University of Antwerp
Faculty of Business and Economics

DEPARTMENT OF ENGINEERING MANAGEMENT

A demand-responsive feeder service with mandatory and optional, clustered bus-stops

Bryan David Galarza Montenegro, Kenneth Sörensen & Pieter Vansteenwegen

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

FACULTY OF BUSINESS AND ECONOMICS

DEPARTMENT OF ENGINEERING MANAGEMENT

A demand-responsive feeder service with mandatory and optional, clustered bus-stops

Bryan David Galarza Montenegro, Kenneth Sörensen & Pieter Vansteenwegen

RESEARCH PAPER 2020-006
NOVEMBER 2020

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

D/2020/1169/006

A demand-responsive feeder service with mandatory and optional, clustered bus-stops

Bryan David Galarza Montenegro*[,1], Kenneth Sörensen[1], and Pieter Vansteenwegen[2]

[1]ANT/OR - Operations Research Group,
Department of Engineering Management, University of Antwerp
[2]Mobility Research Centre - CIB, KU Leuven
*Corresponding author:
bryan.galarzamontenegro@uantwerpen.be (B.D. Galarza Montenegro)
Prinsstraat 13, 2000 Antwerp, Belgium

November 14, 2020

Abstract

With the rise of smart cities in the near future, it will be possible to collect relevant data from passengers in order to improve the quality of transport services. In this paper, a mathematical model and algorithm are developed to plan the trips of the buses in a demand-responsive feeder service. A feeder service transports passengers from a low-demand area, like a sub-urban area, to a transportation hub, like a city center. The feeder service modeled in this paper considers two sets of bus stops: mandatory stops and optional stops. Mandatory stops are always visited by a bus, while optional stops are only visited when a client nearby makes a request for transportation. Passengers are assigned to a bus stop within walking distance. This in turn, gives the service both flexibility through the changing timetables and routes of the buses and some predictability due to the mandatory stops. To optimize the performance of the service, mathematical modeling techniques to improve the model's runtime are developed. It is concluded that a combination of column generation and the separation of sub-tour elimination constraints decreases the computing time of small and midsize instances significantly.

Keywords: flexible bus services; on-demand transportation; feeder service; demand-responsive transportation; combinatorial optimization; column generation

1 Introduction

Mobility is essential for the growth of a society as it enables accessibility to opportunities, social networks, goods and services. Local public transport contributes greatly to urban mobility. Over 57.6 billion public transport journeys were recorded in the European Union in 2014, where 55.7% of these journeys were road-based services [1]. An adequate public transport service is thus essential and can reduce social exclusion and poverty [2]. Over the past few decades, mobility has increased substantially across many European cities. However, this has also resulted in an increase in congestion and pollution due to the preference for private transport over the collective use of public transport ([3], [4]), underlying the

need to increase the attractiveness of public transport.

To become more market-oriented and competitive, the service quality of public transport needs to improve, which can only be achieved through a clear understanding of travel behavior, consumer needs and expectations [5]. Research has shown that reliability is a decisive factor. Rather than waiting time, the uncertainty of when transport will arrive is of most importance. Attributes such as frequency, comfort and arrival time at the destination are also highly valued by consumers. These attributes are key elements of consumer satisfaction ([6], [7]). Other attributes may also have a positive effect on satisfaction and can represent great potential for improvement. For instance, service providers should make clear and simple information available to the public ([5], [8]).

In the future, most of the aforementioned issues regarding public transportation can be addressed. This is possible considering the rise of smart cities. A smart city is an innovative city that uses information and communication technologies and other means to increase the effectiveness of urban management and services, and consequently improve the living standards of its citizens [9]. In a smart city scenario, it will be possible to collect real-time data concerning the potential passengers, like their requested arrival time and their current location etc. Service providers will also be able to communicate bus arrival times and other information in real-time to their passengers and drivers through web or mobile interfaces as well as through displays on-site. This will create a two-way communication system between the passenger and the service providers, which offers a number of opportunities to improve the latter's service quality.

Traditional transport services (TTS) are composed of a set of lines that follow predefined routes and timetables. One crucial advantage of TTS is their low operational costs, arising from their ability to transport large groups of passengers collectively. Furthermore, TTS are efficient in high demand areas as they can sufficiently meet the constant demand of all passengers without requiring routes or timetables to be customized ([10], [11]). The predictable nature of TTS due to fixed timetables makes them highly accessible to most commuters. However, the inflexibility of TTS can also be viewed as a limitation, since it renders TTS inadequate in settings where demand for transportation is sparse and/or constantly changing. Furthermore, traditional bus services are developed using historical, aggregate data and therefore might not cater to the requirements caused by less predictable, ever-changing and more geographically and temporally dispersed travel patterns of citizens today [12]. As such, service providers are unable to effectively handle this demand with their current capabilities and resources. One of the consequences is that passengers utilizing TTS frequently experience long travel times which may result in frustration and client dissatisfaction.

Due to the limitations of TTS, on-demand transportation services (ODTS) are popping up to deal with this volatile demand for transportation. On the one hand, ODTS operate only when there is demand for transportation and are thus better equipped to meet the passenger's expectations [13]. On the other hand, ODTS are often quite expensive and are not applicable on a large scale due the high complexity of scheduling the passenger's requests for a ride and the routing of the vehicles. These services also do not offer a solution to deal with unknown demand for transportation, i.e., passengers that do not request a ride because they are unfamiliar with the service, but could still benefit from such a ride. There seems

to be a window of opportunity for public transport services that combine characteristics of both TTS and ODTS.

This research will focus on developing an optimization model for a demand-responsive feeder service (DRFS), integrating the positive characteristics of TTS as well as those of ODTS. A feeder service is defined as a service that transports passengers, from typically sparsely populated areas, to areas with a high demand for transportation, where the passengers can continue their journey. All passengers will thus have the same destination, but different origins. The DRFS can have one or multiple bus lines and each bus line serves two sets of bus-stops: mandatory stops and (clustered) optional stops. The mandatory stops need to be visited by each bus serving a certain line. The optional stops are only visited by a bus when a client, with origin within walking distance of this stop, needs to be picked-up. The service therefore assumes that users can order a ride on the bus line, like e.g., in a taxi. This implies that the buses can have different routes and different timetables according to the demand, while there is still a factor of predictability due to the mandatory stops. The mandatory stops serve as a safety-net for the unknown demand, as potential passengers may simply take the bus at one of the mandatory stops, even without making a formal request for transportation.

In the DRFS, potential passengers make a request for transportation to the transportation hub, by stating their current location and their latest arrival time. We assume that all requests for transportation are known before the start of the first bus. It is further assumed that each line can be optimized separately so only a single line is considered.

In conclusion, this paper proposes a new type of bus service. To optimize the performance of the service, a Mixed Integer Problem (MIP) is developed. However, it was found that the model could not be solved within a reasonable time for large, realistic instances. Therefore, to shorten the runtimes, two techniques are implemented: separation of sub-tour elimination constraints (SSEC) and column generation (CG).

In the next section, a literature review on public transport services is presented. In Section 3, the optimization model related to the DRFS and a mathematical model are presented. In Section 4, three different approaches to optimize the operation of the feeder service are presented. Section 5 discusses the results for several instances, obtained by optimizing the service. In the last section, conclusions are drawn and plans for future research are discussed.

2 Literature Review

The planning of traditional transport services (TTS) happens in several stages, in which many strategical, tactical and operational decisions are taken ([14], [15]). Figure 1 illustrates this process. Clearly, the decisions in earlier stages influence the decisions in the next stages. Long-term, strategical decisions are taken in stage 1. This stage deals with the design of the infrastructure where, for example, the bus stop locations are defined and the type of vehicles is determined. Tactical, mid-term decisions are taken in stage 2 and stage 3. In stage 2, line planning and frequency setting are determined. Line planning consists of defining bus lines such that a given demand for transportation is satisfied. A bus line is a sequence of bus

stops that is operated by a fleet of buses. Much research has been devoted to line planning, an extensive overview is given by Schöbel [16], and Iliopoulou et al. [17] review applications of meta-heuristics for line planning problems. Frequency setting determines how many buses drive along each line during a certain amount of time. In stage 3, the exact arrival times of the buses at every bus stop of the bus line are determined [10]. Decisions on an operational level are made in stage 4 and stage 5. In stage 4, vehicle scheduling defines the assignment of vehicles to bus lines in such a way that all planned trips can be fulfilled. In stage 5, duty scheduling defines the assignments of drivers to vehicles. Finally, in stage 6, real time control deals with any unpredictability during operation, like traffic jams or strikes for example. These stages are increasingly integrated and jointly optimized over the years, in order to improve the global planning as a whole. Timetabling and vehicle scheduling have been integrated by Carosi et al. [18] and solved with a meta-heuristic, for example. Schöbel[19], on the other hand, develops an eigenmodel to optimize line planning, timetabling and vehicle scheduling together.

