value with a larger number of optional stops. On the other hand, the accepted objective value consistently improves with an increasing number of optional stops. The acceptance rate worsens with a larger number of optional stops. This is likely due to the fact that more optional stops result in longer routes for some bus trips, which might make it more difficult to make feasible passenger assignments. The decreasing acceptance rate explains why the global objective value stagnates after a certain number of optional stops, since rejected passengers contribute more towards the global objective value. The decline in acceptance rate also explains the worsening gap with the static model, since the static model is more efficient at accommodating requests regardless of the number of optional stops. Furthermore, the static model actually benefits from a larger number of optional stops. The average departure time difference improves the most, followed by the arrival time difference. It becomes easier to schedule the departure and arrival of the buses according to the passengers' needs when there are more possible departure stops. As can be seen in 7 (b), the average walking time improves significantly as well for the same reason. The average in-vehicle time of the passengers improves slightly with the number of optional stops because the routes can be more customised for each passenger. The effect of the number of optional stops, on all metrics, seems to be less prominent for the set of instances with a larger fleet size and more requests. It needs to be noted that the ratio of the walking times reported in Table 4 remains unaffected by the number of optional stops. When more optional stops are added, the shortest possible walking decreases as well.

6.2.3. Maximum headway

There is clear difference of how much the maximum headway influences the results of both sets of experiments, i.e., experiments with base instances I_1 and I_{18}, as can be seen in Figure 8. In the second set of experiments, with a larger number of buses, the maximum headway has a smaller impact on the results. This is due to the fact that the presence of more buses during the same planning horizon makes it easier

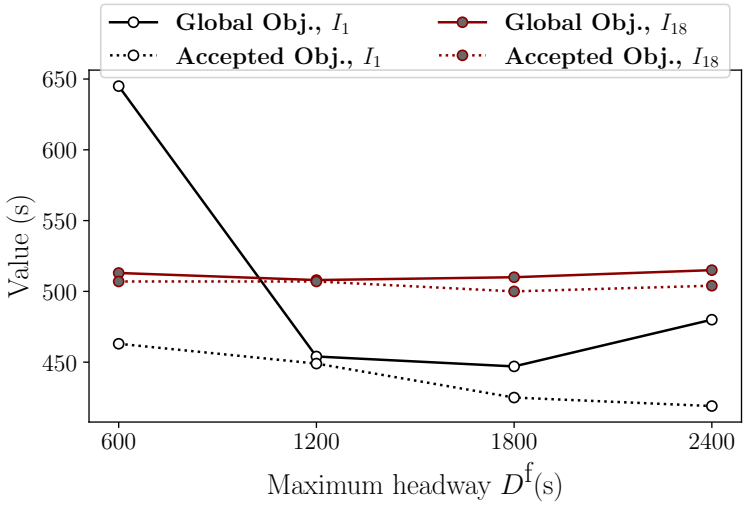


Figure 8.: Influence of the maximum headway on the service quality

to satisfy the headway constraints without compromising the global objective value significantly. Both the acceptance rate and the global objective value are virtually unaffected by the maximum headway. The other metrics do not change much either, and any changes can be attributed to the use of different random numbers. The only notable influence is the worsening in the gap between the static model and the dynamic model with an increasing maximum headway. This implies that the static model benefits more from a larger maximum headway than the dynamic model.

In the first set of experiments, with a relatively low number of buses, the influence of the headway becomes more clear. It is evident that a small headway leads to very low acceptance rates and thus to an worsened global objective value and gap. The accepted objective value remains relatively stagnant. It seems that a smaller headway results in shorter in-vehicle times, with a trade-off for larger differences between the arrival and departure times. A shorter headway forces the buses to make shorter routes in order to maintain the frequency of departing buses at the mandatory stops. This improves the in-vehicle time. At the same time, the shorter routes make it more difficult to satisfy the desired departure times. Furthermore, the more strict departure times at the mandatory stops (due to the smaller maximum headway) make it more difficult to meet the desired arrival times.

6.2.4. Vehicle capacity

Figure 9 shows there is again a clear difference between the two sets of experiments. On the one hand, the set of experiments with a larger number of passengers and buses shows that the vehicle capacity does not influence the results much at all. This is likely due to the fact that the vehicle capacity stays relatively small in comparison to the number of passenger requests. On the other hand, the set of experiments with a low number of requests shows a clear correlation between the vehicle capacity and the results. The results show that a larger vehicle capacity improves the global objective value and the gap. The acceptance rate also increases to nearly 100% when 20 or more passengers can be assigned to a single bus. This is to be expected since a larger capacity offers more feasible assignments for the incoming requests. Since more passenger