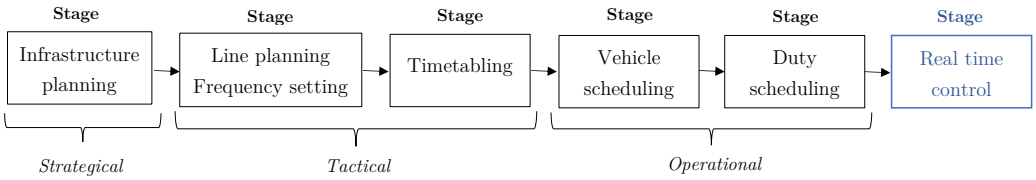

Figure 1: Planning in TTS

The popularity of on-demand transport services (ODTS) has been rising over the years, with several services and models being developed in the last decades. Dynamic ride-sharing services aim to bring together travelers with similar itineraries and time schedules. Agatz [20] gives a review on different operations research models that have been developed in the literature. Classic on-demand services are ordered through a phone call. The dial-a-ride problem (DARP) is a prime example of an optimization model for such services. In the DARP, vehicle routes need to be defined in order to pick-up and drop-off a number of passengers, with the objective to minimize the total cost of transportation. Generally speaking, pick-ups and drop-offs can take place anywhere in the route, i.e., door-to-door service, and each pick-up or drop-off has a time window. An extensive overview of the many variants of the DARP are given by Molenbruch et al. [21]. ODTS also consist of services that are not meant for public transport, for example the school bus routing problem (SBRP). Here, a list of possible bus stops is given as well as a certain number of buses. Each student is assigned to one bus stop and one bus. The objective is to transport students to school while optimizing the bus routes that visit the assigned bus stops [22]. The SBRP can be seen as a vehicle routing problem with stop assignment, which adds an extra layer of complexity to the problem. In the literature, additional constraints are often considered as well, such as bus route scheduling or school bell adjustment. The former specifically considers time windows for arriving at schools and the latter considers the start time or end time of the school.

Services that combine some characteristics of ODTS and TTS have been gaining attention. A large variety of such services exists. For example, passengers can be picked up either from their homes or from serviced bus stops, and routes can be either partially or completely fixed. In some cases, one vehicle is available to serve a dedicated low demand area to bring the passengers to a terminal stop of the fixed-route public transport service [23]. In the

USA, such services have been successfully implemented in recent years [24]. A relevant example of a service that integrates characteristics from TTS and ODTS is the Mobility Allowance Shuttle Transit (MAST) service [25]. In this service, vehicles have a fixed set of bus stops they always need to visit, i.e., a fixed path, and these stops also have fixed timetables. However, the vehicles may deviate from the fixed path. The customers that are served outside of the fixed path are served at their desired location and need to be within a certain radius from the fixed path in a so-called "zone". This service combines the high flexibility of door-to-door services with a fixed main route. This concept has been applied to feeder services as well [26]. Another type of feeder service with flexible characteristics is the so-called demand responsive connector (DRC). In the DRC, buses transport passengers from their origin location to transfer hubs within a pre-defined service area ([27], [28]). In "customized bus" services, the planning of the service is an iterative and interactive process strongly involving the potential passengers. Stops, lines, timetables, etc., are proposed by an operator and then modified until passengers are satisfied [29]. In "variable-type" services, a classical service operates during peak hours and an on-demand service, where skipping fixed routes or portions of a fixed route is possible, operates during low-demand periods [30]. In the 'variable-type" or "customized bus" services, the service provider decides beforehand where, when and how a bus will respond to customer requests. Both services ignore additional information and communication possibilities that are present in a smart city. Fu et al. [31] implement a real-time scheduling model with dynamic stop skipping. This model limits itself to optimizing the schedule of the vehicles just before departure from the depot. The DRFS developed in this paper operates in the same way as Fu et al. [31]. The routing and the schedules of the buses together with the bus stop assignment are optimized a certain amount of time before the departure of the first bus.

Services with flexible characteristics, like MAST, the "variable-type" services and the "customized bus", have more success in low demand areas with a sparse population, while TTS services on the other hand thrive in high-demand and densely populated areas. When and where to use which service is further discussed by Li [11]. Furthermore, research has suggested that an appropriate integration of these services could enhance mobility and increase the use and efficiency of public transport [32].

The DRFS, which is presented in this research, resembles the SBRP and the MAST the most. Just as in the SBRP, the location of the bus stops are predefined and all passengers have the same destination. In the SBRP, however, all passengers in a single bus have the same desired arrival time whereas in the DRFS the desired arrival times differ per passenger. Just as in MAST services, the DRFS has a fixed route where it can deviate from. The main difference is that MAST services provide a door-to-door service to some customers within a certain radius, while the DRFS groups passengers in a number of bus stops. The schedules for the fixed route are also predefined in MAST services, limiting the time they can devote to deviating from the main route to provide the door-to-door service. This is not the case for the DRFS. In contrast to the DRFS, most MAST optimization models do not consider the capacity of the buses and not all passengers have the same destination.

3 Problem description

In this section, the demand-responsive feeder service (DRFS) is described in detail. First, the feeder service is explained and the setting of the problem is determined. Next, an optimization model is defined in more formal terms. Finally, a mathematical model of the optimization problem is presented.

3.1 Description of the demand-responsive feeder service

The feeder service is situated in a residential area, or in general, an area with low demand for transportation. The bus lines of this service are designated shuttle buses that bring the inhabitants of this residential area to transportation hubs or to a nearby city center, i.e., the destination for all passengers is the same. Since all bus lines in such a feeder service are independent, a single bus line is considered, operated by a fleet of vehicles. The set of bus stops S of the bus line consists of: a set of N mandatory stops $F = \{m_0, m_1, \dots m_{N-1}\}$ and a set of optional stops $O = \{o_{0,0}, o_{0,1}, \dots, o_{M-2,K_{M-2}}, o_{M-1,K_{M-1}}\}$. The mandatory and optional stops of a single bus line are illustrated in Figure 2.

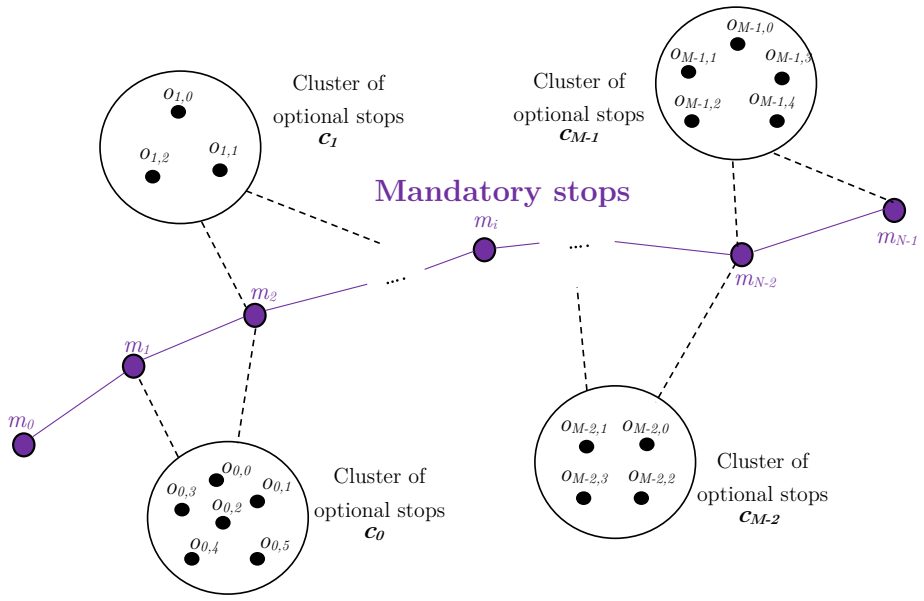

Figure 2: Bus stop structure of the demand-responsive feeder service

The mandatory stops need to be visited by each bus from the line. Mandatory stops can, for example, be placed along some highway or main road. The optional stops are grouped into different clusters. Typically, the optional stops in a cluster will be relatively close to each other and scattered across a small town or neighborhood close to the main road on which the mandatory stops are located. There are $M \leq N-1$ clusters and each cluster c_k has K_k stops. The number of stops in each cluster can vary from cluster to cluster. This implies that there are $|S| = N + \sum_{k=0}^{M-1} K_k$ bus stops in a bus line with N mandatory stops and M clusters. There can be at most one cluster between two mandatory stops. An optional stop can be written as $o_{k,l}$, where k is the index of the cluster and l is the index of the stop

within the cluster c_k.

The buses always start at the first mandatory stop m_0 and end at last mandatory stop m_{N-1}. The buses visit the mandatory stops in the sequence of their index, i.e., a bus cannot visit stop m_i before visiting stop m_{i-1}. A bus route can deviate from the route along the mandatory stops and visit some optional stops in a cluster. From a cluster, the bus can drive to the next mandatory stop, to an optional stop in the same cluster, or to an optional stop in a neighboring cluster. The main restriction is that all mandatory stops have to be visited. The set of buses is labeled B.

It is assumed that a travel matrix is given, containing the travel times between each pair of bus stops, as well as a walking time matrix, containing the walking times between each passenger's origin location and all bus stops within walking distance. Furthermore, a dwell time coefficient, a deceleration time and a acceleration time are given as well. The deceleration and acceleration parameters are the amount of time a bus loses by slowing down to stop at a bus stop and speeding up to leave a bus stop respectively. The dwell time coefficient is the amount of time a bus loses by picking up a single passenger at a bus stop, i.e., if more passengers board the bus in the same bus stop, the dwell time at that bus stop increases. Lastly, the capacity of the buses C is given as well. The total number of passengers requesting a ride during the operating horizon is $|P|$.

A passenger makes a request for a ride via a website, or a mobile application. The passengers state their origin location and desired arrival time at the destination m_{N-1}. This can, in theory, happen until the first bus of the fleet is dispatched from the depot. After all requests are received, the route of each bus is optimized for the whole trajectory and the passengers are notified regarding the departure time of their bus and which bus stop they should go to for their journey. The return trips from the transportation hub to the different stops are not explicitly considered in this paper, but it is easy to see that they result in the same optimization problem.

3.2 Optimization model

Different interdependent decisions need to be made in the DRFS optimization problem. All passengers need to be assigned to an departure bus stop, taking into account the walking time matrix. Consequently, each passenger also needs to be assigned to a bus that will bring him or her to the destination on time. The routing of each bus needs to be determined based on the optional bus stops that are assigned to passengers and that are selected for each bus. Furthermore, the departure time of each bus at the depot needs to be determined. This departure time will then determine the departure time at each bus stop. All these decisions are intertwined and affect one another. This problem can be viewed as an integration of a routing problem, an assignment problem, and a timetabling problem.

These decisions are also subject to a number of restrictions. First, the buses can have different routes according to the demand for transportation but all mandatory stops need to be visited by each bus. Secondly, the capacity of the buses is limited; no more passengers can be accepted in a bus if this capacity is reached. Thirdly, the passengers are not allowed to walk longer than a certain amount of time from their origin location to their assigned departure bus stop. Fourth, given that the requests of the passengers are known before

the optimization process starts, it is assumed that all requests need to be served. As a last restriction, all passengers need to arrive within a certain time window. This means each passenger cannot arrive a certain amount of time earlier or later than his desired arrival time. Arriving too early or too late is not allowed and arriving earlier or later than the desired arrival time is penalized in the objective function. The arrival times of the passengers are thus part of both the constraints and the objective function. This last restriction ensures that the bus arrives at the destination before the latest and after the earliest desired arrival times of all passengers onboard. It needs to be noted, however, that due to the arrival time constraints, the problem can become infeasible if the desired arrival times are spread out in time too much and not enough buses are available.

The objective of this problem is to optimize the service quality of the feeder service. This is modeled by minimizing a weighted sum of three factors. Firstly, the travel time for all buses is minimized. Shorter travel times for the buses will reduce the operating costs and imply shorter ride times for the passengers and thus improve the service quality. Secondly, the total walking time from the origin location of each passenger to the departure bus stop needs to be as low as possible. Thirdly, the absolute time difference between the desired arrival time and the actual arrival time at the destination, of each passenger, needs to be as close to zero as possible. The weights of this sum can be determined depending on the situation and on the preferences of the service provider.

The waiting time between the arrival of the passengers at their departure bus stop and the scheduled departure time of the bus at that bus stop is not considered. In this model, it is assumed that there are no significant delays and that the passengers will make a request beforehand. This means passengers can plan to arrive at the bus stop on time without incurring longer waiting times. Furthermore, the travel time of each bus is chosen over the travel time of each passenger. This is done in order to optimize the routes of the buses rather than the onboard time of each passenger individually. Optimal bus routes, however, also imply shorter travel times for passengers onboard. As discussed in Section 1, the onboard times are typically less important to customers compared to the reliability of the service and arrival time at their destination. Service reliability is inherently part of this service as it deals with passenger requests and a timely arrival at the destination is covered by the third component of the objective function.

3.3 Mathematical Model

For the sake of clarity, the notations of the different sets, parameters and decision variables are listed in Table 1.
The objective z of the mixed integer programming model is to minimize a weighted sum of three terms related to the service quality. The parameters c_1, c_2 and c_3 are weights given to these terms and can be determined by the user. The first term (2) calculates the travel time of each bus. It includes the acceleration and deceleration times of the buses when they leave a bus stop and when they stop at a bus stop respectively. The dwell time of the buses at a bus stop is also taken into account and is dependent on the number of passengers that are picked up at the stop. The second term (3) minimizes the walking time of each passenger from their origin location to their assigned departure bus stop. The last term (4) is the time each passenger will arrive at the destination before or after his or her desired arrival time.

Sets	
B	Set of buses
S	Set of all bus stops
O	Set of optional bus stops
F	Set of mandatory bus stops
P	Set of passengers using the service during the optimization horizon
Parameters	
K_k	Number of optional bus stops in cluster k
M	Number of clusters
tt_{ij}	Travel time from bus stop $i \in S$ to bus stop $j \in S$
τ	Dwell time per passenger boarding
wt_{pi}	Walking time of passenger $p \in P$ to departure bus stop $i \in S$
dat_p	Desired arrival time of passenger $p \in P$ at the destination bus stop m_{N-1}
δ	Average acceleration and deceleration time of a bus
d_w	Maximum value for individual walking time
d_{late}	Maximum value for arriving late
d_{early}	Maximum value for arriving early
C	Capacity of the buses
c_1	Relative weight given to the travel time of the buses
c_2	Relative weight given to the walking time of the passengers
c_3	Relative weight given to the absolute difference in desired and actual arrival time of the passengers
$M_{0,1}$	Large natural numbers for the Big M constraints
Decision Variables	
x_{bij}	0-1 variables determining if bus $b \in B$ visits bus stop $j \in S$ after visiting bus stop $i \in S$
y_{pbi}	0-1 assignment variables which assume value 1 if passenger $p \in P$ is assigned to bus $b \in B$, with departure bus stop $i \in S$
a_p	Arrival time of passenger $p \in P$ in destination bus stop m_{N-1}
D_b	Departure time of bus $b \in B$ at the first mandatory bus stop m_0
la_p	$a_p - dat_p$ when passenger $p \in P$ is late
ea_p	$dat_p - a_p$ when passenger $p \in P$ is early

Table 1: Notation for the MIP

$$\text{Min} \tag{1}$$

$$z = c_1 \left[\sum_{b \in B} \sum_{i \in S} \left(\sum_{j \in S} (tt_{ij} + \delta) x_{bij} + \tau \sum_{p \in P} y_{pbi} \right) \right] \tag{2}$$

$$+ c_2 \left[\sum_{b \in B} \sum_{i \in S} \sum_{p \in P} wt_{pi} y_{pbi} \right] \tag{3}$$

$$+ c_3 \left[\sum_{p \in P} (la_p + ea_p) \right] \tag{4}$$

The first group of constraints deals with the routing of the buses. Constraints (i) ensure that, for optional bus stops, at most one arc enters or leaves any stop. Constraints (ii) ensure that, for each mandatory stop, exactly one arc enters or leaves. If an arc enters the stop, there must be an arc leaving the stop and vice versa (iii). The only exceptions are m_0, where exactly one arc leaves and none enter, and m_{N-1}, where exactly one arc enters and none leave. Constraints (iv) and constraints (v) ensure that no bus ever has stop m_0 as a successor or stop m_{N-1} as a predecessor. Constraints (vi) prevent buses from returning to the same bus stop they have left from. Constraints (vii) are sub-tour elimination constraints.

$$\sum_{j \in S} x_{bij} \leq 1 \qquad \forall\ i \in O,\ b \in B \quad \text{(i)}$$

$$\sum_{j \in S} x_{bij} = 1 \qquad \forall\ i \in F,\ b \in B \quad \text{(ii)}$$

$$\sum_{l \in S} x_{bil} = \sum_{l \in S} x_{bli} \qquad \forall\ i \in S_{0,N-1},\ b \in B \quad \text{(iii)}$$

$$\sum_{i \in S} x_{bi0} = 0 \qquad \forall\ b \in B \quad \text{(iv)}$$

$$\sum_{i \in S} x_{bN-1i} = 0 \qquad \forall\ b \in B \quad \text{(v)}$$

$$x_{bii} = 0 \qquad \forall\ b \in B,\ i \in S \quad \text{(vi)}$$

$$\sum_{i \in S_t} \sum_{j \in S_t} x_{bij} \leq |S_t| - 1 \qquad \forall\ S_t \subset S, S_t \neq \emptyset, b \in B \quad \text{(vii)}$$

A second group of constraints deals with capacities or threshold values. Constraints (viii) ensure that no passenger needs to walk more than a predefined maximum value d_w. Constraints (ix) regulate the number of passengers on each bus, so that buses cannot transport more passengers than a given capacity. Constraints (x) and (xi) ensure that all passengers arrive within the time window $[dat_p - d_{early}, dat_p + d_{late}]$.

$$\sum_{b\in B}\sum_{i\in S} wt_{pi} y_{pbi} \leq d_w \qquad \forall\ p \in P \quad \text{(viii)}$$

$$\sum_{p\in P}\sum_{j\in S} y_{pbi} \leq C \qquad \forall\ b \in B \quad \text{(ix)}$$

$$la_p \leq d_{late} \qquad \forall\ p \in P \quad \text{(x)}$$

$$ea_p \leq d_{early} \qquad \forall\ p \in P \quad \text{(xi)}$$

Constraint (xii) defines the variables ea_p and la_p as positive deviations between the actual arrival a_p time and the desired arrival time dat_p of the passengers. Given the objective function, one of the two variables la_p or ea_p will be zero for each passenger p in the optimal solution.

$$dat_p - a_p + la_p - ea_p = 0 \qquad \forall\ p \in P \quad \text{(xii)}$$

Constraints (xiii) and (xiv) define the variable a_p, the arrival time of a passenger p at the destination. These constraints are big M constraints that link the passengers to the buses. Constraints (xiii) define a_p, in case the passenger arrives late or just on time, i.e., if $a_p - dat_p \geq 0$. Constraints (xiv) define a_p, in the case the passenger arrives early, i.e., if $a_p - dat_p < 0$. Constraints (xv) are big M constraints that link the y and x variables. A passenger with a certain departure bus stop i is only assigned to a bus that visits bus stop i. This constraint is only valid for the optional stops since the mandatory stops are always visited. If these stops were included in this constraint, it would always assign some people to the mandatory stops.