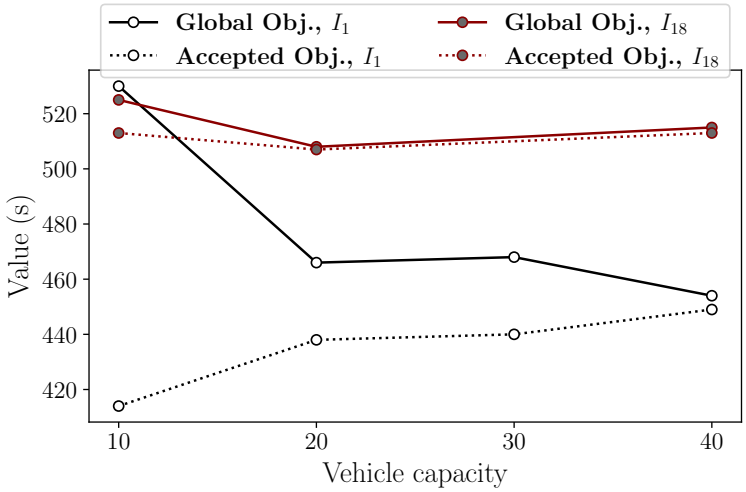


Figure 9.: Influence of the vehicle capacity on the service quality

requests are gradually accepted with an increasing capacity, the accepted objective value worsens slightly. When more passengers are accepted in the service, it becomes more difficult to lower the accepted objective value per passenger.

6.2.5. Number of buses

Figure 10 shows that the global objective value clearly improves when more buses are used. The improvement is linear, but appears to be less significant when the number of buses increases. All performance metrics appear to improve when more buses are used, although lower walking and in-vehicle times are sometimes sacrificed for smaller differences in arrival and departure times. The difference in arrival times and the difference in departure times are the most significant improvements. The in-vehicle time is the second most significant reduction. This is to be expected because more buses allow for more personalised service for each passenger; more available buses allow for more customised routes and timetables. It is evident that the acceptance rate worsens substantially when not enough buses are utilised. The number of buses is the resource parameter that has the most impact on the global objective value. Furthermore, the number of buses is the only resource parameter that improves both the acceptance rate and the accepted objective value at the same time.

6.2.6. Instance parameters for the best service quality

From the analysis of the instance parameters we can determine the values for the resource parameters that lead to the best service quality. It can be concluded that five or more optional stops per cluster, a maximum head way of at least 20 minutes, a bus capacity of at least 20 passengers and a fleet size of at least six buses lead to good service quality. Furthermore, it can be noted that a large fleet size is one of the most important factors for the service quality. When there are more buses available, other instance parameters influence the service quality to a lesser extent.

In terms of external parameters, i.e., the number of passenger requests, we evidently see that the service improves when less passenger requests are received.

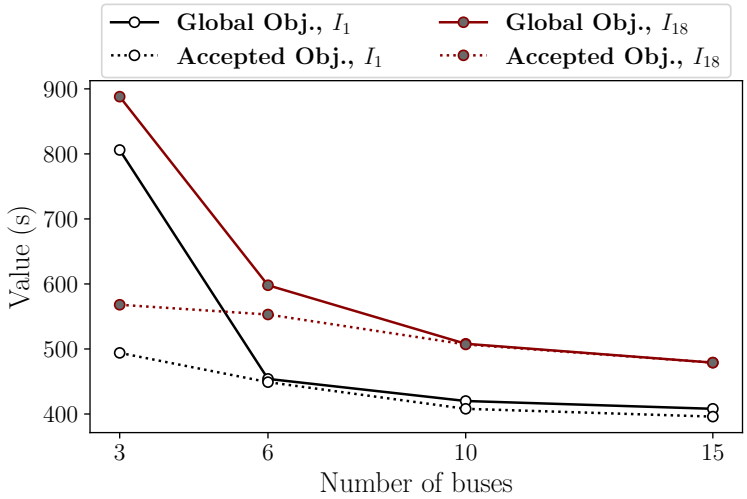


Figure 10.: Influence of the fleet size on the service quality

6.3. *Influence of the freedom of optimization*

In this section we study the influence of the freedom of optimization on the performance metrics. The freedom of optimization refers to how much we allow the service to be flexible with respect to the real-time decisions it has to make, namely by allowing passengers to be reassigned to a different departure stop or by increasing the size of the promised pickup time window Δ_p^{d}. To do this, we select four different instances to run experiments on. We opt to choose instances I_9, I_{15}, I_{18} and I_{20} since they have different attributes such as different number of passenger request, number of buses and different acceptance rates. We discuss the maximum time D^{t} for departing later or earlier than the originally communicated departure time T_p^0, since this is what determines the promised pickup time window Δ_p^{d}. This is done by letting D^{t} take on different values, namely 150s, 300s 600s and 900s. Furthermore we also solve the instances with an additional degree of optimization freedom; we allow the reassignment of departure stops to passengers, in case it might be beneficial. The results of these experiments are summarised in Table 5. The first column shows the instance that is utilised. The second column shows whether or not the stops that are assigned to passengers can be changed afterwards. The third column shows the values of D^{t}. The remaining columns are the same as the columns in Table 4. Table 5 shows the mean values of a set of five experiments that are performed with different random numbers.