$$a_p - \sum_{i\in S}\left(\sum_{j\in S}(tt_{ij}+\delta)x_{bij} + \tau\sum_{c\in P} y_{cbi}\right) - D_b \qquad \forall\ b \in B,\ p \in P \quad \text{(xiii)}$$
$$\leq \left(1 - \sum_{i\in S} y_{pbi}\right) M_0$$

$$\sum_{i\in S}\left(\sum_{j\in S}(tt_{ij}+\delta)x_{bij} + \tau\sum_{c\in P} y_{cbi}\right) + D_b - a_p \qquad \forall\ b \in B,\ p \in P \quad \text{(xiv)}$$
$$< \left(1 - \sum_{i\in S} y_{pbi}\right) M_0$$

$$\sum_{p\in P} y_{pbi} \leq M_1 \sum_{l\in S} x_{bil} \qquad \forall\ i \in O,\ b \in B \quad \text{(xv)}$$

Lastly, a set of constraints ensures that every passenger is assigned to exactly one bus and one departure bus stop.

$$\sum_{b\in B}\sum_{i\in S} y_{pbi} = 1 \qquad \forall\ p \in P \quad \text{(xvi)}$$

The remaining constraints determine the domains of the variables.

$$y_{pbi} \in [0,1] \qquad \forall\ b \in B,\ b \in B,\ p \in P \quad \text{(xvii)}$$

$$x_{bij} \in [0,1] \qquad \forall\ b \in B,\ i \in S,\ j \in S \quad \text{(xviii)}$$

$$a_p,\ la_p,\ ea_p \in \Re^+ \qquad \forall\ p \in P \quad \text{(xix)}$$

$$D_b \in \Re^+ \qquad \forall\ b \in B \quad \text{(xx)}$$

4 Solution methods

In this section, solution methods to solve the optimization model defined in Section 3.2 are described. First, the model is solved using a commercial solver. Then, two improvements are developed in order to reduce the runtime. The first improvement is a separation of constraints, that is utilized to eliminate sub-tours. The second improvement is a combination of column generation and separation of constraints.

4.1 Using CPLEX to solve the MIP

The problem, mathematically modeled in Section 3.3, is solved using the commercial solver CPLEX. Even though this is essentially a black-box procedure, it needs to be noted that the choice of M_0 and M_1 i.e., the large constant values in the model, have a significant impact on the time required to solve the MIP. If these "big M's" are too large, the MIP will have a weak linear relaxation and consequently, a weak lower bound. A weak lower bound in turn, makes the branch-and-bound search that is required to solve this MIP with CPLEX much more time consuming. For this reason, the big M's in these constraints need to be chosen with the lowest possible value while still being larger than any possible value on the left-hand side of the inequality of these constraints [33]. To this end we choose the values for the big natural numbers M_0 and M_1 as:

$$M_0 = \max_{p \in P} dat_p + d_{late} - \left(\min_{p \in P} dat_p - d_{early} \right) \tag{5}$$

$$M_1 = C \tag{6}$$

The reasoning behind formula (5), which sets the value for M_0, is the following. In the case a passenger p is not assigned to bus b, the left-hand side of constraints (xiv) and (xiii) will be equal to the difference of the arrival time of passenger p and the arrival time of bus b, otherwise they will equal to zero. When a passenger is not assigned to a bus, the largest possible difference is the range of the time window any bus can arrive at the destination. This is the difference between a bus arriving d_{early} too early for the passenger that needs to arrive the earliest and a bus arriving d_{late} too late for the passenger that has the latest desired arrival time. The large number M_1 has a value equal to the capacity of the buses (6), since the left-hand side of constraint (xiii) counts the number of passengers that board bus b at bus stop i.

4.2 Separation of sub-tour elimination constraints

In integer programming formulations that involve routing, sub-tour elimination constraints (SECs) are used to ensure connectedness of the routes. The number of SECs is exponential in the size of the considered problem instance, since there are 2^n SECs in a graph with n

nodes. As a result, the runtime of such a model increases considerably. A common strategy to tackle this issue is to solve those SEC formulations with branch-and-cut algorithms. Here, violated SECs are identified dynamically in the course of the branch-and-bound algorithm and are subsequently added to the formulation. This identification process is commonly referred to as separation [33]. The separation of violated SECs (SSEC) is usually performed by means of minimum cut algorithms [34]. This means that as few cuts as possible are added to the formulation. The separation procedure in this research uses a minimum cut approach as well. In the SSEC, the model will be solved as usual using CPLEX but without the sub-tour elimination constraints (vii). At every instance where CPLEX finds a solution that has integer values, i.e., a feasible solution ignoring the SECs, the sub-tour identification algorithm is executed. In this algorithm, the sub-tours are identified and the corresponding cuts are added in order to prevent these sub-tours.

Algorithm 1: Sub-tour identification algorithm

```
for each bus b ∈ B do
    Calculate how many bus stops l were visited by bus b
    Keep bus stops in an array Stops
    Start in bus stop i = m_0, delete i from Stops
    Initialize array Sub, add i to Sub
    Initialize the number of visited stops: v = 1
    while a sub-tour is not detected or not all nodes in Stops are checked do
        Identify successor bus stop j of bus stop i
        Delete j from Stops
        Add j to Sub
        v = v + 1
        if Stops does not contain j then
            j has been visited already → sub-tour detected
            Stop the loop
        else if j = N − 1 then
            if v = l then
                Stop, there are no sub-tours
                Clear Sub
            else if v ≥ N − 1 then
                There are sub-tours but not found yet
                Clear Sub
                Set i=Stops[0] and set v = 1
            end
        end
    end
    if a sub-tour was detected then
        Add it to the model as a constraint using array Sub
    end
end
```

The algorithm to dynamically identify and add SECs is given in Algorithm 1. In this algorithm, the number of bus stops l visited by each bus b is calculated. Then, the algorithm will start in stop m_0 and follow the route in this intermediate solution in order to check

each stop that is visited in the solution. Once it encounters a bus stop that was already checked, it will add this route as a SEC for bus b. If the route reaches the destination m_{N-1} and the number of stops that were visited v is equal to l, then there are no sub-tours and this is a valid route. In case $v \neq l$, there is still a sub-tour in the solution, but it has not been found yet. The algorithm then starts over from a stop that is visited in the solution, but has not been checked yet. The algorithm runs until either it has checked all stops or until it has determined all sub-tours.

4.3 Column Generation

As will be shown in Section 5.3, the SSEC greatly improves the runtime for some instances. However, although the approach manages to find optimal or very good solutions quickly, it still produces solutions with large integrality gaps due to poor lower bounds. In order to remedy this issue, column generation (CG) can be utilized. In CG, only a subset of variables is considered when solving the problem and this can greatly reduce the computational time. The idea underlying CG is that a problem can be restated as a master problem (MP). The MP is a column-wise formulation of the original MIP, i.e., a formulation which can be decomposed into several columns, each corresponding with a variable. The MP is constructed in such a way that selecting the right columns, i.e., selecting which variables are non-zero in the optimal solution, optimizes the original MIP. The MP is then split into two problems; called respectively the restricted master problem (RMP) and the subproblem (SP). The RMP is the same as the MP, with the difference that only a subset of columns are included. Furthermore, the restricted master problem relaxes any integer variable. The SP, on the other hand, is a new problem created to identify a new column corresponding to a new variable.

The original MIP of the DRFS can be reformulated as MP as follows. The service provider has an unlimited number of buses that each have a predetermined timetable, route and a number of passengers onboard, and the problem is determining which buses are chosen. The only restrictions to the MP is that exactly $|B|$ buses need to be chosen and that each passenger is assigned to exactly one bus. In this formulation, the columns correspond with the buses, i.e., adding a new column to the MP is the same as adding an additional bus. Each bus is subject to the same constraints as the buses in the original MIP and the same objective is optimized. The routes, timetables, bus stop assignments and passenger assignments of an initial set of $|B|$ buses are determined using a heuristic. Afterwards, the column generation procedure starts a loop. In the first iteration of the loop, the RMP is solved using only the columns of the initial solution. Afterwards, the dual prices for each of the constraints in the RMP are obtained and utilized in the SP to compute a new column. If the reduced cost of this column is negative, it is added to the RMP in the next iteration of the loop. If the reduced cost is positive, it means the algorithm has found an optimal solution and the loop stops. After the loop stops, the MP is solved once in order to obtain the final solution. The algorithm for column generation is given in Algorithm 2.

Algorithm 2: Algorithm for column generation

```
Generate |B| columns using a heuristic and add them to RMP
while reduced cost of a new column < 0 do
    Solve RMP → dual prices
    Generate a new column with SB using the dual prices
    Add new column to RMP
end
Solve MP
```

In what follows, the algorithm for the construction of the initial solution, the RMP and the SB are discussed in more detail.

4.3.1 Construction of an initial solution

To start the column generation algorithm, an initial solution is needed first. The algorithm for the construction of the initial solution is given in Algorithm 3. The initial solution is constructed by iteratively constructing a route, a timetable and a bus stop assignment for each bus separately. The passengers are sorted according to their desired arrival time dat_p and are added to a bus in this order. The algorithm will stop adding passengers to a bus either when the bus reaches its capacity or when $\max_{p \in P} dat_p + \min_{p \in P} dat_p < d_{late} + d_{early}$. The latter ensures that all passengers reach the destination within the given time window. On each bus, passengers are then assigned to the closest stop to their location, as their departure bus stop. Afterwards, the routes of each bus are determined, taking in mind any optional stop that is assigned to a passenger in a bus. First, all the mandatory stops are added to the route of the bus. Then, all the optional stops that are visited by the bus are inserted in the existing route. Lastly, the departure time of the buses is determined so that each bus arrives within the given time window. This means the bus must arrive within the interval $\left[\max_{p \in P} dat_p - d_{early}, \min_{p \in P} dat_p + d_{late}\right] = [LB, UB]$. The best departure time is determined through a discrete enumeration; all possible departure times, when a time step of 30 seconds is utilized, are checked and the departure time that minimizes the objective is chosen.