Surprisingly, the results for instance I_9 show that the global objective value can worsen when more freedom is given to the optimization of the service, i.e., when D^{t} takes on larger values or when existing stop assignments can be changed. However, the accepted objective value always improves with increasing optimization freedom, which is to be expected. The worsening of the global objective value is a consequence of the decreased acceptance rate. More degrees of freedom in the optimization of the planning leads to a more greedy construction of the timetables and routes. On the one hand, since the planning is re-optimised as new requests arrive, a more greedy construction algorithm can lead to solutions that may improve the quality of the service of passengers already in the service, i.e., the objective value, travel

Inst.	Fixed stops	D^t (s)	Global Obj. (s)	Mean Gap	Best Gap	Acc. Obj. (s)	IVT	WT	Δ^{arr} (s)	Δ^{dep} (s)	Acc. Rate
I_9	Yes	150	628	25%	22%	471	1.12	1.12	93	129	80.7%
	Yes	300	675	31%	26%	459	1.10	1.11	137	68	74.0%
	Yes	600	645	27%	27%	443	1.11	1.10	118	61	76.0%
	Yes	900	630	26%	22%	431	1.11	1.10	78	64	76.7%
	No	600	689	32%	28%	426	1.09	1.10	74	61	69.3%
I_{15}	Yes	150	871	31%	29%	487	1.06	1.44	126	104	52.0%
	Yes	300	886	32%	29%	464	1.04	1.43	97	111	48.7%
	Yes	600	806	26%	24%	494	1.02	1.40	212	91	60.7%
	Yes	900	832	28%	28%	485	1.01	1.41	211	96	56.7%
	No	600	825	27%	22%	452	1.02	1.36	85	110	55.3%
I_{18}	Yes	150	534	8.9%	8.9%	534	1.14	1.15	174	127	100%
	Yes	300	540	10%	5.2%	517	1.11	1.24	140	113	97.0%
	Yes	600	508	4.2%	2.3%	507	1.11	1.18	128	118	99.9%
	Yes	900	508	4.3%	1.7%	502	1.10	1.16	127	119	99.3%
	No	600	509	5.5%	2.2%	508	1.11	1.17	136	117	99.6%
I_{20}	Yes	150	463	5.0%	0.5%	460	1.08	1.14	105	94	99.7%
	Yes	300	462	4.8%	2.2%	457	1.07	1.17	103	82	99.4%
	Yes	600	450	2.4%	0.6%	450	1.08	1.13	75	97	100%
	Yes	900	452	2.7%	1.5%	452	1.10	1.12	72	87	100%
	No	600	451	2.8%	1.5%	451	1.10	1.10	81	86	100%

Table 5.: Dynamic parameter results

time, walking time etc. On the other hand, the more greedy construction makes the assignment of new passengers more difficult, which decreases the acceptance rate. The reason why the static model does not suffer from these downfalls is because all requests are known beforehand, which allows the model to find a balance between the greediness and the feasibility of the solution.

The same holds true for instance I_{15}. However, there seems to be an ideal level of optimization freedom because the best results are observed when $D^t = 600s$ and no change of the bus stop assignment is possible. In this scenario, the acceptance rate is higher in exchange for a less ideal service quality for the passengers already in the service. Because the limiting constraint is a low number of buses in this instance, a sufficient level of optimization freedom allows the algorithm to schedule the fleet of buses in such a way that more requests can be accepted. A higher level of optimization leads again to a overly greedy algorithm that decreases the acceptance rate.

The remaining instances, i.e., instances I_{18} and I_{20}, show the same behaviour as instance I_{15}. Again, there is an ideal level of optimization freedom for these instances. Since the acceptance level of these instances is always very high, the objective values stop improving significantly after a certain level of optimization freedom. Since the limiting factor of these instances is the large number of passenger requests, the service groups passengers to make the planning more efficient. After a certain level of optimization freedom is reached, the grouping of passengers yields diminishing returns.

7. Case study: Antwerp

In order to further validate the usefulness of the dynamic Feeder Service with Mandatory Stops (DFSMS), we test the service on a case study for a potential implementation in the city of Antwerp, Belgium. We simulate this case study by generating instances with the help of REQcreate (Queiroz et al., 2022), which is a tool designed to generate realistic instances for demand-responsive transportation services. REQcreate retrieves real-life city networks from OpenStreetMap (OSM) to generate more realistic instances. The tool also allows the generation of transportation requests, which can mimic realistic mobility patterns. Figure 11 shows the map segment that is relevant for the DFSMS. There are eight mandatory stops equidistantly placed along the N177 highway (blue line), starting in Boom and ending near the centre of Antwerp (both in red squares). There are 56 optional stops scattered across residential neighbourhoods near the N177 (green squares), such as Hemiksem, Schelle or Wilrijk. One hundred passengers requests are sporadically received throughout a planning horizon of two hours. This represents the demand for transportation of people residing in the outskirts of Antwerp and waiting to go to the city centre, for example to go to work. The locations of passengers correspond to realistic origin locations in residential areas. For this case, we assume that half of the passengers have a desired departure time and another half have a desired arrival time. To test this instance, we impose a maximum headway D^{f} of 20 minutes and a vehicle capacity C of 20 passengers. We study the performance of the DFSMS with different fleet sizes.

We compare the performance of the DFSMS with the performance of existing public transit options that are currently available in the region. There are more than 20 bus lines available in the region that can be used by passengers to reach the destination. This means that the existing transit options utilise more resources, when compared to the DFSMS. With the coordinates of each passenger, we can calculate their User Journey Time (URT) to reach the centre of Antwerp using these public transit services. This URT includes walking time to bus stops and in-vehicle time. The DFSMS is optimised five times for each case and the mean value is taken.

Table 6 summarises the results of this case study. The first column (Case) of the table shows the case instance. The first row corresponds with the use of existing public transit options, while the lower rows represent the DFSMS with different fleet sizes. The second column (IVT) shows the average in-vehicle time per passenger. In the third column (WT), the average walking time per passenger is presented. The fourth column shows the average User Ride Time (URT) per passenger, which consist of the walking time and the in-vehicle travel time. The columns Δ^{arr} and Δ^{dep}, again, show the average difference between the desired and actual arrival time per passenger and the average difference between the desired departure time and actual departure time per passenger. Column eight shows the accepted objective, and the ninth column shows the global objective. The last column shows the acceptance rate of the DFSMS.

The acceptance rate in the DFSMS increases significantly with the number of buses. When the fleet size is only three buses, the acceptance rate is relatively low, at 26%. As more buses are available, the acceptance rate increases up until a fleet size of twelve buses. When twelve or more buses are available, the acceptance rate stagnates at around 90%. This indicates that a fleet size of twelve of more buses is needed for a high acceptance rate of passenger requests. Figure 12(a) shows the percent difference