Algorithm 3: Construction of an initial solution

Order passengers according to desired arrival time
for *each bus* $b \in B$ **do**
  while $\sum_{i \in S} \sum_{p \in P} y_{pi}^b < C$ **do**
    Add the next passenger p to bus b
    if $\left(\max_{p \in bus\, b} dat_p + \min_{p \in bus\, b} dat_p \right) < d_{late} + d_{early}$ **then**
      Remove the last passenger p
      Stop the loop
    else
      Determine and assign the closest bus stop to passenger p
    end
  end
  Add a route to bus b: visit all mandatory bus stops $m \in F$
  for $\forall o \in O$ **do**
    if *optional stop* o *is assigned to a passenger onboard bus* b **then**
      Add o to the route
    end
  end
  Calculate the best arrival time for bus b: $a^b = \min_{l \in [LB, UB]} \sum_{p \in P} |l - dat_p|$
  The best arrival time for each passenger $p \in bus\, b$ is then: $a_p^b = \sum_{i \in S} y_{pi}^b a^b$
  Calculate total travel time TT^b on bus b
  The best departure time for bus b is then: $D^b = a^b - TT^b$
end

4.3.2 Restricted master problem

The RMP for CG is given below. The variables λ^b are binary variables that determine whether a bus b is selected or not. The model has two constraints: exactly $|B|$ buses are utilized (xxi), and each passenger must be transported to the destination and is assigned to exactly one bus (xxii). The objective function of the MP is the same as the original MIP, with the difference that there are now several buses and only a subset of these buses are selected by the model. The variables λ^b are binary in the MP, but are relaxed in the RMP. In the RMP, new columns, i.e., new buses generated by the SB, are added iteratively. The number of buses that are present in the RMP at iteration it is B_{it}. It needs to be noted that in the MP and in the RMP, the only variables are λ^b. The variables in the original model are parameters here. The dual prices corresponding with constraints (xxi) and (xxii) are σ and π_p, $\forall p \in P$ respectively.

Minimize

$$z_{RMP} = \sum_{b<I} \left(\sum_{i \in S} \sum_{j \in S} c_1 \left(tt_{ij} + \delta \right) x_{ij}^b + \sum_{i \in S} \sum_{p \in P} y_{pi}^b \left(c_1 \tau + c_2 wt_{pi} \right) + \sum_{p \in P} c_3 \left(la_p^b + ea_p^b \right) \right) \lambda^b$$

S.t.

$$\sum_{b<B_{it}} \lambda^b = |B| \tag{xxi}$$

$$\sum_{b<B_{it}} \sum_{i\in S} y_{pi}^b \lambda^b = 1 \qquad \forall\ p \in P \tag{xxii}$$

$$0 \leq \lambda^b \leq 1 \qquad \forall\ b < B_{it} \tag{xxiii}$$

4.3.3 Subproblem

The subproblem is given below. The SP calculates the route, timetabling and passenger assignment of a single bus, i.e., a column of the RMP. In the objective, the reduced cost of the new bus is calculated using the dual prices σ and π_p. The SP has the same set of variables as in the original problem, except for the fact that variables only pertain to a single bus and thus do not have the b index anymore. The objective function uses the dual prices of the RMP to calculate the reduced cost of this column. The constraints of the SP are the same as in the original problem. The SP excludes the sub-tour elimination constraints because it makes use of the same separation technique as in Section 4.2. Furthermore, some additional constraints (xxxviii) and (xxxix), are added. These are big-M constraints that ensure that passengers that are not picked up by the bus do not affect the objective function. When a passenger p is not picked up by a bus b, a_p^b will be equal to dat_p^b and this will make la_p^b and ea_p^b equal to zero. Consequently, the passenger does not have an impact on the third component of the objective function. Due to the limits set in constraints (xxxii) and (xxxiii), the big M numbers M_2 and M_3 are chosen as d_{late} and d_{early} respectively.

Minimize

$$z_{SP} = \sum_{i\in S} \left(\sum_{j\in S} c_1(tt_{ij} + \delta)x_{ij} + \sum_{p\in P} y_{pi} \left(c_1\tau + c_2 wt_{pi} - \pi_p\right) \right) + \sum_{p\in P} c_3 \left(la_p + ea_p\right) - \sigma$$

S.t.

$$\sum_{\substack{j\in S \\ i<j}} x_{ij} \leq 1 \qquad \forall\ i \in O \tag{xxiv}$$

$$\sum_{j\in S} x_{ij} = 1 \qquad \forall\ i \in F \tag{xxv}$$

$$\sum_{l\in S} x_{il} = \sum_{l\in J} x_{li} \qquad \forall\ i \in S_{0,N-1} \tag{xxvi}$$

$$\sum_{i\in S} x_{i0} = 0 \tag{xxvii}$$

$$\sum_{i\in S} x_{N-1i} = 0 \tag{xxviii}$$

$$x_{ii} = 0 \qquad \forall\ i \in S \tag{xxix}$$

$$\sum_{i\in S} wt_{pi} y_{pi} \leq d \qquad \forall\ p \in P \tag{xxx}$$

$$\sum_{p\in P} \sum_{i\in S} y_{pi} \leq C \tag{xxxi}$$

$$la_p \leq d_{late} \qquad \forall\ p \in P \tag{xxxii}$$

$$ea_p \leq d_{early} \qquad \forall\ p \in P \tag{xxxiii}$$

$$dat_p - a_p + la_p - ea_p = 0 \qquad \forall\ p \in P \tag{xxxiv}$$

$$\sum_{i \in S} \left(\sum_{j \in S} (tt_{ij} + \delta) x_{ij} + \tau \sum_{c \in P} y_{ci} \right) + D - a_p \leq \left(1 - \sum_{j \in S} y_{pi} \right) M_0 \qquad \forall\ p \in P \quad \text{(xxxv)}$$

$$a_p - \sum_{i \in S} \left(\sum_{j \in S} (tt_{ij} + \delta) x_{ij} + \tau \sum_{c \in P} y_{ci} \right) - D < \left(1 - \sum_{j \in S} y_{pi} \right) M_0 \qquad \forall\ p \in P \quad \text{(xxxvi)}$$

$$\sum_{p \in P} y_{pi} \leq M_1 \sum_{l \in S} x_{il} \qquad \forall\ i \in O \quad \text{(xxxvii)}$$

$$la_p \leq M_2 \sum_{i \in S} y_{pi} \qquad \forall\ p \in P \quad \text{(xxxviii)}$$

$$ea_p \leq M_3 \sum_{i \in S} y_{pi} \qquad \forall\ p \in P \quad \text{(xxxix)}$$

$$y_{pi} \in [0, 1] \qquad \forall\ i \in S, p \in P \quad \text{(xl)}$$

$$x_{ij} \in [0, 1] \qquad \forall\ i \in S, j \in S \quad \text{(xli)}$$

$$a_p,\ la_p,\ ea_p \in \Re^+ \qquad \forall\ p \in P \quad \text{(xlii)}$$

$$D \in \Re^+ \quad \text{(xliii)}$$

5 Results

In this section, the results that are obtained with the solution methods defined in Section 4 are discussed. First, a set of instances is generated to test the optimization model. The solution of a single small instance is explained in detail afterwards. Then, the runtimes of the different solution methods are compared, including the original MIP which is solved with CPLEX. Finally, different experiments are conducted in order to determine which parameters of the instances contribute the most to the runtime of each solution method.

5.1 Test instances

To test the different models and algorithms, a set of instances is created. These instances are randomly generated and are listed in Table 2. The instances vary in number of buses $|B|$, requests $|P|$ and bus stops $|S|$. Two further assumptions are that the number of optional stops per cluster K is the same in each cluster and that there is exactly one cluster in between two mandatory stops. In Table 2, the instances are ordered according to the number of variables V in the model. This parameter V is an indication of the size of the instances. The largest instance, I_{14}, is considerably larger than the rest of the instances.

Instance	|B|	|F|	K	|S|	|P|	V
I_1	2	3	3	9	12	1226
I_2	3	3	3	9	12	1335
I_3	2	4	3	13	16	3170
I_4	3	4	3	13	16	3379
I_5	2	5	3	17	20	6522
I_6	3	5	3	17	20	6863
I_7	4	5	3	17	20	7204
I_8	3	6	3	21	25	12678
I_9	4	6	3	21	25	13204
I_{10}	3	6	3	21	30	15213
I_{11}	4	6	3	21	30	15844
I_{12}	4	7	3	25	30	21844
I_{13}	5	7	3	25	30	22595
I_{14}	5	10	3	37	40	62285

Table 2: Test instances of the DRFS

For all instances, the parameters that are used are listed in Table 3. All instances, as well as the solutions discussed in this paper are available in detail online: `https://www.mech.kuleuven.be/en/cib/drbp#section-0`.

Parameter	Value	Unit
C	15	passengers
d_{late}	300	s
d_{early}	900	s
d_w	1200	s
δ	30	s
τ	5	$\frac{\text{s}}{\text{passenger}}$

Table 3: Parameters

5.2 Analysis of a solution

To illustrate what a solution of the DRFS looks like, the optimal solution of instance I_2 is discussed. The routing of the buses, the assignment of passengers to buses and departure bus stops are shown in Figure 3. The solid lines represent the routing of the buses. Each bus has a different route and can be distinguished by the color of the line. Passengers are denoted by a p followed by a number, for example, passenger 1 is denoted as p_1. The passengers walk from their origin location to their departure bus stop, which is displayed as a dotted line. The color of the dotted line of a passenger has the same color as the bus the passenger is boarding.