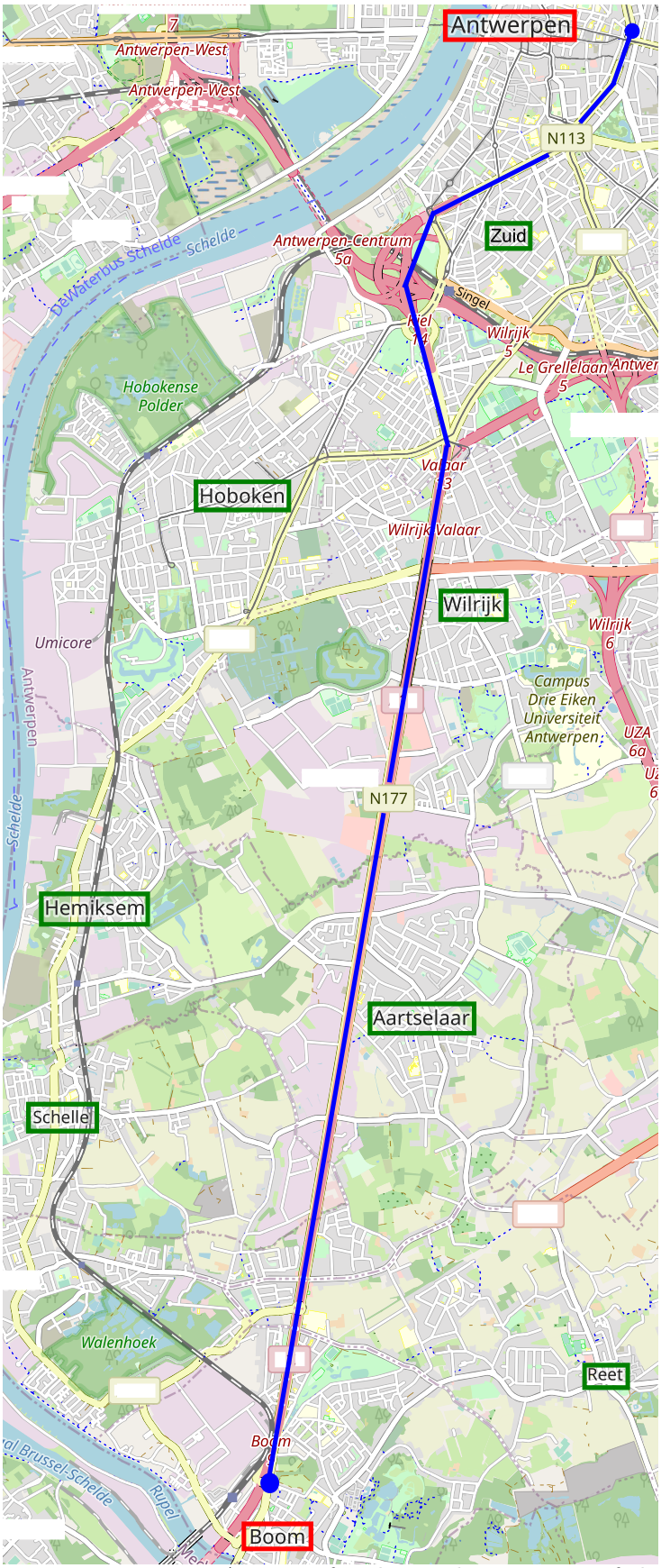
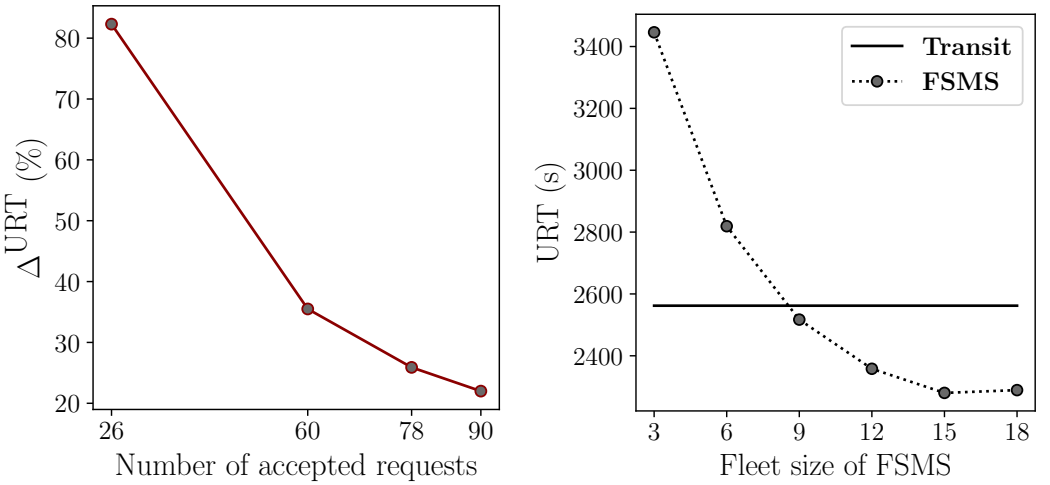


Figure 11.: Map segment in Antwerp for the implementation of the DFSMS

Case	IVT (s)	WT (s)	URT (s)	Δ^{arr} (s)	Δ^{dep} (s)	Acc. Obj. (s)	Global Obj. (s)	Acc.
Transit			2562	750		1101	1101	
$\|B\|=3$	945.5	460	1406	0	287	561	1358	26%
$\|B\|=6$	1370	521	1890	92	148	706	1082	60%
$\|B\|=9$	1488	547	2035	116	125	754	952	77%
$\|B\|=12$	1603	529	2132	106	110	777	872	89%
$\|B\|=15$	1579	526	2106	87	103	759	833	92%
$\|B\|=18$	1552	508	2059	83	100	742	839	89%

Table 6.: Results for case study in Antwerp



(a) URT in function of the number of requests

(b) URT with penalty in function of fleet size

Figure 12.: Influence of the number of requests and the fleet size on the URT

$\left(\Delta^{\text{URT}}\right)$ between the URT of accepted passengers in the DFSMS and the URT of the existing transit options in relation to the number of accepted passenger requests. The higher this difference is, the better the DFSMS performs comparatively. The URT of passengers is consistently lower in the DFSMS than in the existing transit options. When there are less passengers accepted, this difference is larger. This indicates that the DFSMS outperforms the existing transit options in terms of URT in the case of both high and low demand densities. Nevertheless, when demand is low, the DFSMS performs better. Figure 12(b) shows the URT of the DFSMS when rejected passenger requests are penalised with the maximum URT of the existing transit options (4190s). It can be seen that the penalised URT clearly decreases when the fleet size is larger. When nine or more buses are utilized, the DFSMS outperforms the existing transit options in terms of the penalised URT. When fifteen or more buses are used, this difference increases to 12% on average. With fewer buses, the small acceptance rate results in a high penalty cost and the existing transit options are preferred.

When we take the difference in desired and actual arrival/departure times into account, we are able to compare the accepted and the global objective of the DFSMS

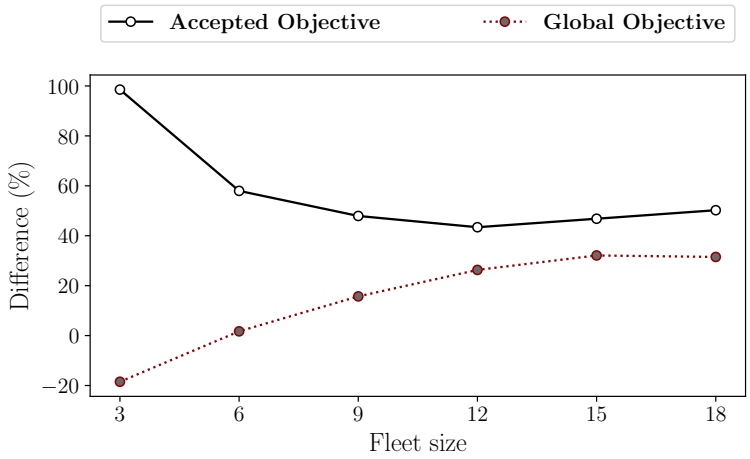


Figure 13.: Difference in global and accepted objective with existing transit options

with the existing transit options. We assume an average difference between the desired and actual arrival/departure times of 12.5 minutes for the existing transit options. This value is chosen because the existing transit options offer a bus departure every 25 minutes on average in the region of this case study, i.e., the average headway is 25 minutes. This implies that passengers depart or arrive, on average, half of this headway (750s) before or after their desired arrival/departure time. For requests in the DFSMS that are not accepted, we assume a penalty per rejected request that is equal to the maximum accepted objective of passengers using the existing transit options (1637s). Figure 13 shows the percent difference of the global and accepted objectives between the DFSMS and the existing transit options, in relation to the fleet size. A higher positive difference indicates that the DFSMS is performing better in comparison to the transit options. Evidently, the accepted objective is better in the DFSMS when there are fewer passengers in the system, i.e., when the acceptance rate is lower due to a small fleet size. With a fleet size of three buses, and an acceptance rate of 26%, the accepted objective is 96% better in the DFSMS compared to the transit options. When the Acceptance rate is around 90%, i.e., when the fleet size is twelve buses or more, the accepted objective of the DFSMS is on average 45% better than in the existing transit options. The difference in the global objective increases when more buses are utilised. When three buses are used, the difference is negative at -18%, which indicates that the existing transit options are outperforming the DFSMS. When six buses are used, the DFSMS starts to outperform the transit options, with a difference of 1.7%. A fleet size of fifteen or more buses gives an average difference of 31.6%.

In conclusion, we can say that the DFSMS is a promising alternative transportation service for passengers that need to reach the centre of Antwerp and live in the area displayed in Figure 11. The DFSMS offers a service with lower average URT and with more customised timetables that better satisfy the needs of the passengers. When the fleet size is large enough, with twelve buses or more, the acceptance rate is also quite high at around 90%. Even when we use a large penalty, i.e., the maximum value of the existing transit options, the DFSMS can outperform the existing options when nine or more buses are utilised. When we look at the global objective, the DFSMS outperforms the transit options when six or more buses are utilised. Furthermore, passengers that

do not make a reservation can still board a bus at the mandatory stops at the N177 and do not need to wait longer than 20 minutes for a bus.

8. Conclusion

This paper makes use of the feeder service with mandatory stops (FSMS), which was previously introduced in Galarza Montenegro et al. (2022b). The FSMS was previously optimised with the assumption that all requests are known beforehand, i.e., the static optimization model of the FSMS. Although not entirely unrealistic, this assumption limits the flexibility of the service. Furthermore, in a real world scenario it might be more convenient to allow last minute requests as well. We contribute to the literature by optimising the FSMS in a real-time manner, where incoming requests can still be processed even after the buses left the depot. We denote this as the dynamic FSMS (DFSMS).

To optimise the dynamic model, a novel two phase heuristic with an insertion phase and an improvement phase is developed. The heuristic is tested on 34 different instances with different instance parameters, such as the number of passenger requests, number of buses etc. It is found that the heuristic performs well for most instances, with an average acceptance rate of 95.1% for passenger requests and an average gap of 6.5% when compared to the static variant of the FSMS. Experiments on the different instance parameters show that the number of available buses and the number of passenger requests per hour affect both the solution quality as well as the acceptance rate the most. In contrast to the static model, the number of optional stops worsens the quality of the solutions by decreasing the acceptance rate of requests.