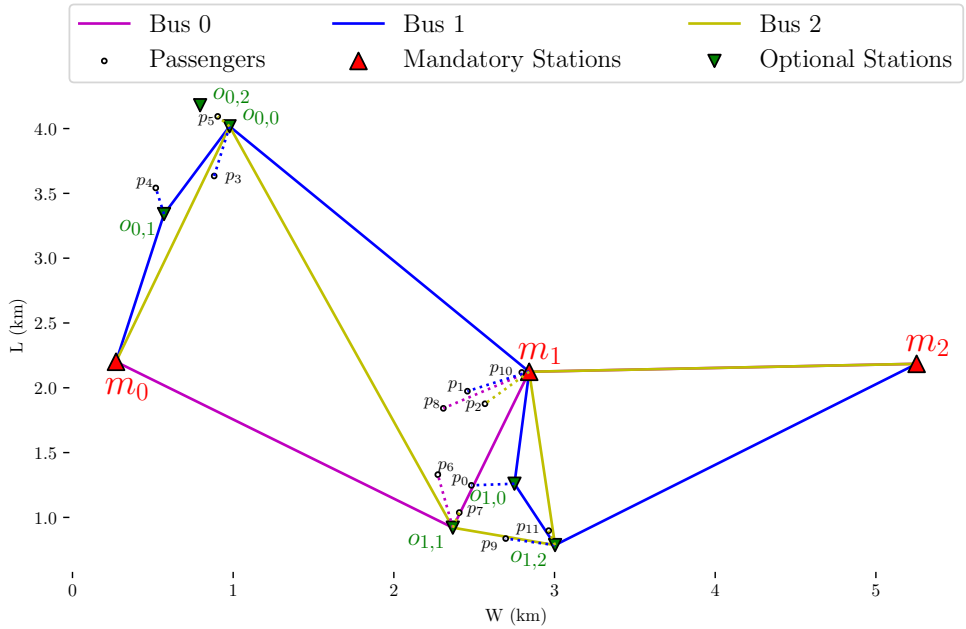

Figure 3: Routes and assignment of passengers to buses and departure bus stops

The timetable for the solution of this instance is shown in Table 4. The departure time D (in minutes), relative to an arbitrary reference point, of each bus is shown in the second column. The passengers that are picked up at a certain bus stop are shown in column PU.

Bus 0			*Bus 1*			*Bus 2*		
Stop	**D(min)**	**PU**	**Stop**	**D(min)**	**PU**	**Stop**	**D(min)**	**PU**
m_0	2		m_0	0		m_0	11	
$o_{1,1}$	7	p_6	$o_{0,1}$	3	p_4	$o_{0,0}$	16	p_5
m_1	11	p_8	$o_{0,0}$	5	p_3	$o_{1,1}$	23	p_7
m_2	15		m_1	11	p_1, p_{10}	$o_{1,2}$	25	p_{11}
			$o_{1,0}$	13	p_0	m_1	28	p_2
			$o_{1,2}$	15	p_9	m_2	32	
			m_2	21				

Table 4: Timetable and passenger pick ups

In Figure 4, the difference between the desired arrival times dat_p and the actual arrival times a_p of the passengers is displayed. A negative value corresponds to the passenger arriving later than he or she desired and positive values indicate early arrivals. It can be seen that the upper limits d_{late} and d_{early} are fully respected in the solution. A characteristic of the optimal solution is that, in each bus, at least one passenger arrives exactly on time.

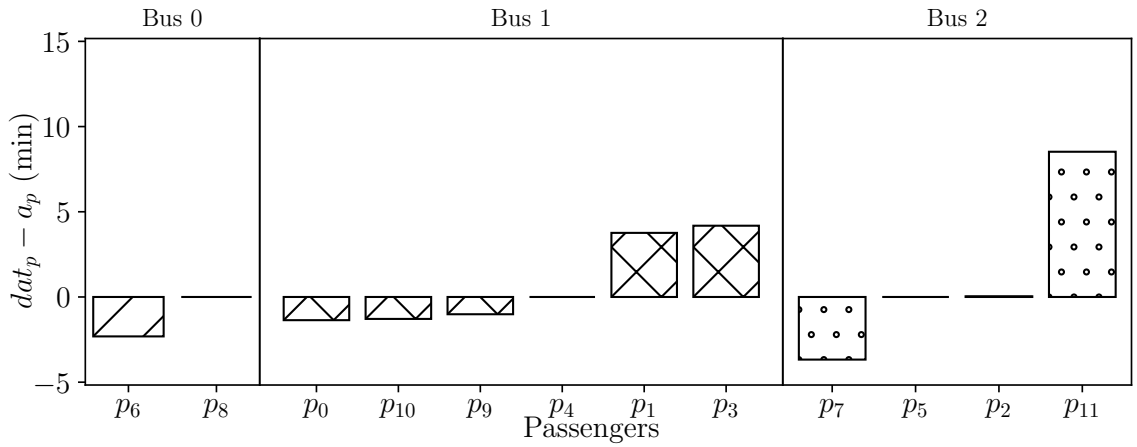

Figure 4: Difference in actual arrival time and desired arrival time

The different walking times, the onboard ride times and the user journey times (UJT) for each passenger are shown in table 5. The onboard ride time of a passenger is the time that the passenger spends onboard the bus, the UJT is the sum of walking time and the onboard ride time. Evidently, the threshold for walking time d_w is respected and the walking times are relatively small. Furthermore the UJT of any passenger never exceeds 24 minutes and the lowest UJT is a little less than 10 minutes. The average UJT is 15.56 minutes.

Passenger	**Bus**	**Walking time (s)**	**Onboard ride time (s)**	**UJT (s)**	$\mathbf{dat_p - a_p}$ **(s)**
p_6	0	502	443	945	-138
p_8		816	257	1073	0
p_0	1	280	464	744	-81
p_1		534	590	1124	225
p_3		478	963	1441	250
p_4		253	1096	1349	0
p_9		357	364	721	-60
p_{10}		50	590	640	-77
p_2	2	522	258	780	2
p_5		150	971	1121	0
p_7		156	532	688	-220
p_{11}		151	428	579	511

Table 5: Journey times of the passengers

With this information, one can determine the journey of each passenger. For example, passenger p_4 is assigned to bus stop $o_{0,1}$, and has to arrive at this stop before minute 3. The passenger is assigned to bus 1, and will arrive at the destination m_2 at minute 21, making his onboard time 18 minutes. Together with a walking time of around 4 minutes, the UJT of p_4 is 22 minutes. The passenger arrives exactly on time at the destination m_3.

5.3 Comparison of the different approaches

The models are solved using CPLEX version 12.9 on a computer with a Windows 10 Enterprise operating system, an Intel Core^{TM} i7-8850H, 2.60Ghz CPU and 16 GB of RAM. All instances are solved with the original MIP, with the original MIP using the SSEC, and with the restated model using CG and SSEC. All runtimes are limited to one hour, i.e., all algorithms stop either after the optimal solution is found, or after one hour of runtime. In case the optimal solution is not found after one hour, the best feasible solution up until that point in time is reported. If no feasible solution could be found after one hour, the objective function value is and the gap are denoted as "/" in Table 6. For the first two models, a gap is displayed as well. This is the ratio between the best feasible solution and the best lower bound found thus far. The last model does not have a gap, due to the nature of column generation; there is no way to calculate a fair lower bound using CG. In Table 6, the results of this experiment are displayed. Column $\mathbf{r_{RC}}$ shows the ratios between the number of requests and the number of available seats in the fleet of buses, in each instance. Objective values that are not optimal after one hour of runtime, are displayed in italic.

		MIP	*SSEC*	*CG + SSEC*	*MIP*	*SSEC*	*CG + SSEC*	*MIP*	*SSEC*
Inst.	$\mathbf{r_{RC}}$	**Runtime (s)**			**Objective function value (s)**			**gap**	
I_1	40,0%	1,2	0,3	4,3	3143,1	3143,1	3143,1	0,0%	0,0%
I_2	26,7%	4,1	20,3	4,1	2932,4	2932,4	2932,4	0,0%	0,0%
I_3	53,3%	27,8	1,1	27,6	4883,2	4883,2	4883,2	0,0%	0,0%
I_4	35,6%	1398	186	19,8	4446,9	4446,9	4446,9	0,0%	0,0%
I_5	66,7%	3600	2,2	74,2	*7433,6*	7294,8	7294,8	43,2%	0,0%
I_6	44,4%	3600	908	81,0	*6688,1*	6117,0	6117,0	37,4%	0,0%
I_7	33,3%	3600	3600	57,7	*6403,00*	5902,4	5902,4	32,0%	14,1%
I_8	55,6%	3600	3600	487	/	*7856,6*	7826,3	/	12,8%
I_9	41,7%	3600	3600	309	/	*7307,5*	7287,9	/	16,8%
I_{10}	66,7%	3600	3600	1822	/	*9488,8*	9483,9	/	4,5%
I_{11}	50,0%	3600	3600	1731	/	*8824,3*	8789,6	/	20,0%
I_{12}	50,0%	3600	3600	2315	/	*9071,1*	9021,1	/	21,3%
I_{13}	40,0%	3600	3600	1450	/	*8998,0*	8937,2	/	22,0%
I_{14}	53,3%	3600	3600	3600	/	*12644,4*	*12542,6*	/	22,5%

Table 6: Results of the instances for the different approaches

It can be seen that the original MIP can only solve the problem to optimality until instance I_4. Larger instances cannot be solved with this model, within one hour. Furthermore, instances larger than instance I_7 are too large for the MIP to even find a gap value. This means that the model could not find a feasible solution for the instances in one hour of runtime. When separation is applied, the problem can be solved to optimality until instance I_6. The runtimes of the models also decrease considerably. The SSEC model did find the optimal solution for instance I_7, but was unable to prove that this solution is optimal and ended with a gap of approximately 14% after one hour. The last model, using CG, could solve all instances except the last one to optimality. The model did manage to find a better feasible solution than the SSEC model for this instance.