Experiments on the degrees of optimization freedom are performed as well. It is found that more optimization freedom improves the service quality of the passengers that have already been accepted by the service. However, in some cases, the global objective function value can worsen with more freedom of optimization. The additional optimization freedom can lead to a more greedy heuristic for the already accepted passengers, which consequently decreases the acceptance rate of later requests. Depending on the instance, there seems to be an ideal level of optimization freedom, which provides the best results.

A case study in Antwerp city shows that the FSMS is a promising alternative to existing transit options in the region. When there are enough resources available, the FSMS provides a service with 22% lower average user ride times and more customized schedules that meet the needs of the passengers better. With a large enough fleet size of twelve buses or more, the acceptance rate is also fairly high at around 90%. When the objectives of the FSMS and the existing transportation services are compared, the FSMS performs 31.6% better on average when enough buses are available to achieve a high acceptance rate.

However, this study still has its limitations and there is still further research that can be done. First, there is a limited number of cases where there is a large gap between the dynamic and the static model. Furthermore, it could be seen that there are diminishing returns when the optimization freedom and the level of flexibility is too high. This indicates that the heuristic can still be improved. This can be done by

forecasting incoming demand and adjusting the planning according to this forecasted demand. Second, passengers that use the service without a reservation are not explicitly considered in the optimization of the service. The mandatory stops are a safety net for these passengers and the use of a capacity for the buses helps in controlling the crowds. However, the demand for transportation of these passengers is much more stochastic in nature and thus deserves more attention. Lastly, we use weight factors in the objective function of both services. This is a good way to improve upon multiple service quality metrics at once. However, a more detailed study is needed to determine the best values of these weights. A Pareto analysis or a multi-criteria analysis can be very interesting in future research.

Acknowledgments

This project was supported by the FWO (Research Foundation Flanders) project G.0759.19N.

References

Alonso-González, M. J., T. Liu, O. Cats, N. Van Oort, and S. Hoogendoorn (2018). The Potential of Demand-Responsive Transport as a Complement to Public Transport: An Assessment Framework and an Empirical Evaluation. *Transportation Research Record 2672*(8), 879–889.

Beirão, G. and J. A. Sarsfield Cabral (2007). Understanding attitudes towards public transport and private car: A qualitative study. *Transport Policy 14*(6), 478–489.

Ceder, A. (2013). Integrated smart feeder/shuttle transit service: simulation of new routing strategies. *Journal of Advanced Transportation 47*, 595–618.

Clarke, G. and J. W. Wright (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations Research 12*(4), 568–581.

Crainic, T. G., F. Errico, F. Malucelli, and M. Nonato (2012). Designing the master schedule for demand-adaptive transit systems. *Annals of Operations Research 194*(1), 151–166.

Crainic, T. G., F. Malucelli, M. Nonato, and F. Guertin (2005). Meta-heuristics for a class of demand-responsive transit systems. *INFORMS Journal on Computing 17*(1), 10–24.

Defryn, C. and K. Sörensen (2017). A fast two-level variable neighborhood search for the clustered vehicle routing problem. *Computers & Operations Research 83*, 78–94.

Dou, X. and Q. Meng (2019). Feeder Bus Timetable Design and Vehicle Size Setting in Peak Hour Demand Conditions. *Transportation Research Record 2673*(10), 321–332.

Dou, X., Q. Meng, and K. Liu (2021). Customized bus service design for uncertain commuting travel demand. *Transportmetrica A: Transport Science 17*(4), 1405–1430.

Freitas, M., J. M. P. Silva, and E. Uchoa (2023). A unified exact approach for clustered and generalized vehicle routing problems. *Computers & Operations Research 149*, 106040.

Fu, L. and Q. Liu (2003). Real-Time Optimization Model for Dynamic Scheduling of Transit Operations. *Transportation Research Record 1*(1857), 48–55.

Galarza Montenegro, B. D., K. Sörensen, and P. Vansteenwegen (2021). A large neighborhood search algorithm to optimize a demand-responsive feeder service. *Transportation Research Part C: Emerging Technologies 127*, 103102.

Galarza Montenegro, B. D., K. Sörensen, and P. Vansteenwegen (2022a). A column generation algorithm for the demand-responsive feeder service with mandatory and optional, clustered bus-stops. *Networks 80*(3), 274–296.

Galarza Montenegro, B. D., K. Sörensen, and P. Vansteenwegen (2022b). A demand-responsive feeder service with a maximum headway at mandatory stops. *Under review, available as a working paper at* `https://hdl.handle.net/10067/1944620151162165141`.

Guo, R., W. Guan, W. Zhang, F. Meng, and Z. Zhang (2019). Customized bus routing problem with time window restrictions: model and case study. *Transportmetrica A: Transport Science 15*(2), 1804–1824.

Hintsch, T. and S. Irnich (2018). Large multiple neighborhood search for the clustered vehicle-routing problem. *European Journal of Operational Research 270*(1), 118–131.

Ibraeva, A., G. H. de Almeida Correia, C. Silva, and A. P. Antunes (2020). Transit-oriented development: A review of research achievements and challenges. *Transportation Research Part A: Policy and Practice 132*, 110–130.

Iliopoulou, C., K. Kepaptsoglou, and E. Vlahogianni (2019). Metaheuristics for the transit route network design problem: a review and comparative analysis. *Public Transport 11*(3), 487–521.

Knowles, R. D., F. Ferbrache, and A. Nikitas (2020). Transport's historical, contemporary and future role in shaping urban development: Re-evaluating transit oriented development. *Cities 99*, 102607.

Lee, A. and M. Savelsbergh (2017). An extended demand responsive connector. *EURO Journal on Transportation and Logistics 6*(1), 25–50.

Leich, G. and J. Bischoff (2019). Should autonomous shared taxis replace buses? a simulation study. *Transportation Research Procedia 41*, 450–460.

Li, X. (2009). *Optimal design of demand-responsive feeder transit services.* Ph. D. thesis, Texas A&M University.

Li, X., Y. Luo, Y. Li, H. Li, and W. Fan (2023). A joint optimisation model for designing demand responsive connectors fed by shared bikes. *Transportmetrica A: Transport Science 19*(2), 2025949.

Li, X. and L. Quadrifoglio (2010). Feeder transit services: Choosing between fixed and demand responsive policy. *Transportation Research Part C: Emerging Technologies 18*(5), 770–780.

Liu, T. and A. A. Ceder (2015). Analysis of a new public-transport-service concept: Customized bus in china. *Transport Policy 39*, 63–76.

Lu, X., J. Yu, X. Yang, S. Pan, and N. Zou (2015). Flexible feeder transit route design to enhance service accessibility in urban area. *Journal of Advanced Transportation 50*(4), 507–521.

Melis, L. and K. Sörensen (2021). The static on-demand bus routing problem: large neighborhood search for a dial-a-ride problem with bus station assignment. *International Transactions in Operational Research 29*(3), 1417–1453.

Mistretta, M., J. A. Goodwill, R. Gregg, and C. DeAnnuntis (2009). Best Practices in Transit Service Planning. Final Report No. BD549-38. Technical report, Center for Urban Transportation Research for the Florida Department of Transportation.

Molenbruch, Y., K. Braekers, and A. Caris (2017). Typology and literature review for dial-a-ride problems. *Annals of Operations Research 259*(1-2), 295–325.

Pratelli, A., M. Lupi, A. Farina, and C. Pratelli (2018). Comparing route deviation bus operation with respect to dial-a-ride service for a low-demand residential area. *DATA ANALYTICS 2018*, Article ID 151.

Qiu, F., W. Li, and A. Haghani (2015). An exploration of the demand limit for flex-route as feeder transit services: a case study in Salt Lake City. *Public Transport 7*(2), 259–276.

Quadrifoglio, L., M. M. Dessouky, and F. Ordóñez (2008). Mobility allowance shuttle transit (MAST) services: MIP formulation and strengthening with logic constraints. *European Journal of Operational Research 185*(2), 481–494.

Quadrifoglio, L. and X. Li (2009). A methodology to derive the critical demand density for designing and operating feeder transit services. *Transportation Research Part B: Methodological 43*(10), 922–935.

Queiroz, M., F. Lucas, and K. Sörensen (2022). Instance generation tool for on-demand transportation problems. *Preprint available on arXiv*.

Schöbel, A. (2012). Line planning in public transportation: models and methods. *OR spectrum 34*(3), 491–510.

Sevaux, M. and K. Sörensen (2008). Hamiltonian paths in large clustered routing problems. In *Proceedings of the EU/MEeting 2008 workshop on Metaheuristics for Logistics and Vehicle Routing, EU/ME*, Volume 8, pp. 411–417.

Sun, B., M. Wei, and S. Zhu (2018). Optimal design of demand-responsive feeder transit services with passengers' multiple time windows and satisfaction. *Future Internet 10*(3).

Tang, C., J. Liu, A. A. Ceder, and Y. Jiang (2023). Optimisation of a new hybrid transit service with modular autonomous vehicles. *Transportmetrica A: Transport Science 0*(0), 1–23.

Vansteenwegen, P., L. Melis, D. Aktaş, B. D. Galarza Montenegro, F. Veiera, and K. Sörensen (2022). A survey on demand-responsive public bus systems. *Transportation Research Part C: Emerging Technologies 137*, 103573.

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

Zhao, J., S. Sun, and O. Cats (2021). Joint optimisation of regular and demand-responsive transit services. *Transportmetrica A: Transport Science 0*(0), 1–24.