In Figure 5, runtimes for the different models are displayed on a logarithmic scale. The runtime of the original MIP increases exponentially and monotonically with the size of the instances. The runtime of the SSEC model also increases exponentially, but the increase is less steep and this allows the model to solve more instances. For the CG model, the increase in runtime is much less steep. This allows the model to solve almost every instance within one hour. The runtimes of the CG model and the SSEC model do not increase monotonically. A possible cause for this non-monotonic increase can be found in the number of buses: the number of ways in which passengers can be assigned to buses increases steeply with the number of buses and this results in a less tight lower bound for the relaxed model. This, in turn, leads to larger gaps and to a longer search in the branch-and-bound tree to prove optimality. The separation of the SEC makes it relatively easy to obtain high-quality feasible solutions or optimal solutions in a short amount of time, but due to the poor lower bound, it takes longer to prove optimality. Further, the CG model only becomes faster than the other models when the instances increase in size. The reason for this behavior is the fact that the CG algorithm requires several iterations to converge and, in the small instances, this additional time is not sufficiently compensated by the faster convergence.

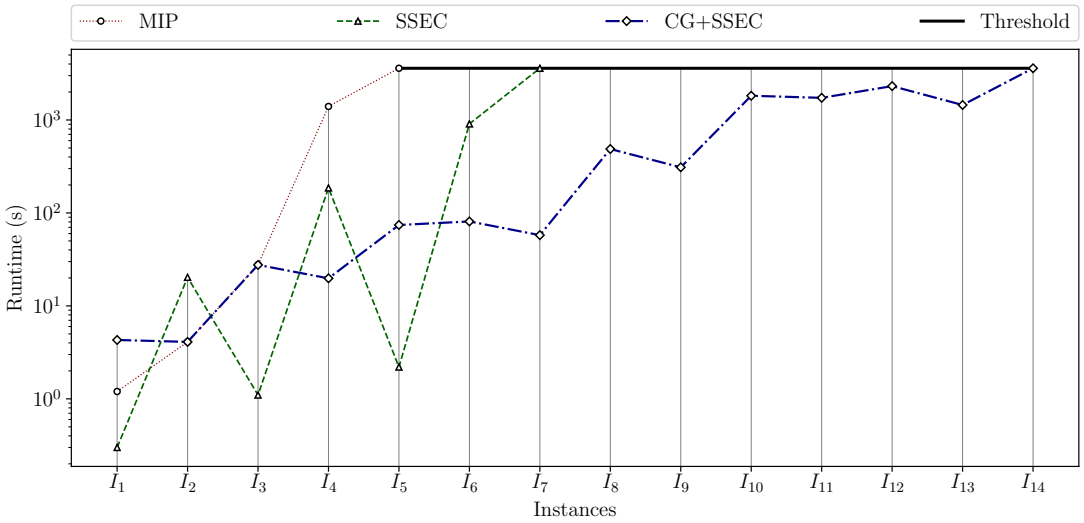

Figure 5: Runtimes of the different instances

5.4 Influence of instance parameters on runtime

In this section, the influence of the different instance parameters on the runtime are discussed. Several experiments are conducted in which four to five instances are solved with the original MIP and the two improvements. Each instance has the same set of parameters except for the parameter in question, which varies from instance to instance in order to isolate the influence of said parameter. The "base instance" I_{base} has 2 buses, 4 mandatory stops, 3 optional stops per cluster, 3 clusters, 16 requests and a capacity of 15 passengers per bus. This instance is modified to create four sets of instances. In each set, either the number of passenger requests, the number of buses, the bus capacity, or the number of optional stops per cluster varies, while the other instance parameters are constant. The choice of I_{base} is arbitrary since we are only interested in the relative increase or decrease

of runtime.

In Table 7, the influence of the number of requests on the runtime is shown. It can be seen that the runtimes of the SSEC and CG solution methods increase with the number of requests. The increase in runtime is, however, greater for the CG improvement. It needs to be noted that, in these small instances, the SSEC solution method has lower runtimes than the CG solution method. This is explained by the low number of buses: the number of ways in which passengers can be assigned to buses is limited and this results is tight lower bounds for the relaxed model. Tight lower bounds result in a fast convergence for the SSEC, while the CG still needs several iterations to converge. Results, which are not displayed in this paper, show that when the number of buses for these instances is raised from two to three, the CG becomes faster than SSEC in all instances.

			MIP	*SSEC*	*CG + SSEC*	
Inst.	**|P|**	**V**	**Runtime (s)**			$\mathbf{r_{RC}}$
$I_{r,1}$	12	2378	25,87	0,39	6,50	40,00%
$I_{r,2}$	16	3170	27,83	1,05	27,55	53,33%
$I_{r,3}$	20	3962	32,18	1,05	52,67	66,67%
$I_{r,4}$	26	5150	28,78	1,60	226,00	86,67%

Table 7: Runtimes for instances with different number of requests

In Table 8, the influence of the number of buses on the runtime is shown. Remarkably, the runtimes of the CG solution method decrease with the number of buses even though the number of variables increases. This is likely due to the nature of the restated model, in which a specific number of buses needs to be chosen among a large number of buses. When more buses are used, the number of non-zero variables in the solution of the RMP is higher, which means that it is less likely to obtain degenerate solutions in the RMP that slow down the convergence of the CG algorithm. The runtimes of the MIP and the SSEC increase steeply with the number of buses. It can be seen that CG is faster than the MIP and SSEC when 3 or more buses are utilized.

			MIP	*SSEC*	*CG + SSEC*	
Inst.	**|B|**	**V**	**Runtime (s)**			$\mathbf{r_{RC}}$
$I_{b,1}$	2	3170	27,83	1,05	27,55	53,33%
$I_{b,2}$	3	3379	485,47	45,16	17,35	35,56%
$I_{b,3}$	4	3588	3600	245,61	12,30	26,67%
$I_{b,4}$	5	3797	3600	3600	10,20	21,33%
$I_{b,5}$	6	4006	3600	3600	8,40	17,78%

Table 8: Runtimes for instances with different number of buses

In Table 9, the influence of the bus capacity is shown. It is clear that the capacity does not have a significant impact on the runtime of MIP and SSEC. CG is, however, affected by the capacity, higher capacities make the runtime increase. This can be explained by the

fact that more capacity implies more possibilities to distribute the passengers among the buses and thus more options need to be explored.

			MIP	*SSEC*	*CG + SSEC*	
Inst.	**C**	**V**	**Runtime (s)**			$\mathbf{r_{RC}}$
$I_{c,1}$	9	3170	29,84	0,86	3,68	88,89%
$I_{c,2}$	12	3170	29,38	1,16	6,47	66,67%
$I_{c,3}$	15	3170	27,83	1,05	27,55	53,33%
$I_{c,4}$	20	3170	28,84	0,81	53,56	40,00%

Table 9: Runtimes for instances with different bus capacities

In Table 10, the influence of the number of optional bus stops is shown. It can be seen that the SSEC model deals well with an increase in optional bus stops, and is faster than the CG algorithm for most of the instances. The SSEC was implemented to deal with SECs and thus to deal with an increase in bus stops. However, beyond two bus stops, the CG becomes faster than the SSEC.

			MIP	*SSEC*	*CG + SSEC*	
Inst.	**K**	**V**	**Runtime (s)**			$\mathbf{r_{RC}}$
$I_{s,1}$	1	1058	0,32	0,28	3,89	53,33%
$I_{s,2}$	2	1970	2,02	0,36	13,88	53,33%
$I_{s,3}$	3	3170	28,41	1,05	28,32	53,33%
$I_{s,4}$	4	4658	3600	27,14	93,26	53,33%
$I_{s,5}$	5	6434	3600	3600	219,97	53,33%

Table 10: Runtimes for instances with different number of optional bus stops per cluster

It is clear that the largest influencing factors for the runtime of CG are the number of buses and the number of requests. More requests slow down the CG algorithm while more buses increase its speed. For the MIP and the SSEC, its the number of stops and the number of buses are the biggest contributors to an increase of runtime. On the other hand, the capacity and the number of requests do not have an effect on MIP and SSEC but do have an influence on the runtimes of CG.

6 Conclusion

In this paper, a new type of feeder service with both optional and mandatory bus stops is proposed together with three different exact approaches to optimally plan the operation of the buses in this service. The service is demand-responsive since it selects which of the clustered optional bus stops to visit, based on passenger requests. On the other hand, there is predictability in the mandatory bus stops, which are always visited by each bus. While passengers can have a customized service by making requests, passengers that do not make such requests can still catch a bus at a mandatory stop. The service incorporates positive characteristics from both traditional transport services and on-demand transport services,

with the aim of improving service quality.

In order to plan the operation of the buses in such a feeder service, a mixed integer programming model is developed. Solving this mathematical model with CPLEX, however, results in large runtimes for non-trivial instances of the problem. To reduce the runtimes, two different techniques are proposed: a separation of sub-tour elimination constraints and a column generation algorithm. Separation allows the model to find good, feasible solutions in a short amount of time, in comparison to the original mixed integer programming model. However, in some cases, the algorithm does not converge quickly, due to bad lower bounds, leading to high integrality gaps. To redirect the gap difficulty towards a number of variable difficulty, the column generation procedure is developed, which requires a reformulation of the problem. The separation algorithm is still used in the subproblem to increase the speed of the algorithm. All models are tested on a set of instances, with varying sizes. It is found that the column generation model outperforms the other models in most cases, and is able to successfully solve larger instances.

The main contribution of this paper is the design of a new type of feeder service. This service could improve service quality in a future, smart city scenario. Different methods to optimize this service were presented. With techniques like separation and column generation, mid-sized instances of this problem can now be solved within reasonable time. This paper limits itself to exact methods. However, further research should focus on developing heuristics to solve larger, more realistic instances. The results of the exact methods in this paper will serve as benchmarks for the solutions obtained by heuristics. Further research will focus on developing algorithms to solve this problem in a dynamic environment, where not all requests are known during the planning phase.

Acknowledgments

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

References

[1] M. Steriu, *Statistics brief - Local public transport trends in the European Union*, 2016. [Online]. Available: `https://www.uitp.org/statistics-brief-public-transport-in-the-EU` (visited on 02/04/2020).

[2] J. Hine and F. Mitchell, "Better for everyone? Travel experiences and transport exclusion," *Urban Studies*, vol. 38, no. 2, pp. 319–332, 2001. DOI: `10.1080/00420980020018619`.

[3] J. Anable, "'Complacent Car Addicts'; or 'Aspiring Environmentalists'? Identifying travel behaviour segments using attitude theory," *Transport Policy*, vol. 12, no. 1, pp. 65–78, 2005. DOI: `10.1016/j.tranpol.2004.11.004`.

[4] S. Handy, L. Weston, and P. L. Mokhtarian, "Driving by choice or necessity?" *Transportation Research Part A: Policy and Practice*, vol. 39, no. 2-3 SPEC. ISS. Pp. 183–203, 2005. DOI: `10.1016/j.tra.2004.09.002`.

[5] G. Beirão and J. A. Sarsfield Cabral, "Understanding attitudes towards public transport and private car: A qualitative study," *Transport Policy*, vol. 14, no. 6, pp. 478–489, 2007. DOI: `10.1016/j.tranpol.2007.04.009`.

[6] L. Dell'Olio, A. Ibeas, and P. Cecin, "The quality of service desired by public transport users," *Transport Policy*, vol. 18, no. 1, pp. 217–227, 2011. DOI: 10.1016/j.tranpol.2010.08.005.

[7] D. A. Hensher, P. Stopher, and P. Bullock, "Service quality - developing a service quality index in the provision of commercial bus contracts," *Transportation Research Part A: Policy and Practice*, vol. 37, no. 6, pp. 499–517, 2003. DOI: 10.1016/S0965-8564(02)00075-7.

[8] M. Friman, B. Edvardsson, and T. Gärling, "Frequency of negative critical incidents and satisfaction with public transport services. I," *Journal of Retailing and Consumer Services*, vol. 8, no. 2, pp. 95–104, 2001. DOI: 10.1016/S0969-6989(00)00003-5.

[9] D. van den Buuse and A. Kolk, "An exploration of smart city approaches by international ICT firms," *Technological Forecasting and Social Change*, vol. 142, no. May 2018, pp. 220–234, 2019. DOI: 10.1016/j.techfore.2018.07.029.

[10] A. Ceder and N. H. M. Wilson, "Bus Network Design," *Transportation Research Part B: Methodological*, vol. 208, no. 4, pp. 331–344, 1986.

[11] X. Li and L. Quadrifoglio, "Feeder transit services: Choosing between fixed and demand responsive policy," *Transportation Research Part C: Emerging Technologies*, vol. 18, no. 5, pp. 770–780, 2010. DOI: 10.1016/j.trc.2009.05.015.

[12] J. D. Nelson, S. Wright, B. Masson, G. Ambrosino, and A. Naniopoulos, "Recent developments in Flexible Transport Services," *Research in Transportation Economics*, vol. 29, no. 1, pp. 243–248, 2010. DOI: 10.1016/j.retrec.2010.07.030.

[13] M. J. Alonso-González, T. Liu, O. Cats, N. Van Oort, and S. Hoogendoorn, "The Potential of Demand-Responsive Transport as a Complement to Public Transport: An Assessment Framework and an Empirical Evaluation," *Transportation Research Record*, vol. 2672, no. 8, pp. 879–889, 2018. DOI: 10.1177/0361198118790842.

[14] A. Ceder, "Introduction into transit service planning," in *Public transit planning and operation: modeling, practice and behavior*, second, Boca Ranton: CRC Press, 2016, ch. 1, pp. 1–21.

[15] R. M. Lusby, J. Larsen, and S. Bull, "A survey on robustness in railway planning," *European Journal of Operational Research*, vol. 266, no. 1, pp. 1–15, 2018. DOI: 10.1016/j.ejor.2017.07.044.

[16] A. Schöbel, "Line planning in public transportation: Models and methods," *OR Spectrum*, vol. 34, no. 3, pp. 491–510, 2012. DOI: 10.1007/s00291-011-0251-6.

[17] C. Iliopoulou, K. Kepaptsoglou, and E. Vlahogianni, "Metaheuristics for the transit route network design problem: a review and comparative analysis," *Public Transport*, vol. 11, no. 3, pp. 487–512, 2019.

[18] S. Carosi, A. Frangioni, L. Galli, L. Girardi, and G. Vallese, "A matheuristic for integrated timetabling and vehicle scheduling," *Transportation Research Part B: Methodological*, vol. 127, pp. 99–124, 2019. DOI: 10.1016/j.trb.2019.07.004.

[19] A. Schöbel, "An eigenmodel for iterative line planning, timetabling and vehicle scheduling in public transportation," *Transportation Research Part C: Emerging Technologies*, vol. 74, pp. 348–365, 2017. DOI: 10.1016/j.trc.2016.11.018.

[20] N. Agatz, A. Erera, M. Savelsbergh, and X. Wang, "Optimization for dynamic ride-sharing: A review," *European Journal of Operational Research*, vol. 223, no. 2, pp. 295–303, 2012. DOI: 10.1016/j.ejor.2012.05.028.

[21] Y. Molenbruch, K. Braekers, and A. Caris, "Typology and literature review for dial-a-ride problems," *Annals of Operations Research*, vol. 259, no. 1-2, pp. 295–325, 2017. DOI: 10.1007/s10479-017-2525-0.

[22] W. A. Ellegood, S. Solomon, J. North, and J. F. Campbell, "School bus routing problem: Contemporary trends and research directions," *Omega (United Kingdom)*, vol. 95, p. 102 056, 2020. DOI: 10.1016/j.omega.2019.03.014.

[23] M. G. Speranza, "On-demand public transportation," no. January, pp. 1–26, 2015.

[24] J. F. Potts, M. A. Marshall, E. C. Crockett, and J. Washington, "Best Practices of Successful Flexible Public Transportation Services," in *Transit Cooperative Research Program (TCRP) Report 140: A Guide for Planning and Operating Flexible Public Transportation Services*, Washington, D.C.: Transportation Research Board, 2010, ch. 4, pp. 42–88.

[25] L. Quadrifoglio, M. M. Dessouky, and F. Ordóñez, "Mobility allowance shuttle transit (MAST) services: MIP formulation and strengthening with logic constraints," *European Journal of Operational Research*, vol. 185, no. 2, pp. 481–494, 2008. DOI: 10.1016/j.ejor.2006.12.030.

[26] X. Lu, J. Yu, X. Yang, S. Pan, and N. Zou, "Flexible feeder transit route design to enhance service accessibility in urban area," *Journal of Advanced Transportation*, no. 50, pp. 507–521, 2015.

[27] X. Li, "Optimal design of demand-responsive feeder transit services," Ph.D. dissertation, Texas A&M University, 2009.

[28] A. Ceder, "Integrated smart feeder/shuttle transit service: simulation of new routing strategies," *Journal of Advanced Transportation*, vol. 47, no. June 2010, pp. 595–618, 2013.

[29] T. Liu and A. Ceder, "Analysis of a new public-transport-service concept: Customized bus in China," *Transport Policy*, vol. 39, no. 2015, pp. 63–76, 2015. DOI: 10.1016/j.tranpol.2015.02.004.

[30] M. Kim and P. Schonfeld, "Conventional, flexible and variable-type bus services," *Journal of Transportation Engineering*, pp. 263–273, 2012.

[31] L. Fu and Q. Liu, "Real-Time Optimization Model for Dynamic Scheduling of Transit Operations," *Transportation Research Record*, no. 1857, pp. 48–55, 2003. DOI: 10.3141/1857-06.

[32] M. Stiglic, N. Agatz, M. Savelsbergh, and M. Gradisar, "Enhancing urban mobility: Integrating ride-sharing and public transit," *Computers and Operations Research*, vol. 90, pp. 12–21, 2018. DOI: 10.1016/j.cor.2017.08.016.

[33] L. Wosley, *Integer Programming*. New York, United States: John Wiley & Sons Inc, 1998.

[34] M. Fischetti and P. Toth, "A polyhedral approach to the asymmetric traveling salesman problem," *Management Science*, vol. 43, no. 11, pp. 1520–1536, 1997. DOI: 10.1287/mnsc.43.11.1520.